Point-based Real-Time Recalibration for Infrared Multi-Camera Systems

Benyamin Mehmandar

A Thesis

in

The Department

of

Computer Science and Software Engineering

Presented in Partial Fulfillment of the Requirements

for the Degree of

Master of Computer Science at

Concordia University

Montréal, Québec, Canada

June 2024

Abstract

Point-based Real-Time Recalibration for Infrared Multi-Camera Systems

Benyamin Mehmandar

This thesis addresses the critical challenge of achieving real-time and precise calibration for multi-camera systems, particularly crucial for applications demanding high precision. Departing from conventional methodologies characterized by limited adaptability and incapacity for on-the-fly recalibrations, this research introduces an innovative neural network-based approach designed to facilitate dynamic real-time calibration. This work incorporates the camera pose synthesis pipeline to simulate real-world camera parameters, introducing various levels of perturbation to the camera parameters to enhance model robustness. Also, a differentiable projection model is utilized to establish a direct correlation between 3D geometries and their 2D image projections, thereby facilitating concurrent optimization of intrinsic and extrinsic camera parameters. This solution entails a multi-headed deep neural network regression model and is tailored for multi-camera systems equipped with onboard processing capabilities, leveraging direct 2D projections of 3D fiducials. A comprehensive series of experiments are conducted to demonstrate the superior efficacy of the proposed method over traditional calibration techniques. This research contributes significantly to the advancement of real-time multi-camera system calibration, offering promising implications for diverse domains reliant on precise temporal synchronization and spatial accuracy.

Acknowledgments

I would like to express my deepest gratitude to my supervisor, Professor Charalambos Poullis, for their invaluable guidance, unwavering support, and insightful feedback throughout the entire process of conducting this research and writing this thesis. Their expertise and encouragement have been instrumental in shaping the direction of this work and pushing me to strive for excellence. Also, I am deeply grateful to my parents for their unwavering love, support, and understanding, which have sustained me through the ups and downs of this academic endeavor, and also my whole life.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Abstract

Camera calibration is a crucial process in computer vision, essential for transforming 3D points in the world coordinate system into 2D points in the image plane. It involves estimating intrinsic parameters, such as focal length and lens distortion, and extrinsic parameters, including the camera's position and orientation. Accurate calibration is fundamental for various applications like overlaying virtual objects, lane detection, surgical planning, and robotic navigation, where precision is critical. Traditional calibration methods, while effective, are time-consuming and often fail to capture the complexities of real-world camera systems. To address these limitations, machine learning-based approaches have emerged, offering the potential for more efficient calibration by learning the mapping between image features and camera parameters. However, these methods can struggle with large data requirements and generalization to new environments, aimed at improving calibration efficiency and reliability in real-world applications. This thesis proposes a novel real-time multi-camera calibration methodology that leverages neural networks for continuous re-calibration, enhancing adaptability to dynamic environments.

1.1 Importance of Camera Calibration

Camera calibration is a process that involves estimating the intrinsic and extrinsic parameters of a camera, which are essential for transforming 3D points in the world coordinate system to 2D points in the image plane [3]. The intrinsic parameters include the focal length, principal point, and lens distortion coefficients, while the extrinsic parameters describe the camera's position and orientation in the world coordinate system [4].

The accuracy of camera calibration directly affects the performance of computer vision systems. Inaccurate calibration can lead to errors in depth estimation, object localization, and pose estimation, which are critical tasks in many computer vision applications. For example, in augmented reality applications, accurate camera calibration is essential for overlaying virtual objects onto the real world in a visually seamless manner [5]. Similarly, in autonomous driving systems, accurate calibration is crucial for tasks such as lane detection, object detection, and pedestrian tracking [6].

1.2 Challenges in Camera Calibration

Traditional camera calibration methods, such as Zhang's method [3], require capturing images of known calibration patterns from multiple viewpoints and solving nonlinear optimization problems to estimate the camera parameters [3, 4]. While these methods are effective, they can be time-consuming and labor-intensive, especially when calibrating multiple cameras or cameras with complex optical systems [7]. Additionally, traditional calibration methods may not fully capture the complexities of real-world camera systems, such as lens distortions and non-linearities, leading to inaccuracies in calibration.

Furthermore, traditional calibration methods such as [7] may not be suitable for real-time applications where calibration needs to be performed quickly and continuously. In dynamic environments, such as outdoor robotics or surveillance systems, conditions may change rapidly, requiring the calibration parameters to be updated in real-time to maintain accuracy.

1.3 Machine Learning-Based Approaches to Camera Calibration

To address the limitations of traditional calibration methods, researchers have explored machine-learning-based approaches to camera calibration. These methods leverage neural networks to learn the mapping between image features and camera parameters, offering the potential for faster and more efficient calibration. By training neural networks on large datasets of images and corresponding camera parameters, these methods can learn complex mappings between image features and camera parameters, allowing them to generalize to unseen environments or conditions.

However, machine learning-based calibration methods may face challenges such as the need for large amounts of training data, susceptibility to overfitting, and difficulties in generalization to unseen environments or conditions [8]. Furthermore, these methods may lack interpretability, making it challenging to understand the underlying factors influencing calibration accuracy [9].

1.4 Online Recalibration Methods

Online recalibration methods offer another avenue to address the challenges of maintaining accurate calibration in dynamic environments [10]. These methods, such as [11], continuously update calibration parameters based on feedback from the environment, ensuring that the system remains accurate and robust over time. By continuously monitoring the environment and adjusting the calibration parameters accordingly, online recalibration methods can adapt to changing conditions and maintain accuracy in dynamic environments.

However, online recalibration methods may introduce additional computational overhead and complexity to the system, and they may require careful tuning to balance accuracy with computational efficiency. Furthermore, online recalibration methods may require sophisticated algorithms to detect and correct calibration errors in real-time, adding to the complexity of the system.

1.5 Proposed Approach

The necessity for real-time recalibration arises from the dynamic nature of real-world environments and the desire for computer vision systems to maintain accurate and reliable performance over time. By enabling systems to adapt to changing conditions and continuously update calibration parameters, online recalibration methods ensure that computer vision applications can meet the demands of complex and evolving scenarios.

This work proposes a novel real-time multi-camera calibration methodology that leverages neural networks to enable on-the-fly recalibration. This approach integrates a differentiable projection model, enabling direct optimization of camera parameters by propagating gradients between 3D geometries and their 2D projections. Dynamically synthesizing camera poses and introducing perturbations to camera parameters enhances the robustness and adaptability of this method to real-world operational challenges.

1.6 Contributions

This thesis presents the manuscript under review, ”Neural Real-Time Recalibration for Infrared Multi-Camera Systems”, Benyamin Mehmandar, Reza Talakoob, Charalambos Poullis. My contributions to this work include conducting background research on camera calibration for infrared systems, designing a point-based neural model using Transformers, and experimenting with artifacts of the point-based method. Additionally, I conducted experiments utilizing different encoders (Conv-1D, Graph Neural Network, and LSTM) for the point-based method, explored various loss strategies for the point-based version, implemented Bundle Adjustment, and carried out relevant experimentation. I also wrote the sections of the manuscript related to the point-based model, as presented in Chapter 3.

1.7 Thesis Organization

The organizational structure of this thesis is as follows: Section 2 provides a foundational overview of camera calibration and its significance across various applications, presenting traditional and deep learning-based methods. Section 3 delineates the proposed methodology, encompassing dynamic camera pose synthesis, network architecture, training loss functions and strategies, and rigorous experimentation. Furthermore, complementary experimental results are presented in

Section 4. Finally, ablation studies on various network architectures and loss strategies are summarized in Section 5.

Chapter 2

Literature Review

Abstract

This chapter provides an overview of traditional camera calibration methods and recent advancements in deep learning-based approaches. Traditional methods have been instrumental in geometric parameter estimation, often requiring manual intervention and extensive calibration procedures. On the other hand, deep learning-based approaches offer automation, efficiency, and adaptability to diverse imaging scenarios. Intrinsic parameter calibration methods aim to directly estimate parameters like focal length from raw image data, while extrinsic parameter calibration methods focus on estimating camera poses from image observations. Furthermore, joint calibration methods aim to simultaneously estimate both intrinsic and extrinsic parameters using deep learning techniques. This exploration aims to provide a comprehensive understanding of camera calibration methodologies spanning traditional techniques and recent advancements in deep learning.

2.1 Traditional Camera Calibration Methods

2.1.1 Radial Distortion Model

One of the earliest challenges addressed in camera calibration is lens distortion correction. The Radial Distortion Model, proposed by Brown in 1971 [12], introduces a mathematical model to account for radial distortions commonly observed in camera lenses. These distortions, including barrel and pincushion distortions, are characterized by non-linear effects on image geometry, necessitating corrective measures during calibration.

2.1.2 Direct Linear Transform (DLT) Algorithm

The Direct Linear Transform (DLT) algorithm, first introduced by Abdel-Aziz and Karara [13], is a cornerstone in camera calibration methodologies. DLT aims to estimate the intrinsic parameters of a pinhole camera from correspondences between 2D image points and their corresponding 3D world points. Despite its simplicity, DLT requires at least six point correspondences to determine the camera parameters accurately.

2.1.3 Tsai's Method

Tsai's approach [14] extends the principles of DLT by incorporating non-coplanar points for calibration. By considering a broader range of geometric configurations, Tsai's method enhances the versatility of camera calibration techniques, enabling accurate parameter estimation in more complex scenarios.

2.1.4 Zhang's Method

Zhang's method [3] introduces the concept of planar calibration patterns and homography for camera calibration. This approach simplifies the calibration process by utilizing planar surfaces as calibration targets, facilitating robust parameter estimation through homographic transformations.

2.1.5 Multi-Camera Calibration

Further to the single-camera calibration methods, several scholars have shifted their research focus towards multi-camera self-calibration, motivated by the prevalence of multiple-camera setups across diverse scenes and their broad applications. One such method is introduced in [15] by Svoboda et al., which extends his previous work in [16], enabling it to perform in multiple-camera designs more conveniently. This method enables simultaneous calibration of multiple cameras located in the same environment, only requiring a standard laser pointer with the presence of at least three cameras, utilizing the MATLAB software presented in [17], leading to various applications such as 3D reconstruction and multi-view analysis. By capturing images with all cameras in the scene, extracting corresponding feature points, and establishing known 3D world coordinates, this technique derives intrinsic and extrinsic parameters for all cameras at the same time, allowing for accurate reconstruction of all cameras' relative positions and orientations in addition to their internal characteristics.

Heikkilä et al. [18] present another notable multi-camera calibration technique which adds the following steps to the classic camera calibration methods: (1) compensating for distortion caused by circular features and (2) correcting the distorted image coordinates. This method requires capturing multiple images of a planar calibration object, such as a checkerboard or grid as shown in Figure 2.1, from varied viewpoints using all the cameras. Corresponding 2D points extracted from the images are paired with the known 3D points, which are then used for estimating the intrinsics and extrinsics of all the cameras. In order to achieve accurate calibration results, this method utilizes a nonlinear optimization algorithm, which refines the initial parameter estimates, compensating for any inaccuracies introduced during the initial parameter estimation process. This optimization uses the parameters from the DLT method as the initial values to prevent them from sticking in the local minima. The Heikkilä method is known for its robustness to lens distortion and ability to handle a wide range of camera models, including pinhole and fish-eye lenses. Since its emergence, it has become popular for camera calibration in various computer vision applications, including 3D reconstruction, object tracking, and augmented reality.

2.2 Deep Learning-Based Camera Calibration Methods

Although the aforementioned methods have been widely used since their emergence and can provide sufficient solutions in various scenarios, they need more labor-intensive, time-consuming, and sometimes expensive procedures. In light of the exceptional performance exhibited by Artificial Neural Networks across various tasks, researchers began to think of applying these powerful networks to tackle the camera calibration problem.

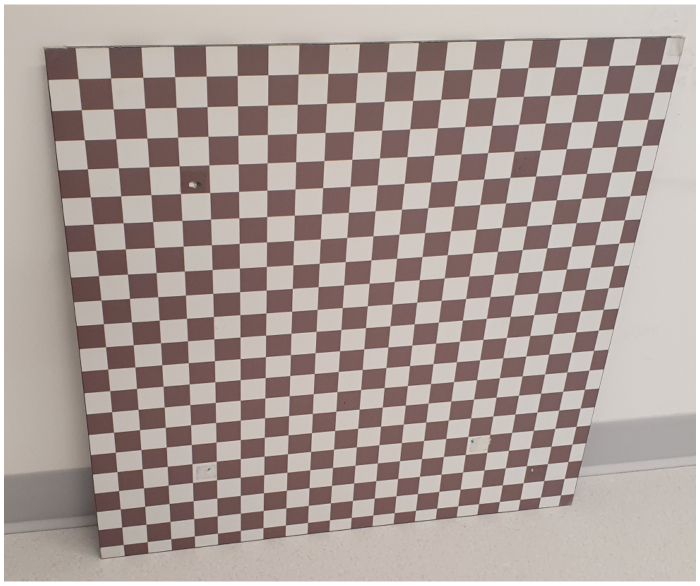


Figure 2.1: Checkerboard calibration object used for camera calibration. The checkerboard pattern facilitates the accurate estimation of the camera's intrinsic and extrinsic parameters by providing well-defined reference points.

Transformers, as one of these influential architectural paradigms, introduced by Vaswani et al. in [1], has fundamentally reshaped the landscape of deep learning, transcending the confines of traditional recurrent and convolutional architectures. Positioned initially for sequence-to-sequence tasks like machine translation, Transformers stand out for their revolutionary self-attention mechanisms. These mechanisms endow the model with the remarkable ability to dynamically weigh the significance of different elements within a sequence, thus effectively capturing intricate relationships and dependencies over long ranges. Figure 2.2 shows the architecture of the Transformer network.

