Investigation on Relative Pose Regression for Multi-Camera Setups with Neural Radiance Fields

Timo Kullmann[1], Patrick Hübner[1], Tristan Wirth[4], Arjan Kuijper[2,3], Dorota Iwaszczuk[1]

[1] Technical University of Darmstadt, Dept. of Civil and Environmental Engineering Sciences,
Remote Sensing and Image Analysis, Darmstadt, Germany
(timo.kullmann, patrick.huebner, dorota.iwaszczuk)@tu-darmstadt.de
[2] Technical University of Darmstadt, Dept. of Computer Science, Mathematical and Applied Visual Computing,
Darmstadt, Germany
[3] Fraunhofer IGD, Darmstadt, Germany, arjan.kuijper@igd.fraunhofer.de
[4] Technical University of Darmstadt, Dept. of Computer Science, Interactive Graphics Systems,
Darmstadt, Germany, tristan.wirth@gris.tu-darmstadt.de

Keywords: 3D Scene Reconstruction, Pose Estimation, Multi-Sensor Platform, Dual-Camera, Indoor

Abstract

Neural Radiance Fields (NeRFs) are a novel approach that is being intensively investigated in 3D scene reconstruction and similar fields to overcome challenges of conventional methods. In this paper, we address the problem of estimating missing camera poses in a six degrees of freedom setting, pushing the capabilities of NeRFs to address scenarios where only the primary camera's pose is known. Specifically, we focus on dual-camera setups with this constraint. Our core contribution is a novel pose correction model that operates alongside an unmodified NeRF model, for which we have chosen Nerfacto in the Nerfstudio framework. The pose correction model learns the necessary relative translation and rotation adjustments for the secondary camera solely through the NeRF loss. This allows us to integrate our correction model directly into the Nerfacto training pipeline without altering the core functionality. Through extensive experiments on different camera configurations in a synthetic scene, we rigorously evaluate our model's performance across diverse scenarios, pushing it to its limits. Our findings reveal that our model can effectively learn pose correction parameters within a constrained range, with increased sensitivity to larger translations and particular challenges in rotation corrections. This research highlights the potential of NeRFs for machine learning-driven 3D reconstruction on dual- and multi-camera platforms, expanding the applicability of NeRFs to more complex, real-world setups despite the inherent challenges.

1. INTRODUCTION

Low-cost multi-camera systems are widely utilized for 3D scene reconstruction across diverse applications, including mobile mapping systems (MMS), which support tasks like creation and maintenance of Building Information Modeling (BIM) (Lehtola et al., 2021; Hou et al., 2024) or vegetation mapping (Lechner et al., 2020). Cameras in MMS are affordable and capable of delivering high-resolution data, with flexible setups and custom configurations (Elhashash et al., 2022; Goebel and Iwaszczuk, 2023). Also vehicles nowadays function as multi-sensor platforms, increasingly equipped with cameras for a range of purposes, from parking assistance to the complex demands of autonomous driving systems (Häne et al., 2017; Indu et al., 2021; Hee Lee et al., 2014). Neural Radiance Fields (NeRFs), introduced by Mildenhall et al. (2020), are an innovative machine learning approach for 3D scene reconstruction using multi-view RGB images or videos from a standard camera. Unlike traditional geometric methods, NeRFs learn a continuous representation, which can generate highly realistic, photo-consistent renderings from any viewpoint under various camera settings.

In multi-camera systems, NeRFs can offer significant benefits by synthesizing realistic views from sparse input images, overcoming the limitations of traditional 3D reconstruction in challenging environments. Applied to systems like MMS or automotive platforms, NeRFs can enhance the quality and density of point clouds, producing precise 3D models for applications such as BIM and vegetation analysis without requiring expensive equipment (Hachisuka et al., 2023). Its usability and ease of setup also make NeRFs a promising tool for broadening access to 3D reconstruction technologies. NeRF models take a five-dimensional input, ray origin and direction, and output the optical density and RGB color at each sampling point along the ray. Optical density reflects the radiance contribution, which is accumulated via ray marching to compute each ray's final color. NeRFs generate these outputs in two stages, creating coarse and fine scene representations. By casting rays for each pixel based on a defined camera setup, NeRFs render accumulated color values to reproduce the scene from any viewpoint, as illustrated in Figure 1.

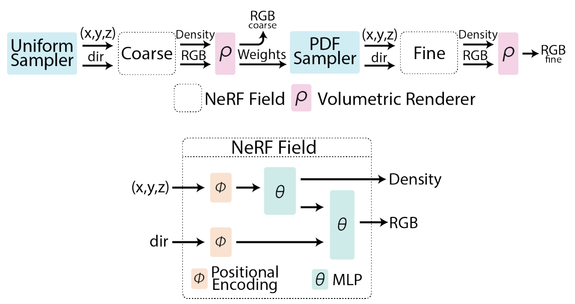

Figure 1. Visualization of the NeRF pipeline and field (nerfstudio Team, 2022).

Despite their potential, NeRFs face practical challenges. Accurate 3D camera poses are critical for learning a scene, but in multi-camera setups, particularly low-cost configurations, pose data may be missing for some cameras. These setups prioritize flexibility, often making precise measurements impractical and leaving some poses unknown. Addressing these challenges is key to expanding the use of NeRFs in accessible, multi-camera systems.

1.1 Related Work

The introduction of X-NeRF represents an advancement in sensor pose estimation. It introduced sensor pose optimization for sensors in parallel image planes and improved rendering across spectral modalities (Poggi et al., 2022). X-NeRF's Normalized Cross-Device Coordinates (NXDC) support multi-sensor setups without cross-spectral calibration, enabling translation-based alignment and novel view synthesis.

