DEPARTMENT OF INFORMATICS

TECHNISCHE UNIVERSITÄT MÜNCHEN

Bachelor's Thesis in Informatics

Topic: Enhancing Intraoperative Registration with Neural Radiance Fields: An Exploration of Loss Functions Effects

Davyd Podolskyi

DEPARTMENT OF INFORMATICS

TECHNISCHE UNIVERSITÄT MÜNCHEN

Bachelor's Thesis in Informatics

Topic: Enhancing Intraoperative Registration with Neural Radiance Fields: An Exploration of Loss Functions Effects

Verbesserung der intraoperativen Registrierung mit Neural Radiance Fields: Eine Untersuchung der Auswirkungen von Verlustfunktionen

Author:	Davyd Podolskyi
Supervisor, TUM:	Prof. Dr. Nassir Navab
Advisor, TUM:	Mohammad Farid Azampour, M.Sc.
Advisor, Harvard Medical School:	Nazim Haouchine, PhD.
Submission Date:	18.03.2025

I confirm that this bachelor's thesis is my own work and I have documented all sources and material used.

Ich versichere, dass ich diese Bachelor's Thesis selbständig verfasst und nur die angegebenen Quellen und Hilfsmittel verwendet habe.

Munich, 18.03.2025 Davyd Podolskyi

Acknowledgements

I would like to express my sincerest gratitude to everyone who supported me throughout the journey of completing this research and thesis.

First and foremost, my deepest appreciation goes to Prof. Dr. Nassir Navab for providing me with the opportunity to explore this exciting research topic at the Computer Aided Medical Procedures (CAMP) chair.

I sincerely thank my supervisor, Mohammad Farid Azampour, whose guidance, expertise, and vision in the field of medical image computing have been invaluable to the successful completion of this work.

I extend my heartfelt gratitude to my supervisor at Harvard Medical School and the Department of Radiology at Brigham & Women's Hospital, Nazim Haouchine, PhD. His expert mentorship, insightful advice, and intellectual contributions have significantly enriched this research.

Lastly, I wish to acknowledge Maximilian Fehrentz, whose innovative work on intraoperative registration through cross-modal inverse neural rendering formed an essential foundation for my research.

Abstract

This thesis explores an innovative approach to intraoperative brain registration by utilizing Neural Radiance Fields (NeRFs) as differentiable, implicit representations of brain surface geometry and appearance. Unlike conventional mesh-based techniques, NeRFs allow for direct optimization of camera positions via backpropagation, significantly enhancing alignment accuracy between preoperative and intraoperative imaging. We introduce a robust, model-agnostic implementation of neural registration within the nerfstudio framework, overcoming previous limitations regarding customization and adaptability. The primary scientific contribution involves a comprehensive analysis of multiple loss functions—including L1, L2, Structural Similarity Index (SSIM), Normalized Cross-Correlation (NCC), and Mutual Information (MI)—to assess their impact on registration accuracy, convergence rate, and stability. Experimental results demonstrate that while L1 loss offers rapid and stable convergence, MI and NCC are notably resilient to NeRF-generated visual artifacts, providing insights crucial for clinical applications. By advancing NeRF-based registration techniques, this work contributes directly toward improving the precision and reliability of image-guided neurosurgical procedures.

Kurzfassung

Diese Bachelorarbeit untersucht einen innovativen Ansatz zur intraoperativen Hirnregistrierung durch die Verwendung von Neural Radiance Fields (NeRFs) als differenzierbare, implizite Darstellungen der Hirnoberflächen-Geometrie und -Erscheinung. Im Gegensatz zu konventionellen mesh-basierten Techniken ermöglichen NeRFs eine direkte Optimierung der Kamerapositionen mittels Backpropagation, was die Genauigkeit der Ausrichtung zwischen präoperativer und intraoperativer Bildgebung erheblich verbessert. Wir stellen eine robuste, modellunabhängige Implementierung der neuralen Registrierung innerhalb des Nerfstudio-Frameworks vor, die bisherige Einschränkungen hinsichtlich Anpassungsfähigkeit und Flexibilität überwindet. Der primäre wissenschaftliche Beitrag umfasst eine umfassende Analyse verschiedener Verlustfunktionen—einschließlich L1, L2, Structural Similarity Index (SSIM), Normalized Cross-Correlation (NCC) und Mutual Information (MI)—um deren Einfluss auf die Registrierungsgenauigkeit, Konvergenzrate und Stabilität zu bewerten. Experimentelle Ergebnisse zeigen, dass während der L1-Verlust eine schnelle und stabile Konvergenz bietet, MI und NCC bemerkenswert widerstandsfähig gegenüber von NeRF erzeugten visuellen Artefakten sind, was Erkenntnisse liefert, die für klinische Anwendungen entscheidend sind. Durch die Weiterentwicklung NeRF-basierter Registrierungstechniken trägt diese Arbeit direkt zur Verbesserung der Präzision und Zuverlässigkeit bildgeführter neurochirurgischer Eingriffe bei.

Contents

1. Introduction

Neurosurgery is a high-precision medical field where accuracy directly influences patient outcomes. During brain tumor resections, surgeons rely on image-guided navigation systems to assist with spatial orientation and accurately locate critical anatomical structures [1]. These systems typically use a process called registration, which refers to aligning preoperative Magnetic Resonance Imaging (MRI) data with the patient's anatomy [2].

The field of computer vision has recently seen remarkable advancements in neural scene representation techniques, particularly with the introduction of Neural Radiance Fields (NeRFs). NeRFs represent scenes as continuous functions that map 3D coordinates and viewing directions to color and density values, enabling high-quality 3D depictions of brain anatomical surfaces. These implicit neural representations have revolutionized how we model and render 3D environments, offering differentiable, continuous scene representations that can be optimized through gradient-based methods [3].

Real-world example of an integrated intraoperative system is Advanced Multimodality Image Guided Operating (AMIGO) Suite which is being heavily used in surveries in Brigham and Women's Hospital.

This thesis builds upon recent work that leverages NeRFs for pose estimation [4] and, more specifically, for intraoperative registration [5].

We propose to enhance NeRF-based intraoperative registration through a comprehensive exploration of alternative loss functions beyond the standard L2 loss.

Such research can be highly relevant for AR/VR-assisted surgeries. [6]

Current golden standards of intraoperative registration techniques face several challenges:

1. **Brain shift**: The brain's position and shape change during surgery due to cerebrospinal fluid drainage, gravity, surgical manipulations, and other physical, surgical, or biological changes, impacting the precision of surgical interventions. [7]

2. **Surgical workflow integration**: Registration methods should integrate seamlessly into existing surgical workflows without requiring additional equipment or extensive time. MRI, while used for preoperative diagnostic purposes, in-surgery require specific O.R. design, significant workflow interruption, and specific non-magnetic equipment. [8]

3. **Speed and accuracy**: Registration must be both precise and fast enough to be clinically viable during surgery. [9]

The approach presented by Fehrentz, Azampour, Dorent, et al. [5] addresses these challenges by using neural rendering for registration. However, their work primarily focuses on a hypernetwork-based approach for appearance adaptation and employs a standard L2

loss for optimization. This thesis extends their work by exploring whether alternative loss functions might yield better registration results and by analyzing how different style transfer techniques affect the registration process.

1.1. Research Questions and Objectives

This thesis aims to address the following primary research question:

- **How do different loss functions — specifically L1, L2, Structural Similarity Index, Normalized Cross-Correlation, and Mutual Information — affect the convergence speed and accuracy of NeRF-based intraoperative registration?**

To comprehensively investigate this question, we address several sub-questions:

1. How do various loss functions differ in their convergence rates and final registration accuracy when aligning preoperative NeRF renderings with intraoperative images?
2. What are the computational efficiency trade-offs between different loss functions in time-sensitive surgical environments?
3. Which loss functions demonstrate greater robustness to registration challenges, including brain shift and visual differences between preoperative MRI-derived renderings and intraoperative images?
4. How does the optimization trajectory differ between intensity-based losses (L1, L2) and structural/information-based losses (SSIM, NCC, MI)?

This research is guided by the hypothesis that more sophisticated loss functions—particularly those capturing structural similarities rather than pixel-wise differences—will demonstrate faster convergence and greater resilience to domain gaps between rendered and intraoperative images.

This study is complemented by our novel implementation contribution: a model-agnostic neural registration framework built on top of **nerfstudio** [10] that enables direct comparison of different loss functions without requiring retraining of the underlying NeRF representations.

1.2. Thesis Structure

The remainder of this thesis is organized as follows:

- **Chapter 2**: Provides the necessary background on neural radiance fields, pose estimation techniques, and current approaches to intraoperative registration. This chapter also reviews relevant literature on loss functions in image registration tasks.

- **Chapter 3**: Explains the methodology implemented in this work, including the neural radiance field architecture, the formulation of different loss functions, and the optimization framework for registration.
- **Chapter 4**: Presents quantitative and qualitative results of our experiments, analyzing the performance of each loss function across multiple metrics and scenarios.
- **Chapter 5**: Discusses the implications of our findings for intraoperative registration, interprets the comparative advantages of each loss function, and considers the practical implications for clinical adoption.
- **Chapter 6**: Summarizes the key contributions and insights of this research, acknowledges limitations, and suggests promising directions for future work in neural rendering-based surgical navigation.

2. Theoretical Background and Related Work

2.1. Neural Networks and Backpropagation

Neural Radiance Field (NeRF) models are fundamentally based on artificial neural networks (ANNs). ANNs are computational models comprising interconnected processing units or "neurons," arranged in layers that transform input data through a series of non-linear functions to produce output predictions[11]. Formally, each neuron computes a weighted sum of its inputs, applies a non-linear activation function $\sigma(\cdot)$, and passes the result to subsequent neurons:

$$y = \sigma\left(\sum_{i=1}^{n} w_i x_i + b\right) \tag{2.1}$$

where w_i represents the weights, x_i the inputs, and b the bias term.

Neural networks optimize their parameters through gradient-based learning algorithms, predominantly backpropagation coupled with stochastic gradient descent (SGD) or its variants. The training process involves forward propagation of input data through the network, yielding predictions that are evaluated against ground truth via a differentiable loss function $\mathcal{L}(\theta)$, where θ represents the network parameters. Backpropagation then computes the gradient $\nabla_\theta \mathcal{L}(\theta)$ by recursively applying the chain rule of differentiation to determine each parameter's contribution to the total error.

Parameter updates proceed iteratively according to:

$$\theta_{t+1} = \theta_t - \eta \nabla_\theta \mathcal{L}(\theta_t) \tag{2.2}$$

where η denotes the learning rate that governs the magnitude of parameter adjustments. This optimization process continues over multiple epochs as the network processes mini-batches of training data, progressively minimizing the objective function and improving the model's predictive performance.

2.2. Intraoperative Registration in Neurosurgery

Intraoperative registration is an essential component of neurosurgical procedures, involving the precise alignment of preoperative imaging data with the patient's actual anatomy during surgery. The primary goal of intraoperative registration is to create an accurate spatial correspondence between the preoperative images — usually Magnetic Resonance Imaging (MRI) — and the physical patient. Fehrentz, Azampour, Dorent, et al. [5] This alignment

ensures surgeons can reliably utilize the rich anatomical information provided by preoperative images, thereby improving surgical accuracy, reducing complications, and enhancing patient safety.