The attention mechanism in Transformers allows the model to dynamically focus on different parts of the input sequence, capturing complex relationships and dependencies. In the Scaled Dot-Product Attention, queries (Q), keys (K), and values (V) are used to compute attention scores through dot products, which are then scaled by the square root of the key dimension. These scores are normalized via a softmax function to create attention weights, which are used to form a weighted sum of the values, effectively highlighting the most relevant parts of the input.

Multi-Head Attention extends this concept by applying multiple attention heads in parallel, each with its own learned linear projections of Q, K, and V. This enables the model to capture diverse features from different parts of the sequence simultaneously. The outputs of these attention heads are concatenated and passed through a final linear layer, enhancing the model's ability to understand and process complex data structures effectively. Figure 2.3 depicts the diagram of both mentioned attention mechanisms, as presented in [1].

Transformers, with their exceptional capacity for attending to relevant parts of the input, offer unmatched benefits in regression tasks. Their capability extends far beyond sequential data, permeating into diverse domains such as computer vision, finance, and geospatial analysis. In computer vision, for instance, Transformers have shown promising results in tasks such as object detection, image regression, and even intricate geometric parameter estimation tasks like camera calibration [19]. Their adeptness in capturing spatial dependencies and global context contributes significantly to the advancement of such applications.

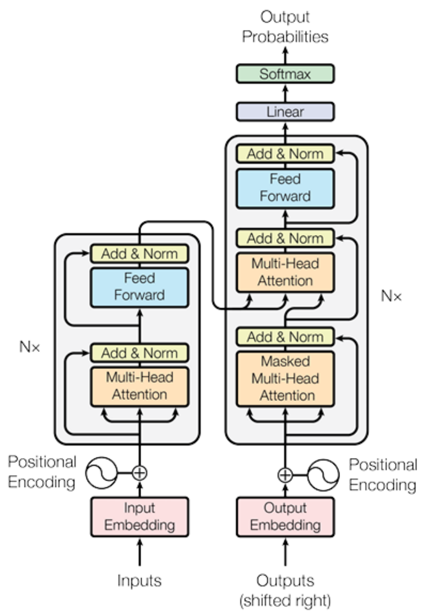


Figure 2.2: Transformer architecture, as shown in [1]. The Transformer uses stacked self-attention and point-wise, fully connected layers in both the encoder (left) and decoder (right), allowing for efficient processing of input and output sequences.

In the realm of time series forecasting, Transformers have emerged as a game-changer, leveraging their attention mechanisms to capture temporal dependencies and subtle patterns within sequential data. This fusion of temporal awareness and attentional focus enables accurate predictions in domains like financial forecasting, where even subtle trends and irregularities can have significant implications.

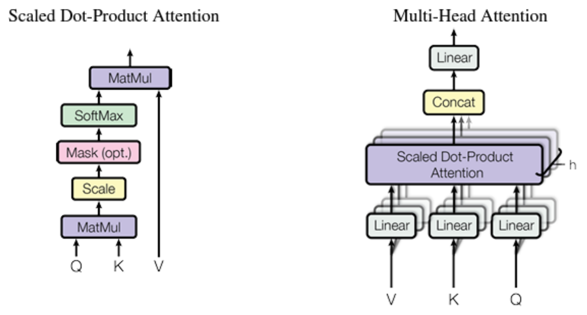


Figure 2.3: The left side illustrates the Scaled Dot-Product Attention mechanism, which computes attention scores by taking the dot product of queries and keys, scaling them, and applying a softmax function. The right side shows the Multi-Head Attention mechanism, which runs multiple attention heads in parallel, allowing the model to capture diverse features and dependencies across different parts of the input sequence, as introduced in [1].

Complementing the Transformer's capabilities, Multilayer Perceptron (MLP) stacks function as the primary tool for regression modeling, adept at approximating complex nonlinear functions. With multiple layers of densely connected neural units, MLPs offer high flexibility, enabling them to capture intricate mappings between input features and target variables across diverse domains.

In regression tasks, MLP stacks serve as powerful function approximators, mapping learned representations from the input space to the target space. Their capacity for modeling complex nonlinear relationships makes them indispensable in domains like healthcare, where predicting patient outcomes based on diverse clinical variables demands robust and flexible regression models. Additionally, the utilization of multiheaded MLP regressors offers several distinct advantages in regression tasks. By employing multiple parallel MLP subnetworks, each focusing on different aspects of the input data, multihead MLP regressors can effectively capture complex nonlinear relationships within heterogeneous datasets. This approach facilitates enhanced model flexibility and adaptability, enabling more robust predictions across diverse domains, which could improve accuracy, efficiency, and interpretability in various real-world applications.

The integration of Transformers and MLP stack architectures has yielded remarkable advancements in regression modeling. In computer vision tasks such as object detection, image regression, and pose estimation, this combination enables models to capture both local features and global context, yielding superior performance in tasks requiring precise geometric parameter estimation, particularly in camera calibration [20]. Furthermore, in natural language processing (NLP), Transformers have revolutionized tasks such as text classification, sentiment analysis, scene understanding [21], and language modeling [22]. Their ability to capture contextual information and long-range dependencies has led to significant improvements in machine translation models, sentiment analysis systems, and question-answering systems.

Employing these powerful networks for camera calibration is a paradigm shift and marks a pivotal advancement in the field, promising to revolutionize the landscape of camera calibration and beyond. Additionally, a subset of these networks demonstrates calibration independent of camera models and labels, exhibiting considerable potential for impactful and significant real-world applications. Learning-based camera calibration problems can be categorized into Intrinsic parameters calibration (including the focal length and principal point), Extrinsic parameters calibration (including the rotation matrix and translation vector), and joint Intrinsic and Extrinsic Calibration.

2.2.1 Intrinsic Parameter Calibration

Deep learning-based approaches for intrinsic parameter calibration aim to directly estimate parameters such as focal length and principal point from raw image data, bypassing the need for explicit feature extraction or manual calibration procedures. Workman et al. [23] proposed a deep neural network trained on natural images to regress focal lengths, demonstrating the feasibility of learning intrinsic camera parameters from raw pixel intensities. Similarly, Cramariuc et al. [24] introduced a data-driven approach to detect intrinsic miscalibration in cameras, leveraging deep learning to assess the degree of calibration errors caused by external disturbances or sensor imperfections.

2.2.2 Extrinsic Parameter Calibration

Deep learning-based methods for extrinsic parameter calibration focus on estimating camera poses, including position and orientation, from image observations. Kendall et al. [25] employed a deep convolutional neural network to regress 6 degrees of freedom camera poses, enabling accurate localization of cameras in 3D space. By learning complex spatial relationships directly from image data, these approaches offer robustness to noise and variability in camera poses, enhancing the accuracy of extrinsic parameter estimation.

2.2.3 Joint Intrinsic and Extrinsic Calibration

Some recent works have explored joint calibration of intrinsic and extrinsic parameters using deep learning techniques. Hold-Geoffroy et al. [26] trained a deep convolutional neural network to simultaneously estimate both intrinsic and extrinsic parameters from images, leveraging large-scale panorama datasets for robust calibration. Similarly, Butt et al. [27] proposed a multi-task learning approach for joint prediction of intrinsic and extrinsic parameters, demonstrating improved performance through integrated optimization of calibration tasks.

2.3 Conclusion

In conclusion, this chapter has explored traditional camera calibration methods alongside recent advances in deep learning-based approaches. While traditional techniques lay the groundwork for geometric parameter estimation, deep learning offers automation and adaptability to diverse scenarios. This section discussed intrinsic and extrinsic parameter calibration methods, as well as joint calibration approaches, showcasing the potential of deep learning in addressing complex challenges. This chapter serves as a comprehensive overview, bridging traditional methods with innovative deep-learning solutions.

Chapter 3

Neural Real-Time Recalibration for Infrared Multi-Camera Systems

The following is a verbatim copy of the manuscript currently under review, titled ”Neural Real-Time Recalibration for Infrared Multi-Camera Systems”, authored by Benyamin Mehmandar[1], Reza Talakoob[2], and Charalambos Poullis. My contributions include conducting experiments utilizing different encoders (Transformer, Conv-1D) for the point-based method, exploring various loss strategies for the point-based version, making comparisons with Bundle Adjustment, and carrying out relevant experimentation.

Abstract

In this paper, we address the challenge of real-time, highly-accurate calibration of multi-camera infrared systems, a critical task for time-sensitive applications. Unlike traditional calibration techniques that lack adaptability and struggle with on-the-fly recalibrations, we propose a neural network-based method capable of dynamic real-time calibration. The proposed method integrates a differentiable projection model that directly correlates 3D geometries with their 2D image projections and facilitates the direct optimization of both intrinsic and extrinsic camera parameters. Key to our approach is the dynamic camera pose synthesis with perturbations in camera parameters, emulating

[1]equal contribution
[2]equal contribution

realistic operational challenges to enhance model robustness. We introduce two model variants: one designed for multi-camera systems with onboard processing of 2D points, utilizing the direct 2D projections of 3D fiducials, and another for image-based systems, employing color-coded projected points for implicitly establishing correspondence. Through rigorous experimentation, we demonstrate our method is more accurate than traditional calibration techniques *with or without* perturbations while also being *real-time*, marking a significant leap in the field of real-time multi-camera system calibration.

3.1 Introduction

Camera calibration is the fundamental process of estimating the camera's intrinsic and extrinsic parameters and is an essential part of many computer vision systems. Accurate calibration is important in various applications, ranging from 3D reconstruction to augmented reality, especially in settings demanding high accuracy, like surgical environments. Traditional calibration methods provide analytical frameworks for addressing camera calibration. However, they require capturing an object of known geometry from multiple viewpoints, then extracting points and establishing correspondences. The inherent computational complexity of these conventional methods tends to increase dramatically with the number of cameras, images, and correspondences, making them impractical for real-time applications.

In time-critical applications requiring high accuracy, standard commercial multi-camera systems have a fixed camera configuration and come pre-calibrated from manufacturers. However, the calibration of these systems deteriorates over time because of wear and tear and, more commonly, because of the buildup of debris on critical components like the fiducials or lenses. In the field of vision-based computer-assisted surgery, calibration problems frequently prevent multi-camera systems from meeting the exact specifications and stringent accuracy standards required for surgical procedures. Such discrepancies can compromise the effectiveness and safety of medical procedures.

This deterioration in calibration emphasizes the need for sophisticated calibration methods. Coordinate Measuring Machines (CMM) are commonly used to improve calibration accuracy, offering a potential solution to this problem. Nonetheless, CMMs are expensive, and manufacturers typically calibrate the multi-camera system before delivery, failing to account for potential discrepancies after deployment. Traditional methods for detecting calibration errors, such as those based on epipolar geometry, face significant computational challenges in multi-camera setups and do not support on-the-fly recalibration, making them ineffective in dynamic environments.

System Context and Problem Statement. This work addresses the calibration challenges inherent in infrared multi-camera systems designed for time-critical applications, specifically vision-based computer-assisted surgery. These systems function exclusively within a fixed distance from a central point of interest, allowing rotation while maintaining a constant radius to ensure comprehensive visual coverage. This operational design is very important for three main reasons: first, to comply with the manufacturer's operating specifications and maintain system integrity and performance; second, to ensure that the area of interest is in-focus across all cameras for accurate tracking of medical instruments; and third, to minimize occlusions and reduce instances where tracked markers are not visible in all camera views. By customizing our approach to accommodate these multi-camera configurations and optimizing calibration for a fixed range of motion, we are able to significantly improve the accuracy and achieve the real-time calibration required for such time-critical applications.

In this paper, we introduce a novel real-time multi-camera calibration method that leverages neural networks to provide on-the-fly recalibration. Our model is trained on synthesized camera poses resulting from OEM calibration parameters, with perturbations applied to the intrinsic and extrinsic parameters. The perturbations emulate real-world operational challenges, thereby enhancing the model's practical applicability. We demonstrate, through rigorous experimentation, that our method not only adapts to alterations in calibration parameters in real time but also surpasses conventional calibration techniques in accuracy. Our key technical contributions are threefold:

- First, we introduce a real-time neural calibration method for multi-camera systems, marking a departure from traditional offline calibration methods. Our method employs a differentiable projection model to flow gradients between 3D geometries and their 2D projections, allowing for direct optimization of camera parameters.

- Second, we enhance the robustness and applicability of our method by dynamically synthesizing camera poses at each epoch and incorporating perturbations to the OEM calibration intrinsic and extrinsic parameters to simulate realistic operational challenges.
- Finally, we introduce two variants of our model: the first is designed for multi-camera systems equipped with onboard processing, directly outputting the 2D projections of the 3D fiducials; the second variant is designed for image-based multi-camera systems.

The paper is structured as follows: Section 3.3 provides a brief overview of the pinhole camera model. Section 3.4 discusses our methodology, including the dynamic camera pose synthesis for multi-camera systems and details on the loss functions and training strategy. Finally, Section 3.5 reports on our experimental results.

3.2 Related Work

Camera calibration is fundamental in computer vision for determining geometric parameters essential for image capture, playing a vital role in applications that require accurate scene measurements. Despite the development of various methods to calculate camera parameters, traditional techniques like the Radial Distortion Model [12], Direct Linear Transform (DLT) [13], Tsai's method [14], Zhang's approach [3], and [28] rely heavily on handcrafted features and model assumptions. These methods, while effective, are often labor-intensive and not suited for real-time or multi-camera calibration due to their complexity and the static nature of their required setup. The perspective-n-point (PnP) problem has seen advancements with [29], [30], [31], and further improvements in [32] through direct optimization methods. However, these solutions still face challenges in terms of real-time execution.