Other recent methods, which tackle sensor pose estimation, have explored pose estimation for six degrees of freedom (6DoF), in contrast to the three degrees of freedom in the work by Poggi et al. (2022). iNeRF, for example, focuses on 6DoF pose estimation, aligning input images with pre-trained NeRF models using gradient-based optimization for an efficient, RGB-only approach (Yen-Chen et al., 2021). Lin et al. (2023) further adapt this to Instant-NGP to accomplish faster results. NeRF-Pose also handles 6DoF, training a pose regression model alongside NeRF with PnP+RANSAC for increased accuracy (Li et al., 2023).

Missing or inaccurate pose data is addressed as well in recent publications. Wang et al. (2022) optimize missing intrinsics and 6DoF poses as learnable parameters. BARF enables NeRF training with imperfect poses through coarse-to-fine refinement (Lin et al., 2021), and CBARF cascades these refinements (Fu et al., 2023). For complex trajectories, NoPe-NeRF leverages depth prior estimating relative poses between frames (Bian et al., 2023).

Finally, Omni-NeRF expands NeRF's scope to 360° images with imperfect pose or intrinsics, broadening pose estimation applications (Gu et al., 2022).

1.2 Contribution

In this paper, we build upon the work of Poggi et al. (2022) by introducing a model that learns relative transformation and rotation parameters to reconstruct missing pose data. Our approach specifically addresses scenarios where only the main camera's poses are available, enabling the estimation of missing poses for secondary cameras. Unlike Poggi et al. (2022), our method extends beyond the restriction of parallel image planes to include rotation correction, making it applicable to a broader range of multi-camera configurations.

We investigate the potential of NeRFs as a reconstruction method in such scenarios and assess its performance in dual-camera setups with missing pose data. Our contributions include a detailed evaluation of our pose correction model, highlighting its strengths and limitations, and providing valuable insights into its capability to estimate missing poses.

2. METHODS

In this section, we explain the methods applied in this study in detail, covering data generation, framework selection, model configuration, pose correction approach, and training process, which together form the basis of our experiments and pose estimation pipeline.

2.1 Data

Machine learning models rely heavily on datasets tailored to the specific task. To ensure a consistent and reproducible setting, while also allowing flexibility in modifying scene details, we used a synthetic scene, which is modified for creating datasets with dual-camera configurations. Each dataset includes a main and a secondary camera to simulate scenarios with incomplete pose data.

For the 3D scene creation, we used the open-source software Blender (Blender Foundation, 2024). Using a royalty-free, user-generated scene from the freely available platform BlenderKit, we created an adapted version of this scene to allow for easy modifications and dataset generation (Viriyahirunpaiboon, 2024). In Blender, custom camera placements and paths were created, enabling precise control over camera configurations and rendering at the desired frame rate. The BlenderNeRF plugin by Raafat (2024) allowed us to export the data directly into a NeRF-compatible format, providing a flexible dataset with a customizable number of images, and included pose information and camera intrinsics for each frame. The camera poses are 4x4 matrices that position each camera in 3D space, enabling controlled multi-camera datasets with precise reference data.

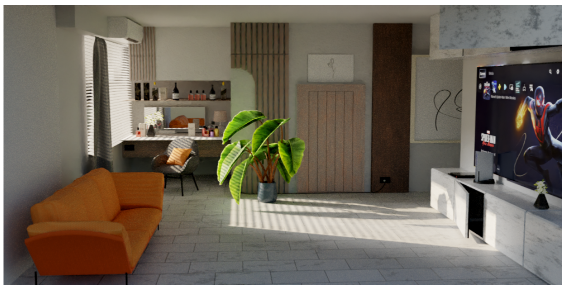
Figure 2. Example rendering of the synthetic Blender scene.

2.2 Framework

For all experiments and implementations presented in this paper, Nerfstudio was chosen as the NeRF-Framework (Tancik et al., 2023). Nerfstudio is an open-source framework implemented in the Python programming language and offers an end-to-end pipeline for NeRF training with a wide variety of NeRF models to chose from. Overall, this framework is well-suited for customization, which enables the implementations of desired adaptions and offers a complete training structure, ensuring an easy-to-use environment.

2.3 Model

The NeRF model used in this paper, Nerfacto, was introduced with the Nerfstudio framework as a comprehensive NeRF implementation. Primarily based on MipNeRF-360 (Barron et al., 2022), it also incorporates features from NeRF-- (Wang et al., 2022), Instant-NGP (Müller et al., 2022), NeRF-W (Martin-Brualla et al., 2021), and Ref-NeRF (Verbin et al., 2021).

One improvement of Nerfacto over MipNeRF-360 was the extension of proposal sampling by introducing a piece-wise sampler. While proposal sampling directs the ray sampling to key areas for reconstruction, the piece-wise sampling process adapts the sampling based on the distance. Near objects are uniformly and densely sampled and for larger distances, the step size is changed, improving the efficiency. A second improvement, hash encoding developed for Instant-NGP by Müller et al. (2022), encodes data in a higher-dimensional space, allowing for a smaller, more efficient MLP.

These enhancements, shown in Figure 3, significantly boost performance while maintaining high-quality results.

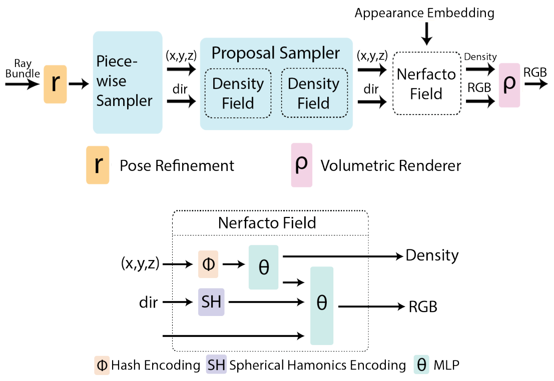

Figure 3. Visualization of the Nerfacto pipeline and field (nerfstudio Team, 2022).