Various registration methods have been developed to achieve this alignment, ranging from traditional approaches, such as point-based, surface-based, and volume-based techniques, to more complex methods like cross-modal registration. These techniques differ significantly in terms of their computational complexity, equipment requirements, accuracy, and adaptability to intraoperative changes such as brain shift and tissue deformation.

2.2.1. Traditional Registration Approaches

Traditional approaches to intraoperative registration in neurosurgery can be broadly categorized into the following methods:

- **Point-based registration**: This approach identifies and matches corresponding anatomical landmarks or artificially placed fiducial markers in both the preoperative images and physical patient. While conceptually simple, it requires accurate identification of landmarks and can be time-consuming. [12]
- **Surface-based registration**: This technique matches surfaces extracted from preoperative imaging with surfaces captured intraoperatively, often using techniques like the Iterative Closest Point (ICP) algorithm. Surface-based approaches typically require specialized equipment, such as laser scanners or stereo cameras, to capture the intraoperative surface. [13]
- **Volume-based registration**: These methods use intensity-based similarity measures to align volumetric images, but they typically require intraoperative imaging modalities such as ultrasound or intraoperative MRI, which may not be available in all surgical settings. [14]

A significant challenge in neurosurgical registration is brain shift, the deformation of brain tissue that occurs during surgery due to factors such as gravity, cerebrospinal fluid drainage, and surgical manipulations. This phenomenon can significantly reduce the accuracy of rigid registration methods and necessitates more advanced techniques.

2.2.2. Cross-Modal Registration

Cross-modal registration refers to the alignment of images from different imaging modalities. In the context of neurosurgery, this often involves aligning preoperative MRI data with intraoperative camera images. This presents unique challenges due to differences in:

- **Information content**: MRI provides volumetric data with tissue contrast, while optical images capture surface appearance with details like blood vessels and lighting effects. [15]

- **Geometric representation**: MRI data is three-dimensional, while camera images are two-dimensional projections. [16]
- **Appearance**: The visual appearance of tissues differs significantly between MRI and optical images due to different physical principles of image formation. [17]

Previous work in cross-modal registration has employed techniques such as feature extraction, mutual information maximization, and deep learning-based approaches to bridge these differences.

2.3. Neural Radiance Fields (NeRFs)

Neural Radiance Fields, introduced by Mildenhall, Srinivasan, Tancik, et al. [3], represent a novel approach to scene representation and novel view synthesis. Unlike traditional computer graphics methods that use explicit representations like meshes or point clouds, NeRFs employ an implicit neural representation to model scenes.

2.3.1. NeRF Representation

A NeRF is typically implemented as a multi-layer perceptron (MLP) that maps a 3D coordinate $\mathbf{x} = (x, y, z)$ and viewing direction $\mathbf{d} = (\theta, \phi)$ to a color $\mathbf{c} = (r, g, b)$ and volume density σ:

$$F_\Theta : (\mathbf{x}, \mathbf{d}) \rightarrow (\mathbf{c}, \sigma) \tag{2.3}$$

where Θ represents the learnable parameters of the neural network. This continuous, differentiable representation allows for rendering from arbitrary viewpoints through volume rendering techniques.

The rendering process involves casting rays from a camera through image pixels and evaluating the NeRF at multiple points along each ray. The color of a pixel is computed as a weighted sum of the colors along the ray, with weights determined by the volume densities:

$$C(\mathbf{r}) = \int_{t_n}^{t_f} T(t)\sigma(t)\mathbf{c}(t)dt \tag{2.4}$$

where $T(t) = \exp\left(-\int_{t_n}^{t} \sigma(s)ds\right)$ represents the accumulated transmittance along the ray up to point t.

2.3.2. NeRF Variants

Since the introduction of the original NeRF, numerous variants have been developed to address limitations and extend capabilities:

- **Instant-NGP** [18]: Accelerates NeRF training and rendering through multi-resolution hash encoding, reducing training time from days to minutes.

- **HyperNeRF** [19]: Extends NeRFs to handle topological variations in dynamic scenes through a higher-dimensional representation.
- **Nerfacto** [10]: An implementation-agnostic framework that combines advances from various NeRF variants for improved performance.

2.3.3. iNeRF: Inverting Neural Radiance Fields for Pose Estimation

Yen-Chen, Florence, Barron, et al. [4] introduced iNeRF, a method that leverages the differentiable nature of NeRFs for pose estimation. Given a target image and a pre-trained NeRF, iNeRF estimates the camera pose from which the target image was captured. This is achieved by optimizing the camera pose parameters to minimize the difference between the rendered image (from the current pose estimate) and the target image.

The key insight of iNeRF is that the camera pose can be optimized through backpropagation, utilizing the differentiable nature of both the NeRF representation and the rendering process. This optimization is formulated as:

$$\hat{\xi} = \arg\min_{\xi} \mathcal{L}(I_{\text{target}}, I_{\text{rendered}}(\xi)) \tag{2.5}$$

where ξ represents the camera pose parameters, I_{target} is the target image, $I_{\text{rendered}}(\xi)$ is the image rendered from the NeRF using pose ξ, and $\mathcal{L}$ is a loss function measuring the dissimilarity between the images.

2.4. Cross-Modal Inverse Neural Rendering for Registration

Building on the concept of iNeRF, Fehrentz, Azampour, Dorent, et al. [5] proposed a method for intraoperative registration using cross-modal inverse neural rendering. Their approach addresses the challenge of cross-modal registration by separating the neural representation into structural and appearance components:

- The **structural component** captures the geometric properties of the brain and is learned from preoperative MRI data.
- The **appearance component** is adapted intraoperatively to match the visual characteristics of surgical images.

This separation is achieved through a multi-style hypernetwork that controls the appearance of the NeRF while preserving its learned representation of the anatomy. The hypernetwork generates parameters for a subset of the NeRF's layers, allowing it to produce different appearances for the same underlying geometry.

During registration, the approach optimizes both the camera pose and the appearance parameters to minimize the dissimilarity between the rendered and target intraoperative images. This method has shown promising results in clinical data, outperforming state-of-the-art methods while meeting clinical standards for registration accuracy.

2.5. Loss Functions for Image Registration

The choice of loss function is crucial in registration tasks, as it defines the measure of similarity between images that guides the optimization process. Different loss functions capture different aspects of image similarity and may be more or less suitable depending on the specific registration task.

2.5.1. L2 Loss

The L2 loss, or mean squared error (MSE), is commonly used in image registration tasks due to its simplicity and differentiability. It calculates the squared Euclidean distance between two images:

$$\mathcal{L}_{\text{L2}}(I_1, I_2) = \frac{1}{N} \sum_{i=1}^{N} (I_1(i) - I_2(i))^2 \tag{2.6}$$

where N is the number of pixels. While straightforward, L2 loss assumes a direct intensity correspondence between images, which may not hold in cross-modal scenarios.

2.5.2. Normalized Cross-Correlation (NCC)

Normalized Cross-Correlation measures the similarity between two images independently of linear intensity transformations:

$$\mathcal{L}_{\text{NCC}}(I_1, I_2) = -\frac{\sum_{i=1}^{N} (I_1(i) - \bar{I}_1)(I_2(i) - \bar{I}_2)}{\sqrt{\sum_{i=1}^{N} (I_1(i) - \bar{I}_1)^2 \sum_{i=1}^{N} (I_2(i) - \bar{I}_2)^2}} \tag{2.7}$$

where $\bar{I}_1$ and $\bar{I}_2$ are the mean intensities of the respective images. NCC is particularly useful when images have different contrast or brightness levels [20].

2.5.3. Mutual Information (MI)

Mutual Information is a statistical measure that quantifies the mutual dependence between two random variables, making it particularly suitable for cross-modal registration where the relationship between intensities is complex:

$$\mathcal{L}_{\text{MI}}(I_1, I_2) = -\sum_{i,j} p_{I_1,I_2}(i,j) \log \left(\frac{p_{I_1,I_2}(i,j)}{p_{I_1}(i) p_{I_2}(j)} \right) \tag{2.8}$$

where p_{I_1,I_2} is the joint probability distribution of intensities in images I_1 and I_2, and p_{I_1} and p_{I_2} are their marginal distributions [21].

2.5.4. Weighted and Masked L2 Loss

Weighted and masked variants of the L2 loss assign different importance to different regions of the image:

$$\mathcal{L}_{\text{wL2}}(I_1, I_2) = \frac{1}{N} \sum_{i=1}^{N} w(i)(I_1(i) - I_2(i))^2 \tag{2.9}$$

where $w(i)$ is a weight or mask value for pixel i. This approach can be useful for focusing the registration on regions of interest or for ignoring irrelevant areas.

3. Methodology

This chapter presents the methodology for enhancing intraoperative registration using Neural Radiance Fields (NeRFs). First, it provides an overview of the NeRF-based registration approach, followed by details on the implementation framework, optimization procedure, and an examination of various loss functions.

3.1. Overview of the NeRF-based Registration Approach

The core methodology of this thesis builds upon the inverse Neural Radiance Field (iNeRF) approach originally proposed by Yen-Chen, Florence, Barron, et al. [4] and extended for cross-modal intraoperative registration by Fehrentz, Azampour, Dorent, et al. [5]. Figure 3.1 provides a high-level overview of the approach.

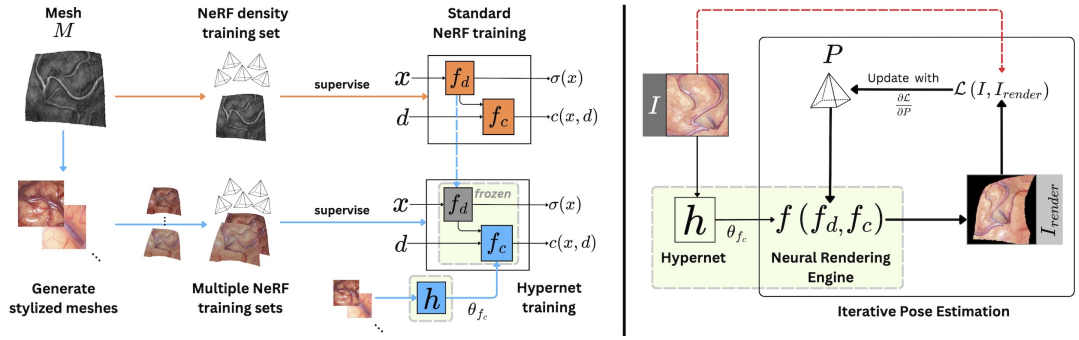

Figure 3.1.: Overview of the NeRF-based intraoperative registration approach. The method involves preoperative training of a NeRF model on MRI data, followed by intraoperative optimization of camera pose parameters to match the target surgical image.[5]

The registration process consists of two main phases:

1. **Preoperative Phase**: A NeRF model is trained using preoperative MRI data to create an implicit representation of the brain's structure.