The advent of artificial neural networks has prompted a significant shift in camera calibration research. Techniques like [33], [34], and PoseNet [25] for extrinsic calibration and others [23], [24], [35], [36] for intrinsic parameter estimation leverage deep learning for more adaptable and potentially real-time calibration. Yet, these neural network approaches require extensive datasets for training, have low accuracy, and cannot generalize well across varying conditions.

In multi-camera calibration, research has extended towards self-calibration methods for setups in shared environments, as seen in the works of Svoboda *et al.*. [15], Heikkilä and Silvén [18], and also [37], [38], [39]. These methods facilitate 3D reconstruction and multi-view analysis but remain challenged by the demands of real-time processing and dynamic scenes. Recent developments have also explored joint estimation of intrinsic and extrinsic parameters, as demonstrated by [26], [27], [40], [41]. These approaches promise more integrated calibration processes through deep learning, highlighting the potential for efficient real-time calibration. Nonetheless, achieving a balance between computational efficiency and accuracy remains a critical challenge for these advanced methods.

In contrast to the aforementioned techniques, our method enables real-time recalibration, effectively predicting camera parameters in the presence of perturbations to the OEM intrinsic calibration parameters. Additionally, a streamlined procedure for dynamic camera pose synthesis facilitates its generalization to arbitrary configurations of multi-camera systems and arbitrary calibration objects.

3.3 Camera Model

For the sake of completeness, this section provides an overview of the classic pinhole camera model, incorporating lens distortion to establish the relationship between the 3D world coordinates of a point $\mathbf{P}$ and its 2D image projection $\mathbf{p} = [x, y]$. Consider $\mathbf{P} = [X, Y, Z]^T$ as a point in world coordinates, with $\mathbf{R}$ and $\mathbf{t}$ representing the camera's rotation matrix and translation vector, respectively. The transformation of the 3D point into camera coordinates $\mathbf{P}^C = [X^C, Y^C, Z^C]^T$ is given by $\mathbf{P}^C = \mathbf{R}\mathbf{P} + \mathbf{t}$. The projection from 3D to 2D coordinates is given by:

$$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \frac{1}{Z^C} \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \mathbf{P}^C, \tag{1}$$

where Z^C is the third component of $\mathbf{P}^C$, f_x and f_y the focal lengths, and c_x, c_y the principal point coordinates.

Lens Distortion. In order to accurately represent lens distortion, which commonly occurs in real camera systems, we include both the radial and tangential distortion. The distortion is represented using a polynomial model, where the radial distortion is captured by a sixth-order polynomial and the tangential distortion is addressed through first-order terms. The equations for the distorted coordinates x_{dist} and y_{dist} are given as follows:

$$\begin{aligned} x_{\text{dist}} &= x(1 + k_1 r^2 + k_2 r^4 + k_3 r^6) + 2p_1 xy + p_2(r^2 + 2x^2), \\ y_{\text{dist}} &= y(1 + k_1 r^2 + k_2 r^4 + k_3 r^6) + p_1(r^2 + 2y^2) + 2p_2 xy, \end{aligned} \tag{2}$$

where $r^2 = x^2 + y^2$, and k_1, k_2, k_3 are the radial distortion coefficients, and p_1, p_2 are the tangential distortion coefficients.

6D Rotation Parameterization. In line with the approach proposed in Zhou *et al.*. [2], our work adopts a 6D parameterization for the representation of 3D rotations, diverging from traditional quaternion and Euler angle representations. The latter are known to introduce parameter space discontinuities, as depicted in Figure 3.1, complicating the learning process for neural networks due to the inherent discontinuities. Furthermore, a critical limitation of these traditional approaches is their inability to guarantee that the network outputs are orthogonal rotation matrices, which is often a desirable property for ensuring the physical plausibility of rotations in 3D space. These limitations are particularly pronounced in tasks requiring the learning of continuous and smooth rotation spaces. For the purpose of training, a 3D rotation matrix R is parameterized by a 6D vector r_{6D}, which encapsulates the elements of R's first two columns. This parameterization provides a direct mapping to a rotation matrix without the need for complex conversions or the risk of introducing discontinuities. Specifically, during loss computation, the r_{6D} vector is seamlessly transformed back into an orthogonal rotation matrix R. This approach not only aligns with the findings from [2] that direct regression on 3x3 rotation matrices can lead to larger errors, but also addresses the need for generating orthogonal matrices directly from the network.

Differentiable Projection. The image formation process with the pinhole camera model is designed to be differentiable, facilitating the backpropagation of gradients from the loss -a function of the difference between the observed and projected points- to the camera parameters.

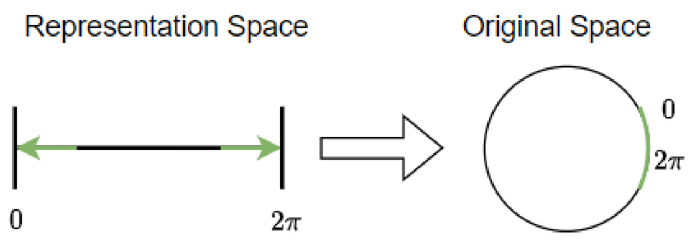


Figure 3.1: Discontinuities introduced by quaternions and Euler angle representations hinder network learning efficiency as shown in [2].

3.4 Methodology

Our methodology employs a neural network designed to estimate the intrinsic and extrinsic parameters of a multi-camera system, leveraging a known calibration object in the scene. This tailored approach ensures that, once trained, the model is specifically attuned to the multi-camera system and calibration object utilized during its training phase. We invert the traditional image formation process, enabling our model to deduce camera parameters—such as rotations (R), translations (t), focal lengths (f_x, f_y), principal points (c_x, c_y), and distortion coefficients ($k_{1,2,3}, p_{1,2}$)—from 2D projections of 3D fiducial points on the calibration object.

At the core of our methodology is the dynamic camera pose synthesis, imperative for simulating realistic conditions that a multi-camera system might encounter. We introduce controlled perturbations into the OEM camera parameters, dynamically synthesizing diverse training samples. Given the synthesized camera parameters, the process starts with projecting the 3D fiducials of the calibration object onto the image plane. These 2D projected points serve as input to our model, which then predicts the intrinsics and extrinsics of the multi-camera system. Utilizing these predicted camera parameters, we project the known 3D fiducials back onto the image plane through a differentiable process. The deviation between the projections from the synthesized and predicted camera parameters is quantified using a loss function. Figure 3.2 provides an overview of our method.

3.4.1 Dynamic Camera Pose Synthesis

Dynamic camera pose synthesis begins with the OEM calibration parameters of a multi-camera setup, typically determined by the manufacturing process. We represent a multi-camera setup of N_C cameras as $C_i, 0 \leq i < N_C$, where C_i^{OEM} denotes the initial calibration of each camera.

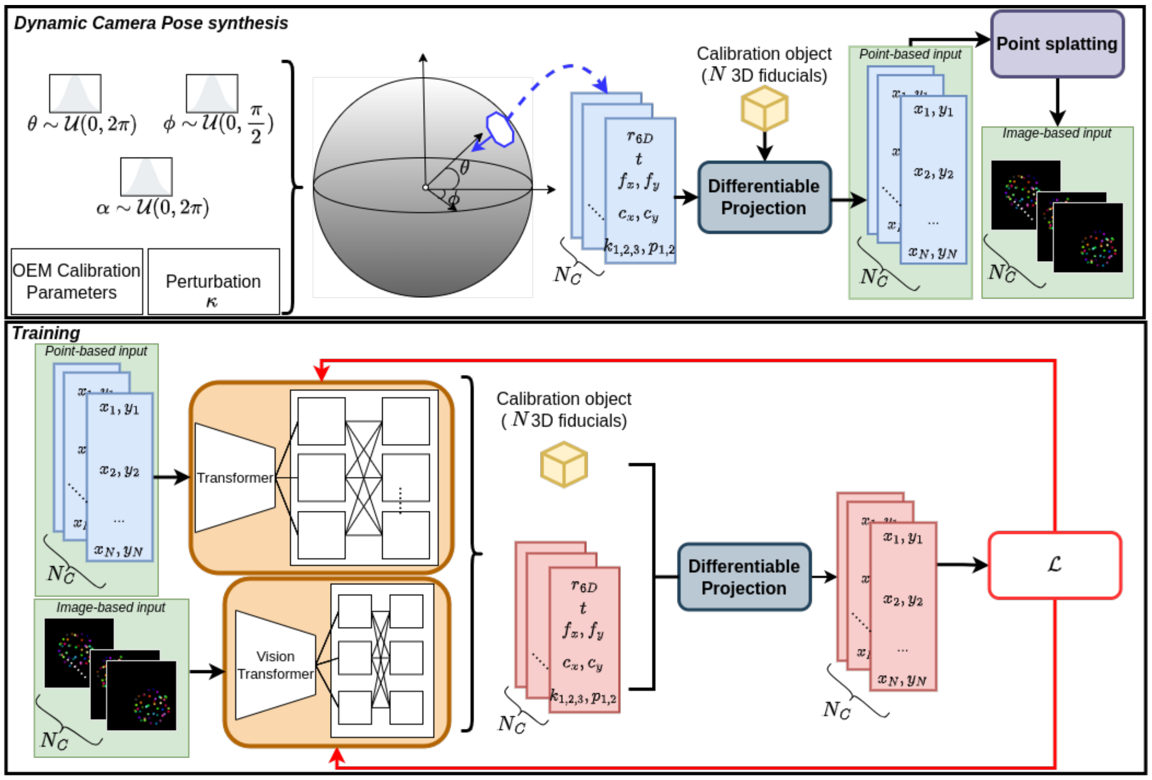

Dynamic Camera Pose synthesis
$\theta \sim \mathcal{U}(0, 2\pi)$
$\phi \sim \mathcal{U}(0, \frac{\pi}{2})$
$\alpha \sim \mathcal{U}(0, 2\pi)$
OEM Calibration Parameters
Perturbation κ
θ
ϕ
r_{6D}
t
f_x, f_y
c_x, c_y
$k_{1,2,3}, p_{1,2}$
N_C
Calibration object (N 3D fiducials)
Differentiable Projection
Point-based input
x_1, y_1
x_2, y_2
x_N, y_N
Point splatting
Image-based input
Training
Transformer
Vision Transformer
$\mathcal{L}$

Figure 3.2: **Technical overview.** Our methodology begins with the synthesis of dynamic camera poses (see top fig.). Given spherical angles ϕ (azimuth), θ (elevation), along with the intrinsic rotation angle α, the OEM calibration parameters, the maximum perturbation limit κ, and known 3D fiducials (e.g. a cube calibration object), this module performs two primary functions: (i) it synthesizes poses for the multi-camera system, and (ii) it computes the projected 2D points. Subsequently, it employs point splatting to render images of these points. During training (see bottom fig.), the synthesizes poses and projected points (alternatively rendered images) are used to train the neural network. A differentiable projection ensures the propagation of gradients from the loss $\mathcal{L}$ back to the predicted camera parameters.

Perturbations

To create a robust model capable of handling changes in calibration, we introduce perturbations to the OEM camera parameters, generating perturbed parameters $C_i^{pert} = C_i^{OEM} \times (1 + \delta)$, where δ represents the perturbation defined as $\delta = \kappa \times \mathcal{U}(0, 1)$, with κ controlling the maximum desired perturbation.

Camera Pose Synthesis

For the synthesis of camera poses, we employ a two-step process, ensuring both focus towards the fiducial points and variability in camera orientation to prevent model overfitting. The procedure is as follows.

The centroid of the camera system is dynamically positioned on a hemisphere's surface, ensuring varied perspectives. The radius of the hemisphere is determined in advance based on the application requirements and the range of motion of the multi-camera system. Specifically, the centroid's position $P_{centroid}$ on the hemisphere is determined by:

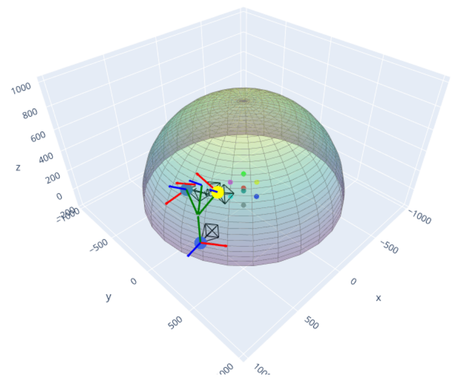
$$P_{centroid} = (x_c, y_c, z_c) = \rho \cdot (\sin(\phi) \cdot \cos(\theta), \sin(\phi) \cdot \sin(\theta), \cos(\phi)), \tag{3}$$

where ρ denotes the hemisphere radius, and $\theta \sim \mathcal{U}(0, 2\pi)$, $\phi \sim \mathcal{U}(0, \pi/2)$ are angles sampled from a uniform distribution, ensuring the centroid is randomly positioned over the hemisphere.

