IMPLICIT POSITIONING AND MAPPING (IMAP) FOR REAL-TIME 3D SCENE RECONSTRUCTION

HARDIK SHUKLA [HARDIKSH@SEAS.UPENN.EDU], JALAJ SHUKLA [JALAJS@SEAS.UPENN.EDU], ADITYA RANGAMANI [ADIVSR@SEAS.UPENN.EDU], SAI CHAND GUBBALA [SAICHAN8@SEAS.UPENN.EDU]

ABSTRACT. Through this project, our objective is to demonstrate the viability of utilizing a multi-layer perceptron (MLP) as the underlying scene representation for a simultaneous localization and mapping (SLAM) algorithm. Our project incorporates the keyframe structure, examines the comparison between positional encoding and Gaussian encoding, and sheds light on the implementation of the algorithm on an F-1/10th car for localization purposes. We operate under the assumption of lacking prior data, with the MLP constructing a dense map dynamically as images become accessible to it.

1. INTRODUCTION

In recent years, advancements in neural network technologies have revolutionized simultaneous localization and mapping (SLAM) systems, paving the way for real-time scene understanding and navigation in dynamic environments. One such innovation, Implicit Mapping and Positioning (iMAP), introduces an approach to perform SLAM by leveraging a multilayer perceptron (MLP) as the sole scene representation. Unlike traditional methods, iMAP operates in real-time, generating a, scene-specific implicit 3D model of occupancy and color on-the-fly without the need for prior training data. By employing dynamic information-guided pixel sampling and a keyframe structure segregation, surpassing the limitations of conventional dense SLAM techniques. The project emphasises to use implement the iMAP framework, highlighting its advantages over existing methods like NeRF SLAM, which rely on offline learning and struggle with the real-time operation and scene complexity. As we delve into the objectives of this project, we aim to harness the capabilities of iMAP and provide a better way to navigate in the context of an F-1/10th car for SLAM, steering away from the constraints of Neural Radiance Field (NeRF) SLAM and presenting a solution for efficient, real-time scene understanding and navigation. The project aims to achieve this by reducing the computational time for rendering by trading off the overall quality of the formed image and reducing the number of features, further segmenting the map just in two regions i.e. the obstacles and the road where it can go.

1.1. **Contributions.** The major contributions of this project are:

(1) To monitor the six degrees of freedom (6-DoF) position and orientation of the robot relative to the origin employing a dense real-time SLAM approach utilizing solely RGB-D images as input.
(2) Implementation of active sampling and adaptive keyframe selection strategies to reduce computational resource usage.

2. BACKGROUND

2.1. **Pipeline.** The iMAP framework operates similarly to NeRF but incorporates specific modifications tailored to its objectives. Illustrated in Figure 1, the pipeline of our code implementation explains key stages. Departing from NeRF's architecture 1, we employ a multilayer perceptron (MLP) featuring four hidden layers, each having a feature size of 256, a significant modification to the eight hidden layers of NeRF. Unlike NeRF, we omit consideration of viewing directions as we are not interested in modeling the reflectance, thereby streamlining the model. To optimize computational efficiency, we experimented with positional encoding techniques which resulted in diminishing performance, prompting us to use Gaussian positional embedding. This embedding, serving as input, is concatenated to the second activation layer of the network, following a stratified and hierarchical volume sampling approach, similar to the NeRF. Further with active sampling and keyframe selections, the iMAP integrates SLAM processes, explaining a refined pathway for real-time scene understanding and navigation helping deploy the model offline.

