Novel View Synthesis with limited number of cameras

Vladimír Hlačina
Institute of Computer Science and Mathematics
Slovak University of Technology
Bratislava, Slovak Republic
xhlacina@stuba.sk

prof. Ing. Gregor Rozinaj, PhD.
Institute of Multimedia Information and Communication Technologies
Slovak University of Technology
Bratislava, Slovak Republic
gregor.rozinaj@stuba.sk

***Abstract*—This article focuses on synthesize new views from a limited number of cameras, typically 2 to 4, using advanced image fusion and perspective transformation techniques. Furthermore, this work studies current methods of view synthesis in order to identify their strengths and weaknesses in the context of a limited number of input cameras. The main goal was to propose an algorithm that will be optimized for efficient fusion of views from a small number of cameras, while emphasizing the quality of the resulting images, the speed of processing, and the scalability of the solution. The proposed algorithm was then compared with another existing method, that solves the mentioned problem. The conclusion consists of evaluating the results of each method and describing their strengths and weaknesses.**

***Keywords*—Novel View Synthesis, computer vision, depth map, Gaussian Splatting**

I. Introduction

In recent decades, information technology has seen rapid development. Modern digital images contain rich data useful for applications like object detection, or view synthetis [1]. This visual information serves as the foundation for advanced computer vision techniques. One promising area is Novel View Synthesis (NVS), which generates new perspectives of a scene from existing images by leveraging spatial relationships and visual cues. Our paper implements an innovative perspective on this topic while leveraging strengths of several traditional methods [2], [3]. The paper is structured as follows: Section II reviews related works; Section III introduces our proposed method; Section IV discusses the experimental setup; Section V analyzes the results; and Section VI concludes the paper.

II. Novel View Synthesis process

In general, the process of NVS from a set of images consists of four consecutive steps:

1) **Feature Matching** - Extract and match visual elements between input images to establish correspondence.
2) **Camera Setup** - Determine camera positions, orientations, and parameters for each input image.
3) **Scene Reconstruction** - Build a 3D representation of the scene geometry from the matched features.
4) **View Synthesis** - Render new perspectives by projecting the 3D scene from desired viewpoints with appropriate texturing.

III. Compared Methods

This section explores two methods for synthetizing new views:

A. Depth-Guided Frontal Face Synthesis

Reconstructs frontal faces by extracting depth-guided foregrounds from multiple viewpoints, aligning them using facial landmarks, and blending facial regions. The method uses MiDaS depth estimation and landmark-based alignment for accurate feature correspondence.

B. 3D Gaussian Splatting

Represents scenes as 3D Gaussian primitives that are efficiently rendered through alpha-blended splatting, enabling real-time novel view synthesis. Unlike NeRF, it achieves fast rendering by rasterizing explicit Gaussians instead of neural network queries [4].

IV. Proposed Solution

The first method, Depth-Guided Multi-View Face Synthesis, was implemented using the pretrained MiDaS (*Multiple Depth Estimation Accuracy with Single Network*) neural network for depth estimation [2]. This approach utilizes Dlib's facial landmark detector with a 68-point model for alignment and custom region blending algorithms implemented with OpenCV (*Open Source Computer Vision*) library. The second method, 3D Gaussian Splatting, was tested using the Jawset Postshot software that enables rapid generation of photorealistic 3D scenes through optimized neural radiance field training.

A. Depth-Guided Multi-View Face Synthesis

At the beginning of our algorithm, the set of images from multiple views are read in color format and equalized. This is an essential step for the quality of the following foreground extraction.

1) Pre-Processing and depth map estimation: As part of the preprocessing, it was necessary to modify the input images in order to enhance their features. We applied adaptive histogram equalization (CLAHE) to enhance the contrast in images characterized by uneven lighting. The necessity of preprocessing depends on the complexity of the scene; however, it is generally regarded as a beneficial practice. For instance, the

more complex real-world image illustrated in Figure 1 exhibits a less detailed depth map, revealing noticeable artifacts.

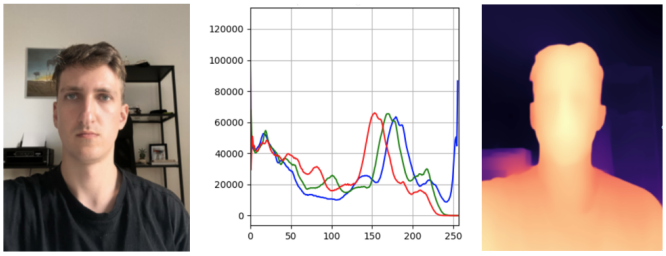


Figure 1. Example down view with respective color histogram and resulting depth map.

