2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
October 14-18, 2024. Abu Dhabi, UAE

Representing 3D sparse map points and lines for camera relocalization

Bach-Thuan Bui[1], Huy-Hoang Bui[1], Dinh-Tuan Tran[2], and Joo-Ho Lee[2]

***Abstract*— Recent advancements in visual localization and mapping have demonstrated considerable success in integrating point and line features. However, expanding the localization framework to include additional mapping components frequently results in increased demand for memory and computational resources dedicated to matching tasks. In this study, we show how a lightweight neural network can learn to represent both 3D point and line features, and exhibit leading pose accuracy by harnessing the power of multiple learned mappings. Specifically, we utilize a single transformer block to encode line features, effectively transforming them into distinctive point-like descriptors. Subsequently, we treat these point and line descriptor sets as distinct yet interconnected feature sets. Through the integration of self- and cross-attention within several graph layers, our method effectively refines each feature before regressing 3D maps using two simple MLPs. In comprehensive experiments, our indoor localization findings surpass those of Hloc and Limap across both point-based and line-assisted configurations. Moreover, in outdoor scenarios, our method secures a significant lead, marking the most considerable enhancement over state-of-the-art learning-based methodologies. The source code and demo videos of this work are publicly available at: https://thpjp.github.io/pl2map/.**

I. INTRODUCTION

Owing to the cost-effectiveness and abundant texture resources of visual features, their application in localization and mapping has gained considerable traction in the realms of robotics and computer vision. In contrast to mere point-based methods, incorporating line-assisted features offers a deeper understanding of scene layouts and geometric cues, paving the way for more versatile and efficient applications.

Recent studies have shown that simultaneous localization and mapping (SLAM) or structure from motion (SfM) performance can be enhanced by integrating both points and lines [1], [2], [3], [4], [5]. However, localization based on maps pre-built using SLAM or SfM-based methods often requires huge computational resources for feature matching (FM) between local images and global maps [6], [2]. This process requires the storage of pre-built 3D maps [6], [7], including detailed descriptor components, which proves to be prohibitively costly for real-time applications, particularly in the context of applications with lightweight robotics platforms.

Therefore, in this study, we introduce PL2Map, a novel neural network tailored for the efficient representation of complex point and line maps. This methodology naturally provides 2D-3D correspondences, simplifying the relocalization task by foregoing the need for expensive feature matching and descriptor management.

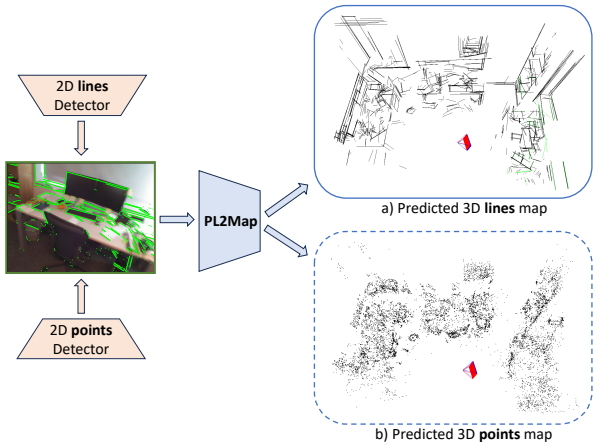


Fig. 1: **Representing 3D point-line maps by PL2Map**. We show an example of the results of the proposed learning method for representing 3D point-line features. The red camera poses in both predicted lines (a) and points (b) map are the ground truth poses of the input image on the left, and the blue ones are the estimated camera poses using predicted lines or points map.

Our method aims to map sparse descriptors directly to 3D coordinates using a neural network, which can encounter several challenges. The primary issue is the variability in the number of points and lines, which fluctuates with changes in the viewpoint. Furthermore, the sequence in which points and lines are arranged is inconsistent, and there is always variability in line lengths [8], [9]. This introduces additional complexity in accurately mapping 2D sparse points and lines to 3D space compared with more uniform approaches [10], [11], [12], [13]