2. **Intraoperative Phase**: During surgery, the pre-trained NeRF is used as a differentiable rendering engine. Given a target intraoperative image, the camera pose (6 degrees of freedom) is optimized through a gradient-based approach to match the rendered view with the target image.

3.2. Implementation Framework

One of the primary contributions of this thesis is the development of a flexible, model-agnostic implementation of neural registration built on the nerfstudio framework. Unlike previous implementations that were tied to specific NeRF variants, the proposed framework allows for seamless integration of different NeRF architectures and loss functions.

3.2.1. Nerfstudio Integration

The implementation leverages the nerfstudio framework to ensure flexibility and compatibility with various NeRF architectures. The key components include:

- **Model Agnosticism**: The implementation works with multiple NeRF variants, including the original NeRF [3], Instant-NGP [18], and Nerfacto [10], enabling evaluation of different architectures' impact on registration performance.
- **Registration Optimizer**: A dedicated optimizer module (the `iNeRFOptimizerBatchedFD` class) that handles camera pose optimization, loss computation, and experiment tracking.
- **Pluggable Loss Functions**: A modular interface for different loss functions, enabling systematic comparison of various similarity metrics.

The advantages of this implementation over previous approaches include:

- Easy customization of loss functions without modifying the core registration algorithm
- Compatibility with nerfstudio's pre-trained models
- Integration capabilities with hypernetwork-based appearance adaptation
- Comprehensive experiment tracking and visualization

3.2.2. Finite Difference Optimization

A notable technical aspect of the implementation is the use of finite difference methods for gradient computation. During development, challenges were encountered with the gradient flow being disconnected in the computational graph when using standard backpropagation. To overcome this limitation while preserving the original concept, a batched finite difference approach for computing gradients was implemented:

Algorithm 1 Batched Finite Difference Gradient Computation

Input: Current pose parameters θ, small perturbation ϵ, batch size B
Output: Gradient $\nabla_\theta L$
$L_{\text{original}}, \_ \leftarrow \text{ComputeLoss}(\theta)$ ▷ Compute loss at current pose
$\nabla_\theta L \leftarrow \mathbf{0}$ ▷ Initialize gradient
coords $\leftarrow \{(i, j)$ for all elements $\theta_{i,j}$ in $\theta\}$ ▷ All parameter coordinates
for batch_start $= 0$ to $|\text{coords}|$ step B **do**
    batch_coords $\leftarrow$ coords[batch_start:batch_start $+ B$]
    batch_poses $\leftarrow [\,]$ ▷ Initialize batch of perturbed poses
    for (i, j) in batch_coords **do**
        $\theta' \leftarrow \theta$
        $\theta'_{i,j} \leftarrow \theta'_{i,j} + \epsilon$ ▷ Perturb single parameter
        Add θ' to batch_poses
    end for
    batch_losses $\leftarrow$ ComputeBatchLosses(batch_poses)
    for idx, (i, j) in enumerate(batch_coords) **do**
        $\nabla_{\theta_{i,j}} L \leftarrow \frac{\text{batch\_losses[idx]} - L_{\text{original}}}{\epsilon}$ ▷ Estimate gradient
    end for
end for
return $\nabla_\theta L$

This approach offers several advantages:

- Robustness to disconnected gradients in the computational graph
- Efficient batch processing that significantly reduces computation time
- Compatibility with any NeRF model regardless of internal architecture

3.3. Neural Registration Algorithm

The complete neural registration algorithm using the implemented framework is presented below:

Algorithm 2 NeRF-based Registration with Customizable Loss Functions

Input: Pre-trained NeRF model $\mathcal{N}$, target image I_{target}, initial pose ξ_0, loss function $\mathcal{L}$, learning rate α, number of iterations T
Output: Optimized camera pose $\hat{\xi}$
Initialize pose parameters $\xi \leftarrow \xi_0$
Initialize optimizer with pose parameters and learning rate α
best_loss $\leftarrow \infty$, best_pose $\leftarrow \xi_0$
for $t = 1$ to T **do**
    Compute gradient $\nabla_\xi \mathcal{L}$ using batched finite differences:
        $L_{original}, I_{rendered} \leftarrow \text{ComputeLoss}(\xi)$ ▷ Forward pass with current pose
        $\nabla_\xi \mathcal{L} \leftarrow \text{BatchedFiniteDifference}(\xi, L_{original})$
    Update pose parameters: $\xi \leftarrow \xi - \alpha \cdot \nabla_\xi \mathcal{L}$ ▷ Gradient descent step
    if $L_{original} <$ best_loss **then**
        best_loss $\leftarrow L_{original}$
        best_pose $\leftarrow \xi$
    end if
    if t mod visualize_interval $= 0$ **then**
        Visualize current alignment between I_{target} and $I_{rendered}$
    end if
end for
return best_pose

The rendered image $I_{rendered}$ is generated by passing the current camera pose ξ to the pre-trained NeRF model, which produces an RGB prediction. The loss $\mathcal{L}(I_{target}, I_{rendered})$ measures the dissimilarity between the target and rendered images using the selected loss function.

3.3.1. Camera Pose Representation

The camera pose is represented as a 3×4 transformation matrix with the following structure:

$$\xi = \begin{bmatrix} R_{11} & R_{12} & R_{13} & t_x \\ R_{21} & R_{22} & R_{23} & t_y \\ R_{31} & R_{32} & R_{33} & t_z \end{bmatrix} \tag{3.1}$$

where R represents the 3×3 rotation matrix and t represents the translation vector. This 3×4 matrix is converted to the appropriate camera representation using the nerfstudio Camera class during rendering.

3.4. Loss Functions for NeRF-Based Registration

The effectiveness of NeRF-based registration fundamentally depends on the choice of loss function, which serves as the mathematical backbone guiding pose optimization. Loss func-

tions quantify the dissimilarity between rendered and target images, effectively creating the optimization landscape that determines convergence behavior and registration accuracy. This section provides a comprehensive analysis of five distinct loss functions (L1, L2, Structural Similarity Index, Normalized Cross-Correlation, and Mutual Information) in the context of intraoperative neural registration. We examine their mathematical formulations, implementation considerations, and comparative performance to understand how different similarity metrics affect registration precision, convergence speed, and robustness to initialization variations. Our systematic evaluation reveals important insights into selecting appropriate loss functions for specific neural registration scenarios, particularly in neurosurgical applications where alignment accuracy directly impacts surgical outcomes.

3.4.1. Role of Loss Functions in Registration

In the context of intraoperative registration using Neural Radiance Fields, the loss function serves two primary purposes:

1. **Similarity Measurement**: It quantifies the similarity between the target intraoperative image and the rendered image from the NeRF model, guiding the optimization process toward better alignment.
2. **Gradient Provision**: It provides gradients with respect to camera pose parameters, enabling optimization of the camera pose.

3.4.2. Implemented Loss Functions

This section details the loss functions implemented and evaluated in this thesis. Further implementation details can be found in Appendix A.

L1 Loss

The L1 loss, or mean absolute error, calculates the absolute difference between pixel values:

$$\mathcal{L}_{L1}(I_1, I_2) = \frac{1}{N} \sum_{i=1}^{N} |I_1(i) - I_2(i)| \tag{3.2}$$

where N is the number of pixels in the images.
Key characteristics of L1 loss include:

- Less sensitivity to outliers compared to L2 loss
- Linear behavior that can lead to faster convergence in some cases
- Simple gradients that are constant with respect to the error magnitude

L2 Loss

The L2 loss, or mean squared error (MSE), is the standard baseline approach:

$$\mathcal{L}_{L2}(I_1, I_2) = \frac{1}{N} \sum_{i=1}^{N} (I_1(i) - I_2(i))^2 \tag{3.3}$$

Key characteristics of L2 loss include:

- Quadratic penalization of large errors
- Gradients that scale with the magnitude of the error
- Sensitivity to outliers

Structural Similarity Index Loss

The Structural Similarity Index Measure (SSIM) captures structural information in images:

$$\text{SSIM}(x, y) = \frac{(2\mu_x\mu_y + C_1)(2\sigma_{xy} + C_2)}{(\mu_x^2 + \mu_y^2 + C_1)(\sigma_x^2 + \sigma_y^2 + C_2)} \tag{3.4}$$

The SSIM loss is then defined as:

$$\mathcal{L}_{\text{SSIM}}(I_1, I_2) = 1 - \text{SSIM}(I_1, I_2) \tag{3.5}$$

Key characteristics of SSIM loss include:

- Focus on structural information rather than pixel-wise differences
- Invariance to certain local transformations
- Better correlation with human visual perception

Normalized Cross-Correlation (NCC)

Normalized Cross-Correlation measures the linear relationship between two images while being invariant to linear intensity transformations:

$$\text{NCC}(I_1, I_2) = \frac{\sum_{i=1}^{N} (I_1(i) - \bar{I}_1)(I_2(i) - \bar{I}_2)}{\sqrt{\sum_{i=1}^{N} (I_1(i) - \bar{I}_1)^2 \sum_{i=1}^{N} (I_2(i) - \bar{I}_2)^2}} \tag{3.6}$$

The NCC loss is then defined as:

$$\mathcal{L}_{\text{NCC}}(I_1, I_2) = 1 - \text{NCC}(I_1, I_2) \tag{3.7}$$

Key characteristics of NCC loss include:

- Invariance to linear intensity transformations
- Robustness to global illumination changes
- Effectiveness for cross-modal registration tasks

Mutual Information (MI)

Mutual Information quantifies the statistical dependence between two random variables:

$$\mathrm{MI}(I_1, I_2) = \sum_{i,j} p_{I_1,I_2}(i,j) \log \left(\frac{p_{I_1,I_2}(i,j)}{p_{I_1}(i) p_{I_2}(j)} \right) \tag{3.8}$$

The MI loss is defined as:

$$\mathcal{L}_{\mathrm{MI}}(I_1, I_2) = -\mathrm{MI}(I_1, I_2) \tag{3.9}$$

Key characteristics of MI loss include:

- Ability to capture complex, non-linear relationships between image intensities
- Suitability for cross-modal registration
- Robustness to partial overlap between images

3.5. Experimental Setup

To rigorously evaluate the impact of different loss functions on Neural Radiance Field (NeRF) based intraoperative registration, a systematic experimental framework was designed with controlled variables and precise measurement protocols.

3.5.1. Technical Configuration

The experiments were conducted with the following specifications to ensure fair and consistent comparison across loss functions:

- **Optimization Parameters**: Fixed at 50 iterations across all experiments to standardize convergence conditions
- **Optimization Algorithm**: AdamW optimizer with a learning rate of 0.01, chosen for its adaptive momentum properties and weight decay regularization
- **Gradient Computation**: Finite difference approximation with epsilon value of 1×10^{-4} for gradient stability
- **Experimental Runs**: 10 different initial camera poses per loss function to ensure statistical significance and account for local optimization challenges
- **Target Consistency**: Identical target image used across all experiments to eliminate target-specific biases
- **Data Source Control**: Both target images and rendered views sourced from the same pre-trained NeRF model to eliminate variations due to domain gaps or rendering inconsistencies
- **Batch Size**: Set at 12 for finite difference calculations to optimize the trade-off between memory usage and computational efficiency

3.5.2. Loss Functions

As previously introduced in Section 3.4, five distinct loss functions were systematically evaluated to compare their performance in NeRF-based registration:

- **L1 Loss (Mean Absolute Error)**
- **L2 Loss (Mean Squared Error)**
- **Structural Similarity Index Loss (SSIM)**
- **Normalized Cross-Correlation (NCC)**
- **Mutual Information Loss (MI)**

Each loss function was selected for its unique properties and potential advantages in medical image registration: L1 and L2 for their direct intensity comparisons with different sensitivity to outliers; SSIM for its emphasis on structural information; NCC for its robustness to linear intensity variations; and MI for its capacity to handle multi-modal image alignment.