Following the implementation of CLAHE, the histogram depicted in Figure 2 exhibits a balanced distribution, leading to a clearer representation of the resulting depth map.

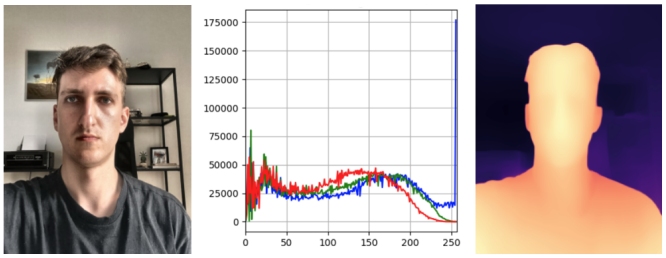


Figure 2. Example down view, color histogram and resulting depth map after applying CLAHE.

It's important to note that this color enhancing technique is just a temporal step helping to create more precise foreground extracting mask. To accurately depict the frontal perspective of the scene with true-to-life colors, we will extract the foreground from the original images rather than utilizing the equalized versions.

2) Extracting Foreground: The subsequent phase of the process involves the extraction of the foreground utilizing the estimated depth map. Given that the depth map is not inherently binary, it is essential to convert this image into a binary format. The binary map is generated by employing the formula 1.

$$binary\_map = depth\_map > threshValue \quad (1)$$

The formula implies that if the intensity value of a pixel in the depth map is greater than a chosen threshold value, it is stored as a value of 1 in the binary map. The result is a mask composed of values $[0, 1]$. By applying the $cv.bitwise\_and$ operation, we can separate foreground area from the unwanted background.

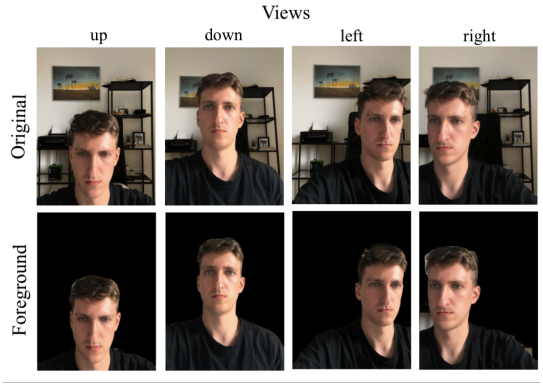


Figure 3. Example of foreground extraction process for full set of images.

The final step is the application of this mask to the respective image using AND operation. Figure 3 illustrates the process of foreground extraction, showcasing an individual in the foreground while the background is simultaneously extracted.

3) Face alignment: The third step involves normalizing the facial orientation across all four perspectives by identifying 68 facial landmarks through Dlib's predictor. By calculating the centers of the eyes and utilizing geometric transformations, the images are aligned to a standardized target position at the center of the image to correspond with the frontal view.

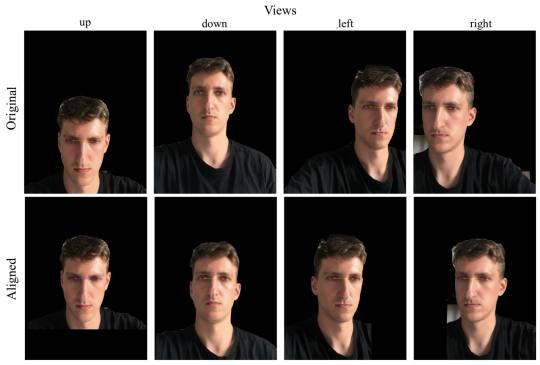
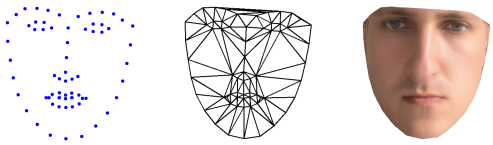


Figure 4. Example of face alignment process for full set of images.

The standardization illustrated in Figure 4 guarantees uniform feature placement from various perspectives prior to blending, with each face being adjusted through a methodical application of rotation, scaling, and translation to ensure accurate alignment according to eye coordinates.