To overcome the aforementioned challenges, we first drew inspiration from the principles of feature matchers [7], [14], treating points and lines as two distinct yet interrelated sets of unordered descriptors. To account for variations in line lengths and ensure their unique features, we adopted a strategy inspired by [8], conceptualizing lines as sequences of words, with each word representing an intra-point descriptor. We then used a transformer encoder model [15] to encode each line sentence as a unique point-like descriptor, thereby streamlining the line descriptor extraction process within the PL2Map's preprocess, allowing for a shared extractor for both points and lines. Subsequently, we utilized self and cross-attention mechanisms within several graph layers to

[1]Graduate School of Information Science and Engineering, Ritsumeikan University, Japan.
[2]College of Information Science and Engineering, Ritsumeikan University, Japan.

facilitate the exchange and refinement of feature descriptors. Following this attention-driven update, the points and line features were split into two separate Multilayer Perceptrons (MLPs) to regress their respective 3D coordinates. The contributions of this study are as follows.

- To the best of our knowledge, our method of direct learning in mapping 2D-3D correspondences for point and line features represents this first attempt at camera relocalization.
- We propose a complete learning pipeline including network architecture, and robust loss functions for learning to represent both points and lines from pre-built SfM models. Through the proposed end-to-end training pipeline, the maps of points and lines can be further refined, leading to improvements in subsequent camera relocalization.
- We set a new record on two localization benchmarks of the 7scenes [16] and Cambridge Landmarks [17], in which, our PL2Map surpasses both FM-based Hloc and Limap performance on 7scenes. For outdoor Cambridge Landmarks, our pipeline marks the most significant enhancement over state-of-the-art learning-based approaches.

II. RELATED WORK

A. Image Retrieval and Pose Regression

Image retrieval-based methods relocalize by simply comparing a query image with posed images in the database [18] and approximating the pose of the query with the highest-ranking retrieved images [19]. Pose regression employs a neural network to predict the absolute camera pose from a query image or the relative pose between a query and retrieved images [17], [20], [13], [19], [21]. However, both image retrieval and pose regression-based methods exhibit lower accuracy than the image structure-based methods discussed subsequently.

B. Feature Matching

Feature-matching-based (FM) approaches are typically divided into direct [22], [23] and indirect methods [6], [2], [24]. Direct methods employ a strategy of matching 2D-3D correspondences directly from query image features to 3D points in SfM models. Although direct approaches can yield accurate camera poses, they are limited in scalability to larger scenes because of memory consumption and ambiguities [6]. Conversely, indirect approaches begin with an image retrieval step against images in the database [6], [2] and then match the features from the query image to those in the retrieved images. This creates a very robust pipeline under challenging conditions [7], [14].

Traditionally, FM-based methods rely solely on point features for relocalization. However, recent advancements suggest that incorporating line features can significantly enhance this pipeline [24], [2], [1]. However, the introduction of additional features increases the complexity and cost of FM-based pipelines owing to the additional matching steps required for line features [25], [8], [9], and additional memory is required for storing line descriptors [2].

C. Learning Surrogate Maps

Another prominent category within relocalization methodologies is scene coordinate regression (SCR) [16], [10], [13], [11]. These techniques infer 3D coordinates within the scene space from dense 2D pixel positions in the images, effectively embedding map representations within the neural network's weights. This approach offers notable benefits, including minimal storage requirements [11], [26] and enhanced privacy due to the implicit nature of the map representation [27]. Nevertheless, recent learning-based approaches [28], [29], [30] have demonstrated that focusing on key landmarks can improve relocalization accuracy and robustness to environmental changes [28], [29]. Specifically, the sparse map learning approach detailed in [28] has shown superior performance compared with dense SCR [13], [10], particularly in scenarios characterized by significant domain shifts or limited training data availability.

Our method aligns closely with this innovative trend as we endeavor to create neural-based surrogate maps that incorporate both point and line features. This, results in significant enhancement of the camera relocalization task.

III. PROPOSED METHOD

A. Problem Statements

Recent advancements in SfM and visual SLAM have been explored with many successful mapping elements, such as points, lines, edges, planes, and objects [1], [5], [31], [2], [3], [32], [33]. With the demand for additional mapping elements, there is a clear need for a more efficient mapping representation strategy that extends beyond the basic storage of descriptor vectors [7], [6], [2]. To address this issue, specifically for point and line maps, we introduce a neural-based surrogate model capable of representing both 3D points and lines through their descriptors. This simplifies the matching process for multiple mapping elements.