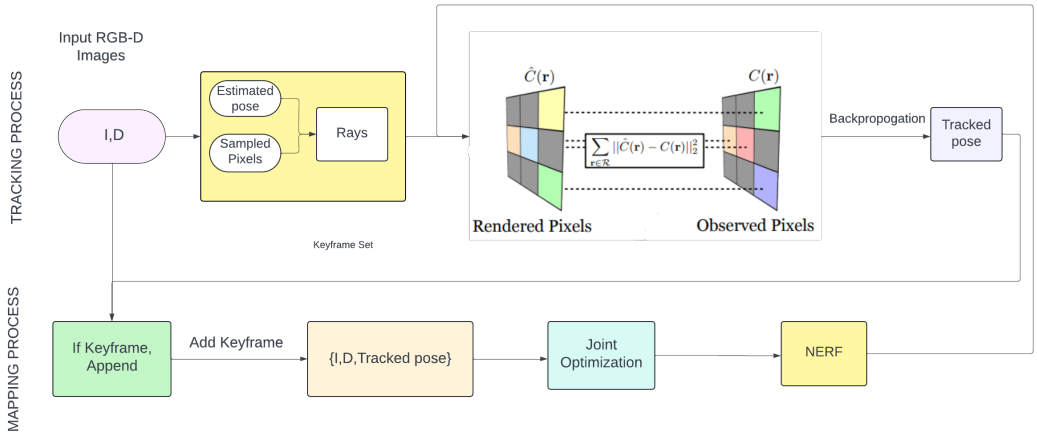

FIGURE 1. Pipeline

2.2. **Tracking Process.** To determine the robot's pose, we assume that the scene's NeRF, after performing joint optimization, with parameters Θ, has been reconstructed. However, the camera pose T corresponding to an image remains unknown. Given the parameters Θ and the image I, the pose determination is expressed as:

$$T = \text{argmin}_{T \in \text{SE}(3)} L(T|I, \Theta)$$

At each optimization step, if T_i represents the estimated camera pose, and I denotes the observed image, with $L(T_i|I, \Theta)$ being the loss function used to train the fine model in NeRF, we employ gradient-based optimization to solve for T_i. We iteratively compute the gradient $\nabla L(T_i|I, \Theta)$ of the loss function through the MLP. We utilize the functionality from NeRF to generate an image observation based on an estimated camera pose T in the coordinate frame of the NeRF model. Subsequently, we employ the same photometric loss function L as used in NeRF. However, rather than backpropagating to update the weights Θ of the MLP, we instead update the pose T to minimize L. The Adam optimizer 6 with a learning rate of 0.05 is utilized for this purpose.

2.3. **Depth and Color Rendering:** With the provided camera pose T_{w_c} and a pixel coordinate $[u, v]$, here u, v is the image pixels, we initiate the process by back-projecting a normalized viewing direction and transforming it into world coordinates using the expression $r = T_{w_c} K^{-1}[u, v]$, where K represents the camera intrinsic matrix. Subsequently, we sample along the ray $P_i = d_i r$ with d_i denoting the depth values. Transitioning to volume rendering, we convert volume density into an occupancy probability by incorporating the inter-sample distance, $\delta_i = d_{i+1} - d_i$, and then passing it through the activation function $o_i = 1 - \exp(-\rho_i \delta_i)$, where ρ is the density. The termination probability of the ray at each sample point can then be determined. This probability is calculated as $w_i = o_i \prod_{j=1}^{i-1}(1 - o_j)$, where o_i is the occupancy probability. Finally, depth and color are rendered as the expectations:

$$\hat{D}[u, v] = \sum_{i=1}^{N} w_i d_i, \quad \hat{I}[u, v] = \sum_{i=1}^{N} w_i c_i.$$

Furthermore, the depth variance along the ray can be calculated as:

$$\hat{D}_{\text{var}}[u, v] = \sum_{i=1}^{N} w_i (\hat{D}[u, v] - d_i)^2.$$

2.4. **Keyframe Selection:** Keyframe selection is a method in video processing aimed at minimizing redundancy by identifying essential frames, termed keyframes, that capture scene information. Instead of processing every frame, keyframes represent the scene sparsely. Starting with the initial frame, subsequent keyframes are added based on new information they provide. Frames are compared to existing snapshots of the scene, and if they capture significantly new regions, they become keyframes. The keyframe set evolves dynamically as new frames are processed, ensuring efficient resource utilization.

It begins by rendering a uniform set of pixel samples, denoted as s, from the frame. For each pixel (u, v) in this set, the normalized depth error is calculated as the absolute difference between the true depth $D[u, v]$ and the estimated depth $\hat{D}[u, v]$, divided by the true depth $D[u, v]$. This normalizes the depth error, allowing for comparison across different regions of the frame. The equation then sums up these normalized depth errors for all pixels in the sample set s, averaging them by dividing by the

total number of samples $|s|$. Finally, the proportion P is calculated by taking the inverse of this average, effectively measuring the fraction of the frame where the depth error is smaller than a predefined threshold t_{depth}.

$$P = \frac{1}{|s|} \sum_{(u,v)\in s} 1\frac{|D[u,v] - \hat{D}[u,v]|}{D[u,v]} < t_{depth}$$

Once this proportion P falls below a second threshold t_P (set at 0.65 in this case), the frame is deemed to provide significant new information and is consequently added to the keyframe set. This mechanism ensures that frames with substantial variations from the existing map snapshot are selected as keyframes, promoting the incremental refinement of the scene representation.