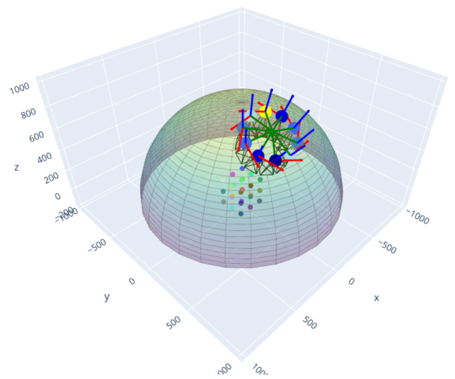
Next, a rotation R_{focus} is applied to align the camera's viewing direction towards the fiducials' centroid, ensuring that the camera is oriented towards the area of interest. This alignment is critical for simulating realistic camera setups where the fiducials are within the camera's field of view. To introduce additional randomness and prevent the network from overfitting to specific camera locations, a secondary random rotation R_{random} is applied by an intrinsic rotation angle $\alpha \sim \mathcal{U}(0, 2\pi)$ around the centroid point. The combination of R_{focus} and R_{random} ensures that each camera is not only oriented towards the fiducials but also positioned and rotated in a manner that provides a diverse set of viewing angles and positions. This diversity is imperative for training a robust model capable of generalizing across various camera orientations and positions.

As a final validation step, we conduct a visibility check to ensure all fiducials are within the field of view of all cameras. This step is essential, since the random placement of cameras on the hemisphere might result in scenarios where not all fiducials are visible from all cameras.

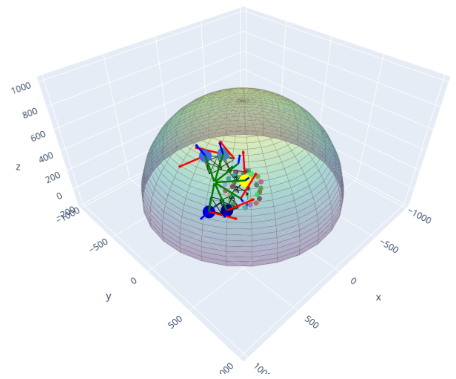
The dynamic camera pose synthesis for each training epoch diversifies the dataset, improving model generalization. Figure 3.3 illustrates synthesized camera poses for various multi-camera systems e.g. T-shape ($N_C = 4$ ⊤), U-shape ($N_C = 7$ ⊔), O-shape ($N_C = 10$ ○), and different calibration objects e.g. cube ($N_{fid.} = 8$), cube ($N_{fid.} = 27$), sphere ($N_{fid.} = 64$).



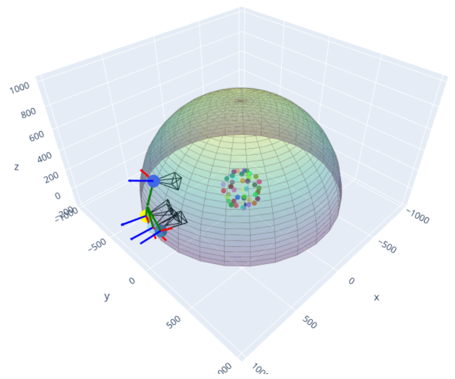
(a) T; cube(8)



(b) O; cube(27)



(c) U; sphere(64)



(d) T; sphere(64)

Figure 3.3: **Dynamic Camera Pose Synthesis.** Our framework supports arbitrary configurations of multiple cameras as well as a wide range of calibration objects. To synthesize camera poses, we employ a random uniform sampling strategy across three dimensions to ensure a comprehensive exploration of the pose space: azimuth (θ), elevation (ϕ), and roll (α), where $\theta \sim \mathcal{U}(0, 2\pi)$, $\phi \sim \mathcal{U}(0, \frac{\pi}{2})$, and $\alpha \sim \mathcal{U}(0, 2\pi)$. Additionally, Original Equipment Manufacturer (OEM) calibration parameters and a predefined maximum perturbation limit (κ) are incorporated.

3.4.2 Network Architecture

We introduce two variants: the first one, known as the point-based model (Pt), is intended for multi-camera systems with onboard processing that directly outputs the 2D projections of the 3D fiducials, which is the typical case for IR multi-camera systems; the second one is the image-based model (Img), designed for multi-camera systems that output images for subsequent processing, rather than the point coordinates. The architecture of both variants is detailed in the supplementary material along with an ablation study on the components.

3.4.3 Loss

Our loss function is crafted to encapsulate multiple facets of camera parameter estimation. The primary term $\mathcal{L}\epsilon$ is incorporated to enforce geometric consistency. It measures the RMSE between 2D points x, x' obtained by projecting the 3D points using the predicted and ground-truth parameters, respectively, and is given by $\mathcal{L}\epsilon = RMSE(x, x')$. For rotation, we use geodesic loss $\mathcal{L}_{geo}$, computed from the predicted and ground-truth rotation matrices, to penalize deviations in the orientation. Additionally, $\mathcal{L}_{diff}$ is the root mean square error (RMSE) that quantifies the difference between the predicted and ground-truth parameters. We also employ a scaling value for the rotation and distortion coefficients since their values are very small compared to other camera parameters, as illustrated by the experiments in the ablations in the supplementary material.

3.4.4 Training

In the initial phase of 10,000 epochs, we utilize a simplified loss function focused on $\mathcal{L}_{diff}$ and $\mathcal{L}_{geo}$ to establish a robust baseline, formulated as $\mathcal{L} = \lambda_1 \times \mathcal{L}_{diff} + \lambda_1 \times \mathcal{L}_{geo}$. Subsequently, we introduce the full compound loss, expressed as $\mathcal{L} = \lambda_1 \times \mathcal{L}_{diff} + \lambda_1 \times \mathcal{L}_{geo} + \lambda_2 \times \mathcal{L}_{\epsilon}$. In all experiments, the coefficients are set to $\lambda_1 = 100, \lambda_2 = 0.01$, with the batch size of 512 and 8 for Pt and Img variants, respectively.

Our training strategy is based on introducing a random perturbation to the camera parameters in every epoch, to simulate a wide spectrum of deviations that may occur due to buildup of debris on critical components like the fiducials or lenses. This methodical addition of perturbation aids in better exploration of the parameter space and enhances the robustness and accuracy of the estimated camera parameters. We employ an Adam optimizer with parameter-specific learning rates. The rates are adaptively adjusted via a ReduceLROnPlateau scheduler based on the compound loss function. Furthermore, the backpropagation step employs gradient clipping for stability.

3.5 Experimental Results

There are no learning-free or neural techniques for *real-time* recalibration of *infrared multi-camera* systems. In the supplementary material, we adopt standard calibration and optimization

Table 3.1: **Experimental results.** $\textbf{RE}_{avg}^{20K}$ is the average RMSE reprojection error on 3 different trials (synthetic test sets, each comprising 20,000 data samples). Training on all models includes adding a **Max Perturb.** $\kappa \in [\textbf{min}\%, \textbf{max}\%]$ to the OEM camera intrinsic and extrinsic parameters respectively. $\boldsymbol{N_{fid.}}$ and $\boldsymbol{N_C}$ are the number of 3D fiducials and the number of cameras, respectively. The rotation angle α remains the same in all experiments i.e., $\alpha \sim \mathcal{U}(0, 2\pi)$. The parameters for Pt and Img models are ∼33m and ∼86m, respectively.

Row #	**Variant**	$\boldsymbol{\theta}$	$\boldsymbol{\phi}$	$\boldsymbol{N_{fid.}}$ (object)	$\boldsymbol{N_C}$ (config.)	**Max Perturb.** $\kappa_{int.}, \kappa_{ext.} \in$ [min%, max%]	$\textbf{RE}_{avg}^{20K}$ (pixels)
1	Pt	$\sim\mathcal{U}(0,2\pi)$	$\sim\mathcal{U}(0,\pi/2)$	8 ▣	10 ○	$\pm2.5, \pm2.5$	$9.93\pm 1.4 \times 10^{-4}$
2	Pt	$\sim\mathcal{U}(0,2\pi)$	$\sim\mathcal{U}(0,\pi/2)$	8 ▣	10 ○	$\pm5, \pm5$	$11.53\pm 1.5 \times 10^{-4}$
3	Pt	$\sim\mathcal{U}(0,2\pi)$	$\sim\mathcal{U}(0,\pi/2)$	8 ▣	10 ○	$\pm10, \pm10$	$14.21\pm 1.8 \times 10^{-4}$
4	Pt	$\sim\mathcal{U}(0,2\pi)$	$\sim\mathcal{U}(0,\pi/2)$	8 ▣	7 ⊔	$\pm2.5, \pm2.5$	$14.67\pm 3.5 \times 10^{-4}$
5	Pt	$\sim\mathcal{U}(0,2\pi)$	$\sim\mathcal{U}(0,\pi/2)$	8 ▣	4 ┬	$\pm2.5, \pm2.5$	$12.89\pm 2.1 \times 10^{-4}$
6	Pt	$\sim\mathcal{U}(0,2\pi)$	$\sim\mathcal{U}(0,\pi/2)$	64 ●	10 ○	$\pm2.5, \pm2.5$	$14.92\pm 7.3 \times 10^{-5}$
7	Pt	0	0	8 ▣	6 ○	$0, 0$	$1.12\pm 5 \times 10^{-7}$
8	Pt	0	0	8 ▣	6 ○	$\pm2.5, 0$	$3.08\pm 9.5 \times 10^{-6}$
9	Pt	0	0	8 ▣	6 ○	$\pm10, 0$	$8.04\pm 7.2 \times 10^{-4}$
10	Img	0	0	8 ▣	6 ○	$\pm2.5, 0$	$4.55\pm 1.8 \times 10^{-4}$
11	Img	0	0	8 ▣	10 ○	$\pm2.5, \pm2.5$	$4.12\pm 2.4 \times 10^{-6}$
12	Img	0	0	8 ▣	10 ○	$\pm5, \pm5$	$6.01\pm 1.6 \times 10^{-5}$
13	Img	0	0	8 ▣	10 ○	$\pm10, \pm10$	$6.97\pm 1.1 \times 10^{-5}$

techniques for infrared cameras and demonstrate the impracticality of using traditional methods for on-the-fly calibration. Below, we provide a comprehensive evaluation of our method's accuracy across various scenarios, demonstrating its suitability for time-sensitive, high-accuracy applications. We conclude with an assessment of the generalizability of our approach to diverse multi-camera system configurations and calibration objects.

In Table 3.1, we detail the performance of our model under various conditions, reporting the average RMSE reprojection error RE_{avg}^{20K} across three trials on synthetic test sets, each comprising 20,000 data samples. Our model demonstrates robust adaptability to different levels of perturbations, where performance gracefully degrades as perturbations reach extreme values. For example, with an O-shape camera system comprising 10 cameras ($N_C = 10$ ○), and a calibration object with 8 fiducials ($N_{fid.} = 8$), our method achieves a RE_{avg}^{20K} of $9.93\pm 1.4\times10^{-4}$, $11.53\pm 1.5\times10^{-4}$, and $14.21\pm 1.8 \times 10^{-4}$, for perturbations of up to $5\%, 10\%, 20\%$, respectively, as shown in rows 1-3. As we explain in the subsequent section, the range of motion is often constrained in real-world scenarios due to operational limitations. Nevertheless, our approach demonstrates robust performance and effectively predicts camera poses across a wide range of motion, i.e., $\theta \sim \mathcal{U}(0, 2\pi)$, $\phi \sim \mathcal{U}(0, \pi/2)$, and $\alpha \sim \mathcal{U}(0, 2\pi)$.

3.5.1 Generalization to Arbitrary Configurations & Calibration Objects

We conducted comprehensive experiments with various camera configurations and calibration objects to evaluate the generalization of our method. Cameras were positioned in geometric configurations common to real-world scenarios, such as the O-shaped ($N_C = 10$ ○), U-shaped ($N_C = 7$ ⊔) and T-shaped ($N_C = 4$ ⊤) arrangements. As shown in Table 3.1 (in rows 1, 4, and 5), even with a small number of fiducials i.e., $N_{fid.} = 8$, the reprojection errors RE_{avg}^{20K} are low with $9.93\pm\ 1.4 \times 10^{-4}$, $14.67\pm\ 3.5 \times 10^{-4}$, $12.89\pm\ 2.1 \times 10^{-4}$, for the O-shape, U-shape, and T-shape configurations, respectively.

The versatility of our approach is further demonstrated through tests involving calibration objects with different shapes and numbers of fiducials. Specifically, we evaluated our model using a calibration cube with $N_{fid.} = 8$ and a calibration sphere with $N_{fid.} = 64$ fiducials. As shown in Table 3.1(rows 1, 6), our method maintains a consistent error profile for both calibration objects.

3.5.2 Comparison with CMM-calibrated multi-camera system.

As previously stated, it is important to recognize that regressing camera poses under the conditions that $\theta \sim \mathcal{U}(0, 2\pi)$, $\phi \sim \mathcal{U}(0, \pi/2)$ is more challenging than in typical real-world scenarios where multi-camera systems have predefined operational ranges for θ and ϕ. Our method, when practically tested in a surgical setting using a multi-camera system with a predefined operational range of motion (specifically, an overhead fixed O-shaped system ($N_C = 6$ ○) where $\theta = \phi = 0$, and a calibration object with $N_{fid.} = 8$), demonstrated superior performance. This system was previously calibrated (offline) using a high-end CMM. Our approach surpassed the CMM calibration in terms of reprojection error within this motion range. CMM calibration parameters *without perturbations* led to an average reprojection error of 1.80 pixels. In contrast, our method led to an average reprojection error of $1.12\pm\ 5 \times 10^{-7}$ on a model trained *without perturbations* (Table 3.1; row 7), and an average reprojection error of $3.08\pm 9.5 \times 10^{-6}$ (Table 3.1; row 8) and $8.04\pm\ 7.2 \times 10^{-4}$ (Table 3.1; row 9) for *perturbations of up to 5% and 20%*, respectively. Additionally, with an average inference time of 0.0026 seconds on a system equipped with a Nvidia RTX 4090 GPU, our method proves capable of supporting real-time applications.

