Head-Worn Camera Image Stabilization using Neural Radiance Field

Boxiang Rong
ETH Zürich
borong@ethz.ch

Zilong Deng
Universität Zürich
dengzi@ethz.ch

Ziyao Shang
ETH Zürich
zshang@ethz.ch

Minjing Shi
ETH Zürich
shimin@ethz.ch

Abstract

Motion blur happens with fast head-camera movement and long exposure times. In this project, we proposed to reconstruct the indoor environment in advance and render clear images to replace blurry camera views to achieve the goal of image stabilization. We used four pipelines, including traditional reconstruction and NeRF-based ones (Depth-Supervised NeRF, Deblur NeRF, Instant NGP). Experiments were done on three scenes to test the stabilizing performance and analyze the pros and cons of each pipeline. We also proposed adding camera motion information to Deblur NeRF for better deblurring and verified the performance of our modification.

1. Introduction

With its huge application potential in various fields, Augmented Reality (AR) has been gaining increasingly greater public attention. However, there are currently various issues regarding this technology. One of which is motion blur: During fast head movements, the scenes captured by the camera on AR devices tend to be blurry[17]. In addition, the current processing capacity of AR devices cannot support producing high-quality 3D reconstructions on a real-time basis. Therefore, a pre-built model would be desirable to compensate for these blurry images during runtime.

For this project, we aim to create a pipeline for generating offline scene representations for indoor scenery. Using the volumetric rendering technique, the pre-created scene representations could be used as a supplement when an AR camera device experiences fast movement. In this case, the representation provides an online-render result of the current scene given the camera pose, so the otherwise blurry images could be replaced by clear scene captures.

Given its ability to represent complicated environments, Neural Radiance Field (NeRF)[12] has been widely used in reconstructing 3D objects. We attempted to reconstruct the scene in both traditional ways and NeRF-based methods. For the traditional explicit method, a point cloud of the scene is created using RGB-D images. The cloud is then smoothed into a mesh representation. For the implicit methods, we created implicit scene reconstructions by adapting three proposed NeRF methods and then analyzing their accuracy, efficacy, and usability.

2. Related work

2.1. Non-NeRF-Based Methods

Point cloud reconstruction has many applications, such as 3D architectural modeling[9], terrestrial surveying[3], simultaneous Localization and Mapping (SLAM) for autonomous vehicles, etc. One approach to point cloud reconstruction is Structure from Motion (SFM)[5] algorithms, which looks for corresponding points among overlapping images using SIFT descriptor[10] and recovers both 3d coordinates and camera poses. COLMAP[15, 16] is a reconstruction pipeline utilizing Structure-from-Motion and Multi-View Stereo that takes in images and outputs scene reconstructions. Moreover, if depth maps are available, one can directly project points into 3d space and generates a dense point cloud using OPEN3D[22]. In our project, based on dense point-cloud, we further use Poisson reconstruction[6] to recover mesh and then use color map optimization[21] to improve the texture mapping.

2.2. NeRF Based Methods

In recent years, NeRF has emerged as a prominent method for encoding 3D scene representations using neural networks. While the original NeRF framework achieved remarkable results in novel-view synthesis, several recent developments have aimed to address its limitations and enhance its capabilities. PixelNeRF[20] introduces a learning framework that incorporates spatial image features aligned to each pixel. MetaNeRF[18] focuses on meta-learning, aiming to reconstruct novel scenes with only a few input images. MVS-NeRF[2] leverages multi-view stereo (MVS) techniques to improve the reconstruction quality of NeRF. NeRF with depth prior[4, 14] addresses the challenge of insufficient views by leveraging depth information from images and camera poses, significantly reducing the required number of viewing angles for reliable reconstruction. Deblur-NeRF[11] tackles the issues of motion blur and image noise by training an additional deformable sparse kernel on top of the NeRF framework, resulting in convincing reconstructions of scenes with blurry images. Instant NGP[13] efficiently encodes the 3d feature vectors of scenes using a multi-resolution hash table and greatly accelerates the training speed of NeRF. DP-NeRF[8] focuses on mitigating view-dependent effects in NeRF reconstructions, leveraging a differentiable projection module to achieve superior results. These recent advancements collectively contribute to expanding the capabilities and improving the performance of NeRF-based methods, enabling more accurate and detailed 3D scene reconstructions.