2.5. **Active sampling:** The goal of active sampling is to reduce computational and memory costs while focusing computational resources on informative areas of the scene. The three active sampling techniques used in the project are discussed below:

(1) **Image Active Sampling:** In the initial stage of each optimization iteration, a sparse pixel set is uniformly sampled from depth and color images of keyframes. These pixels update scene network parameters, camera poses, and calculate loss statistics, comprising geometric (e_{g_i}) and photometric (e_{p_i}) errors. Loss within each [8x8] grid region guides active sampling. The average loss per region is normalized into a probability distribution, favoring regions with higher losses. Samples are then re-sampled accordingly, focusing computational efforts on areas needing precise reconstruction or higher detail.

(2) **Keyframe Active Sampling:** The process involves allocating samples to each keyframe proportional to the loss distribution across keyframes. It ensures that the computational resources are focused on updating keyframes that are most relevant for scene reconstruction.As iMAP continuously optimizes the scene map using a set of selected keyframes, which acts as a memory bank to prevent network forgetting. More samples are allocated to keyframes with higher losses, indicating regions that are newly explored, highly detailed, or prone to network forgetting.

(3) **Bounded Keyframe Selection:** As the camera moves to new and unexplored regions, the keyframe set grows over time. To limit computational complexity, a fixed number of keyframes is selected at each iteration based on the loss distribution which includes the last keyframe and the current live frame in joint optimization, forming a bounded window with a constant number of keyframes.

2.6. **Joint Optimization:** It involves the simultaneous optimization of two key components: the parameters of the implicit scene network θ i.e. basically the weights and biases of the network, and the camera poses for a growing set of keyframes $\{I_i, D_i, T_i\}$. Each keyframe includes associated color I_i and depth measurements D_i along with an initial pose estimate T_i. The optimization process leverages the differentiability of the rendering function with respect to these variables. This enables iterative optimization aimed at minimizing both geometric and photometric errors for a selected number of rendered pixels in each keyframe.

The photometric loss, denoted by L_p, measures the L_1-norm between the rendered and measured color values $e_{p_i}[u,v] = I_i[u,v] - \hat{I}_i[u,v]$ for M pixel samples. The geometric loss, denoted by L_g, measures the depth difference $e_{g_i}[u,v] = D_i[u,v] - \hat{D}_i[u,v]$ and utilizes the depth variance as a normalization factor to down-weight the loss in uncertain regions such as object borders.

The joint optimization process combines both losses with a weighted sum, where the importance given to the photometric error is adjusted by a factor λ_p. The ADAM 6 optimizer is used on weighted sum of of both losses. Mathematically, the objective of the joint optimization can be expressed as:

$$\min_{\theta,\{T_i\}} (L_g + \lambda_p L_p)$$

Additionally, in the context of camera tracking, close-to-frame-rate optimization of smaller displacements is crucial for robustness. Therefore, a parallel tracking process continuously optimizes the pose of the latest frame with respect to the fixed scene network at a much higher frame rate compared to joint optimization. Without joint optimization, the scene representation and camera poses may become inconsistent, leading to drift and inaccuracies in the SLAM system. Without camera tracking, SLAM systems may struggle to accurately estimate the camera's pose, especially during rapid motion or in dynamic environments, resulting in reduced localization accuracy and increased drift over time.

3. Literature Survey

In this section, we present the literature review. Mildenhall et al. 1 established the groundwork novel view synthesis utilizing a Multi-Layer Perceptron (MLP). This approach, when implemented with dense monocular SLAM as discussed in the work done by Rosinol et al. 2, serves for localization and mapping purposes. However, this method operates offline and uses positional encoding inhibiting smooth transition and exhibiting extended computational time, thus diminishing efficiency. The emergence of Inverting Neural Radiance Fields for Pose Estimation, a work done by Yen-Chen et al. 3 introduces an avenue for analysis-by-synthesis employing NeRF for mesh-free, RGB-only six degrees of freedom (6DoF) pose estimation i.e. obtaining the translation and rotation of a camera concerning a 3D object or scene given an image. This work can be used for SLAM, as