Assume that we have a set of 2D keypoints $\{\mathbf{p}_i\}^N$ and a set of 2D line segments $\{\mathbf{l}_i\}^M$ extracted from image $\mathbf{I}^r$, each associated with visual descriptors $\{\mathbf{d}_i^p\}^N$ and $\{\mathbf{d}_i^l\}^M$ respectively. Here, r denotes the image sourced from the reference database used to construct the 3D points and line map. We aim to develop a learning function $\mathcal{F}(.)$ that inputs the two sets of *visual descriptors* $\{\mathbf{d}_i^p\}^N$ and $\{\mathbf{d}_i^l\}^M$, and outputs the corresponding sets of 3D points $\{\mathbf{P}_i \in \mathbb{R}^3\}^N$ and lines $\{\mathbf{L}_i \in \mathbb{R}^6\}^M$ sets in the world coordinates system. The ultimate goal is to estimate a six degrees of freedom (6 DOF) camera pose $\mathbf{T} \in \mathbb{R}^{4\times4}$ for any new query image $\mathbf{I}$ from the same environment.

B. PL2Map

This section presents in detail the PL2Map model designed to learn the representation of the sparse 2D-3D correspondences for both points and lines. Both points and lines possess interchangeable features, such as line endpoints and adjacent points, which can be integrated to enhance the

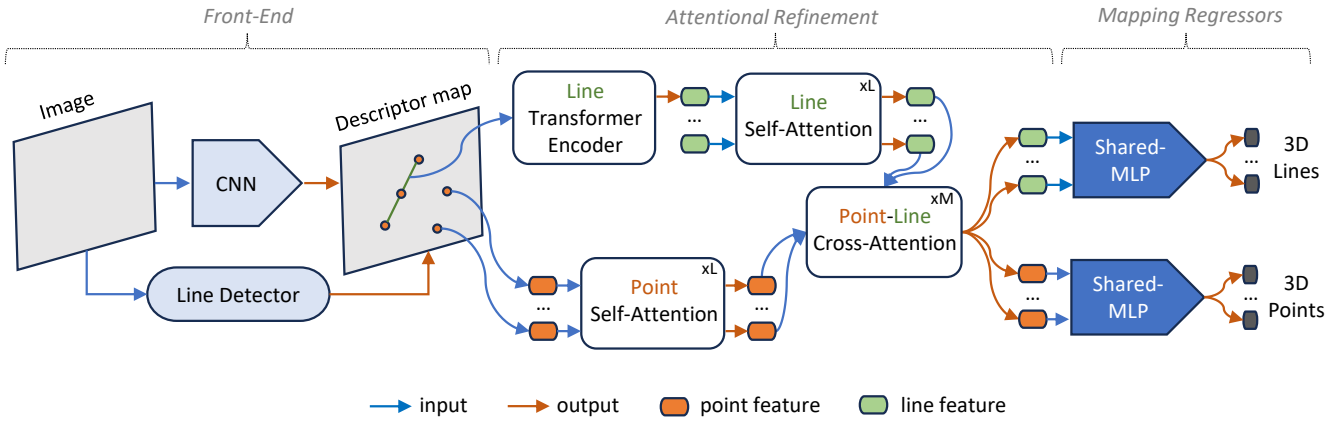


Fig. 2: **PL2Map pipeline**. We illustrate the architecture of PL2Map, which consists of three main components: *Front-End*, *Attentional Refinement*, and *Mapping Regressors*.

development and accuracy of the final 3D map [25]. Fig. 2 shows the proposed architecture including three sub-blocks.

- *Front-End*: We use the available 2D detectors to extract both the point and line positions and their descriptors from the input image.
- *Attentional Refinement*: Points and line features/descriptors are boosted by a subsequent attentional refinement module, targeting the awareness of the surrounding point and line features [28].
- *Mapping Regressors*: The last module consists of two MLPs that are used to regress the 3D map lines and points.

1) Front-End: The proposed system inputs the available 2D point and line descriptor sets of $\{\mathbf{d}^p\}^N$ and $\{\mathbf{d}^l\}^M$, extracted from the images. To gather these inputs, we rely on off-the-shelf 2D point and line detectors, either hand-crafted strategies or learning-based approaches, such as SIFT [34], SuperPoint [35], LSD [36], and DeepLSD [37].