2.4 Relative Pose Correction

The constraint of having no pose information of secondary cameras limits the known poses to those of the main camera. Therefore, the poses of the additional cameras have to be estimated. For this purpose, we introduce a new pose correction model, which is able to learn a relative pose correction. The estimation of a relative pose correction is significantly less complex than an absolute pose estimation. Making use of this is possible through having the main camera data available. The main camera's pose can therefore be used as the initial value for a secondary camera's pose. From that starting point the relative pose correction parameters have to be estimated.
By splitting the pose correction into a separate model the main NeRF model itself can stay mostly unchanged.
The correction is split into translation and rotation correction. Those parameters are optimized separately for each secondary camera in the training process. The corrections are applied to the input data of the NeRF model in each iteration. By this, the correction model does not require a separate loss function and relies solely on the loss calculated by the NeRF model. Through the pre-training introduced in Section 2.6 it is assumed that the quality of the result of the NeRF model is mainly influenced by the quality of the relative pose correction parameters.

2.5 Relative Pose Correction Implementation

The relative pose correction model is implemented as a separate module. As explained in Section 2.4 the pose correction is split into translation and rotation correction. The translation correction parameter is represented as a three dimensional vector and the rotation correction as a quaternion. Both are separately optimized as learnable parameters in the pose correction model. All rays in the ray bundle, which is processed in the NeRF model, consist of an origin and a direction vector. The origin vector is therefore adjusted by the use of the translation correction vector, shifting the ray origin in space with a regular matrix multiplication. The rotation correction is applied to the direction vector through a quaternion rotation. All rays in the input ray bundle for the NeRF model are modified by the relative pose correction model and passed on to the main NeRF model. No changes are made to the processing of the input ray bundle done by the main NeRF model. Figure 4 showcases the adapted Nerfacto pipeline, where the pose refinement step, which is included in Nerfacto, was replaced by our pose correction model.

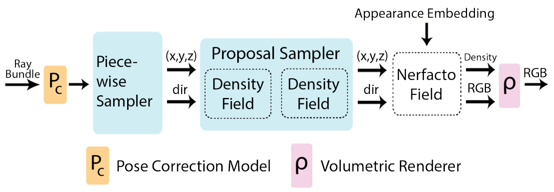

Figure 4. The adapted Nerfacto pipeline with our pose correction model.

2.6 Training Process

The training of the pose correction model and the NeRF model is split and handled in a separate, alternating way. First, the NeRF model is pre-trained using only data from the main camera. This creates the baseline Nerfacto field, and thus the 3D representation, which was trained on correct pose information. Second, this baseline field can then be used to train the pose correction model. In this step the parameters of the NeRF model are frozen and therefore can not be corrupted by a potentially incorrect pose correction. Through this separation the pose correction parameters are optimized depending on the NeRF output without negatively influencing the Nerfacto field. Overall, this training approach allows for both the NeRF model and the pose correction model to be trained efficiently.

3. EXPERIMENTS AND EVALUATION

3.1 Hyperparameters

To achieve the best possible comparability between all evaluation configurations, the hyperparameters were pre-selected and fixed over all experiments. The hyperparameters were selected using OpTuna, a hyperparameter tuning tool (Akiba et al., 2019). Table 1 below lists all used training details and hyperparameters.

Training Setup Overview	
GPU	Nvidia RTX 4090
NeRF Studio Version	0.3.4
Total Number Of Iterations	30000
Pre-training Steps	4000
NeRF Field Initial Learning Rate	0.01
NeRF Field Final Learning Rate	0.0001
Pose Correction Initial Learning Rate	0.005
Pose Correction Final Learning Rate	0.0005
NeRF Field Training Steps	400
Pose Correction Training Steps	1100
Combined Training Percentage	3%
Rays per batch	16384
Camera Resolution	500x500px

Table 1. Overview of all important training details and hyperparameters.

3.2 Experiment Structure

The goal of the evaluation process is to explore the functionality and limits of the proposed pose correction. For this, the datasets were split into five different categories. While all poses from the main camera are known to the model, the secondary camera poses are unknown. Ground truth poses are available for all poses. The created datasets consist of 34 images per camera, with a resolution of 500 $\times$ 500 pixels.

Translation The first category only contains datasets with a translation transformation between the cameras. The extent of the translation is again split into steps, ranging from a smaller to a larger translation. The first part of this category only contains translations along a single axis, again reducing the complexity to a minimum. The second part contains combinations of translations in multiple axes.

Rotation The second category works analogously to the first category, but focuses exclusively on rotation, instead of translation.

Combination The third category combines translation and rotation to increase complexity. Additionally, the goal is to test the limits for the translation and rotation and to determine whether these limits shift when two types of transformation are combined. Again, the extent of the transformations is staggered into multiple datasets.

Multi-Camera This additional category focuses on using three cameras. It combines and repeats tests from previous categories in the same manner. Therefore, this category helps to understand, how multiple optimizations at once affect the quality of the reconstruction. Finally, these experiments further explore the robustness of the model.

Additional Experiments These additional experiments aim to investigate performance improvement possibilities and the robustness of the model. Therefore, three additional experiments were conducted. The first experiment tested the hypothesis that more data leads to better performance, which is a general assumption for machine learning tasks. In the second experiment, we tested whether increasing the number of rays per batch above 16,384, the count used in all our other experiments, would lead to a significant improvement over Nerfacto's default value of 4,096. Lastly the model was also tested with secondary camera data with a lower resolution of 200 $\times$ 200 pixels.