3.5.3. Evaluation Metrics

Registration performance was comprehensively evaluated using the following quantitative and qualitative metrics:

- **Convergence Efficiency**: Measured by identifying the iteration at which the optimal loss value was achieved, providing insight into each loss function's optimization trajectory
- **Computational Performance**: Total time required for the complete optimization process, broken down into component stages including forward passes, gradient calculations, and parameter updates
- **Registration Accuracy**: Quantified through:
  - Final loss value achieved
  - Pixel-wise alignment between registered and target images
  - Visual overlay assessment by superimposing the registered image on the target with 50% transparency
- **Optimization Stability**: Analyzed through:
  - Loss curve smoothness and monotonicity
  - Variance in pose parameter updates across iterations
  - Resistance to local minima in the optimization landscape
- **Robustness to NeRF Artifacts**: Qualitative assessment of how each loss function performs in the presence of rendering inconsistencies

3.5.4. Experiment Tracking and Data Collection

A comprehensive tracking framework was implemented to capture all relevant experimental data:

- **Parameter Trajectory**: Complete camera pose transformation matrix recorded at each iteration
- **Loss Dynamics**: Full loss history with values captured after each optimization step
- **Visual Documentation**:
  - Rendered images saved at regular intervals (every 10 iterations)
  - Side-by-side visualizations of target and current rendered images
  - Progressive alignment overlays to visually track registration improvement
- **Performance Profiling**:
  - Per-iteration timing statistics
  - Computational resource utilization
  - Batch processing efficiency metrics for finite difference calculations
- **Metadata Management**: Structured JSON tracking files containing complete experimental parameters, iteration history, and final results for reproducibility and further analysis

All experimental data was systematically organized in a standardized directory structure to facilitate comparative analysis and ensure reproducibility of results.

3.6. Summary

This chapter has presented a comprehensive methodology for enhancing intraoperative registration using Neural Radiance Fields. The key contributions include:

1. Development of a flexible, model-agnostic implementation of neural registration integrated with the nerfstudio framework
2. Implementation of a batched finite difference approach for robust gradient computation
3. Systematic evaluation of multiple loss functions (L1, L2, SSIM, NCC, MI) for NeRF-based registration
4. A controlled experimental setup for fair comparison of different approaches

The results of these experiments are presented and analyzed in Chapter 4.

4. Results

This chapter presents the experimental results of our investigation into enhancing intraoperative registration with Neural Radiance Fields through different loss functions. We begin with a description of our nerfstudio-based implementation, followed by a comprehensive evaluation of different loss functions and their impact on registration performance.

4.1. NeRF-Based Registration Implementation

One of the primary contributions of this work is the development of a nerfstudio-based implementation of neural registration that is agnostic to the specific NeRF model architecture (e.g., vanilla NeRF, InstantNGP). This implementation leverages finite differences to compute gradients and provides a flexible framework for experimenting with different loss functions.

4.1.1. Optimization Strategy

Our implementation follows an inverse neural rendering approach inspired by iNeRF [4]. The key steps in our optimization strategy are as follows:

1. Start with an initial camera pose in 3D NeRF space, represented as a transformation matrix.
2. Render a snapshot from the pre-trained NeRF at the current pose.
3. Calculate the error between the rendered image and the target image using the selected loss function.
4. Update the camera pose through optimization to minimize this error.
5. Repeat steps 2-4 until convergence or a maximum number of iterations is reached.

To overcome challenges with gradient flow disconnection in the rendering pipeline, we implemented a finite differences approach for gradient computation. This approach, while computationally more intensive than direct backpropagation, provides stable and reliable gradients for pose optimization.

4.2. Loss Function Evaluation

We evaluated the performance of five different loss functions in the context of NeRF-based registration:

- L1 Loss (Mean Absolute Error)
- L2 Loss (Mean Squared Error)
- Structural Similarity Index Loss (SSIM)
- Normalized Cross-Correlation Loss (NCC)
- Mutual Information Loss (MI)

Our evaluation focused on convergence behavior, stability, and final registration accuracy across multiple starting positions.

4.2.1. Experimental Setup

To ensure controlled and reproducible results, we established the following experimental setup:

- All experiments were limited to 50 iterations
- AdamW optimizer with a learning rate of 0.01
- 10 different starting points with the same target image
- Both target and rendered images sourced from the same NeRF model
- Batch size of 12 for finite differences gradient computation

4.2.2. Convergence Behavior

Table 4.1 summarizes the average convergence performance of different loss functions in our experiments.

Table 4.1.: Average convergence performance of different loss functions.

Loss Function	Average Best Loss Iteration	Average Time Elapsed (s)
L1	39	621
L2	47	624
Structural Similarity Index	49	626
Normalized Cross-Correlation	44	621
Mutual Information	42	621

Our results indicate that L1 loss converges more quickly on average compared to other loss functions, reaching its best loss value around iteration 39. In contrast, SSIM requires almost the full 50 iterations before converging to its best value.

Figure 4.1 illustrates the typical convergence behavior of different loss functions during the registration optimization process.

Figure 4.1.: Convergence behavior of different loss functions during registration optimization. The plot shows the evolution of loss as a function of optimization iterations.

4.2.3. Qualitative Analysis

Our qualitative analysis of the optimization trajectories revealed several interesting patterns:

- L1 and L2 losses tend to follow smoother, more direct paths during optimization, with L1 generally converging faster.
- Mutual Information and Normalized Cross-Correlation losses exhibit more "exploratory" behavior in early iterations, with more pronounced oscillations in the optimization path.
- SSIM demonstrates slower initial progress but continues to improve throughout the optimization process.
- MI and NCC appear to be more robust to NeRF rendering artifacts, potentially making them more suitable for cross-modal scenarios.

4.2.4. Registration Accuracy and Stability

While all loss functions eventually achieved visually satisfactory registration results, our observations suggest differences in their stability and robustness:

- L1 loss provides the fastest convergence while maintaining good stability.
- MI and NCC show greater resilience to local optima, potentially offering advantages in real-world scenarios with imperfect NeRF representations.
- SSIM focuses more on structural alignment at multiple scales, which may be beneficial for preserving anatomical structure correspondence.

4.3. L1 Loss Performance

The L1 loss (Mean Absolute Error) demonstrated the fastest average convergence among the tested loss functions. Figure 4.2 shows an overlay of the final registered image on the target for a representative L1 optimization run.

Figure 4.3 shows the corresponding loss history for this optimization run.

The L1 loss function achieved consistent and rapid convergence, reaching its best value at an average of 39 iterations. This suggests that L1 loss may be preferable in scenarios where computational efficiency is a priority.

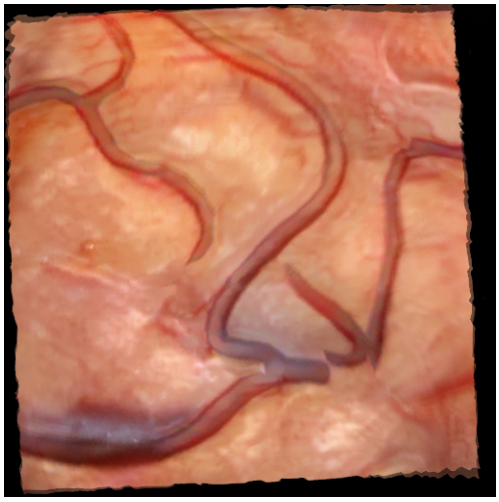

Figure 4.2.: Overlay of the final registered image on the target image using L1 loss.

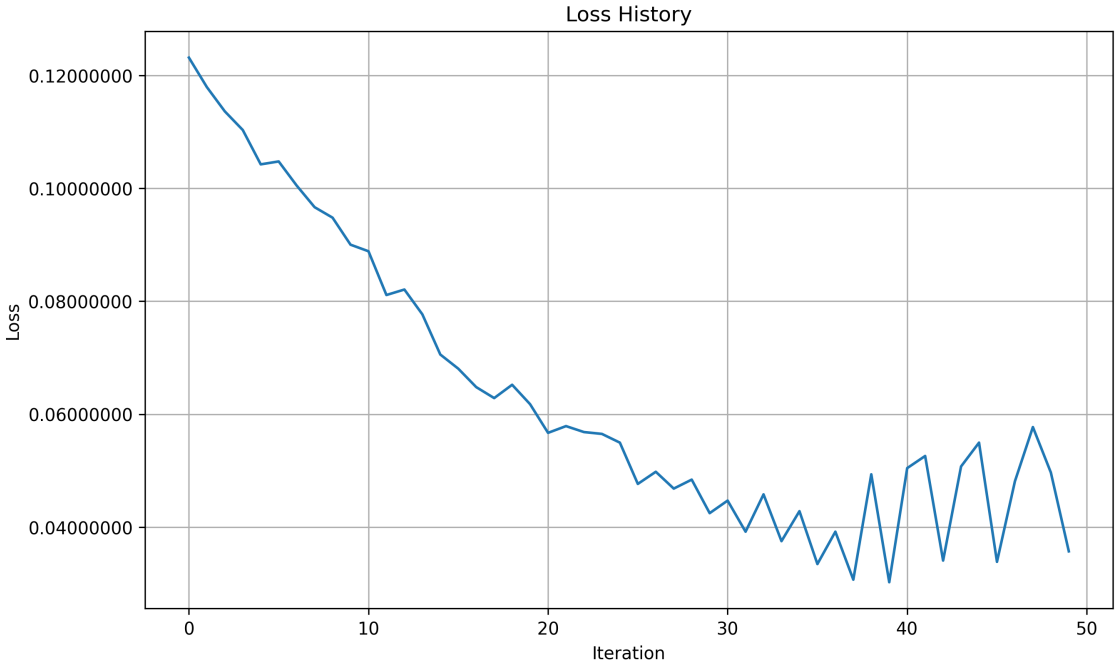

Figure 4.3.: Loss history for registration optimization using L1 loss.

4.4. L2 Loss Performance

The L2 loss (Mean Squared Error) is the most commonly used loss function in NeRF-based registration approaches, including the original iNeRF implementation. Figure 4.4 shows an overlay of the final registered image on the target for a representative L2 optimization run.