3.6 Conclusion

Our work presents a neural calibration method tailored for the real-time and adaptive calibration of multi-camera systems. Central to our approach is the combination of dynamic camera pose synthesis with a differentiable projection model, which facilitates the direct optimization of camera parameters from image data. Comprehensive experimental analysis demonstrated the robustness of the method and its capacity to accurately predict calibration parameters while accommodating random perturbations.

We further elaborated on the practicality of our method, contrasting the impracticality of recalibrating or optimizing camera parameters at each step in real-time applications. Our evaluations revealed our method's superior accuracy and generalizability across different scenarios. The introduction of two variants—catering to systems with either direct 2D projections of 3D fiducials or color-coded 2D projected points—proves our method's flexibility and broad applicability in diverse operational contexts. Future research directions include the integration of perturbations into the extrinsic parameters of the camera models. This holds the potential to significantly broaden the applicability of our approach in more dynamically varied environments.

Chapter 4

Additional Experimental Results

The following is a verbatim copy of the manuscript submitted as supplementary material for the paper titled ”Neural Real-Time Recalibration for Infrared Multi-Camera Systems”.

Abstract

In the supplementary material, we present details on the network architectures and an ablation on their components, comparisons with standard calibration and optimization techniques, qualitative results and training nuances of the real-time neural multi-camera system calibration method. Additionally, through a series of visualizations, we illustrate the precision of our model in predicting camera poses against ground truth, with an emphasis on robustness in the presence of perturbations. Lastly, we address the limitations and discussions.

4.1 Network Architecture

We introduce two variants: the first one, known as the point-based model (Pt), is intended for multi-camera systems with onboard processing that directly outputs the 2D projections of the 3D fiducials, which is the typical case for IR multi-camera systems; the second one is the image-based model (Img), designed for multi-camera systems that output images for subsequent processing, rather than the point coordinates. The architecture of both variants is summarized in the supplementary material.

4.1.1 Point-based variant.

Given an input tensor $X \in \mathbb{R}^{N_C \times N_{fid.} \times 2}$, where N_C is the number of cameras, and $N_{fid.}$ is the number of fiducials, the point-based model performs the following operations:

(1) **Embedding Layer f_{embed}:** Maps input fiducial points to a higher-dimensional space: $X_{enc} = f_{embed}(X)$, where $X_{enc} \in \mathbb{R}^{N_C \times 512}$.

(2) **Camera Identity Encoding CIE:** Incorporates camera-specific information: $X^{'}_{enc} = X_{enc} + CIE(N_C, 512)$, where $X^{'}_{enc} \in \mathbb{R}^{N_C \times 512}$.

(3) **Transformer Encoder f_{TE}:** Processes embeddings through a transformer encoder to model complex relationships: $X^{''}_{enc} = f_{TE}(X^{'}_{enc})$, $X^{''}_{enc} \in \mathbb{R}^{N_C \times 512}$.

(4) **Downstream Heads h^{M}_{*}:** Process the encoder output through separate prediction heads for different outputs $X^{*}_{out} \in \mathbb{R}^{M}$ (rotation r_{6D}, translation t, focal lengths $fc = (f_x, f_y)$, principal point $pp = (c_x, c_y)$, and distortion coefficients $kc = (kc_{1,2,3}, p_{1,2})$):

$$\begin{aligned} X^{r_{6D}}_{out} &= h^{6}_{R}(X^{''}_{enc}), \quad X^{t}_{out} = h^{3}_{t}(X^{''}_{enc}), \\ X^{fc}_{out} = h^{2}_{fc}(X^{''}_{enc}), \quad X^{pp}_{out} &= h^{2}_{pp}(X^{''}_{enc}), \quad X^{kc}_{out} = h^{5}_{kc}(X^{''}_{enc}) \end{aligned}$$

(5) In a final step, the outputs X^{*}_{out} are combined to form the output tensor $\hat{X}_{out}$ representing the combined camera parameters:

$$\hat{X}_{out} = (\gamma(X^{r_{6D}}_{out}), X^{t}_{out}, X^{fc}_{out}, X^{pp}_{out}, X^{kc}_{out})$$

where $\hat{X}_{out} \in \mathbb{R}^{N_C \times 21}$, $\gamma(.)$ is a function that expands the 6D parameterized rotation to a rotation matrix as described in Section 3 in the main paper, and 21 is the number of calibration parameters per camera.

Incorporating camera identity encoding CIE into our architecture significantly improves performance by disambiguating camera perspectives, as shown in the ablations in Section 4.1.3. The CIE applies a simple one-hot encoding strategy where each camera is assigned a unique identifier in this embedding space. To accommodate scenarios where the number of cameras exceeds the embedding dimension, it introduces a small amount of noise to each encoding, ensuring that each camera's encoding remains distinct. This enhances robustness and accuracy for spatially-aware tasks and supports generalization to new configurations and arbitrary camera setups.

4.1.2 Image-based variant.

The image-based variant utilizes a Vision Transformer (ViT) to process the input image tensor $X_{in} \in \mathbb{R}^{N_C \times Ch \times H \times W}$ and predict camera parameters.

(1) **Encoder $f_{encoder}$.** The input image tensor is resized to 224×224 pixels and passed through a ViT encoder. The ViT model is adapted by replacing its classification head with an identity layer, allowing the model to output a feature representation directly $X_{enc} = f_{encoder}(X_{in})$, where $X_{in} \in \mathbb{R}^{N \times 3 \times 224 \times 224}$, and $X_{enc} \in \mathbb{R}^{N_C \times 768}$.

(2) **Bottleneck Layer $f_{bottleneck}$.** The encoded features are further processed through a bottleneck layer to reduce dimensionality and focus on relevant features for parameter prediction: $X_{bottleneck} = f_{bottleneck}(X_{enc})$, and comprises a linear layer, reducing features to $\mathbb{R}^{N_C \times 128}$.

(3) **Downstream Heads:** Similar to the point-based model variant, separate heads are used for predicting the camera parameters from the bottleneck features $X_{bottleneck}$, and combined to form the output tensor $\hat{X}_{out} \in \mathbb{R}^{N_C \times 21}$.

4.1.3 Ablation Study

We conducted an ablation study to demonstrate the impact and significance of individual components of our network. All models have been trained following the training strategy in Section 4.4 in the main paper, given the conditions explained in Table 4.1.

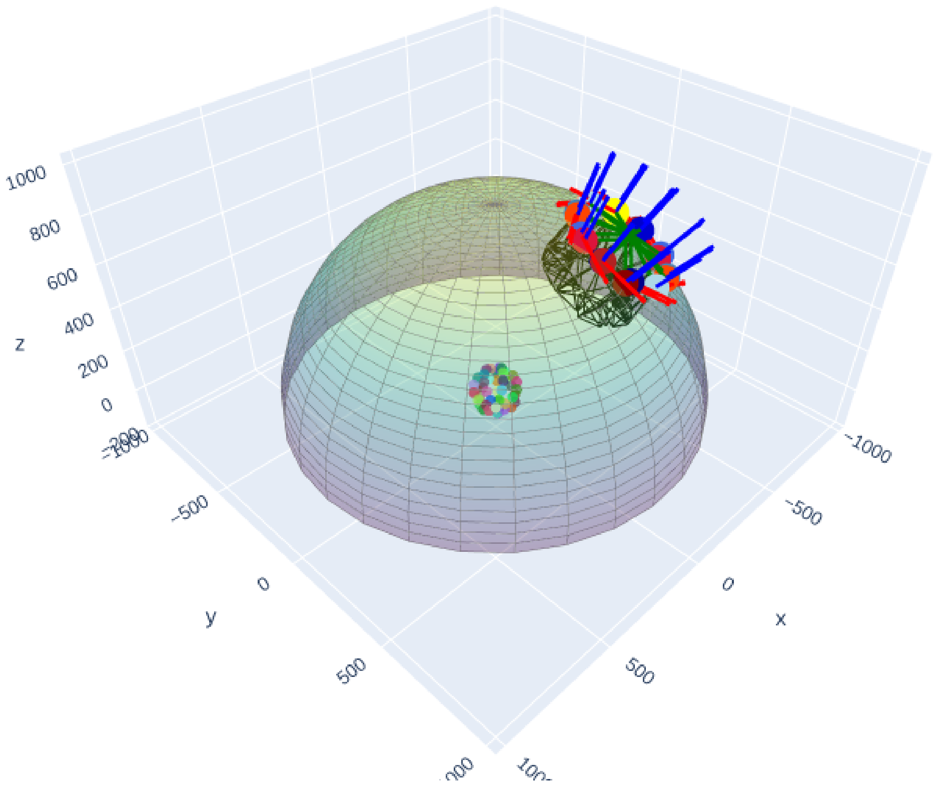


Figure 4.1: **Visualization of predicted vs ground truth camera poses.** The calibration object is a sphere with 64 fiducials. The multi-camera system configuration is O-shaped comprising 10 cameras. For closer inspection please refer to the interactive visualization in the **cameras.html** file.

Impact of Value Scaling

We highlight the significant role of scaling rotation and distortion coefficients prior to RMSE loss calculation. Given the inherently small magnitudes of these coefficients—elements of the rotation matrix ($0 < r_i < 1$) and distortion parameters ($kc_{1,2,3}, p_{1,2} \lesssim 0.1$)—their contribution to the overall loss is minimal. Introducing a scaling factor, λ_{scale}, amplified their effect, as evidenced by the increased prediction errors observed when omitting this factor, detailed in Table 4.1.

Camera Identity Encoding CIE

We showcase the significance of Camera Identity Encoding (CIE). The results in Table 4.1 clearly indicate that incorporating CIE leads to notable performance improvements. By integrating CIE, our model distinctly differentiates between camera perspectives, leading to marked improvements in accuracy for spatially-aware tasks.

Table 4.1: **Ablation study results**. Training on all models includes a perturbation of **Max Perturb.** $\kappa \in [\mathbf{-2.5}\%, \mathbf{+2.5}\%]$ of the OEM intrinsic parameters, utilizing 10 cameras ($N_C = 10$) and a calibration object with 8 fiducials ($N_{fid.} = 8$). Pt models have been trained for 300k epochs and Img models for 150k epochs.

Row #	**Variant**	**# Params**	$\boldsymbol{\theta}$	$\boldsymbol{\phi}$	**Encoder**	**Scaled** $\lambda_{scale} = 1000$	**Camera Identity Encoding** CIE	$\mathbf{RE}_{avg}^{20K}$ (pixels)
1	Pt	~ 33m	$\sim \mathcal{U}(0, 2\pi)$	$\sim \mathcal{U}(0, \pi/2)$	Transformer	✓	✓	$12.47 \pm 6 \times 10^{-4}$
2	Pt	~ 33m	$\sim \mathcal{U}(0, 2\pi)$	$\sim \mathcal{U}(0, \pi/2)$	Transformer		✓	$421.26 \pm 9.6 \times 10^{-3}$
3	Pt	~ 33m	$\sim \mathcal{U}(0, 2\pi)$	$\sim \mathcal{U}(0, \pi/2)$	Transformer	✓		$25.49 \pm 2 \times 10^{-2}$
4	Pt	~ 8m	$\sim \mathcal{U}(0, 2\pi)$	$\sim \mathcal{U}(0, \pi/2)$	CNN-1D	✓	✓	$96.93 \pm 3.9 \times 10^{-2}$
5	Img	~ 86m	0	0	ViT	✓	-	$6.89 \pm 5.8 \times 10^{-5}$
6	Img	~ 24m	0	0	CNN-2D	✓	-	$20.62 \pm 2.8 \times 10^{-5}$

Impact of Encoder

We contrast the performance impact between utilizing a transformer encoder and a 1D convolutional (CNN-1D) encoder within our model. As detailed in Table 4.1, the integration of a transformer encoder significantly enhances model accuracy. Similarly, for the image-based model, the ViT encoder results in increased precision when compared with CNN-2D.

4.2 Comparison with Standard Calibration and Optimization Techniques

In this section, we explore the differences between recalibrating and optimizing infrared cameras at every step in real-time applications. We start our discussion in Section 4.2.1, examining the drawbacks of recalibrating at each step using standard methods. Through experiments, we demonstrate the significant time required for this process, thereby illustrating the impracticality of using traditional methods for on-the-fly calibration in multi-camera systems. Following this, Section 4.2.2 presents experiments that emphasize the difficulties in minimizing reprojection errors using Bundle Adjustment (BA).

4.2.1 Inference vs. Recalibration

In this section, we address the limitations of traditional calibration methods. Traditional calibration methods require a large number of images, especially for applications demanding high accuracy. In such scenarios, the requirement can escalate to thousands of images, leading to a prohibitive increase in execution time (i.e., > 1000 images for vision-based computer-assisted surgical multi-camera systems). Even in the hypothetical scenario where capturing thousands of images instantaneously were possible, the calibration process itself remains time-consuming and scales poorly as image quantity increases. As depicted in Figure 4.2 for a single camera, this execution time grows exponentially with the number of images, rendering a recalibration at every step impractical within real-time applications. Additionally, increasing the number of iterations in the Levenberg-Marquardt refinement step leads to a linear time increase, as illustrated by the blue curve in Figure 4.2.