3.3 Evaluation Process

The evaluation process consists of two steps. First, after each experiment, the final pose correction parameters are compared to the ground truth values generated from the Blender scene. The generated 4 $\times$ 4 transformation matrix from the Blender data is decomposed into translation and rotation elements. The translation vectors are compared using Euclidean distance, and the rotation is represented as a quaternion to calculate the angular difference in degrees. Thus, smaller error metric values indicate a more accurate estimation of the relative pose correction. Since the translation values are defined by the scale of the Blender scene, they do not relate to a real-world unit. These error values are used to assess performance. Additionally, selected results are rendered from a baseline NeRF model for visual inspection and interpretation.

4. RESULTS

This section presents the results of the conducted experiments, with abbreviations used to improve readability due to space constraints. Within each category, the applied transformation is classified into relative sizes: S (small), M (medium), L (large), and XL (extra-large). Note that these categorizations are relative within each category and are not comparable across categories. All calculations were performed using all decimal places, with rounding only applied in the tables for readability. Labels x, y, and z denote the relevant axes for each transformation, indicating either a translation along or a rotation around the specified axis. Additionally, trans. stands for translation, and rot. for rotation. These abbreviations are used consistently throughout the paper.

4.1 Translation Results

Experiments across individual axis translations show that the model can effectively learn pose corrections within limited translation ranges. Most of the tested translations were accurately estimated, resulting in robust alignment with the expected ground truth translations. For small translations the results were still good when combining translations over two axes, but would result in incorrect estimations for medium to large translation combinations.

Exp.	Result	Ground Truth	Error
S x trans.	$(0.36, 0, 0)$	$(0.35, 0, 0)$	0.01
No rot.	$(1, 0, 0, 0)$	$(1, 0, 0, 0)$	0°
M x trans.	$(0.67, 0, 0)$	$(0.66, 0, 0)$	0.01
No rot.	$(1, 0, 0, 0)$	$(1, 0, 0, 0)$	0°
L x trans.	$(1.33, 0.01, -0.01)$	$(1.32, 0, 0)$	0.02
No rot.	$(1, 0, 0, 0)$	$(1, 0, 0, 0)$	0°
XL x trans.	$(1.13, -0.60, 0.20)$	$(1.93, 0, 0)$	0.60
No rot.	$(1, 0.03, -0.04, 0.02)$	$(1, 0, 0, 0)$	5.38°
S y trans.	$(-0.05, 0.01, 0.40)$	$(0, 0, 0.40)$	0.05
No rot.	$(1, 0, 0, 0)$	$(1, 0, 0, 0)$	0°
M y trans.	$(-0.05, -0.10, 1.02)$	$(0, 0, 0.97)$	0.12
No rot.	$(1, 0, 0, 0)$	$(1, 0, 0, 0)$	0°
L y trans.	$(-0.04, -0.02, 1.52)$	$(0, 0, 1.49)$	0.05
No rot.	$(1, 0, 0, 0)$	$(1, 0, 0, 0)$	0°
S z trans.	$(-0.05, 0.01, 0.40)$	$(0, 0, 0.40)$	0.05
No rot.	$(1, 0, 0, 0)$	$(1, 0, 0, 0)$	0°
M z trans.	$(-0.05, -0.10, 1.02)$	$(0, 0, 0.98)$	0.12
No rot.	$(1, 0, 0, 0)$	$(1, 0, 0, 0)$	0°
L z trans.	$(-0.04, -0.02, 1.52)$	$(0, 0, 1.49)$	0.05
No rot.	$(1, 0, 0, 0)$	$(1, 0, 0, 0)$	0°
S xy trans.	$(0.58, 0.18, 0)$	$(0.57, 0.18, 0)$	0.01
No rot.	$(1, 0, 0, 0)$	$(1, 0, 0, 0)$	0°
M xy trans.	$(0.08, -0.47, 0.85)$	$(0.76, 0.32, 0)$	1.34
No rot.	$(1, 0.04, -0.04, 0.01)$	$(1, 0, 0, 0)$	5.61°
S xz trans.	$(0.58, 0, -0.22)$	$(0.57, 0, -0.22)$	0.01
No rot.	$(1, 0, 0, 0)$	$(1, 0, 0, 0)$	0°
M xz trans.	$(0.70, 0, -0.35)$	$(0.46, -0.59, 0.16)$	0.81
No rot.	$(1, 0.03, -0.01, 0.01)$	$(1, 0, 0, 0)$	4.29°
S yz trans.	$(0, 0.11, -0.17)$	$(0, 0.11, -0.17)$	0
No rot.	$(1, 0, 0, 0)$	$(1, 0, 0, 0)$	0°
M yz trans.	$(0, 0.30, -0.40)$	$(0, 0.31, -0.40)$	0.01
No rot.	$(1, 0, 0, 0)$	$(1, 0, 0, 0)$	0°

Table 2. Pose correction model results for translations. See Section 4 for abbreviation explanation.

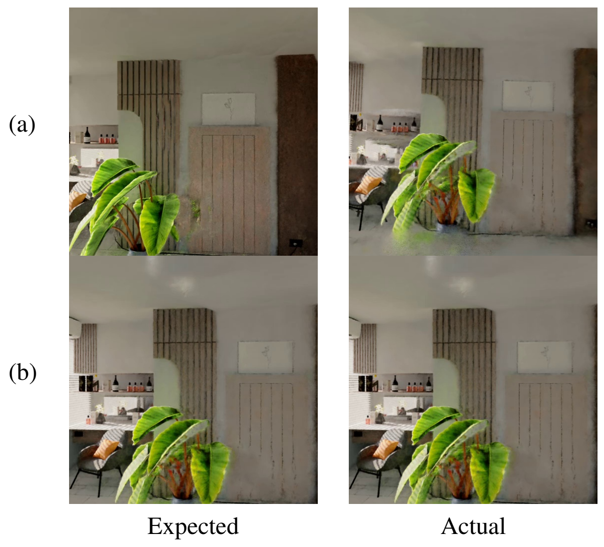

Figure 5. Renderings of x-axis translation correction results: incorrect (a) and correct (b).