3. Methodology

To generate offline scene representations of the indoor environment, we use both explicit and implicit modeling methods. The explicit method stands for traditional point cloud and mesh reconstruction, while the implicit ones are NeRF-based methods, which encode information inside the MLP networks. Once the scene is reconstructed, we input the camera pose and render clear views during run-time.

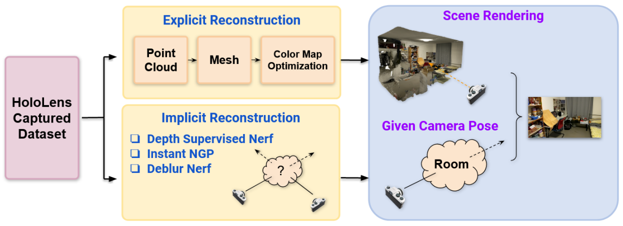


Figure 1: Overview of pipelines. The HoloLens captured data is fed into two streams of methods, explicit and implicit reconstruction, to build environment models. The pre-build models will render clear images to replace blurry camera images.

3.1. Color Map Optimization

Compared with learning-based methods, traditional reconstruction algorithms require much less computing resources and is faster to build simple scenes with few-shot images. The pipeline is shown in Fig 2.

Our Room dataset is captured by HoloLens, which uses two devices to capture RGB and depth images with different resolutions asynchronously. Therefore, we first prepossess depth information by cropping and interpolating to align each clear RGB with a corresponding depth map. After that, we use the paired RGB-D to build point clouds for each image. Then, stitching all point clouds together and applying Poisson surface[6] reconstruction, we get the room model represented by a mesh. Finally, clear images are selected as

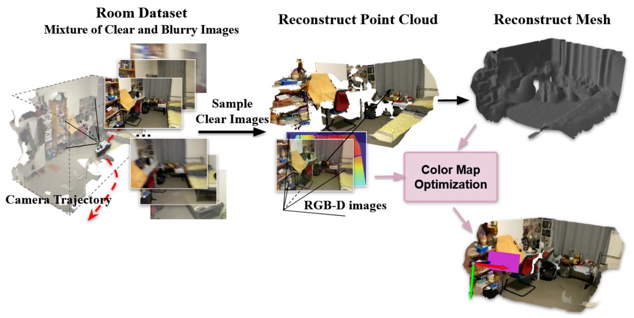


Figure 2: Traditional reconstruction pipeline. Upon stitching multiple point clouds, Poisson surface reconstruction is used to recover mesh for the whole room given clear RGB-D images. Finally, use color map optimization to refine the color on the mesh.

input to color map optimization[21], which will generate a room model with fine-painted color.

To add more technical details, each image in the dataset has a timestamp. When aligning depth to RGB, images with the nearest timestamps will be paired to generate an RGB-D file. Since depth map resolution is way lower than RGB's, we use nearest-neighbor interpolation to get dense and smooth depth distribution. Meanwhile, we select only a small portion of images to reconstruct, and for stitched point clouds, we need to filter noisy points. However, the filtering process requires heavy parameter engineering, and surfaces with complex structures are always badly recovered.

3.2. Depth-Supervised NeRF

The main advantage of Depth-supervised NeRF (DS-NeRF) is that it leverages both the training images and the depth prior of the rendered points, resulting in a NeRF model that learns the underlying scene geometry with much larger accuracy. We use two losses defined in the original DS NeRF paper[4]. These two losses are combined through a pre-defined weight λ_D: $l = l_{color} + \lambda_D l_{Depth}$. Where the depth loss:

$$l_{\text{depth}} = \mathbb{E}_{x_i \in X_j} \sum_k \log h_k \exp\left(-\frac{(t_k - D_{ij})^2}{2\sigma_i^2}\right) \Delta t_k \tag{1}$$

Here, for a ray $r(t) = o + t * d$, The ray distribution, which denotes the likelihood the ray ends at time t, is characterized by $h(t) = \sigma(t) * \exp(-\int_0^t \sigma(s)ds)$ with σ as the density function. X_j is the set of all points in the point cloud that is visible in view j. For each point i and associated camera view j, the depth distribution is estimated by a normal distribution: $\mathbb{D}_{ij} \sim \mathbb{N}(D_{ij}, \sigma_i)$.

The main pipeline of DS NeRF is shown in Fig 3. All training images are fed into the Structure-from-Motion

(SFM) application called COLMAP, which generates a point cloud of the scene and the estimated camera poses of each image view.