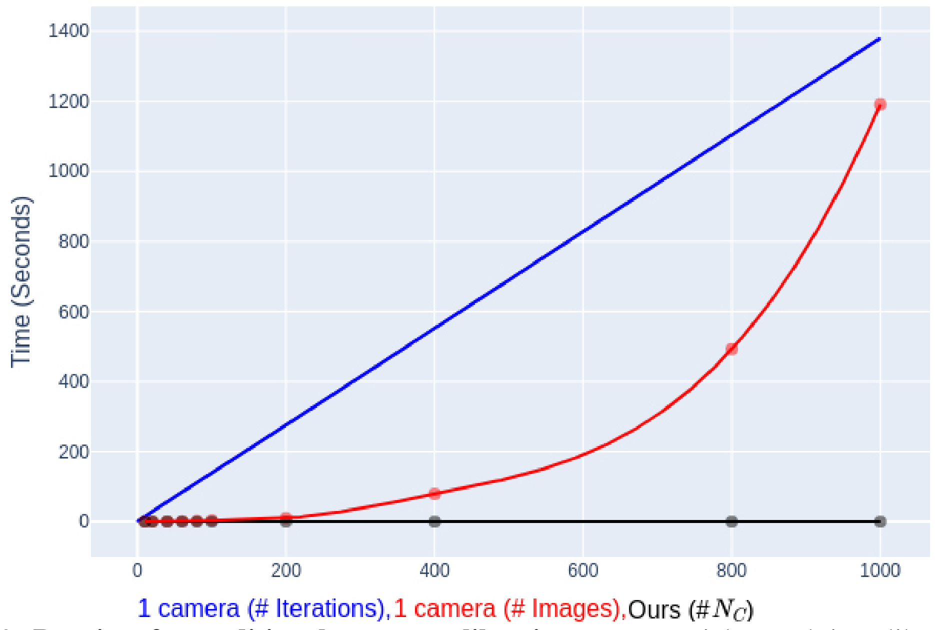


Figure 4.2: **Runtime for traditional camera calibration.** Exponential growth in calibration time with increasing number of images (red; 1 camera). Linear increase w.r.t. LM iterations on 100 images (blue; 1 camera). Ours; real-time ($\tau_{min}^{i=1} = 0.0026s$, $\tau_{max}^{i=10} = 0.012s$) for increasing number of cameras $2^i, 1 \leq i \leq 10$ (black).

In contrast, our method operates on a fixed number of cameras, N_C, which inherently caps the inference time to $\tau_{min}^{i=1} = 0.0026s$, $\tau_{max}^{i=10} = 0.012s$ for increasing number of cameras $2^i, 1 \leq i \leq 10$ on an Nvidia RTX 4090, as shown in Figure 4.2 by the black curve. This translates to ~ 387 inferences per second, demonstrating a significant advantage in the context of real-time applications.

4.2.2 Inference vs. Minimizing Reprojection Errors

We now pivot our experiments to investigate the feasibility of minimizing reprojection error at every step using Bundle Adjustment (BA) as an alternative to recalibrating at every step, in the context of real-time optimization amidst decalibration.

We conducted 1000 trials, each applying random perturbations within a range of $\kappa \in [-10\%, 10\%]$ to the OEM intrinsic parameters. For every trial, our objective was to minimize the reprojection error with BA, utilizing data from $N_{fid.} = 8$ points and $N_C = 6$ cameras.

Our findings highlight a significant constraint: the time taken for a single BA iteration, averaging 0.0385 seconds, substantially exceeds our execution time of 0.0026 seconds for a single inference, and proves that optimizing camera parameters by minimizing reprojection errors at every step is not a viable choice for real-time applications.

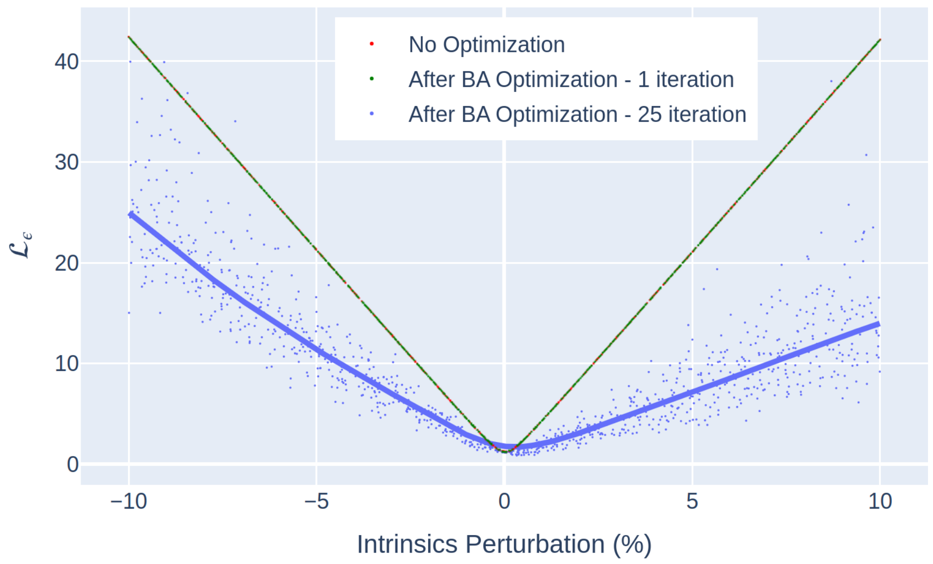


Figure 4.3: **Effect of decalibration on reprojection error (RMSE).** OEM intrinsic parameters are perturbed by 20%, i.e., $\kappa \in [-10\%, 10\%]$, simulating potential decalibration. Reprojection error without intervention (red), with 1 iteration of Bundle Adjustment (green), with 25 iterations (blue); 1000 trials.

Furthermore, a single iteration of BA resulted in minimal improvement in reprojection error, as illustrated in Figure 4.3 in green vs red. Subsequent experiments showed that achieving a noticeable reduction in reprojection error required an average of 25 BA iterations per system's capture, culminating in an untenable average execution time of 1.2892 seconds for real-time processing. Despite these efforts, the reprojection error remained significantly high (Figure 4.3; blue), compared to our method (Table 1 in the main paper; rows 9, 10).

4.3 Qualitative Results

In Figure 4.1, we show a qualitative result of the calibration from the multi-camera system calibration. The blue-shaded spheres represent the ground truth values. The red-shaded spheres represent the predicted values. The single yellow sphere per multi-camera system represents the first camera, and is singled out for visualization purposes only, to demonstrate the in place rotation R_{random} i.e. that the multi-camera system does not always have the same vertical orientation. For closer inspection please refer to the interactive visualization in the supplementary material.

Figure 4.5 illustrates the reprojection accuracy of 3D fiducials using predicted camera poses (white dots) contrasted against the ground-truth projections (color-coded). In this example, a point-based model was trained with 5% perturbation, and the input points to the model were generated with *no perturbation* to the camera parameters. The multi-camera system is arranged in an O-shape, consisting of 10 cameras, with a cube with 8 fiducials as the calibration object.

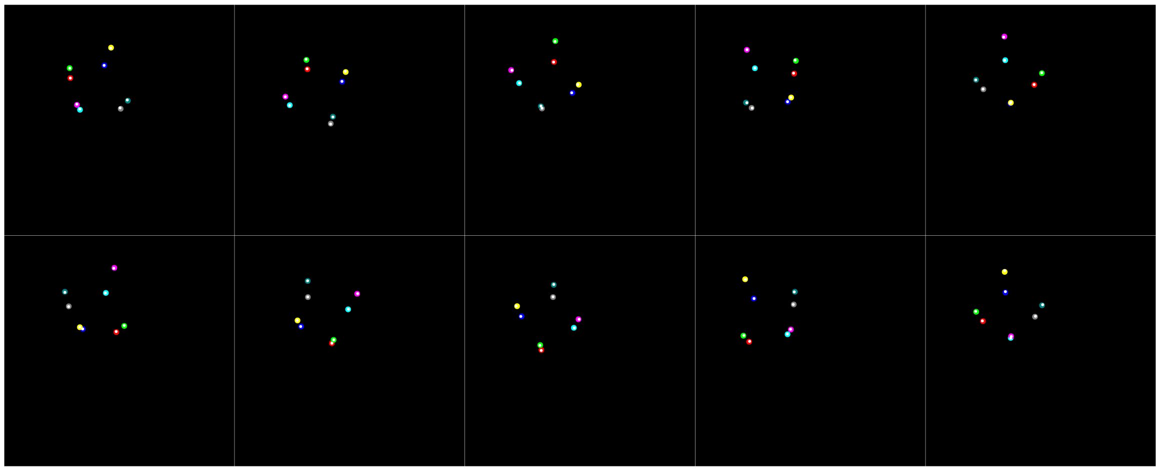


Figure 4.4: **Reprojection of 3D Fiducials with Predicted Camera Poses (With 5% Perturbation).** This figure illustrates the resilience and accuracy of our pose estimation model during decalibration of the camera parameters of up to 5% perturbation. The calibration cube and the O-shaped arrangement of 10 cameras remain constant as in Figure 4.5, allowing for a direct comparison across different testing conditions. Note: The reprojected points are shown in white. For optimal visibility, please zoom in.

Figure 4.4 shows the same model as Figure 4.5, i.e., trained with a 5% perturbation. However, in this example we simulate a decalibration by *generating the test data from perturbed camera parameters of up to 5%*. Beyond the quantitative evaluation presented in the previous section, these figures facilitate a direct qualitative comparison of model performance and clearly demonstrate the robustness of our pose estimation technique under conditions of uncertainty.

4.4 Training: Less Effective Strategies

In this section, we describe our exploration of various loss and regularization terms intended to improve model performance. Despite their theoretical potential, these methods did not enhance our results in practice.

4.4.1 Regularization terms

- A regularization term, denoted as $\mathcal{L}_{\kappa_c}$, was introduced to encourage minimal distortion coefficients (k_1, k_2, k_3, k_4, k_5) by minimizing the sum of the absolute values of these coefficients.
- A regularization term, $\mathcal{L}_{fc}$, was designed to ensure consistency in the focal lengths f_x and f_y by minimizing the squared difference between these focal lengths.

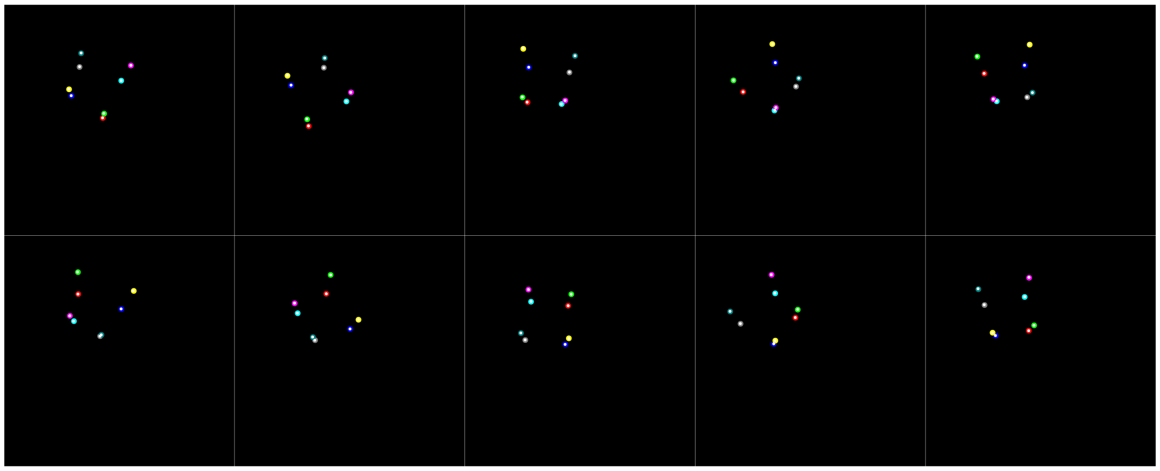


Figure 4.5: **Reprojection of 3D Fiducials with Predicted Camera Poses (Test Data Generated Without Perturbation).** This figure demonstrates the reprojection accuracy in a O-shaped multi-camera setup comprising 10 cameras, with the calibration object being a cube with 8 fiducials. The comparison between the predicted (in white) and ground-truth (color-coded; enlarged for visualization purposes) projections demonstrates the precision of our model in the absence of perturbation. Note: The reprojected points are shown in white. For optimal visibility, please zoom in.

- The regularization term $\mathcal{L}_{pp}$ aimed to align the principal points (c_x, c_y) with the image center by minimizing the squared Euclidean distance between the principal points and the center of the image.
- A regularization term, $\mathcal{L}_{largefc}$, is formulated to discourage small focal lengths by imposing a penalty on the inverse of the sum of the focal lengths along both the x and y axes.
- Applying an L1 norm regularization to our model's parameters aimed to encourage sparsity and reduce overfitting by penalizing large weights, thus simplifying the model. This technique, intended to improve generalization, unfortunately did not yield the expected performance enhancements in our camera parameter prediction task.

4.4.2 Losses

We investigated the log-cosh loss function, attracted by its theoretical benefits such as smooth gradients, outlier robustness, and balanced error sensitivity. These characteristics suggested that log-cosh could enhance prediction precision and stability. The log-cosh loss is given by,

$$\mathcal{L}_{logcosh} = \frac{1}{N}\sum_{i=1}^{N}((y_i - \hat{y}_i)\tanh(y_i - \hat{y}_i) - \log(2) + \log\left(1 + e^{-2|y_i - \hat{y}_i|}\right))$$

where y_i represents the target values, and $\hat{y}_i$ represents the predicted values. However, contrary to our expectations, empirical testing revealed that log-cosh loss did not outperform our existing loss function described in Section 4.3 in the main paper.