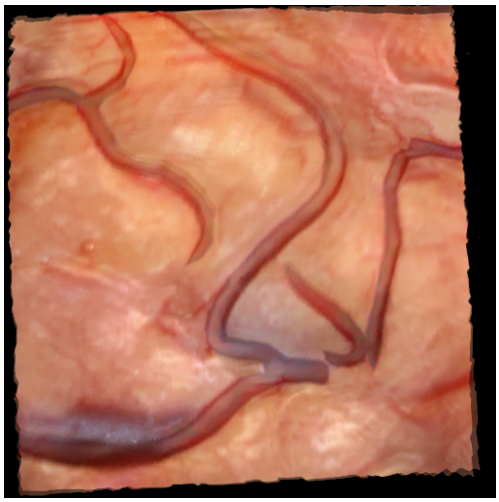

Figure 4.4.: Overlay of the final registered image on the target image using L2 loss.

Figure 4.5 shows the corresponding loss history for this optimization run.

The L2 loss required more iterations to converge compared to L1, with an average best loss occurring at iteration 47. This suggests that while L2 can achieve high-quality registration, it may require more computational resources to reach convergence.

4.5. Structural Similarity Index Loss Performance

The Structural Similarity Index (SSIM) loss focuses on preserving structural information rather than pixel-wise differences. Figure 4.6 shows an overlay of the final registered image on the target for a representative SSIM optimization run.

Figure 4.7 shows the corresponding loss history for this optimization run.

The SSIM loss function required the most iterations to reach its best value, with an average of 49 iterations. This suggests that while SSIM can provide high-quality registration with emphasis on structural correspondence, it may require more computational time to converge fully.

4.6. Normalized Cross-Correlation Loss Performance

Normalized Cross-Correlation (NCC) is often used in multi-modal registration scenarios due to its invariance to linear intensity changes. Figure 4.8 shows an overlay of the final registered image on the target for a representative NCC optimization run.

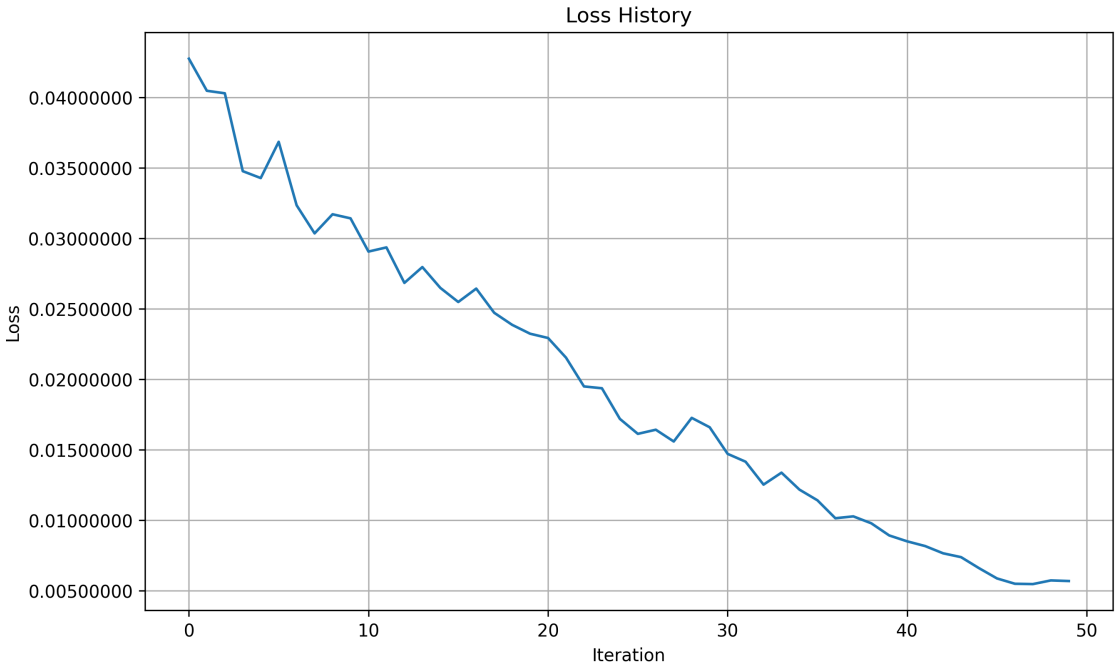

Figure 4.5.: Loss history for registration optimization using L2 loss.

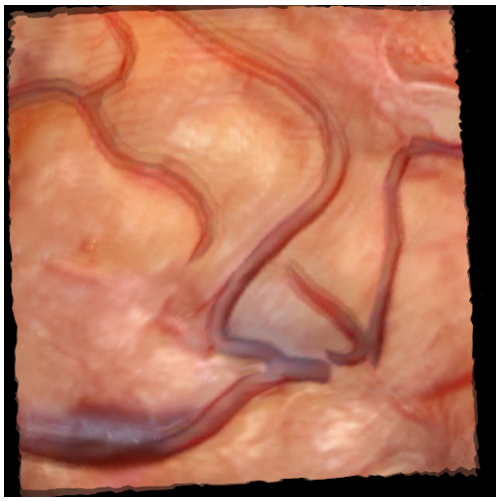

Figure 4.6.: Overlay of the final registered image on the target image using SSIM loss.

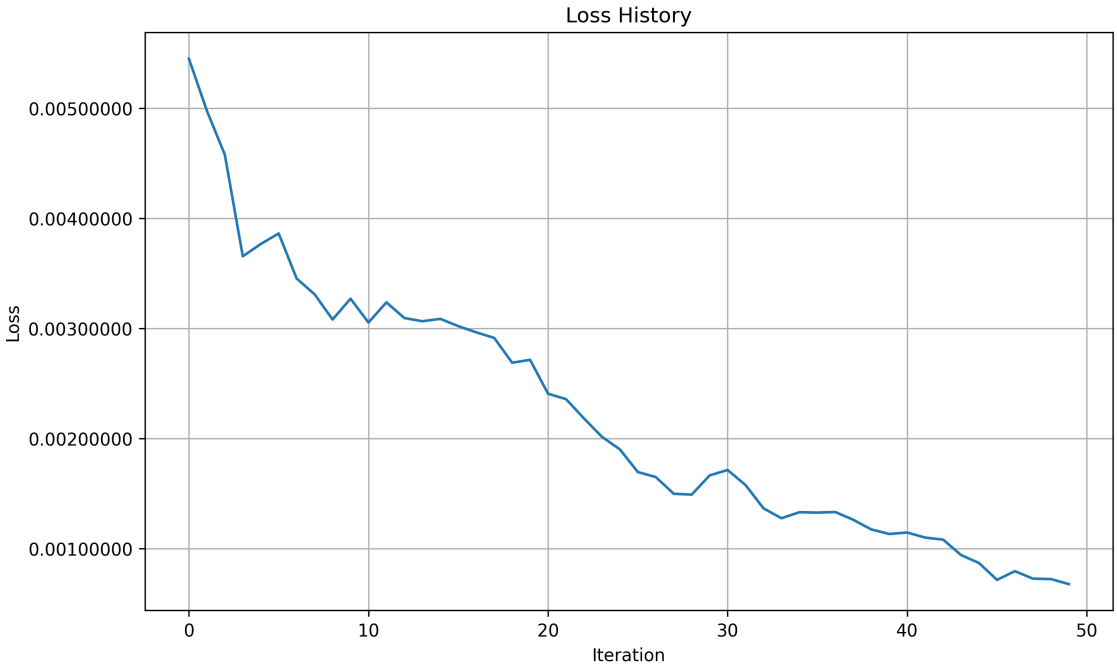

Figure 4.7.: Loss history for registration optimization using SSIM loss.

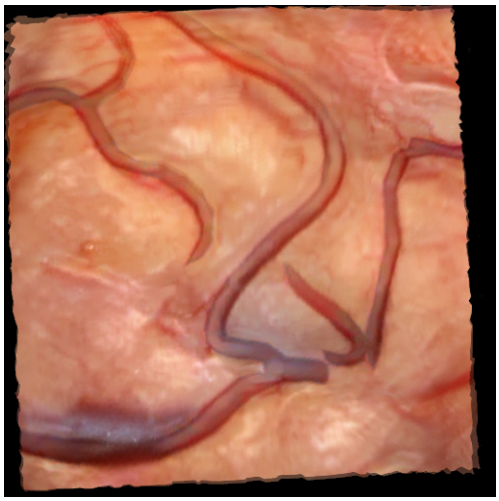

Figure 4.8.: Overlay of the final registered image on the target image using NCC loss.

Figure 4.9 shows the corresponding loss history for this optimization run.

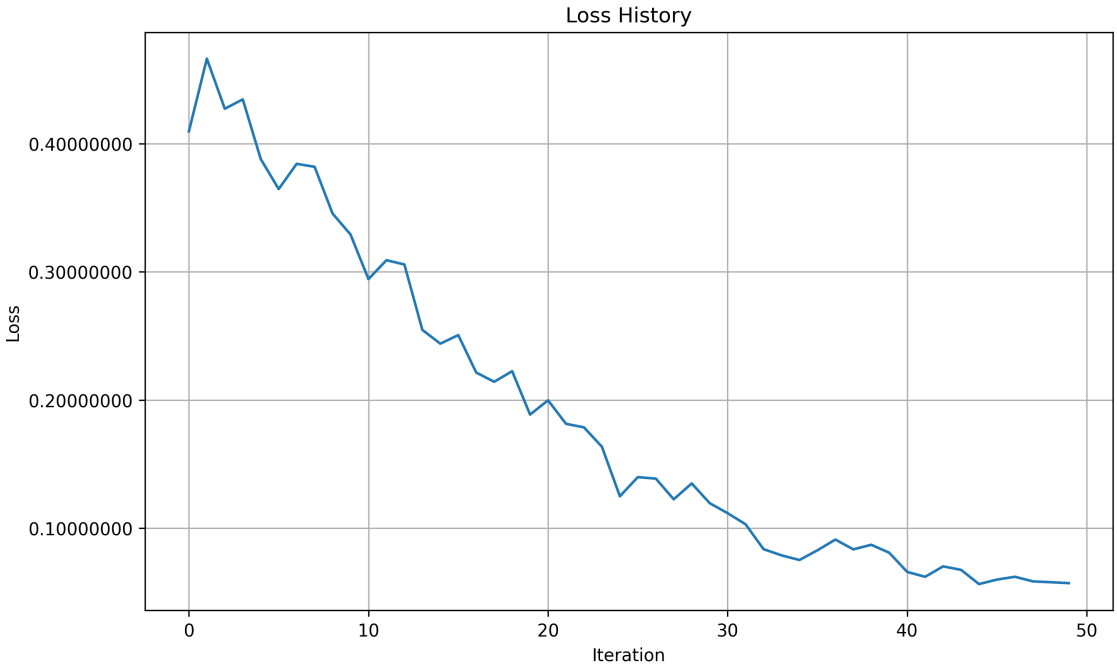

Figure 4.9.: Loss history for registration optimization using NCC loss.

The NCC loss function demonstrated moderate convergence speed, with its best value occurring at an average of 44 iterations. While slightly slower than L1, NCC showed good stability and robustness to intensity variations.

4.7. Mutual Information Loss Performance

Mutual Information (MI) is a widely used similarity measure for multi-modal registration. Figure 4.10 shows an overlay of the final registered image on the target for a representative MI optimization run.

Figure 4.11 shows the corresponding loss history for this optimization run.

The MI loss function reached its best value at an average of 42 iterations, placing it between L1 and NCC in terms of convergence speed. The more exploratory optimization path of MI may contribute to its robustness in complex registration scenarios.