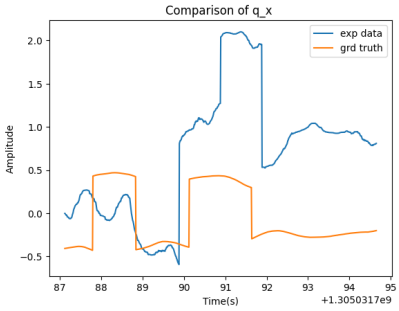

FIGURE 5. Comparison of q_x

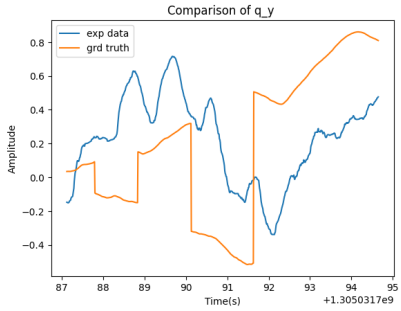

FIGURE 6. Comparison of q_y

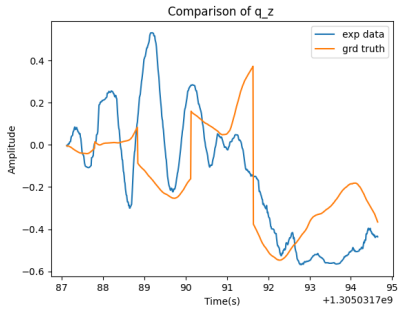

FIGURE 7. Comparison of q_z

discussed by Sucar et al. 4 in his work, demonstrating the applicability of an MLP as the sole scene representation in a real-time SLAM system, which this project investigates. Initially, we implemented positional encoding to increase the feature space, but on observing the negative impact of it due to the susceptibility to noise from positional encoding as discussed by Lin et al. 5, we used Gaussian encoding.

4. APPROACH

As per our requirement to develop an offline real-time SLAM system tailored for F1/10th race car, we opted for iMAP as it provides efficient scene representation and navigation. The approach follows the pipeline mentioned in the above section and has mentioned specifics along with the block diagram figure 1 Firstly, we optimized the parameters such as reducing the number of features and adjusting the positional encoding to better suit the dynamics of the F1/10th race car environment. Additionally, we adapted a dynamic frequency for adaptive keyframe selection to align with the speed and motion characteristics of the vehicle. Moreover, we reduced the number of features keeping it to a minimum to give the output suitable for further simple segmentation. Our approach initiates by acquiring RGBD frames as input data. These frames are then processed using a Multilayer Perceptron (MLP), serving as the backbone of the iMAP algorithm, to generate a dense, scene-specific implicit 3D model of occupancy and color. The model is continuously updated and refined as the vehicle navigates through its environment, providing real-time feedback for dynamic path planning and obstacle avoidance.

5. EXPERIMENTAL RESULTS

Given the focus of our project on localization and scene reconstruction, we have considered RGB and Depth reconstructed images, along with pose results from the MLP. These are compared with the ground truth data from the TUM dataset, specifically the 'freiburg1_rpy' sequence. This dataset offers a diverse environment conducive to training the MLP for improved outcomes. Training involved 2869 images with varied translations and rotations. Since we are using the keyframe selection using confidence the number of images that we extract from this drastically reduces to about 750 which helps in the computational efficiency of the implementation of iMAP. To assess pose accuracy, we analyzed translation vectors (x, y, z) and quaternions representing relative camera rotations compared to their initial position, totaling 7 parameters. Additionally, comparison with ground truth involved evaluating reconstructed RGB and depth images to gauge the efficacy of our iMAP algorithm implementation.