The resulting renderings for an incorrectly and a correctly estimated pose correction can be seen in Figure 5. The latter is representative for all correct estimations since in those cases the expected results is optimally replicated.

4.2 Rotation Results

Regarding the correction of rotations around a single axis, the results were less promising compared to the translation experiments. Only small rotations up to 15 degrees could be accurately estimated. For any larger rotation for each of the axes the resulting correction estimation was unsuccessful. One additional observation is the tendency of the model to apply a translation correction, even though only a rotation would have been necessary. One reason might be the increased complexity of rotations compared to translations, resulting in a tendency towards translations instead of rotations. This will be further discussed in Section 5.

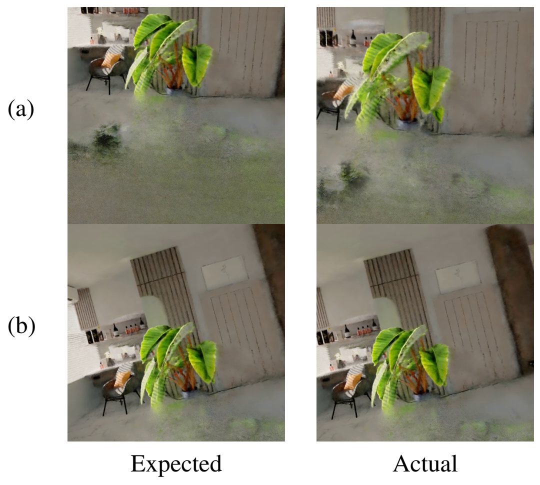

Figure 6. Renderings of z-axis rotation correction results: incorrect (a) and correct (b).

Exp.	Result	Ground Truth	Error
No trans.	$(0, 0.01, 0.13)$	$(0, 0, 0)$	0.13
$5°$ x rot.	$(1, 0.03, 0, 0)$	$(1, 0.04, 0, 0)$	$2.13°$
No trans.	$(-2.05, -0.71, -0.96)$	$(0, 0, 0)$	2.37
$20°$ x rot.	$(0.70, -0.35, 0.42, 0.46)$	$(0.99, 0.17, 0, 0)$	$102.31°$
No trans.	$(2.01, -0.89, -0.15)$	$(0, 0, 0)$	2.20
$30°$ y rot.	$(0.84, -0.34, 0, -0.42)$	$(0.97, 0, 0.26, 0)$	$71.48°$
No trans.	$(2.63, -1.04, -1.43)$	$(0, 0, 0)$	3.17
$60°$ y rot.	$(0.94, 0.25, -0.13, 0.21)$	$(0.87, 0, 0.5, 0)$	$83.60°$
No trans.	$(-0.01, 0, 0.01)$	$(0, 0, 0)$	0.01
$15°$ z rot.	$(0.99, 0, -0.03, -0.12)$	$(0.99, 0, 0, -0.13)$	$2.36°$
No trans.	$(0.75, -2.31, -4.53)$	$(0, 0, 0)$	5.14
$35°$ z rot.	$(0.31, 0.50, -0.18, -0.79)$	$(0.95, 0, 0, -0.30)$	$115.29°$

Table 3. Pose correction model results for rotations. See Section 4 for abbreviation explanation.

The images in (a) in Figure 6 show a result, which is visually close to the correct pose correction, but the accurate values were not achieved. The images in (b) in Figure 6 on the other hand are representative for all accurate pose correction estimations which only include rotations.

4.3 Combination Results

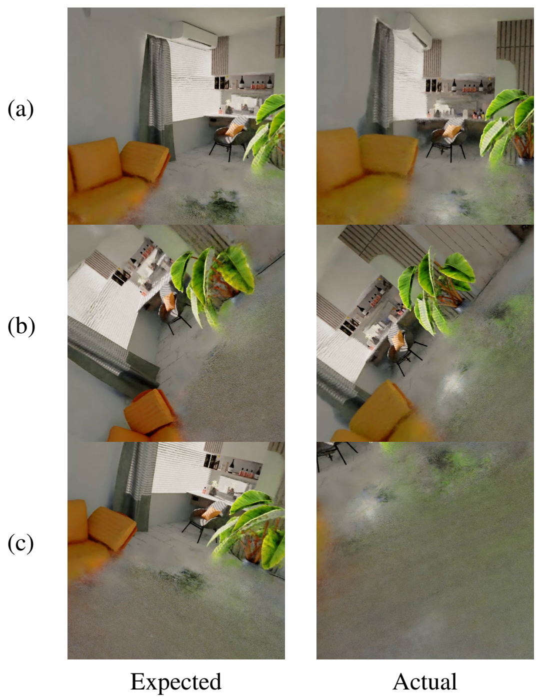

Figure 7. Renderings of pose correction estimations for a combination of translation and rotation.

When combining translations and rotations the overall error further increases compared to the results for a single transformation. As seen before, small combinations can be estimated, but the error increases quickly with increasing transformations. Figure 7 visualize different possibilities for resulting renderings incorrect relative pose correction estimations. The implications of these results and their possible explanations are discussed in Section 5.