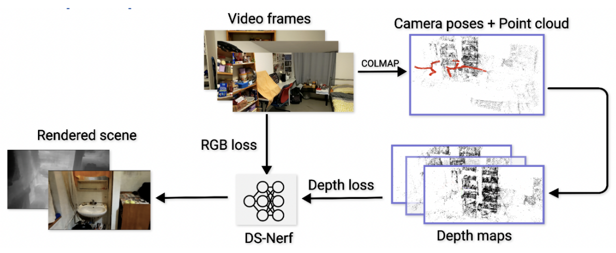


Figure 3: Pipeline of Depth-Supervised NeRF. SFM pipeline is applied to the RGB images to create a point cloud, which will be projected onto each image afterward to generate its sparse depth map. Both the depth map and the RGB image are fed for training

Then, the depth map of each scene is mapped onto its RGB image to form its depth map and error. This information is fed together into DS-NeRF, resulting in an implicit scene representation that is also able to generate reliable depth information.

Due to the complicated nature of the test scenes, we also fine-tuned the parameters of the feature extraction process in SFM, so we could capture more features by making the feature detection more sensitive to edges and color variations, thus adapting the feature matching (otherwise suitable for a small number of contiguous training images) to the large and complicated data sets.

We also attempted to convert the monocular depth map generated by HoloLens into our training depth. However, due to the incompatibility of the coordination systems of the provided pose, the resulting dataset could not train DS NeRF in good quality.

3.3. Deblur NeRF

NeRF-based pipelines can be easily affected by blurry images, which requires manually filtering blurry data. However, Deblur NeRF can recover a clear scene from only blurry images by adding a 'deformable' kernel[11]. The pipeline of Deblur-NeRF is shown in Fig 4.

During training, each ray will be optimized into several rays through a deformable kernel. The optimization process is described by Eq. 2.

$$(\Delta q, \omega_q) = G_\phi(p, q', l), \text{ where } q' \in \mathcal{N}'(p) \tag{2}$$

$$q = q' + \Delta q \tag{3}$$

$$b_p = \sum_{q \in \mathcal{N}(p)} \omega_q c_q, \text{ w.r.t } \sum_{q \in \mathcal{N}(p)} \omega_q = 1 \tag{4}$$

Here, $\mathcal{N}'(p)$ is the predefined kernel, with fixed position shift; l is the view embedding for distinguishing different views, and G_ϕ is the MLP block which outputs position shift Δq and weights ω_q for each optimized rays.

Then, each optimized ray q, going through the NeRF block, gets a color value c_q, which will then be mixed to produce a blurry image. In other words, the forward process simulates how blur is produced, with ground truth being the actual blurry images to supervise training. After that, the deformable kernel will be discarded during testing, and only basic NeRF will produce a clear scene. Meanwhile, an align loss is designed to ensure the optimized rays are sampled around the input ray. More details can be referred to in the original Deblur-NeRF paper[11].

Motivated by the fact that motion blur usually follows a specific blurry pattern, which is aligned with camera movement[1], we proposed to add trajectory information to supervise the training of the deformable kernel based on the aforementioned Deblur NeRF. We can easily acquire trajectory information from the dataset by computing the translation and rotation between continuous frames. In our case, we use the velocity vector (3 values) and quaternions (4 values) to represent the camera's motion and encode them into high-dimensional vectors. Finally, initialize the view embedding with encoded velocity and quaternions. Experimental results are shown in 4.4.1.

3.4. Instant NGP

For the Instant Neural Graphics Primitives (Instant NGP)[13], we used the standard framework to train our dataset on the model. We applied the recommended scale and lighting settings of the dataset, and we pre-processed our dataset prior to training to fit the prerequisites of the model. We modified several hyperparameters of the NeRF model within the framework, including color activation units (from Sigmoid to ReLU), total loss calculation (from Huber to L2), and density activation units (from exponential to ReLU).

4. Experiments

4.1. Datasets

Our dataset is gathered from a previous Mix Reality course project[7]. It consists of two video recordings of two different room scenes captured using Hololens2. Each capture contains thousands of RGB video frames in 1280×720, monocular depth frames in a lower capturing frequency, the intrinsic parameters of the camera, and the corresponding camera poses and timestamp for each RGB frame. For the first capture, the HoloLens has a relatively slow movement, which results in a dataset containing less motion blur. This dataset is used for the main scene experiments as well as Instant-NGP parameterization testing. The second capture contains more motion blur and is used for experiments for Deblur-NeRF modifications and ablation studies.