Figure 4.6 illustrates a key observation: minimizing our proposed final loss also reduces the auxiliary losses we initially investigated-except from the L1 norm which relates to the network weights rather than the calibration error, and whose purpose is to enforce sparsity. However, this effect is not reciprocal; directly incorporating these auxiliary terms into our model did not further enhance performance beyond the improvements achieved with the final loss alone described earlier. This indicates that while our final loss effectively captures the essence of the auxiliary losses, adding them explicitly does not provide additional benefits. In the figure, we present the training loss

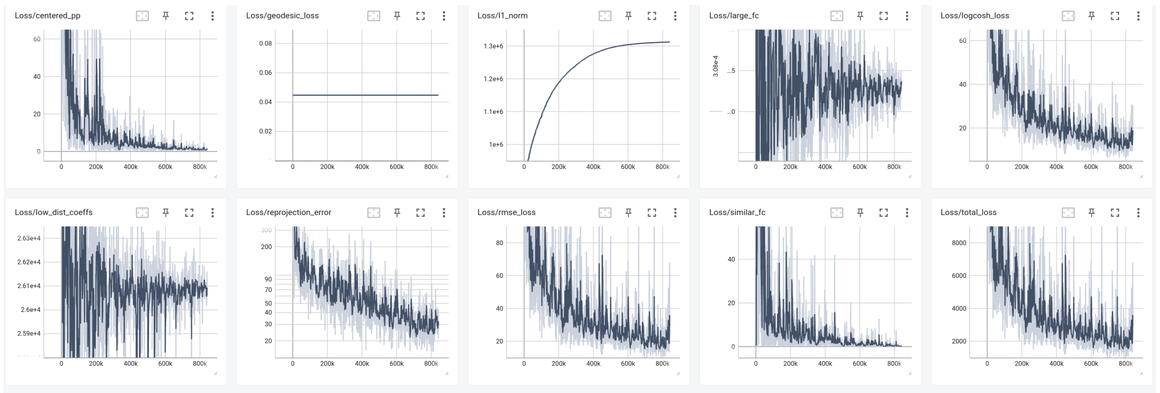


Figure 4.6: **Illustration of the proposed loss compared to auxiliary regularization and loss terms.** This figure demonstrates that minimizing the proposed loss leads to a reduction in auxiliary losses. However, this is a non-reciprocal relationship where the inclusion of auxiliary losses does not further improve model performance beyond the capabilities of the proposed loss which is a combination of the reprojection error, the RMSE error of the parameters, and the geodesic error. Here, we present the training loss graphs for the point-based variant. As previously described, the distortion coefficients are scaled by scaling factor $\lambda_{scale} = 1000$.

patterns for the point-based variant. Comparable trends are observed with the image-based variant.

4.5 Limitations & Discussion

4.5.1 Adhering to Operational Specifications

Our camera pose synthesis methodology is specifically designed to meet the manufacturer's usage requirements, prioritizing high-accuracy predictions in real-world applications, particularly in time-critical contexts such as surgical applications. An important component of our methodology involves simulating a field of view that ensures comprehensive coverage of the area of interest, closely mimicking the actual setup. This is accomplished by ensuring a constant radius for the hemisphere, in accordance with the operational guidelines of the multi-camera system. Unlike some multi-camera systems that are statically affixed with no or limited range of motion, our methodology introduces a significant enhancement by allowing an extended range of motion—albeit at a fixed distance—over the entire hemisphere. The decision to adhere to the specified fixed radius is a deliberate design choice aimed at ensuring that our synthesized camera poses accurately reflect the real-world configuration.

4.5.2 Limited Range of Motion for Image-based

Our assessment of the image-based variant shows promising results in multi-camera setups having a limited range of motion. However, to achieve broader generalization across diverse vantage points -especially at glazing angles-, a more advanced architecture is imperative. This is based on the observation that accuracy drops when using the full range of motion on the hemisphere, attributed to the complexity of input images. Specifically, the challenge arises as only a fraction, $N_{fid.}$ pixels, are relevant to the task amidst the backdrop of the remaining $H \times W - N_{fid.}$ pixels, which significantly narrows the dataset's utility for network learning.

4.5.3 Model Customization

Our methodology demonstrates a tailored approach, achieving high accuracy in predicting camera parameters when applied to the particular multi-camera system, calibration object, and operating distance utilized during its training phase. This high level of accuracy reflects how well the method is customized to fit its specific training conditions, including the perturbations, up to the extent the model has been trained to withstand. However, it is important to note that the model's performance is finely tuned to this particular setup. Any deviation from the original camera setup, use of a different calibration object, or change in the desired maximum perturbation level requires the training of a new model tailored to those new conditions.

4.5.4 Decalibration Detection

Our method also extends beyond real-time calibration to effectively identify decalibration. By continuously comparing the predicted calibration parameters in real-time with those of the OEM, one can swiftly identify any deviations indicative of decalibration. This dual functionality positions our method as both a calibration tool and an operational integrity monitor, ensuring continuous accuracy and reliability of multi-camera systems through early detection and prompt recalibration response.

4.6 Model Training Progression Animation

Included in the supplementary materials is a fast-forward animation, which illustrates the evolution of our network's learning process throughout the training phase. This animation presents key stages of the network's training progression at intervals of every 20 epochs, for a total of 50K epochs.

Chapter 5

Ablation Studies

Abstract

This chapter presents the experimentation, which was not documented in previous chapters. Primarily, it outlines the outcomes of experiments conducted utilizing various proportions of each loss in the compound loss function. Subsequently, it presents findings from experiments employing other encoder networks and comparing their performance with the Transformer encoder. In addition to the Convolutional encoder that was evaluated in Chapter 4, this section employs Graph Neural network and LSTM models as encoders and compares their performance against the Transformer encoder. Finally, the effectiveness of Bundle Adjustment optimization is further evaluated in the presence of simultaneous extrinsic and intrinsic perturbations. This experiment confirms the findings of Section 4.2.2 and extends the results to a more general case involving full parameter perturbation.

5.1 Impact of Loss Term Weights

In this section, the impact of different weights (λ_1 and λ_2) assigned to various components of the loss function on the performance of the camera calibration regression model is explored and reported in Table 5.1. The overall loss function, expressed as $\mathcal{L} = \lambda_1 \times \mathcal{L}_{diff} + \lambda_1 \times \mathcal{L}_{geo} + \lambda_2 \times \mathcal{L}_{\epsilon}$ in Section 3, was altered in the following ways:

- $\lambda_1 \times \mathcal{L}_{geo}$ and $\lambda_2 \times \mathcal{L}_{\epsilon}$ were excluded from the loss function, and $\mathcal{L} = \lambda_1 \times \mathcal{L}_{diff}$ with

Table 5.1: **Experimental results for various loss strategies.** $\mathbf{RE}_{avg}^{20K}$ is the average RMSE reprojection error on 3 different trials (synthetic test sets, each comprising 20,000 data samples). Training on all models includes a **Max Perturb.** $\kappa \in$ [**-2.5**$\%$, **+2.5**$\%$] of the intrinsic and extrinsic parameters. $\boldsymbol{N_{fid.}}$ and $\boldsymbol{N_C}$ are the number of 3D fiducials and the number of cameras, respectively. The intrinsic rotation angle α remains the same in all experiments i.e., $\alpha \sim \mathcal{U}(0, 2\pi)$. Also, $\boldsymbol{\lambda_1}$, $\boldsymbol{\lambda_2}$ determine the contribution of loss components to the total loss after the first 100 epochs.

Row #	**Variant**	$\boldsymbol{\theta}$	$\boldsymbol{\phi}$	$\boldsymbol{N_{fid.}}$ (object)	$\boldsymbol{N_C}$ (config.)	$\boldsymbol{\lambda_1}$, $\boldsymbol{\lambda_2}$ (epoch $>$ 100)	$\mathbf{RE}_{avg}^{20K}$ (pixels)
1	Pt	$\sim \mathcal{U}(0, 2\pi)$	$\sim \mathcal{U}(0, \pi/2)$	8 ▣	10 ○	100, 0.01	9.93$\pm$ 1.4×10^{-4}
2	Pt	$\sim \mathcal{U}(0, 2\pi)$	$\sim \mathcal{U}(0, \pi/2)$	8 ▣	10 ○	100, 0.1	13.98$\pm$ 1.1×10^{-4}
3	Pt	$\sim \mathcal{U}(0, 2\pi)$	$\sim \mathcal{U}(0, \pi/2)$	8 ▣	10 ○	0, 1	**Inf loss**
4	Pt	$\sim \mathcal{U}(0, 2\pi)$	$\sim \mathcal{U}(0, \pi/2)$	8 ▣	10 ○	1, 0	13.35$\pm$ 1.5×10^{-4}

$\lambda_1 = 100$ was crafted as the loss for the whole training process. Excluding the $\mathcal{L}_\epsilon$ from the total loss has led the model to have worse performance with a $\mathbf{RE}_{avg}^{20K}$ average RMSE reprojection error of 13.35 (Table 5.1; row 4). The reason might stem from the fact that only minimizing $\mathcal{L}_{diff}$ will try to regress parameters close to the ground truth. However, the nature of the task cannot be fully represented by this loss, since small changes in parameters such as the rotation angles could lead to large reprojection errors, which is not fully reflected in $\mathcal{L}_{diff}$.

- After the first 100 epochs, the loss function was adjusted to include only the reprojection error, defined as $\mathcal{L} = \lambda_2 \times \mathcal{L}_\epsilon$ with $\lambda_2 = 1$. While this strategy was expected to significantly reduce the reprojection error, it led to $\mathcal{L}_\epsilon$ increasing infinitely and the model becoming unstable, as the loss became so large that NaN values were encountered, despite clipping the gradients.

- Finally, the value of λ_2 which was 0.01 in all the experiments presented in Section 3 and Section 4, was changed to 0.1. This strategy led to $\mathbf{RE}_{avg}^{20K}$ average RMSE reprojection error of 13.98 (Table 5.1; row 2). Although this strategy has resulted in a comparative result to the main approach, it shows that $\lambda_2 = 0.01$ is a better choice.

5.2 Graph Neural Network Encoder

In the quest to enhance camera calibration through innovative machine learning approaches, a Graph Neural Network (GNN) encoder was tested within the regression pipeline. The GNN encoder employs a two-layer Graph Convolutional Network (GCN) [42] architecture, and input features are processed through successive GCN layers, each followed by ReLU activations, to learn meaningful representations from graph-structured data. The features extracted by this GNN encoder are then fed into multi-layer perceptron (MLP) heads to predict camera calibration parameters.

Despite the theoretical strengths of GNNs in capturing spatial relationships and their potential in handling non-Euclidean data structures, the GNN encoder underperformed compared to the transformer-based model, with the $\mathbf{RE}_{avg}^{20K}$ average RMSE reprojection error of 249.93 (Table 5.2; row 2). This discrepancy in performance can be attributed to several factors. Firstly, GNNs might struggle with the specific characteristics of the camera calibration task, where the ability to model long-range dependencies and global context, as Transformers do, is crucial. Additionally, GNNs are typically more sensitive to the quality of the graph structure and edge information, which might not be as effectively leveraged in this application compared to the self-attention mechanism of Transformers. The comparative analysis underscores the importance of selecting model architectures that align well with the nature of the task, illustrating why the Transformer-based approach yielded superior results in this scenario.

5.3 LSTM Encoder

Given that the input to the camera calibration task consists of point coordinates, utilizing an LSTM [43] encoder was considered a promising approach. The sequential nature of object points can be effectively captured by LSTMs, which are designed to handle temporal dependencies and sequential data. This capability makes LSTMs a potentially suitable choice for modeling the ordered structure of point coordinates, allowing the network to learn the inherent relationships between successive points.

The RNNEncoder was constructed using PyTorch, implementing an LSTM with two layers to process the sequential input data. The LSTM layers aim to capture the temporal dependencies in

Table 5.2: **Experiment Results with Various Encoders**. Training on all models includes a perturbation of **Max Perturb.** $\kappa \in [\textbf{-2.5}\%, \textbf{+2.5}\%]$ of the OEM intrinsic and extrinsic parameters, utilizing 10 cameras ($N_C = 10$) and a calibration object with 8 fiducials ($N_{fid.} = 8$).

Row #	**Variant**	**# Params**	θ	ϕ	**Encoder**	**Scaled** $\lambda_{scale} = 1000$	**Camera Identity Encoding** CIE	**RE**$^{20K}_{avg}$ (pixels)
1	Pt	~ 33m	$\sim \mathcal{U}(0, 2\pi)$	$\sim \mathcal{U}(0, \pi/2)$	Transformer	✓	✓	$9.93 \pm 1.4 \times 10^{-4}$
2	Pt	~ 13m	$\sim \mathcal{U}(0, 2\pi)$	$\sim \mathcal{U}(0, \pi/2)$	GNN	✓	✓	$249.93 \pm 1.7 \times 10^{-4}$
3	Pt	~ 9m	$\sim \mathcal{U}(0, 2\pi)$	$\sim \mathcal{U}(0, \pi/2)$	LSTM	✓	✓	$14.11 \pm 1 \times 10^{-4}$

the point coordinates, transforming these into meaningful feature representations. These features are then used by multi-layer perceptron (MLP) heads to predict camera calibration parameters.

Despite the theoretical suitability of LSTMs for this task, the LSTM encoder did not perform as well as the transformer-based model, resulting in a higher reprojection error of 14.11 compared to 9.93 (Table 5.2; row 3) with the transformer encoder. This underperformance could be attributed to several factors. LSTMs, while effective in capturing local sequential dependencies, may struggle with the long-range dependencies and global context that are crucial for accurate camera calibration. Transformers, with their self-attention mechanisms, excel at capturing such long-range dependencies, leading to more robust feature representations. Additionally, LSTMs are more prone to issues like vanishing gradients, which can hinder the learning process for deeper dependencies. This comparison underscores the necessity of selecting model architectures that are well-aligned with the task's requirements, highlighting the superior performance of the Transformer-based approach in this scenario.