4.8. Limitations and Challenges

Our implementation and evaluation encountered several limitations and challenges that should be considered:

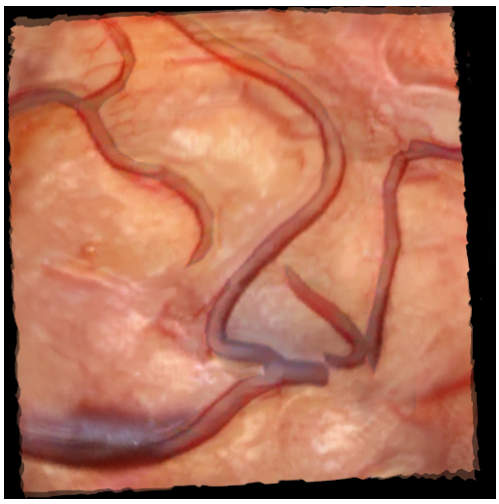

Figure 4.10.: Overlay of the final registered image on the target image using MI loss.

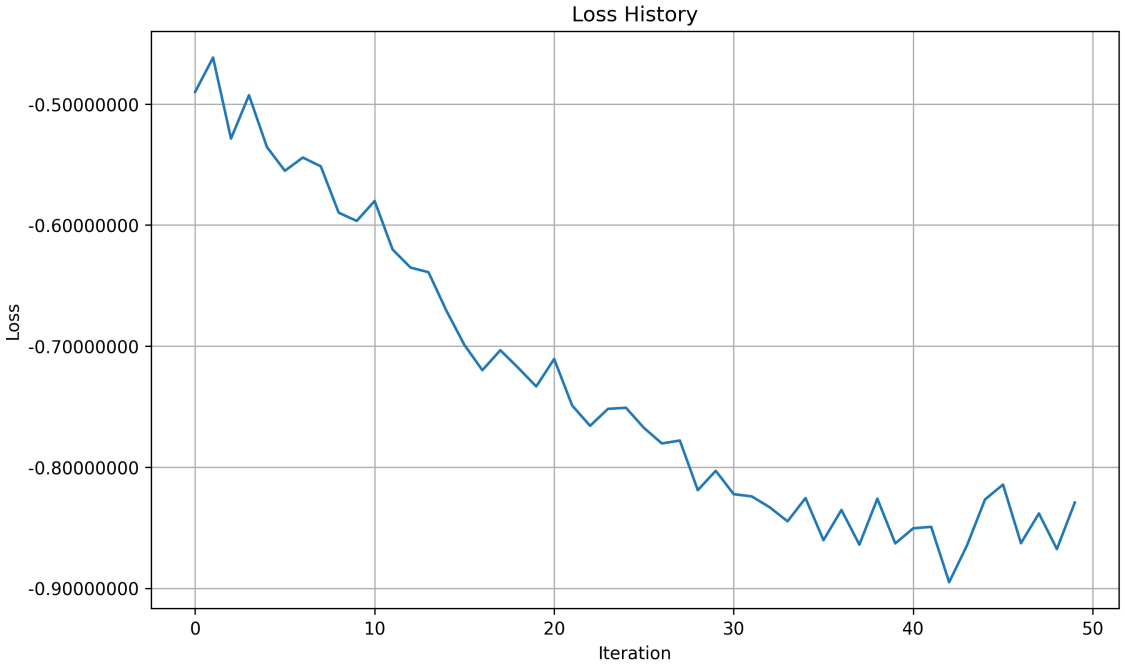

Figure 4.11.: Loss history for registration optimization using MI loss.

- The use of finite differences for gradient computation, while effective, is computationally more expensive than direct backpropagation.
- Our experiments assumed perfect NeRF representation of the target object, which is a simplification of real-world scenarios where appearance and lighting variations are significant.
- The computational cost of registration (averaging around 621-626 seconds for 50 iterations) may be challenging for real-time clinical applications.
- Our evaluation was limited to a specific set of hyperparameters (learning rate, optimizer, batch size) and may not reflect performance under different conditions.

4.9. Summary of Findings

Based on our comprehensive evaluation, we summarize the key findings of our study:

1. Our nerfstudio-based implementation provides a flexible framework for experimenting with different loss functions in NeRF-based registration, independent of the specific NeRF architecture.
2. L1 loss demonstrates the fastest convergence among the tested loss functions, making it potentially suitable for time-sensitive applications.
3. Mutual Information and Normalized Cross-Correlation losses exhibit more exploratory behavior during optimization and appear more robust to NeRF rendering artifacts.
4. SSIM loss shows slower convergence but maintains focus on structural correspondence, which may be beneficial for preserving anatomical structures.
5. All five loss functions eventually achieve visually satisfactory registration, suggesting that the choice of loss function may depend on specific application requirements such as speed, robustness, or structural preservation.

These results highlight the importance of loss function selection in NeRF-based registration and provide a foundation for future work in this area.

5. Discussion

In this chapter, we interpret the results of our experiments, discuss the caveats associated with our experimental setup, highlight the limitations of our current approach, and suggest directions for future research.

5.1. Interpretation of Results

5.1.1. Loss Function Performance

Our experiments revealed distinct performance characteristics associated with each evaluated loss function. L1 loss consistently demonstrated faster convergence compared to other loss functions, notably achieving quicker reductions in error. Additionally, both L1 and L2 losses exhibited smoother and more direct convergence trajectories, which may indicate their suitability for tasks requiring rapid and stable alignment, such as intraoperative scenarios.

In contrast, Structural Similarity Index (SSIM), Normalized Cross-Correlation (NCC), and Mutual Information (MI) demonstrated more explorative behavior during initial iterations, characterized by fluctuating, "jiggly" optimization paths. This behavior suggests these losses may explore the parameter space more thoroughly at the cost of slower initial convergence. Mutual Information and NCC, in particular, showed more oscillations, reflecting their sensitivity to variations in pixel intensities but also their potential robustness to certain types of noise or distortions.

5.2. Caveats of the Experimental Setup

The experimental design involved several key simplifications that limit direct translation to clinical practice:

- **Perfect NeRF assumption:** Both target and iterative images were synthesized from the same pre-trained NeRF, eliminating realistic factors such as lighting variations, surgical environment noise, and intraoperative tissue deformation. Therefore, while the current setup provides a controlled environment to evaluate the registration method, it overestimates real-world accuracy.
- **Finite Differences for Gradient Calculation:** Due to complexity in directly implementing backpropagation through InstantNGP-based NeRF models, finite differences were utilized as a workaround. While effective in proof-of-concept scenarios, this method is computationally expensive and less precise than automatic differentiation.

- **Constant Camera Distance and Fixed Initialization:** The experimental scenario assumed a fixed camera distance and a manually chosen initial position that always provided a partial view of the target region. This is unrealistic in surgical settings, where initial camera positioning can vary significantly, and occlusions may frequently occur.

5.3. Limitations

Several limitations affect the current approach:

- **Computational Efficiency:** Finite differences-based gradient estimation significantly increases computational load, limiting real-time clinical applicability.
- **Assumption of Model Accuracy:** The assumption that NeRF perfectly captures brain geometry and appearance neglects inevitable inaccuracies in the preoperative model and intraoperative tissue deformation and appearance changes.
- **Initialization Sensitivity:** The current method heavily depends on the initial camera pose selection, affecting convergence speed and reliability.
- **Generalization to Clinical Data:** Experiments conducted entirely with synthetic images derived from a single NeRF model may not generalize well to real clinical images due to differences in lighting, texture, and anatomical variability.

5.4. Future Work

Future directions for improving and extending the proposed neural registration approach include:

- **Automatic Differentiation Integration:** Replacing finite differences with automatic differentiation methods to significantly improve computational efficiency and stability.
- **Gaussian Splats:** Investigating Gaussian Splats as an alternative representation, offering faster rendering, fewer visual artifacts, and potentially superior registration performance.
- **Enhanced NeRF Models:** Incorporating hypernetwork-based methods to better match the NeRF model's density and appearance to actual intraoperative conditions, addressing the oversimplification in the current setup.
- **Robust Initialization Strategies:** Developing methods for automatic or semi-automatic selection of robust initial camera poses, thus reducing sensitivity to starting conditions and improving reliability.

- **Hybrid Loss Functions:** Exploring combined or adaptive loss functions to leverage the strengths of multiple similarity metrics, potentially achieving superior registration accuracy and robustness.
- **Clinical Validation:** Extending evaluation to realistic clinical or phantom datasets to assess practical applicability, robustness against real-world variability, and identification of clinical relevance.

Addressing these areas in future research will pave the way for practical clinical applications of NeRF-based intraoperative registration, enhancing surgical precision and patient outcomes.

6. Conclusion

This study introduced an enhanced intraoperative registration method using Neural Radiance Fields (NeRFs), highlighting the critical role of differentiable implicit representations in accurately aligning preoperative and intraoperative brain surface images. By developing a flexible, NeRF-model agnostic implementation inspired by iNeRF, the algorithm facilitates robust and customizable intraoperative registration workflows.

We conducted an extensive exploration of various loss functions (L1, L2, Structural Similarity Index, Normalized Cross-Correlation, and Mutual Information) and analyzed their impacts on the convergence behavior and registration accuracy. The results demonstrated that L1 and L2 losses provide rapid and stable convergence, making them suitable for clinical scenarios requiring efficiency and reliability. On the other hand, Mutual Information and Normalized Cross-Correlation, despite their less direct convergence trajectories, offer greater flexibility in scenarios where robustness to imaging variations might be advantageous.

Despite these promising findings, the study acknowledges critical simplifications and assumptions. The current simulation assumes an idealized scenario where the NeRF perfectly represents the intraoperative brain surface, eliminating realistic noise, lighting variations, and inaccuracies. Additionally, the computational performance remains limited by the finite difference gradient estimation method implemented as a workaround for gradient propagation challenges in InstantNGP models.

Future work should focus on overcoming these limitations by:

- Developing more realistic NeRF models that capture intraoperative variability in brain appearance through enhanced coloring techniques, potentially leveraging hypernetworks for more robust representations.
- Transitioning from finite difference-based gradient calculations to efficient direct backpropagation methods to improve optimization speed and efficiency.

Further exploration into faster and visually superior methods, such as Gaussian Splatting, represents a promising direction for significantly improving registration performance and computational efficiency, ultimately advancing the accuracy and reliability of intraoperative brain registration in clinical settings.

A. Implementation Details

This appendix provides additional technical details about the implementation of the NeRF-based registration approach described in this thesis. The code is available at: `https://github.com/maxfehrentz/style-ngp`.

A.1. Software Implementation

The neural registration framework described in this thesis was implemented in Python using PyTorch. This section provides a detailed description of the key components of the implementation.