Exp.	Result	Ground Truth	Error
S_1	$(-0.33, -0.02, 0.23)$	$(0.53, 0, 0)$	0.89
	$(0.85, -0.46, 0.23, 0.14)$	$(0.98, 0.17, 0.09, -0.02)$	$79.08°$
S_2	$(0.41, -0.01, -0.28)$	$(0.65, 0.07, -0.05)$	0.34
	$(1, -0.01, -0.04, -0.02)$	$(1, -0.03, -0.03, -0.03)$	$0°$
M_1	$(0.42, 0.09, 0.31)$	$(0.78, 0, 0)$	0.49
	$(1, 0.03, -0.02, 0)$	$(1, 0.06, 0, 0)$	$2.62°$
M_2	$(-0.57, -0.26, 0.38)$	$(0.79, 0.29, 0.32)$	1.46
	$(1, -0.01, 0.09, 0.02)$	$(0.97, -0.05, 0.25, 0.01)$	$18.91°$
L	$(-0.22, -0.21, 0.32)$	$(1.41, 0.57, 0.21)$	1.81
	$(0.93, -0.30, -0.10, -0.21)$	$(0.95, -0.16, 0.23, 0.15)$	$57.86°$
XL	$(0.05, -0.02, 0.60)$	$(1.74, 1.30, -0.08)$	2.25
	$(0.90, -0.08, -0.04, -0.42)$	$(0.87, -0.28, 0.11, -0.38)$	$28.77°$

Table 4. Pose correction model results for combinations of rotation and translation. See Section 4 for abbreviation explanation.

4.4 Multi-Camera Results

Exp.	Result	Ground Truth	Error
S z trans.	$(0.06, 0.11, 0.43)$	$(0, 0, 0.97)$	0.55
No rot.	$(1, -0.02, 0, 0)$	$(1, 0, 0, 0)$	$3.62°$
L z trans.	$(0.04, 0.13, 0.90)$	$(0, 0, 1.49)$	0.61
No rot.	$(1, -0.02, 0, 0)$	$(1, 0, 0, 0)$	$2.81°$
M x trans.	$(0.35, -0.02, -0.12)$	$(0.97, 0, 0)$	0.63
No rot.	$(1, 0, 0, 0)$	$(1, 0, 0, 0)$	$0°$
L x trans.	$(0.67, -0.02, -0.09)$	$(1.32, 0, 0)$	0.66
No rot.	$(1, 0, -0.04, 0)$	$(1, 0, 0, 0)$	$4.58°$
S y trans.	$(0, -0.44, -0.02)$	$(0, -0.48, 0)$	0.05
No rot.	$(1, 0, 0, 0)$	$(1, 0, 0, 0)$	$0°$
M y trans.	$(0, -0.59, -0.03)$	$(0, -0.64, 0)$	0.05
No rot.	$(1, 0, 0, 0)$	$(1, 0, 0, 0)$	$0°$
No trans.	$(-0.01, 0, -0.03)$	$(0, 0, 0)$	0.03
15° z rot.	$(0.99, 0.00, -0.03, -0.12)$	$(0.99, 0, 0, -0.13)$	$3.86°$
No trans.	$(0, 0, 0.12)$	$(0, 0, 0)$	0.12
5° x rot.	$(1, 0.03, 0, 0)$	$(1, 0.04, 0, 0)$	$2.30°$

Table 5. Pose correction model results for combinations of three cameras. See Section 4 for abbreviation explanation.

All other experiments were conducted with a dual-camera setup, requiring only one pose correction estimation. To further test the model's robustness with an increased number of cameras, additional experiments were performed using a three-camera configuration. While the accuracy of pose estimations declined slightly compared to the dual-camera results, the model continued to perform well in estimating pose corrections for both additional cameras when the transformations were kept moderate.

4.5 Results of Additional Experiments

More training data. For the dataset used in the experiment „XL x translation“ shown in Table 2 the number of frames was doubled. Unfortunately this did not result in an improvement and the resulting pose estimation was equal to using the original number of frames.

More rays per batch. Doubling the number of rays per batch did not lead to a significant improvement in accuracy. Larger increases were not tested due to the resulting increase in required computational time.

Lower secondary camera resolution. Lowering the resolution of the secondary camera to 200x200 pixels did not negatively affect the models ability to correctly estimate the pose correction parameters. Therefore, if the pose estimation was correct for a dataset with a resolution of 500x500, the same dataset with a resolution of 200x200 lead to the same correct results.

5. DISCUSSION

Overall, the results show that our approach is suitable for relative pose correction between two cameras. Using only the NeRF-loss value to optimize the pose correction model in combination with the alternating training approach is sufficient for accurately converging towards correct relative pose correction parameters in a given range. The following parts of this discussion section dissect the findings into different data-dependent categories with the goal of showcasing the possibilities of the pose correction model, as well as introducing possible explanations for the found limitations.

5.1 Translation Correction

Our pose relative pose correction model can effectively learn a wide range of translation corrections (Table 2). This shows that the model is highly usable for relative pose correction tasks within this range of possible translation values.

Local optimum due to visual similarity: On the other hand, the images in (a) in Figure 5 reflect a different phenomenon. Here, the pose correction model produced inaccurate pose correction values, yet these results appear explainable upon visual inspection. Specifically, the main error involves a backward z-axis translation, causing the camera's perceived field of view to expand, thereby increasing the field of view for the processed rays in the NeRF model. This incorrect translation and field of view adjustment enhanced the visual overlap between the expected and actual renderings, thus lowering the NeRF loss. The hypothesis for this observation is that the model may have converged to a local optimum where loss was minimized due to improved visual overlap.

Local optimum due to pose out of bounds: A second observation shows the pose correction parameters positioning the camera outside the scene, resulting in renderings dominated by the background. In this particular scene, the background is predominantly gray, which is the most common color in the scene. The hypothesis here is that the model converged to a local optimum. In this scenario, the loss may be lower than for an entirely incorrect pose within the scene, as the gray background likely overlaps more with the expected image due to its prevalence. This tendency is most likely related to the observation in Figure 8 and the explanations in Section 5.3.

Similar observations can be made for datasets that feature translations in more than one axis. Either the resulting pose correction parameters were estimated correctly, or the results were similar to the ones showcased by the images in (a) in Figure 5, reinforcing the proposed hypotheses.