5.4 Inference vs. Minimizing Reprojection Errors with Presence of Extrinsic and Intrinsic Parameters Perturbation

As part of this thesis, Bundle Adjustment (BA) was employed to optimize the calibration of a multi-camera setup. This approach was chosen to evaluate its capability for online calibration and to compare its performance with a deep-learning model. This method optimizes camera parameters and the 3D scene structure by minimizing the discrepancy between observed image points and their projections. Implementing this approach requires capturing multiple images of a planar calibration object, such as a checkerboard or grid, from varied viewpoints using all the cameras. Corresponding 2D points extracted from the images are paired with known 3D points, which are then used to estimate the intrinsics and extrinsics of all the cameras. To achieve accurate calibration results, this method utilizes a nonlinear optimization algorithm that refines the initial parameter estimates, compensating for any inaccuracies introduced during the initial parameter estimation process. Leveraging iterative optimization techniques, BA is crucial for refining camera calibration and enhancing accuracy in applications such as augmented reality and 3D mapping. In this section, as an extension to the experiment done in 4.2.2, after perturbing all the camera parameters, the effectiveness of BA to refine the camera parameters is evaluated. Figure 5.1 shows how intrinsic and extrinsic parameter perturbation degrades the reprojection error and highlights the limited effectiveness of performing BA with 1 and 25 iterations in improving the reprojection error. Therefore, this experiment confirms the findings of 4.2.2 and underlines the fact that optimizing camera parameters by minimizing reprojection errors at every step is not a viable choice for real-time applications.

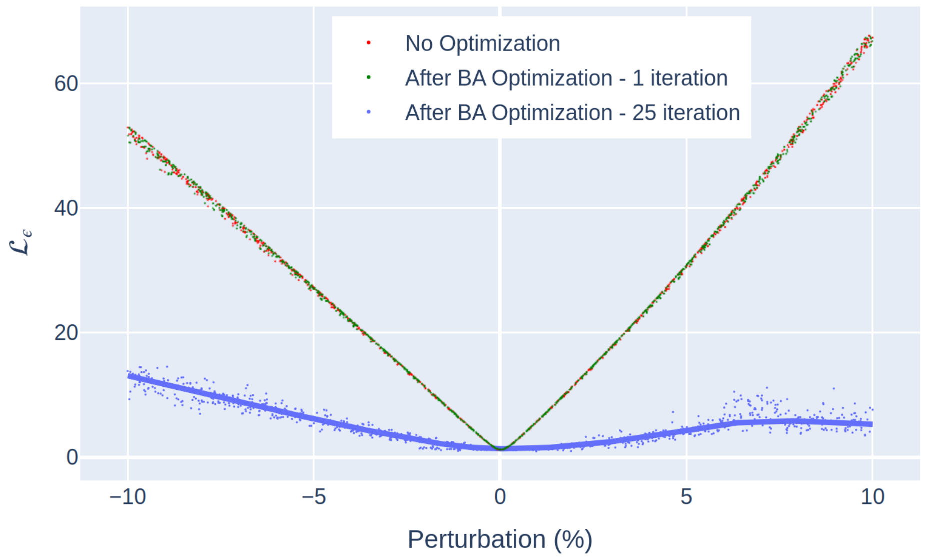


Figure 5.1: **Effect of all parameters' decalibration on reprojection error (RMSE).** OEM intrinsic and extrinsic parameters are perturbed by 20%, i.e., $\kappa \in [-10\%, 10\%]$, simulating potential decalibration. Reprojection error without intervention (red), with 1 iteration of Bundle Adjustment (green), with 25 iterations (blue); 1000 trials.

Chapter 6

Conclusion

In this thesis, critical challenges in the calibration and real-time optimization of vision-based computer-assisted surgical systems with multi-camera setups have been addressed. Traditional calibration methods, although effective, are hindered by the exponential increase in execution time as the number of images grows, making them impractical for real-time applications. By leveraging an Nvidia RTX 4090, a novel approach has been developed that maintains a consistent inference time, achieving approximately 387 inferences per second, representing a significant improvement for real-time scenarios.

Experiments have demonstrated that minimizing reprojection errors through Bundle Adjustment is not feasible for real-time applications due to the substantial time required for each iteration. In contrast, the proposed point-based model has shown robustness and accuracy, maintaining high performance even with up to 20% perturbation in camera parameters. Various regularization terms and loss functions were explored, revealing that while theoretical enhancements did not yield practical improvements, the proposed loss function effectively reduced calibration errors.

The method has been designed to adhere to operational specifications crucial for surgical applications, ensuring high accuracy and real-time performance. The customization of the methodology to specific setups has highlighted its high accuracy, though retraining is necessary for different configurations. The regression accuracy achieved by the point-based model further underscores its effectiveness in real-world applications.

Additionally, the approach effectively detects decalibration, ensuring continuous accuracy and reliability of multi-camera systems. This dual functionality enhances the operational integrity of these systems.

In summary, a robust, efficient, and scalable solution for real-time calibration of multi-camera systems has been presented in this thesis. The significant reduction in execution time, coupled with high accuracy and resilience under perturbation, makes the proposed method a valuable tool for advancing computer-assisted surgical systems and other real-time applications requiring precise multi-camera calibration.

Bibliography

[1] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need in advances in neural information processing systems, 2017," *Search PubMed*, pp. 5998–6008, 2017.

[2] Y. Zhou, C. Barnes, J. Lu, J. Yang, and H. Li, "On the continuity of rotation representations in neural networks," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

[3] Z. Zhang, "A flexible new technique for camera calibration," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 22, no. 11, pp. 1330–1334, 2000.

[4] R. Hartley and A. Zisserman, *Multiple view geometry in computer vision*. Cambridge University Press, 2004.

[5] R. T. Azuma, "A survey of augmented reality," *Presence: teleoperators & virtual environments*, vol. 6, no. 4, pp. 355–385, 1997.

[6] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? the kitti vision benchmark suite," in *2012 IEEE conference on computer vision and pattern recognition*, pp. 3354–3361, IEEE, 2012.

[7] D. Scaramuzza, A. Martinelli, and R. Siegwart, "A toolbox for easily calibrating omnidirectional cameras," in *2006 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 5695–5701, IEEE, 2006.

[8] Y. Cai, L. Ge, J. Cai, and J. Yuan, "Weakly-supervised 3d hand pose estimation from monocular rgb images," in *Proceedings of the European conference on computer vision (ECCV)*, pp. 666–682, 2018.

[9] M. D. Zeiler and R. Fergus, "Visualizing and understanding convolutional networks," in *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part I 13*, pp. 818–833, Springer, 2014.

[10] F. Devernay and O. Faugeras, "Straight lines have to be straight," *Machine vision and applications*, vol. 13, pp. 14–24, 2001.

[11] F. Shu, W. Neddermeyer, and J. Zhang, "Online approaches to camera pose recalibration," in *2006 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 61–66, IEEE, 2006.

[12] C. B. Duane, "Close-range camera calibration," *Photogramm. Eng*, vol. 37, no. 8, pp. 855–866, 1971.

[13] Y. Abdel-AzizandH and M. Karara, "Direct linear transformation into object space coordinates in close-range photogrammetry," in *Proceedings of the Symposium Close-Range Photogrammetry*, pp. 1–18, 2015.

[14] R. Tsai, "A versatile camera calibration technique for high-accuracy 3d machine vision metrology using off-the-shelf tv cameras and lenses," *IEEE Journal on Robotics and Automation*, vol. 3, no. 4, pp. 323–344, 1987.

[15] T. Svoboda, D. Martinec, and T. Pajdla, "A convenient multicamera self-calibration for virtual environments," *Presence: Teleoperators & virtual environments*, vol. 14, no. 4, pp. 407–422, 2005.

[16] T. Svoboda, H. Hug, and L. Van Gool, "Viroom—low cost synchronized multicamera system and its self-calibration," in *Pattern Recognition: 24th DAGM Symposium Zurich, Switzerland, September 16–18, 2002 Proceedings 24*, pp. 515–522, Springer, 2002.

[17] T. Svoboda, “Quick guide to multi-camera self-calibration,” *ETH, Swiss Federal Institute of Technology, Zurich, Tech. Rep. BiWi-TR-263, http://www. vision. ee. ethz. ch/svoboda/SelfCal*, 2003.

[18] J. Heikkila and O. Silvén, “A four-step camera calibration procedure with implicit image correction,” in *Proceedings of IEEE computer society conference on computer vision and pattern recognition*, pp. 1106–1112, IEEE, 1997.

[19] J. Lee, H. Go, H. Lee, S. Cho, M. Sung, and J. Kim, “Ctrl-c: Camera calibration transformer with line-classification,” in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 16228–16237, 2021.

[20] A. Ghofrani, R. M. Toroghi, and S. M. Tabatabaie, “Catiloc: Camera image transformer for indoor localization,” in *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 1450–1454, IEEE, 2021.

[21] Y.-H. Wu, Y. Liu, X. Zhan, and M.-M. Cheng, “P2t: Pyramid pooling transformer for scene understanding,” *IEEE transactions on pattern analysis and machine intelligence*, 2022.

[22] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, “Bert: Pre-training of deep bidirectional transformers for language understanding,” *arXiv preprint arXiv:1810.04805*, 2018.

[23] S. Workman, C. Greenwell, M. Zhai, R. Baltenberger, and N. Jacobs, “Deepfocal: A method for direct focal length estimation,” in *2015 IEEE International Conference on Image Processing (ICIP)*, pp. 1369–1373, IEEE, 2015.

[24] A. Cramariuc, A. Petrov, R. Suri, M. Mittal, R. Siegwart, and C. Cadena, “Learning camera miscalibration detection,” in *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 4997–5003, IEEE, 2020.

[25] A. Kendall, M. Grimes, and R. Cipolla, “Posenet: A convolutional network for real-time 6-dof camera relocalization,” in *Proceedings of the IEEE international conference on computer vision*, pp. 2938–2946, 2015.

[26] Y. Hold-Geoffroy, K. Sunkavalli, J. Eisenmann, M. Fisher, E. Gambaretto, S. Hadap, and J.-F. Lalonde, “A perceptual measure for deep single image camera calibration,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2354–2363, 2018.

[27] T. H. Butt and M. Taj, “Camera calibration through camera projection loss,” in *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 2649–2653, IEEE, 2022.

[28] S. Shah and J. Aggarwal, “A simple calibration procedure for fish-eye (high distortion) lens camera,” in *Proceedings of the 1994 IEEE international Conference on Robotics and Automation*, pp. 3422–3427, IEEE, 1994.

[29] G. Nakano, “A versatile approach for solving pnp, pnpf, and pnpfr problems,” in *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part III 14*, pp. 338–352, Springer, 2016.

[30] S. Urban, J. Leitloff, and S. Hinz, “Mlpnp-a real-time maximum likelihood solution to the perspective-n-point problem,” *arXiv preprint arXiv:1607.08112*, 2016.

[31] S. Haner and K. Astrom, “Absolute pose for cameras under flat refractive interfaces,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1428–1436, 2015.

[32] M. W. Cao, W. Jia, Y. Zhao, S. J. Li, and X. P. Liu, “Fast and robust absolute camera pose estimation with known focal length,” *Neural Computing and Applications*, vol. 29, pp. 1383–1398, 2018.

[33] A. D. Nguyen and M. Yoo, “Calibbd: Extrinsic calibration of the lidar and camera using a bidirectional neural network,” *IEEE Access*, vol. 10, pp. 121261–121271, 2022.

[34] R. Itu, D. Borza, and R. Danescu, “Automatic extrinsic camera parameters calibration using convolutional neural networks,” in *2017 13th IEEE International Conference on Intelligent Computer Communication and Processing (ICCP)*, pp. 273–278, IEEE, 2017.

[35] A. Hagemann, M. Knorr, and C. Stiller, "Deep geometry-aware camera self-calibration from video," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 3438–3448, 2023.

[36] O. Bogdan, V. Eckstein, F. Rameau, and J.-C. Bazin, "Deepcalib: A deep learning approach for automatic intrinsic calibration of wide field-of-view cameras," in *Proceedings of the 15th ACM SIGGRAPH European Conference on Visual Media Production*, pp. 1–10, 2018.

[37] O. D. Faugeras, Q. T. Luong, and S. J. Maybank, "Camera self-calibration: Theory and experiments," in *Computer Vision—ECCV'92: Second European Conference on Computer Vision Santa Margherita Ligure, Italy, May 19–22, 1992 Proceedings 2*, pp. 321–334, Springer, 1992.

[38] C. S. Fraser, "Digital camera self-calibration," *ISPRS Journal of Photogrammetry and Remote sensing*, vol. 52, no. 4, pp. 149–159, 1997.

[39] R. I. Hartley, "Self-calibration from multiple views with a rotating camera," in *Computer Vision—ECCV'94: Third European Conference on Computer Vision Stockholm, Sweden, May 2–6, 1994 Proceedings, Volume I 3*, pp. 471–478, Springer, 1994.

[40] M. Lopez, R. Mari, P. Gargallo, Y. Kuang, J. Gonzalez-Jimenez, and G. Haro, "Deep single image camera calibration with radial distortion," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 11817–11825, 2019.

[41] N. Wakai and T. Yamashita, "Deep single fisheye image camera calibration for over 180-degree projection of field of view," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 1174–1183, 2021.

[42] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," *arXiv preprint arXiv:1609.02907*, 2016.

[43] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.