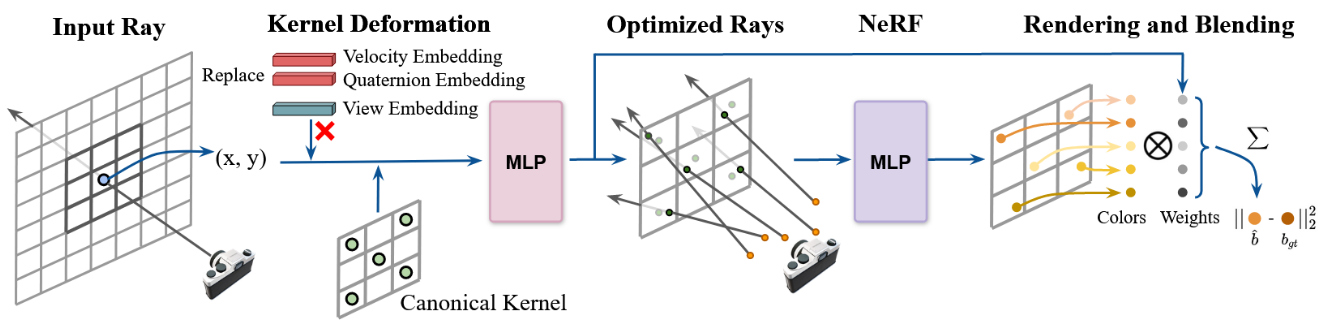


Figure 4: Deblur NeRF pipeline. When training, for input ray, the position, view embedding, and predefined kernel offset are fed in MLP to generate multiple optimized rays. Colors of all optimized rays will be mixed to recover ground truth(blurry image). Our modification replaced view embedding with trajectory information. Only NeRF will be used when testing.

4.2. Main testing scenes

We conducted three experiments on subsets of the more detailed room dataset. First, the **poster** scene contains diverse and intricate variations in its colors but not many variations in its depth. Then, the **bookshelf** contains a complex depth distribution, as well as a somewhat diverse RGB variation. Last, the whole **room**[1], with a much larger spatial extent than the other two scenes, is rendered [2].

4.3. Results

The results can be found in table 1.Fig 5 shows a side-to-side comparison of each method and the ground truth on scenes **poster** and **bookshelf**. Videos of the whole room can be found here [3].

In the poster scene, color details are generally well preserved for Color Map Optimization, Depth-Supervised NeRF, and Deblur NeRF. We can still well observe the ”TROPHY” characters on the poster. However, explicit mesh reconstruction is found to be badly influenced by the noisy point clouds, especially in the bookshelf scene, due to its large depth variance. Considering the time and computing resources, the traditional reconstruction is still more convenient for simple and plain scenes.

DS NeRF generally gets better performance in the Bookshelf and Poster scene than others. Both DS NeRF and Deblur-NeRF have a greater capability in reconstructing complicated scenes, despite the slow inference speed compared to Instant-NGP and Color map optimization. For all scenes, Deblur NeRF actually generates the clearest image among all pipelines but doesn't get high scores when evaluated by metrics. We will discuss the analysis of that in the next section.

[1]For the whole room, each method is not trained/tested on the exact same images, but they all come from the same scene.

[2]To train DS NeRF on the room scene, we removed images with little texture to guarantee a more stable feature matching.

[3]YouTube Video of Room Scene https://youtube.com/playlist?list=PLUffCQyBEYtbOQg4-66ZrcuNmsX0OXVKv

Poster	**MSE**	**PSNR**	**SSIM**	**LPIPS**
ColorMap Opt.	0.099	16.195	0.546	0.383
Instant NGP	0.048	19.532	0.688	0.526
DS NeRF	0.009	26.950	0.813	0.247
Deblur-NeRF	0.034	23.172	0.730	0.316

Bookshelf	**MSE**	**PSNR**	**SSIM**	**LPIPS**
ColorMap Opt.	0.159	14.119	0.422	0.480
Instant NGP	0.033	20.402	0.661	0.423
DS NeRF	0.021	23.106	0.709	0.344
Deblur-NeRF	0.052	19.322	0.630	0.335

Room	**MSE**	**PSNR**	**SSIM**	**LPIPS**
ColorMap Opt.	0.110	16.426	0.484	0.393
Instant NGP	0.028	21.585	0.690	0.405
DS NeRF	0.028	21.879	0.637	0.456
Deblur-NeRF	0.057	19.614	0.621	0.361

Table 1: **View generation:** We applied all four methods to each of the three scenes.