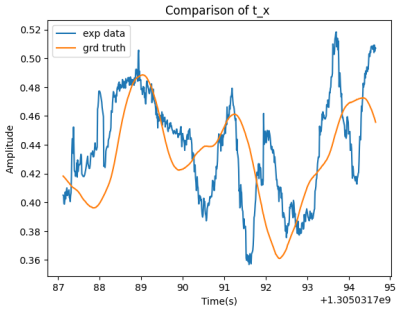

FIGURE 2. Comparison of t_x

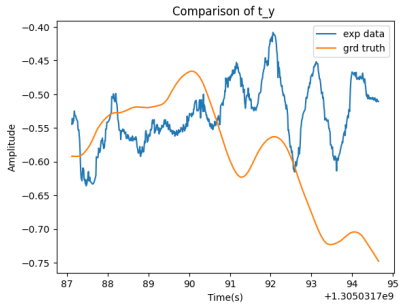

FIGURE 3. Comparison of t_y

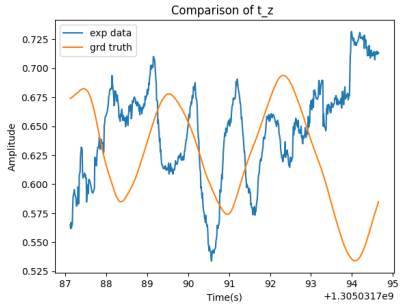

FIGURE 4. Comparison of t_z

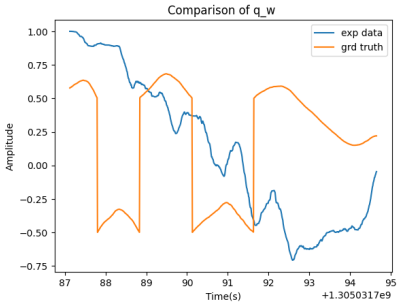

FIGURE 8. Comparison of q_z

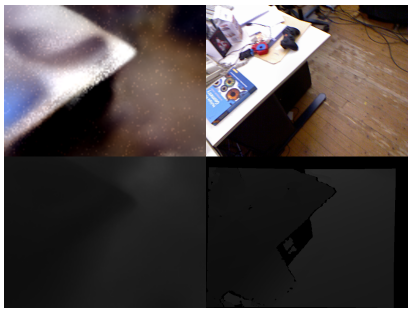

FIGURE 9. RGB-D reconstruction

FIGURE 10. RGB-D reconstruction

6. DISCUSSION

If we examine the plots comparing translation and rotations, we observe similarities in trends between the experimental and ground truth data. However, the experimental data is affected by noise, leading to somewhat inaccurate results. The comparisons between RGB and depth reconstruction images show similarity, especially in scenes with abundant features. Distortions in the graph come into play due to the MLP not being able to reproduce perfect results. A couple of other things that we tried and failed include increasing the number of layers in the MLP. Increasing the layers caused an increase in computational effects which is adverse to the real-time network we are building. We tried playing around with the input feature space post-Gaussian embedding. We found that too less or too many input features cause distortions in the reconstructed image, hence we settled on an empirical optimal value. Incorporating local MLPs, akin to NiceSLAM's implementation, could significantly enhance reconstruction performance compared to iMAP. Additionally, adapting active sampling based on the car's speed could focus sampling efforts on crucial frames. Dynamic thresholding for keyframe selection might further optimize computation by selecting fewer frames based on the thresholding algorithm. While deploying this algorithm on an F1/10th car represents the next step, time constraints prevented its implementation. Furthermore, segmenting the scene into traversable and non-traversable areas could aid in navigation.

REFERENCES

(1) Mildenhall, B., Srinivasan, P.P., Tancik, M., Barron, J.T., Ramamoorthi, R. and Ng, R., 2021. Nerf: Representing scenes as neural radiance fields for view synthesis. Communications of the ACM, 65(1), pp.99-106.
(2) Rosinol, A., Leonard, J.J. and Carlone, L., 2022. Nerf-slam: Real-time dense monocular slam with neural radiance fields. In 2023 IEEE. In RSJ International Conference on Intelligent Robots and Systems (IROS) (pp. 3437-3444).
(3) Yen-Chen, L., Florence, P., Barron, J.T., Rodriguez, A., Isola, P. and Lin, T.Y., inerf: Inverting neural radiance fields for pose estimation. In 2021 IEEE. In RSJ International Conference on Intelligent Robots and Systems (IROS) (pp. 1323-1330).
(4) Sucar, E., Liu, S., Ortiz, J. and Davison, A.J., 2021. imap: Implicit mapping and positioning in real-time. In Proceedings of the IEEE/CVF International Conference on Computer Vision (pp. 6229-6238).
(5) Lin, C.H., Ma, W.C., Torralba, A. and Lucey, S., 2021. Barf: Bundle-adjusting neural radiance fields. In Proceedings of the IEEE/CVF International Conference on Computer Vision (pp. 5741-5751).
(6) Diederik P. Kingma and Jimmy Ba. ADAM: A method for stochastic optimization. In Proceedings of the International Conference on Learning Representations (ICLR), 2015