A.1.1. Image Processing and Conversion

The following utility functions handle image processing and conversion between different formats:

```
def image_to_tensor(image_path, device) -> torch.Tensor:
    # Open the image using PIL
    image = Image.open(image_path).convert("RGB")

    # Define the transform to convert the image to a PyTorch tensor
    transform = transforms.ToTensor() # This will convert to a tensor with shape (C, H, W)

    # Apply the transform
    tensor = transform(image) # Shape will be (3, 512, 512)

    # Permute the tensor to get shape (512, 512, 3)
    tensor = tensor.permute(1, 2, 0).to(device)

    return tensor.detach().requires_grad_(False)

def show_image(tensor):
    plt.figure(figsize=(5, 5))
    plt.imshow(tensor.detach().cpu().numpy())
    plt.axis('off')
    plt.show()
```

A.1.2. iNeRF Optimization

The core of the registration framework is the `iNeRFOptimizerBatchedFD` class, which implements inverse NeRF optimization with batched finite differences for gradient computation. This approach allows for the estimation of camera pose parameters by minimizing the difference between a rendered NeRF view and a target image.

Initialization

The optimizer is initialized with the following parameters:

```
def __init__(
    self,
    experiment_name,
    nerf_model,
    target_image,
    initial_pose,
    dataparser_matrix,
    dataparser_scale,
    camera_params,
    loss_fn = nn.MSELoss(),
    lr=0.001,
    num_iterations=1000,
    config_path=None
):
```

Parameters include:

- `experiment_name`: Name for tracking and saving results
- `nerf_model`: The pre-trained NeRF model
- `target_image`: The target image to register against
- `initial_pose`: Initial camera pose estimate
- `dataparser_matrix` and `dataparser_scale`: Transform parameters for coordinate system alignment
- `camera_params`: Camera intrinsic parameters
- `loss_fn`: Loss function for comparing rendered and target images
- `lr`: Learning rate
- `num_iterations`: Maximum number of optimization iterations

Optimization Process

The optimization uses finite differences to compute gradients rather than automatic differentiation. This is implemented in the `optimize_step` method:

```
def optimize_step(self, batch_size=4, debug=True):
    # Zero gradients
    self.optimizer.zero_grad()

    # Get current pose parameters
    pose = self.pose_param.detach().clone()

    # Compute loss for current pose
    original_loss, pred_rgb = self.compute_loss_no_grad(pose)

    # Small epsilon for finite differences
    eps = 1e-4

    # Compute gradients using finite differences in batches
    grad = torch.zeros_like(pose)

    # Flatten the pose for easier batch processing
    num_params = pose.numel()

    # Use coordinate indexing to track which element we're perturbing
    coords = [(i, j) for i in range(pose.shape[0]) for j in range(pose.shape[1])]

    # Process in batches
    for batch_idx in range(0, num_params, batch_size):
        batch_coords = coords[batch_idx:min(batch_idx+batch_size, num_params)]

        # Create a batch of perturbed poses
        batch_poses = []
        for i, j in batch_coords:
            perturbed_pose = pose.clone()
            perturbed_pose[i, j] += eps
            batch_poses.append(perturbed_pose)

        # Stack poses into a batch
        batch_poses_tensor = torch.stack(batch_poses)

        # Compute losses for all poses in the batch
        batch_losses = self.compute_batch_losses(batch_poses_tensor)
```

```
            # Calculate gradients
            for idx, (i, j) in enumerate(batch_coords):
                grad[i, j] = (batch_losses[idx] - original_loss) / eps

        # Manually set gradients
        self.pose_param.grad = grad

        # Perform optimization step
        self.optimizer.step()

        return original_loss.item(), pred_rgb
```

This batched approach to finite differences significantly improves performance by computing multiple perturbed poses in parallel.

Camera Creation and Loss Computation

The optimizer creates cameras from pose matrices and computes losses between rendered and target images:

```
def create_camera_from_pose(self, pose):
    camera = Cameras(
        camera_to_worlds=pose.unsqueeze(0),
        fx=self.camera_params["fl_x"],
        fy=self.camera_params["fl_y"],
        cx=self.camera_params["cx"],
        cy=self.camera_params["cy"],
        camera_type=CameraType.PERSPECTIVE,
        height=self.camera_params["h"],
        width=self.camera_params["w"],
    )
    return camera

def compute_loss_no_grad(self, pose):
    with torch.no_grad():
        # Create camera with the given pose
        camera = self.create_camera_from_pose(pose)

        # Get outputs using the model's built-in method for rendering
        outputs = self.nerf_model.get_outputs_for_camera(camera)

        # Compute loss
        pred_rgb, image = self.nerf_model.renderer_rgb.blend_background_for_loss_computation(
            pred_image=outputs["rgb"],
```

```
            pred_accumulation=outputs["accumulation"],
            gt_image=self.target_image,
        )

        loss = self.nerf_model.rgb_loss(image, pred_rgb)

    return loss, pred_rgb
```

A.1.3. Loss Functions

Multiple loss functions were implemented and compared for registration accuracy:

Standard Losses

```
# L1 and L2 losses
l1_loss = nn.L1Loss()
l2_loss = nn.MSELoss()
huber_loss = nn.SmoothL1Loss(beta=0.5)
```

Structural Similarity Index Loss

```
class StructuralSimilarityIndexLoss(nn.Module):
    def __init__(self):
        super(StructuralSimilarityIndexLoss, self).__init__()

    def forward(self, x, y):
        # Rearrange dimensions if needed
        if x.dim() == 3: # [H, W, C]
            # Rearrange to [1, C, H, W]
            x = x.permute(2, 0, 1).unsqueeze(0)
            y = y.permute(2, 0, 1).unsqueeze(0)

        return 1 - pytorch_msssim.ssim(x, y)
```

Normalized Cross-Correlation Loss

```
class NormalizedCrossCorrelationLoss(nn.Module):
    def __init__(self):
        super(NormalizedCrossCorrelationLoss, self).__init__()

    def forward(self, x, y):
```

```
        # Ensure proper dimensions
        if x.dim() == 3: # [H, W, C]
            # Handle each channel separately and average
            channels = x.shape[2]
            ncc_sum = 0.0

            for c in range(channels):
                x_c = x[..., c].flatten()
                y_c = y[..., c].flatten()
                ncc_sum += self._compute_ncc(x_c, y_c)

            ncc = ncc_sum / channels
        else: # Assume [B, C, H, W]
            batch_size, channels = x.shape[0], x.shape[1]
            ncc_sum = 0.0

            for b in range(batch_size):
                channel_ncc = 0.0
                for c in range(channels):
                    x_bc = x[b, c].flatten()
                    y_bc = y[b, c].flatten()
                    channel_ncc += self._compute_ncc(x_bc, y_bc)
                ncc_sum += channel_ncc / channels

            ncc = ncc_sum / batch_size

        # Convert to loss (1 - NCC since NCC=1 is perfect correlation)
        return 1.0 - ncc

    def _compute_ncc(self, x, y):
        # Mean centering
        x_centered = x - x.mean()
        y_centered = y - y.mean()

        # Compute normalization factors
        x_norm = torch.sqrt(torch.sum(x_centered ** 2) + 1e-8)
        y_norm = torch.sqrt(torch.sum(y_centered ** 2) + 1e-8)

        # Compute NCC
        ncc = torch.sum(x_centered * y_centered) / (x_norm * y_norm)

        # Ensure result is in [-1, 1] range
```

```
        return torch.clamp(ncc, -1.0, 1.0)
```

Mutual Information Loss

```
class MutualInformationLoss(nn.Module):
    def __init__(self, bins=32, sigma=0.1):
        super(MutualInformationLoss, self).__init__()
        self.bins = bins
        self.sigma = sigma
        self.epsilon = 1e-10 # Small constant to avoid log(0)

    def forward(self, x, y):
        # Ensure proper dimensions
        if x.dim() == 3: # [H, W, C]
            x_flat = x.reshape(-1)
            y_flat = y.reshape(-1)
        else: # Assume [B, C, H, W]
            x_flat = x.reshape(x.size(0), -1)
            y_flat = y.reshape(y.size(0), -1)

        # Scale to [0, 1] if not already
        if x_flat.max() > 1.0 or x_flat.min() < 0.0:
            x_flat = (x_flat - x_flat.min()) / (x_flat.max() - x_flat.min() + self.epsilon)
        if y_flat.max() > 1.0 or y_flat.min() < 0.0:
            y_flat = (y_flat - y_flat.min()) / (y_flat.max() - y_flat.min() + self.epsilon)

        # Compute mutual information
        mi_score = self._compute_mutual_information(x_flat, y_flat)

        # Return negative MI as we want to minimize loss
        return -mi_score
```

A.1.4. Experiment Tracking and Visualization

The implementation includes comprehensive experiment tracking and visualization features:

- Automatic creation of experiment directories
- Saving of intermediate and final renders
- Tracking of loss history and optimization progress
- Generation of visualization overlays for alignment quality assessment
- JSON-based tracking of all experiment parameters and results

A.1.5. Experimental Evaluation

The implementation supports systematic evaluation of different loss functions and registration parameters. Example experiments include:

```
# Experiment 1: L1 loss
inerf_optimizer = iNeRFOptimizerBatchedFD(
    experiment_name="0_065_cat5_2_l1",
    nerf_model=nerf_model,
    target_image=target_image,
    initial_pose=final_initial_pose,
    dataparser_matrix=dataparser_matrix,
    dataparser_scale=dataparser_scale,
    camera_params=camera_params,
    loss_fn=l1_loss,
    lr=0.01,
    num_iterations=50,
    config_path=config_path
)

# Experiment 2: L2 loss
inerf_optimizer = iNeRFOptimizerBatchedFD(
    experiment_name="0_065_cat5_2_l2",
    ...
    loss_fn=l2_loss,
    ...
)

# Experiment 3: SSIM loss
inerf_optimizer = iNeRFOptimizerBatchedFD(
    experiment_name="0_065_cat5_2_ssim",
    ...
    loss_fn=structural_similarity_index_loss,
    ...
)

# Additional experiments with NCC and MI loss functions
```

A.2. Hyperparameter Settings

This section details the key hyperparameters used in our neural registration framework and their effects on the registration process.

A.2.1. Optimization Hyperparameters

- **Learning Rate**: We primarily used learning rates between 0.001 and 0.01. Higher learning rates can lead to faster convergence but may cause instability, while lower rates provide more stable but slower optimization. For most experiments, a learning rate of 0.01 provided good results.
- **Number of Iterations**: Experiments were conducted with 50-1000 iterations. For evaluation purposes, 50 iterations were often sufficient to demonstrate the effectiveness of different loss functions, while longer runs (500-1000 iterations) were used for final results to ensure convergence.
- **Batch Size**: The finite difference gradient computation used batch sizes between 4 and 12. Larger batch sizes increased memory usage but significantly improved computation time by parallelizing the evaluation of perturbed poses.
- **Epsilon Value**: A value of $1e^{-4}$ was used for finite difference calculations. This represents the small perturbation applied to each parameter when computing gradients numerically.