4.4. Extended Experiments

4.4.1 Trajectory + Deblur NeRF

In Fig.6, we compared the rendering quality of Deblur-NeRF before and after adding embedded trajectory information. The modified Deblur-NeRF seems to overfit on the training set and performs badly on the test set. We also tried to normalize trajectory embedding, but it's not working. One possible direction is that instead of directly using trajectory as an embedding, we can use it to design a deformable kernel for each view, following the simulation

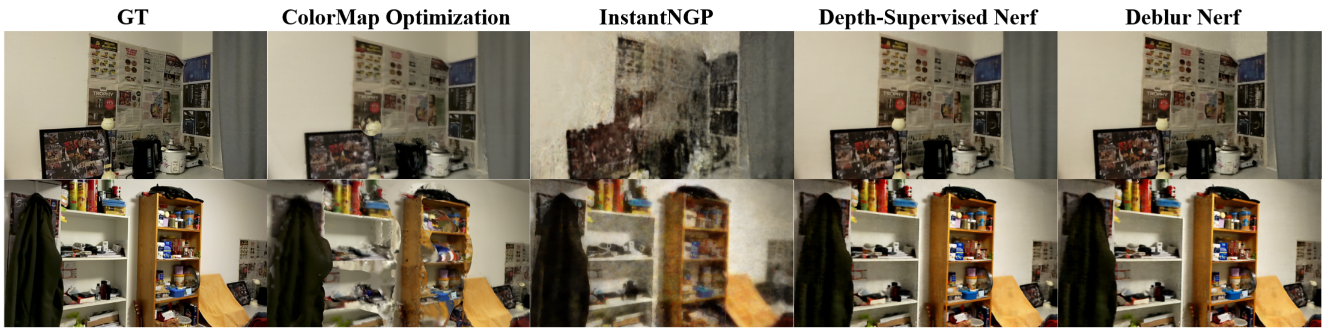


Figure 5: Comparing the rendering quality of each method on the scenes **Poster** and **bookshelf**. The ground truth images are shown on the top left.

process in computer graphics[1].

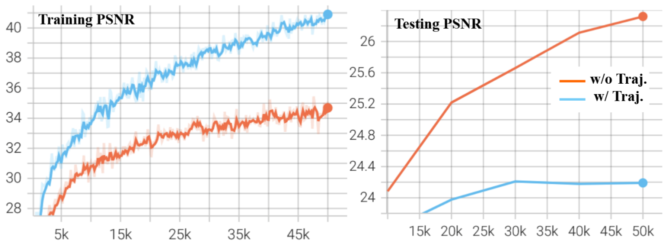


Figure 6: PSNR of training and testing. The model overfit the training set after using the trajectory embeddings.

4.4.2 Few-shot Scene Reconstruction

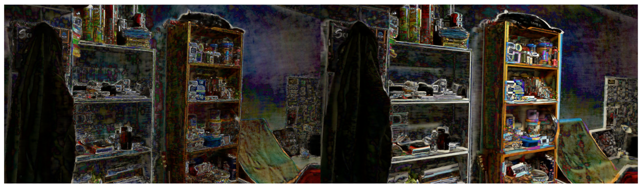
Depth information is known to be able to reduce the training burden on the number of images. We investigated its validity in the context of our project. We reduced the number of training images for the poster scene from 33 to 7. As shown in table 2, the reconstruction quality of DS NeRF is much better than Deblur-NeRF in all metrics. In a word, DS NeRF only needs a few pictures for training and can still clearly reconstruct the scene when data is limited.

Poster	MSE	PSNR	SSIM	LPIPS
DS NeRF	0.207	13.224	0.382	0.599
Deblur-NeRF	0.018	24.324	0.782	0.248

Table 2: Few-shot training: a comparison between DS-NeRF and Deblur-NeRF

4.4.3 Shifted outputs of Deblur-NeRF

To find out why clear outputs of Deblur NeRF get a lower metrics value, we calculated the channel difference between ground truth and output images. It is found that Deblur NeRF generates a slightly shifted image, shown by the bold edges in Fig. 7. However, when only blurry images are available, Deblur NeRF is still the best choice to recover clear ones. We show its ability to correct blurry images in Fig. 8.