5.2 Rotation Correction

In contrast to the translation correction, the rotation correction proved to be more error prone. Overall, more incorrect estimations could be observed. Only small rotations were correctly estimated in the training process. A successful relative pose correction and a case, where the model got close to a correct solution, but did not achieve the accurate correction values, can be seen in Figure 6.

Complexity of Rotations: Rotations are more challenging to correctly approximate due to the complex mathematical nature of applying a rotation in comparison to a translation. Firstly, using the quaternion representation requires four values to be optimized, in contrast to the three dimensional vector for a translation. Translations primarily involve straightforward summations, whereas applying a quaternion to a vector $\mathbf{v}$ requires quaternion multiplication, a more complex operation involving both the quaternion and its inverse.

$$\mathbf{v}' = q \otimes \mathbf{v} \otimes q^{-1}$$

This overall increased mathematical complexity might make it more difficult for the model to correctly optimize the rotation correction parameters in the backpropagation process. Furthermore, a small change in the rotation quaternion can result in significantly larger evaluation changes for the computed rays than an equivalent translation change. This higher sensitivity for changes can cause additional challenges in the optimization process.

5.3 Combinations of Translation and Rotation Correction

The results (a) and (b) in Figure 7 both highlight that the model may converge to incorrect solutions that still produce high visual similarity, as discussed in Section 5.1. The result (c) in Figure 7 illustrate that NeRFs interpret visual similarity differently from humans. The evaluation at each pipeline step compares the input ray bundle's results to the ground truth in a purely pixel-wise manner, without structural context. Thus, NeRFs require only significant pixel-wise overlap between two views to perceive them as visually similar. Comparing the pixel-wise RGB similarity between actual and expected renderings, Figure 8 shows that the images in (b) and (c) in Figure 7 produce similar histograms. Although the renderings differ, their pixel-wise similarity can yield similar loss values, leading to a potential local optimum. Furthermore, the histogram for two mismatched images differs greatly, reinforcing this observation.

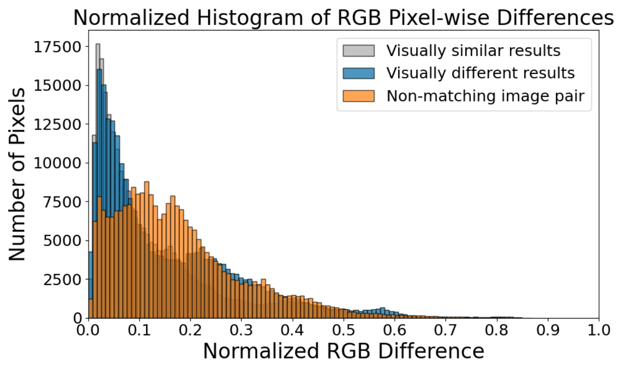

Figure 8. Histograms for the images (b) (visually different) and (c) (visually similar) in Figure 7 and a non-matching image pair.

5.4 Multi-Camera Correction

The results in Table 5 demonstrate that the proposed approach can be effective even in setups involving three cameras. While the addition of extra cameras does further limit the range of pose corrections that can be accurately estimated, the model still creates promising results for both smaller translations and rotations. This indicates that the introduced approach can also handle larger numbers of cameras with only a minor impact on performance, although more extensive testing is needed.

5.5 Additional Experiments

The additional experiments generally showed no improvements in the pose correction model's performance. Specifically, adding more training images did not significantly boost accuracy. While increasing rays per batch did improve accuracy, it also raised training time substantially, making this option impractical with the available hardware. Notably, the model successfully converged on accurate pose correction parameters even with lower-resolution images from secondary cameras, indicating a degree of robustness across varying resolutions. This resilience should be tested further with additional datasets.

5.6 Summary

Our findings show that the model effectively learns translation corrections, especially for smaller translations. Larger translations are more challenging but still yield reasonable estimates. Rotation corrections are more difficult, with the model often converging to local optima rather than the correct values, especially for larger angles. The model only succeeds for small angle adjustments, highlighting its limitations and the need for further refinement. Despite this, the model's success in correcting translations and small rotations shows promise for NeRF in multi-camera setups.

6. CONCLUSION AND OUTLOOK

In this paper, we investigated the feasibility to utilize NeRFs in multi-camera setups, where only the main camera's pose is known, in particular the ability of NeRFs to manage relative translation and rotation corrections for one or more secondary cameras. The research demonstrated that the proposed model is able to learn pose correction parameters within a limited range. Therefore, the sensitivity to large translations and challenges with rotation corrections suggest the need for further optimization and testing. Additionally, the model demonstrated some robustness to varying resolutions and camera counts, underscoring the potential of the proposed approach. Despite these challenges, the promising results showcase the possibilities, as well as the importance of continued research in this area. Ultimately, this research provides the groundwork for advancements in enabling NeRF as an innovative machine learning driven reconstruction method for practical multi-camera applications.

This research opens several pathways for future work. First, additional experiments could examine various camera configurations, types, and scene complexities to further assess scalability and refine the model's effectiveness. Alternative training strategies, such as fully decoupling the training of pose parameters or incorporating a specific loss function for pose correction, also offer potential performance improvements. Another critical area is applying the framework to real-world data from complex and lower-resolution cameras. Only when the model is applied to real-world data can it be evaluated for its applicability in the diversity of real-world scenarios.

References

Akiba, T., Sano, S., Yanase, T., Ohta, T., Koyama, M., 2019. Optuna: A next-generation hyperparameter optimization framework. *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, KDD '19, Association for Computing Machinery, New York, NY, USA, 2623–2631.

Barron, J. T., Mildenhall, B., Verbin, D., Srinivasan, P. P., Hedman, P., 2022. Mip-nerf 360: Unbounded anti-aliased neural radiance fields.

Bian, W., Wang, Z., Li, K., Bian, J.-W., Prisacariu, V. A., 2023. Nope-nerf: Optimising neural radiance field with no pose prior. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 4160–4169.