4) Average face reconstruction: This phase synthesizes a standardized frontal facial reference by warping aligned faces to a common landmark configuration. A triangular mesh was created using Delaunay triangulation based on average landmarks. Affine transformations were applied to the triangles of each input face, resulting in a consistent facial structure that preserves features from all angles. On figure 5, we displayed the individual stages of how the frontal face was created.



(a) (b) (c)

Figure 5. 5a displays the average landmarks, 5b representing the triangulated average landmarks and 5c shows the synthetized frontal face.

5) Region-based blending: The final stage involves integrating facial features from multiple angles using multi-scale blending techniques to merge aligned faces with the average facial area. This includes creating masks for specific regions of the foreground individual and aligning the features with the frontal view. Frequency-based decomposition is used to preserve details and ensure smooth transitions between regions..

Figure 6. Synthetized frontal view using proposed method.

Figure 6 displays the final synthetized frontal view created from four input images.

B. 3D Gaussian Splatting

This method represents a novel approach to volumetric rendering in Jawset PostShot's computational framework. Scenes are parametrized via optimized 3D Gaussian primitives, enabling high-fidelity representation. It demonstrate significant performance advantages over traditional mesh-based and voxel methodologies. The implementation exhibits real-time rendering capabilities [5].

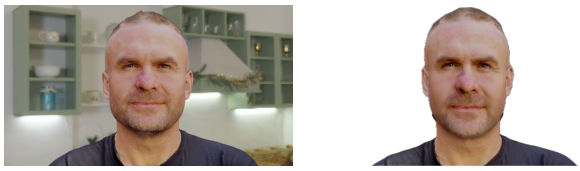
Figure 7. Synthetized frontal view using 3DGS.

V. Experiment Results

Two methodologies were assessed using two datasets. The first dataset included Blender-designed scenes from IMICT FEI STU, each showcasing unique scenarios affecting light intensity and foreground/background variations, with a resolution of 1920x1080 pixels. The second dataset aimed to test the methodologies' limitations using real-world photographs taken at 2k resolution with a smartphone. Results were evaluated based on processing time and image similarity techniques, compared to the reference frontal view.

A. First Dataset (Blender Scenes)

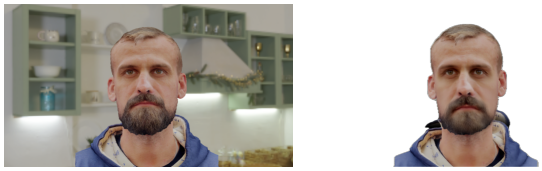
1) Proposed method: In all synthetic scenes, a camera shift was applied alongside x and y axes. The results, as observed, are nearly flawless. The algorithm effectively generated a frontal perspective. In Figure 8, it is evident that employing a highly detailed model in the foreground posed no issues.



(a) Base image (b) Synthetized image

Figure 8. An example of a scene with high detailed model and result of synthetization.

The second illustration in Figure 9 depicts a scenario in which the clothing interacts with the neck region, resulting in a ghosting effect due to imperfect treatment of the neck area in the presence of the clothing.



(a) Base image (b) Synthetized image

Figure 9. An example of a scene with clothing colliding with a neck area and a result of synthetization.

2) Method using 3D Gaussian Splatting: The application of Gaussian Splatting is anticipated to yield a high degree of accuracy across various adverse conditions, regardless of the simplicity or complexity of the scenes involved. Nevertheless, it is projected that several hundred images will be necessary to adequately depict a three-dimensional scene. Given our constraints of limited camera availability and a low-resolution background, this has led to the emergence of numerous popping artifacts.

(a) With background (b) Without background

Figure 10. Results of synthetization of the first scene using 3DGS

Figure 10 displays synthesized frontal views of the scene. We tested input images with (Figure 10a) and without (Figure

10b) background. The average similarity for the image with the background was 0.5394. Removing the background decreased popping artifacts and improved similarity by 0.24, resulting in a new value of 0.7819.

The same phenomenon was observed in other scenes, with highest average similarity reaching a value of 0.9513, with an average training time of 15.6135 minutes. Additionally, Figure 11 demonstrates a scenario where the clothing collided with the neck region was also showing problems with popping artefacts for the scene with background.

(a) With background (b) Without background

Figure 11. Results of synthetization of the second scene using 3DGS.

Given that multiple training settings were tested, we ended up with similar results for each set of images.