(a) Depth-Supervised NeRF (b) Deblur NeRF

Figure 7: Channel difference between the ground truth and the output of DS NeRF and Deblur NeRF. The darker the image of channel difference, the closer the original image to the g.t. is.

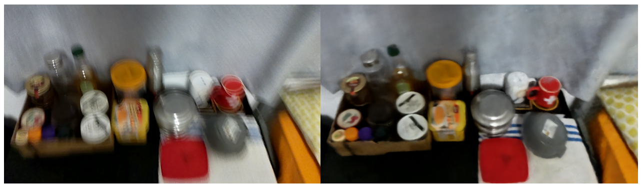


Figure 8: Comparison between training set image(left), and output image from trained Deblur NeRF(right)

4.4.4 Different Instant NGP Training Setting

We tried different training settings of Instant NGP to analyze how different Instant-NGP parameterizations (ReLU RGB Activation, L2 Total Loss, and ReLU Density Activation) could affect the results. The results, shown in table 3, evidence a preference for L2 total loss.

Poster	PSNR	SSIM	LPIPS
ReLU RGB Activation	15.647	0.643	0.484
L2 Total Loss	21.258	0.679	0.425
ReLU Density Activation	16.183	0.628	0.482

Table 3: Comparison of different Instant-NGP parametrizations

5. Discussion

NeRF-based methods have demonstrated high-quality reconstruction capabilities in indoor environments. Unlike traditional explicit methods such as point clouds and mesh, NeRF's continuous implicit representation model enables clear and smooth view synthesis with adaptable accuracy and the ability to learn novel views. Explicit methods remain more cost-effective and faster, although they may struggle with complex surface conditions.

The Depth-Supervised NeRF leverages the depth maps to provide an assumed depth prior for each 3D point during training, so it needs fewer images than other methods to train an average representation and has a more precise depth map in rendering views. Since the features inside the room have many occlusions and depth cliffs, the DS NeRF has an advantage in learning these kinds of scenes.

The Deblur-NeRF, which aims to train a representation given blurry images, has shown its ability to reduce the blur and obtain a good, blurry-free representation. Combined with the pre-deformable kernel and the motion embedding of the camera, Deblur-NeRF can learn the movement direction that causes the blur and reproduce a clear scene.

The Instant NGP pipeline is performed as a baseline for NeRF-based methods. It adds a HASH table to the original NeRF to speed up the rendering. As the experiment result shows, the original NeRF has a limited ability to reconstruct room scenes and will produce a lot of artifacts.

5.1. Future Scope

For future works, one direction is to combine DS-NeRF and Deblur NeRF, which show great capabilities in reconstructing complicated scenes given limited and blurry training images. We can also add the HASH table designed in the Instant NGP for faster rendering, which may make it possible to integrate the run-time rendering of the implicit model into the AR device itself.

Another direction is to predict blur patterns with trajectory information in a computer graphics manner. We can directly use the poses to calculate the exact motion of the camera and use that to design the deformable kernels.

We can also try to add a Fourier embedding[19] to the NeRF to let the network learn more high-frequency features of the scene.

5.2. Limitation

Our proposed pipelines also have several limitations. The Instant NGP pipeline, compared with other pipelines, is featured in fast training speed but can not recover clear scenes after the same iterations of training. For the Deblur-NeRF, although it can deblur by using a deformable kernel, this kernel also produces slight shifting and blending of the view synthesis, which causes the decrease of its testing PSNR. But anyway, the deblur NeRF did render the clearest images to the human eye. In DS NeRF, we tried to use the monocular depth map to train the model, but it failed due to the mismatch of the provided poses' coordinate system and the camera's coordinate system.

6. Conclusion

In this project, we built several pipelines to reconstruct the indoor environment and render clear images. Through the NeRF-based pipeline, we can collect data and train the representation offline. During high-speed movements in run-time, we can replace the blurry image with the rendering image from the representation, thus achieving the stability of the head-on camera.

6.1. Work split and Modifications

Work split is shown in table 4. We carry out experiments with different methods. We built a traditional pipeline from scratch while inheriting the existing datasets from the previous project group. Based on the original DS NeRF, we modified the data loader and tried using both COLMAP and monocular depth maps as the depth supervision. We add trajectory information to the Deblur NeRF for training. For the Instant NGP, we run it using the whole dataset and use it as the baseline of the implicit method set.