Blender Foundation, 2024. Blender. `https://www.blender.org`. Accessed: 2024-11-11.

Elhashash, M., Albanwan, H., Qin, R., 2022. A Review of Mobile Mapping Systems: From Sensors to Applications. *Sensors*, 22(11). https://www.mdpi.com/1424-8220/22/11/4262.

Fu, H., Yu, X., Li, L., Zhang, L., 2023. Cbarf: Cascaded bundle-adjusting neural radiance fields from imperfect camera poses.

Goebel, M., Iwaszczuk, D., 2023. BACKPACK SYSTEM FOR CAPTURING 3D POINT CLOUDS OF FORESTS. *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, X-1/W1-2023, 695–702. https://isprs-annals.copernicus.org/articles/X-1-W1-2023/695/2023/.

Gu, K., Maugey, T., Knorr, S., Guillemot, C., 2022. Omni-nerf: Neural radiance field from 360° image captures. *2022 IEEE International Conference on Multimedia and Expo (ICME)*, 1–6.

Hachisuka, S., Tono, A., Fisher, M., 2023. Harbingers of nerf-to-bim: a case study of semantic segmentation on building structure with neural radiance fields. *EC3 Conference 2023*, 4, European Council on Computing in Construction, 0–0.

Hee Lee, G., Pollefeys, M., Fraundorfer, F., 2014. Relative pose estimation for a multi-camera system with known vertical direction. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Hou, J., Hübner, P., Schmidt, J., Iwaszczuk, D., 2024. Indoor Mapping with Entertainment Devices: Evaluating the Impact of Different Mapping Strategies for Microsoft HoloLens 2 and Apple iPhone 14 Pro. *Sensors*, 24(4). https://www.mdpi.com/1424-8220/24/4/1062.

Häne, C., Heng, L., Lee, G. H., Fraundorfer, F., Furgale, P., Sattler, T., Pollefeys, M., 2017. 3D visual perception for self-driving cars using a multi-camera system: Calibration, mapping, localization, and obstacle detection. *Image and Vision Computing*, 68, 14–27.

Indu, S., Srivastava, S., Sharma, V., 2021. Optimal camera placement and orientation of a multi-camera system for self driving cars. *Proceedings of the 2020 4th International Conference on Vision, Image and Signal Processing*, ICVISP 2020, Association for Computing Machinery, New York, NY, USA.

Lechner, A. M., Foody, G. M., Boyd, D. S., 2020. Applications in remote sensing to forest ecology and management. *One Earth*, 2(5), 405–412.

Lehtola, V. V., Nikoohemat, S., Nüchter, A., 2021. *Indoor 3D: Overview on Scanning and Reconstruction Methods*. Springer International Publishing, Cham, 55–97.

Li, F., Yu, H., Shugurov, I., Busam, B., Yang, S., Ilic, S., 2023. Nerf-pose: A first-reconstruct-then-regress approach for weakly-supervised 6d object pose estimation.

Lin, C.-H., Ma, W.-C., Torralba, A., Lucey, S., 2021. Barf: Bundle-adjusting neural radiance fields.

Lin, Y., Müller, T., Tremblay, J., Wen, B., Tyree, S., Evans, A., Vela, P. A., Birchfield, S., 2023. Parallel inversion of neural radiance fields for robust pose estimation.

Martin-Brualla, R., Radwan, N., Sajjadi, M. S. M., Barron, J. T., Dosovitskiy, A., Duckworth, D., 2021. Nerf in the wild: Neural radiance fields for unconstrained photo collections.

Mildenhall, B., Mildenhall, B., Srinivasan, P. P., Srinivasan, P. P., Tancik, M., Tancik, M., Barron, J. T., Barron, J. T., Ramamoorthi, R., Ramamoorthi, R., Ng, R., Ng, R., 2020. NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis. *European Conference on Computer Vision*.

Müller, T., Evans, A., Schied, C., Keller, A., 2022. Instant neural graphics primitives with a multiresolution hash encoding. *ACM Transactions on Graphics*, 41(4), 1–15. http://dx.doi.org/10.1145/3528223.3530127.

nerfstudio Team, 2022. Nerfacto. `https://docs.nerf.studio/nerfology/methods/nerfacto.html`. Accessed: 2024-11-08.

Poggi, M., Ramirez, P. Z., Tosi, F., Salti, S., Mattoccia, S., Stefano, L. D., 2022. Cross-spectral neural radiance fields. *2022 International Conference on 3D Vision (3DV)*, 606–616.

Raafat, M., 2024. BlenderNeRF. Version 6.0.0. DOI: 10.5281/zenodo.13250725.

Tancik, M., Weber, E., Ng, E., Li, R., Yi, B., Wang, T., Kristoffersen, A., Austin, J., Salahi, K., Ahuja, A., Mcallister, D., Kerr, J., Kanazawa, A., 2023. Nerfstudio: A modular framework for neural radiance field development. *Special Interest Group on Computer Graphics and Interactive Techniques Conference Conference Proceedings*, SIGGRAPH '23, ACM.

Verbin, D., Hedman, P., Mildenhall, B., Zickler, T., Barron, J. T., Srinivasan, P. P., 2021. Ref-nerf: Structured view-dependent appearance for neural radiance fields.

Viriyahirunpaiboon, P., 2024. Blenderkit asset gallery. [Accessed 02-05-2024].

Wang, Z., Wu, S., Xie, W., Chen, M., Prisacariu, V. A., 2022. Nerf--: Neural radiance fields without known camera parameters.

Yen-Chen, L., Florence, P., Barron, J. T., Rodriguez, A., Isola, P., Lin, T.-Y., 2021. inerf: Inverting neural radiance fields for pose estimation. *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, IEEE, 1323–1330.