B. Second Dataset (Real Life Scenes)

1) Proposed method: In the course of evaluating this method, we integrated an enhanced blending technique specifically in the neck region. The reason for their inclusion was that these images had some rotation to it, and it was necessary to minimize its impact on the final result.

As we displayed on Figure 6, the final results were not completely accurate when compared to the initial dataset. The average processing duration recorded was 11.2566 seconds across sixteen distinct scenes.

2) Method using 3D Gaussian Splatting: Applying 3DGS on a second dataset, the situation was somewhat different from the first. In these scenes, we were not able to accurately represent a 3d model using only four images.

As illustrated in Figure 7, the outcomes derived from the real-life dataset exhibited numerous artifacts. Despite our attempts to adjust various training parameters, there was no noticeable enhancement in the results. Even after nearly doubling the number of Splats, the results remained largely unchanged.

VI. Comparison of Results

The proposed method, enhanced through various modifications for improved outcomes, demonstrates an average execution duration of 11.2566 seconds for the initial dataset. The overall average similarity score of 0.8442 is notably high, especially given the time required to obtain this result. The 3DGS had an average execution time of 952.2843 seconds, far surpassing the proposed method. Figure efgraph1 compares the similarity scores from each method across 16 scenes with the background removed.

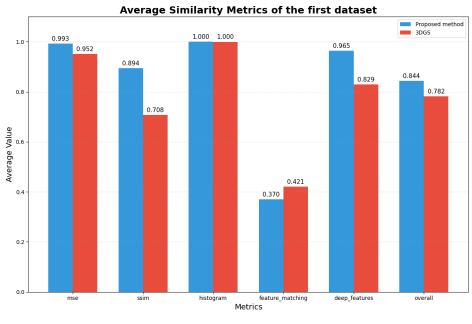


Figure 12. Graph of average similarity scores for the first dataset

By evaluating the results, we can conclude that our proposed method performed better than the 3DGS in handling of these scenarios. Our proposed method was approximately 12.82% more successful. The average values are presented in Table I.

Method	Average runtime (s)	Average similarity
Disparity	11.2566	0.8442
3DGS	952.2843	0.7819

TABLE I. Results of similarity scores for proposed method and 3DGS.

Applying the methods to real scenes, our evaluation was limited to comparing runtime and conducting a visual assessment of the outcomes. In these cases, we were unable to evaluate similarity due to the absence of a reference frontal view.. Our findings indicate that the proposed method effectively utilized data from four cameras, outperforming the 3DGS, which necessitates significantly more information to achieve high-quality results. Given the inability to utilize similarity as a metric, we can regard the runtime of each method as the primary parameter. In this context, the proposed method demonstrates a significant increase in speed and requires less computational resources.

VII. Conclusion and Future Work

The analysis of the graphs and figures indicates that the proposed method outperformed on the initial dataset and demonstrated significantly faster processing times across both datasets. Furthermore, it yielded more precise outcomes despite the limited number of input images, although it is restricted to synthesizing only the frontal view. While PostShot offers a more straightforward setup, it necessitates a broader scene coverage. As mentioned earlier, the result's quantity is more important in this case which our method sucessfully demonstrated.

For future work, we intend to investigate the integration of synthesizing various views of the scene while still relying on just four images. Our proposed strategy includes the introduction of input parameters to select dynamic viewpoint angles. This opens up the possibility for possible use in real-time video conferencing. By simulating physical co-presence through synthesized arbitrary angles relative to participants, our method aims to transform remote communication into natural interactions that closely resemble in-person meetings.

ACKNOWLEDGMENT

This work has been supported by the projects DISIC 09I05-03- V2, VEGA 1/0605/23 InteRViR, and Erasmus+ NEXT.

REFERENCES

[1] S. T. Barnard and W. B. Thompson, "Disparity analysis of images," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. PAMI-2, no. 4, pp. 333–340, 1980.

[2] R. Ranftl, K. Lasinger, and D. Hafner, "Towards robust monocular depth estimation: Mixing datasets for zero-shot cross-dataset transfer," 2020.

[3] J. Covas, *Photogrammetry as a Surveying Technique Applied to Heritage Constructions Recording – Advantages and Limitations – Volume I*. PhD thesis, University of Lisbon, 2018.

[4] B. Kerbl, G. Kopanas, T. Leimkühler, and G. Drettakis, "3d gaussian splatting for real-time radiance field rendering," 2023.

[5] Jawset Visual Computing, "Postshot: End-to-End Radiance Field Software," 2024.

This page is intentionally left blank.