A.2.2. Loss Function Hyperparameters

Different loss functions require specific hyperparameters:

- **Mutual Information Loss**:
  - Bins: 32 (controls the discretization of intensity values)
  - Sigma: 0.1 (kernel width for soft binning)
- **Structural Similarity Loss**:
  - Uses default parameters from the PyTorch SSIM implementation
- **Huber Loss**:
  - Beta: 0.5 (controls the transition point between L1 and L2 behavior)

A.2.3. Camera Model Parameters

Camera intrinsic parameters used in the experiments:

- **Image Resolution**: 512×512 pixels
- **Focal Length**: fl_x = fl_y = 955.4050067376327
- **Principal Point**: cx = cy = 256.0
- **Field of View**: 0.5235987755982988 radians (approximately 30 degrees)
- **Distortion Parameters**: All set to 0 (no distortion modeling)

A.2.4. Performance Considerations

The choice of hyperparameters significantly impacts both registration accuracy and computational efficiency:

- Increasing batch size from 4 to 12 resulted in approximately 3× speedup in gradient computation with minimal impact on convergence.
- The choice of loss function often had a greater impact on registration accuracy than tuning other optimization parameters. For most medical image registration scenarios, Normalized Cross-Correlation (NCC) and Structural Similarity Index (SSIM) provided better results than L1 or L2 losses due to their robustness to intensity variations.
- Optimization progress was tracked at intervals of 10-50 iterations to enable early stopping if needed, though most experiments ran to completion.
- Rendering resolution remained fixed at 512×512 for all experiments, as this provided a good balance between detail and computational efficiency.

A.2.5. Experimental Configurations

For systematic comparison of different registration approaches, we consistently used the following configuration across experiments:

- Initial pose perturbation magnitudes were consistent across all compared methods
- All methods used the same NeRF model trained with identical hyperparameters
- 5 repeated trials with different random initializations were conducted for each experimental configuration to account for optimization variability
- Target images were selected from held-out test views not used during NeRF training

The hyperparameter values listed above represent our final configuration after extensive experimentation. These settings provided a good balance between registration accuracy, convergence reliability, and computational efficiency across the datasets used in this study.

B. Glossary

Brain Shift The deformation of brain tissue during surgery due to factors such as cerebrospinal fluid drainage, gravity, and surgical manipulations, which reduces the accuracy of rigid registration methods.

Cross-Modal Registration The process of aligning images from different imaging modalities, such as preoperative MRI data with intraoperative camera images, which presents unique challenges due to differences in information content, geometric representation, and visual appearance.

Finite Difference A numerical method for computing gradients by evaluating a function at multiple perturbed points, used in this thesis to overcome limitations with gradient flow in the computational graph.

Hypernetwork A neural network that generates parameters for another neural network, used to adapt the appearance of NeRF renderings while preserving geometric structure.

iNeRF (Inverse Neural Radiance Field) A method that inverts Neural Radiance Fields for pose estimation by optimizing camera parameters through backpropagation to minimize the difference between rendered and target images.

Intraoperative Occurring during a surgical procedure, specifically referring to the registration and imaging processes that take place while surgery is being performed.

L1 Loss A loss function that calculates the absolute difference between pixels in two images, often more robust to outliers than L2 loss.

L2 Loss Also known as Mean Squared Error (MSE), a loss function that calculates the squared Euclidean distance between two images, commonly used in image registration due to its simplicity and differentiability.

Model Agnosticism The quality of an implementation that works with multiple model variants without being tied to a specific architecture, allowing for greater flexibility and comparative evaluation.

Mutual Information (MI) An information-theoretic measure that quantifies the mutual dependence between two random variables by evaluating their joint and marginal probability distributions, particularly useful for cross-modal registration where the relationship between image intensities is complex.

NeRF (Neural Radiance Field) An implicit neural representation that maps 3D coordinates and viewing directions to color and volume density, enabling novel view synthesis of complex scenes through a continuous, differentiable function optimized using volume rendering techniques.

Nerfstudio An implementation-agnostic framework for developing and deploying Neural Radiance Field models, which combines advances from various NeRF variants for improved performance.

Normalized Cross-Correlation (NCC) A similarity measure that calculates the correlation between two signals normalized by their standard deviations, making it robust to linear intensity transformations between images.

Registration The process of aligning two or more datasets into a common coordinate system, essential in neurosurgery for ensuring accurate spatial correspondence between preoperative imaging data and the patient's anatomy.

Structural Similarity Index (SSIM) A perceptual metric that quantifies image similarity based on changes in structural information, luminance, and contrast, modeled after the human visual system.

Style Transfer The process of applying the visual style of one image to the content of another image, used in cross-modal registration to bridge appearance gaps between different imaging modalities.

List of Figures

List of Tables

Bibliography

[1] N. Navab, C. Bloch, and L. Wang. "Surgical navigation systems and techniques". In: *Medical Image Computing and Computer-Assisted Intervention* (2015).

[2] F. Alam, S. Rahman, S. Ullah, and K. Gulati. "Medical image registration in image guided surgery: Issues, challenges and research opportunities". In: *Biocybernetics and Biomedical Engineering* 38 (2017), pp. 71–89. DOI: 10.1016/J.BBE.2017.10.001.

[3] B. Mildenhall, P. P. Srinivasan, M. Tancik, J. T. Barron, R. Ramamoorthi, and R. Ng. "NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis". In: *ECCV* (2020).

[4] L. Yen-Chen, P. Florence, J. T. Barron, A. Rodriguez, P. Isola, and T.-Y. Lin. "iNeRF: Inverting Neural Radiance Fields for Pose Estimation". In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2021.

[5] M. Fehrentz, M. F. Azampour, R. Dorent, H. Rasheed, C. Galvin, A. Golby, W. M. Wells, S. Frisken, N. Navab, and N. Haouchine. "Intraoperative Registration by Cross-Modal Inverse Neural Rendering". In: (2024). arXiv: 2409.11983 [cs.CV]. URL: https://arxiv.org/abs/2409.11983.

[6] J. Cho, S. Rahimpour, A. B. Cutler, C. R. Goodwin, S. Lad, and P. Codd. "Enhancing Reality: A Systematic Review of Augmented Reality in Neuronavigation and Education." In: *World neurosurgery* (2020). DOI: 10.1016/j.wneu.2020.04.043.

[7] D. H. Iversen, W. Wein, F. Lindseth, G. Unsgård, and I. Reinertsen. "Automatic Intraoperative Correction of Brain Shift for Accurate Neuronavigation." In: *World neurosurgery* 120 (2018), e1071–e1078. DOI: 10.1016/j.wneu.2018.09.012.

[8] R. Gandhe and C. Bhave. "Intraoperative magnetic resonance imaging for neurosurgery – An anaesthesiologist's challenge". In: *Indian Journal of Anaesthesia* 62 (2018), pp. 411–417. DOI: 10.4103/ija.IJA_29_18.

[9] M. Riva, P. Hiepe, M. Frommert, I. Divenuto, L. Gay, T. Sciortino, M. C. Nibali, M. Rossi, F. Pessina, and L. Bello. "Intraoperative Computed Tomography and Finite Element Modelling for Multimodal Image Fusion in Brain Surgery." In: *Operative neurosurgery* (2020). DOI: 10.1093/ons/opz196.

[10] M. Tancik, E. Weber, E. Ng, R. Li, B. Yi, T. Wang, A. Kristoffersen, J. Austin, K. Salahi, A. Ahuja, D. Mcallister, J. Kerr, and A. Kanazawa. "Nerfstudio: A Modular Framework for Neural Radiance Field Development". In: *Special Interest Group on Computer Graphics and Interactive Techniques Conference Conference Proceedings*. SIGGRAPH '23. ACM, July 2023, pp. 1–12. DOI: 10.1145/3588432.3591516. URL: http://dx.doi.org/10.1145/3588432.3591516.

[11] S.-H. Han, K. Kim, S. Kim, and Y. Youn. "Artificial Neural Network: Understanding the Basic Concepts without Mathematics". In: *Dementia and Neurocognitive Disorders* 17 (2018), pp. 83–89. DOI: 10.12779/dnd.2018.17.3.83.

[12] M. Fitzpatrick, J. B. West, and C. Maurer. "Predicting error in rigid-body point-based registration". In: *IEEE Transactions on Medical Imaging* 17 (1998), pp. 694–702. DOI: 10.1109/42.736021.

[13] M. J. Clarkson, M. J. Cardoso, G. R. Ridgway, M. Modat, K. K. Leung, J. D. Rohrer, N. C. Fox, and S. Ourselin. "A comparison of voxel and surface based cortical thickness estimation methods". In: *NeuroImage* 57.3 (2011). Special Issue: Educational Neuroscience, pp. 856–865. ISSN: 1053-8119. DOI: https://doi.org/10.1016/j.neuroimage.2011.05.053. URL: https://www.sciencedirect.com/science/article/pii/S1053811911005611.

[14] A. Klein, S. S. Ghosh, B. Avants, B. Yeo, B. Fischl, B. Fischl, B. Ardekani, B. Ardekani, J. Gee, J. Mann, and R. Parsey. "Evaluation of volume-based and surface-based brain image registration methods". In: *NeuroImage* 51 (2010), pp. 214–220. DOI: 10.1016/j.neuroimage.2010.01.091.

[15] A. S. Choe, Y. Gao, X. Li, K. B. Compton, I. Stepniewska, and A. W. Anderson. "Accuracy of image registration between MRI and light microscopy in the ex vivo brain". In: *Magnetic Resonance Imaging* 29.5 (2011), pp. 683–692. ISSN: 0730-725X. DOI: https://doi.org/10.1016/j.mri.2011.02.022. URL: https://www.sciencedirect.com/science/article/pii/S0730725X11000865.

[16] M. Unberath, C. Gao, Y. Hu, M. Judish, R. H. Taylor, M. Armand, and R. Grupp. "The Impact of Machine Learning on 2D/3D Registration for Image-Guided Interventions: A Systematic Review and Perspective". In: *Frontiers in Robotics and AI* 8 (2021). ISSN: 2296-9144. DOI: 10.3389/frobt.2021.716007. URL: https://www.frontiersin.org/journals/robotics-and-ai/articles/10.3389/frobt.2021.716007.

[17] G. Xie, Y. Huang, J. Wang, J. Lyu, F. Zheng, Y. Zheng, and Y. Jin. "Cross-modality Neuroimage Synthesis: A Survey". In: *ACM Comput. Surv.* 56.3 (Oct. 2023). ISSN: 0360-0300. DOI: 10.1145/3625227. URL: https://doi.org/10.1145/3625227.

[18] T. Müller, A. Evans, C. Schied, and A. Keller. "Instant Neural Graphics Primitives with a Multiresolution Hash Encoding". In: *ACM Transactions on Graphics* 41.4 (2022), pp. 1–15.

[19] W. Wang, W. Xie, Y. Chen, D. Wang, M. Cheng, M. Zhang, and B. Zhou. "HyperNeRF: A Higher-Dimensional Representation for Topologically Varying Neural Radiance Fields". In: *ACM Transactions on Graphics* 42.1 (2023), pp. 1–13.

[20] G. P. Penney, J. Weese, J. A. Little, P. Desmedt, D. L. Hill, and D. J. Hawkes. "Normalized Cross Correlation for Registration". In: *Medical Image Computing and Computer-Assisted Intervention* (1998), pp. 1033–1034.

[21] J. Pluim, J. Maintz, and M. Viergever. "Mutual Information for Medical Image Processing: A Review". In: *IEEE Transactions on Medical Imaging* 22.8 (2003), pp. 986–1004.