TASKS	Members
Literature study, Results analysis	All Members
Explicit Reconstruction and Color Map Optimization	Boxiang Rong
SFM and DS NeRF training with sparse depth prior	Zilong Deng, Ziyao Shang
DS NeRF training with monocular depth map	Zilong Deng, Ziyao Shang
Deblur NeRF dataloader and experiments	Boxiang Rong
Trajectory Deblur implementation and experiments	Boxiang Rong
Instant NGP experiments	Minjing Shi

Table 4: Work split of team members.

References

[1] James F Blinn. Modeling motion blur in computer-generated images. *ACM SIGGRAPH Computer Graphics*, 16(3):137–141, 1982.

[2] Anpei Chen, Zexiang Xu, Fuqiang Zhao, Xiaoshuai Zhang, Fanbo Xiang, Jingyi Yu, and Hao Su. Mvsnerf: Fast generalizable radiance field reconstruction from multi-view stereo. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 14104–14113, 2021.

[3] S.I. Deliry and U. Avdan. Accuracy of unmanned aerial systems photogrammetry and structure from motion in surveying and mapping: A review. *J Indian Soc Remote Sens*, 49:1997–2017, 2021.

[4] Kangle Deng, Andrew Liu, Jun-Yan Zhu, and Deva Ramanan. Depth-supervised NeRF: Fewer views and faster training for free. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2022.

[5] Richard Hartley and Andrew Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, 2003.

[6] Michael Kazhdan, Matthew Bolitho, and Hugues Hoppe. Poisson surface reconstruction. In *Proceedings of the fourth Eurographics symposium on Geometry processing*, pages 61–70, 2006.

[7] Annamalai Lakshmanan, Florence Kissling, Gowtham Senthil, and Siva Vignesh Krishnan Chidambaram. Image stabilization for hololens camera. Virtual Reality course project, ETH Zürich, 2022.

[8] Dogyoon Lee, Minhyeok Lee, Chajin Shin, and Sangyoun Lee. Dp-nerf: Deblurred neural radiance field with physical scene priors. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 12386–12396, June 2023.

[9] Massimiliano Lo Turco and Cettina Santagati. From sfm to semantic-aware bim objects of architectural elements. In *Digital Heritage. Progress in Cultural Heritage: Documentation, Preservation, and Protection*, 2016.

[10] David G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 2004.

[11] Li Ma, Xiaoyu Li, Jing Liao, Qi Zhang, Xuan Wang, Jue Wang, and Pedro V. Sander. Deblur-nerf: Neural radiance fields from blurry images. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 12851–12860, 2022.

[12] 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.

[13] 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.

[14] Barbara Roessle, Jonathan T. Barron, Ben Mildenhall, Pratul P. Srinivasan, and Matthias Nießner. Dense depth priors for neural radiance fields from sparse input views. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2022.

[15] Johannes Lutz Schönberger and Jan-Michael Frahm. Structure-from-motion revisited. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[16] Johannes Lutz Schönberger, Enliang Zheng, Marc Pollefeys, and Jan-Michael Frahm. Pixelwise view selection for unstructured multi-view stereo. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2016.

[17] Haruo Takemura and Hirokazu Kato. Feature extraction in augmented reality. *arXiv preprint arXiv:1911.09177*, 2019.

[18] Matthew Tancik, Ben Mildenhall, Terrance DeVries, Sara Fridovich-Keil, Nithin Raghavan, Utkarsh Sinha, Jonathan T. Barron, and Gordon Wetzstein. Learned initializations for optimizing coordinate-based neural representations. In *International Conference on Learning Representations (ICLR)*, 2021.

[19] Matthew Tancik, Pratul P. Srinivasan, Ben Mildenhall, Sara Fridovich-Keil, Nithin Raghavan, Utkarsh Singhal, Ravi Ramamoorthi, Jonathan T. Barron, and Ren Ng. Fourier features let networks learn high frequency functions in low dimensional domains. *NeurIPS*, 2020.

[20] Alex Yu, Vickie Ye, Matthew Tancik, and Angjoo Kanazawa. pixelNeRF: Neural radiance fields from one or few images. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021.

[21] Qian-Yi Zhou and Vladlen Koltun. Color map optimization for 3d reconstruction with consumer depth cameras. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 638–645, 2014.

[22] Qian-Yi Zhou, Jaesik Park, and Vladlen Koltun. Open3D: A modern library for 3D data processing. *arXiv:1801.09847*, 2018.