For the descriptor features of the points, we utilized the direct results produced by the extractors. For lines, rather than employing separate line descriptors, we opted to use point descriptors to represent the lines. This approach is more convenient and cost-effective for subsequent inference processes. To achieve this, we uniformly sampled T point descriptors to represent a 2D line, which subsequently served as the input for the attentional refinement module. This sampling process is shown on the left-hand side of Fig. 3.

2) Attentional Refinement: Attentional refinement is a key component of our method and is comprised of three submodules: Line Transformer Encoders, Self-Attention, and Cross-Attention. Each submodule is specifically designed to augment the features of lines and points by utilizing descriptor similarity.

Line Transformer Encoder. Because the point and line features extracted from the *front-end* module have different dimensions, we initially employed a transformer-based encoder [15] to align the dimensions of the line descriptors similar to those of the points. The process for encoding line descriptions is shown in Fig. 3. In our approach, a line

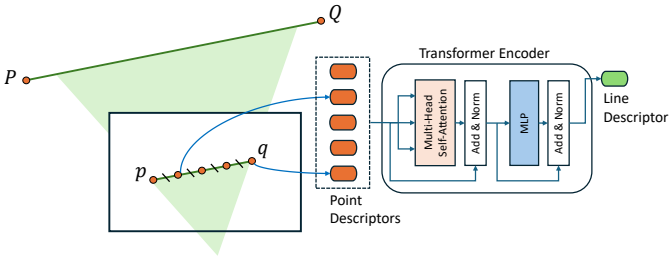


Fig. 3: **Line transformer encoder**. We represent a 2D local line by uniformly sampling $T-2$ number of points inside the line segment of endpoints p and q. A transformer-based model is then used to uniformly transform all point descriptors to a single feature with the same dimension, which can be considered as a line descriptor.

segment is characterized by two endpoints, p and q. Similar to [8], we uniformly sampled $T-2$ interval point tokens and their descriptors for a line segment. This process yields an embedded line token $\mathbf{L} \in \mathbb{R}^{T \times D}$, where D represents the descriptor dimension, which is consistent with that of the point descriptors.

Unlike the approach in [8], we did not incorporate the position and orientation of the line in our method. Instead, the input to the transformer model consisted solely of point descriptors, as illustrated in Fig.3. This decision was based on the observation that a line's appearance in terms of position and orientation can vary significantly with changes in the camera view, whereas its 3D position remains constant in the world coordinate system. Previous studies, such as [28], demonstrated that including positional factors in 3D regression modules can lead to suboptimal performance in mapping regression. We assume that the descriptors of interval points along a corresponding segment follow a similar pattern. The 3D coordinates (PQ) of the 2D line (pq), as shown in Fig. 3, are also illustrated with different reprojection lengths but can be represented by using only 2D sampled points.

To embed the line tokens of $\mathbf{L}$ to have the same dimensions

as the point descriptor, we use a single transformer model $\mathfrak{T}$, which can be written as follows:

$$\mathfrak{T}(\mathbf{L}) = \mathbf{d}^l, \tag{1}$$

where $\mathfrak{T} : \mathbb{R}^{N\times D} \rightarrow \mathbb{R}^{1\times D}$. The architecture of $\mathfrak{T}$ is shown in Fig. 3, which consists of two main components: a multi-head self-attention layer and an MLP layer, whereas each sub-layer is appended by a residual connection and layer normalization.

Self and Cross Attention. Similar to previous studies [7], [28], [25], we also consider the attention module as a complete graph with two types of undirected edges. The self-attention edge $\mathcal{E}_{self}$ connects all the surrounding descriptors of either points or lines in the same image, whereas the cross-attention edge $\mathcal{E}_{cross}$ connects points to lines and lines to points.

Let ${}^{(m)}\mathbf{d}_i$ be the intermediate descriptor for element i in layer m. We initialized ${}^{(0)}\mathbf{d}_i = \mathbf{d}_i$. Then, the residual update for all i is:

$${}^{(m+1)}\mathbf{d}_i = {}^{(m)}\mathbf{d}_i + \phi_m\left(\left[{}^{(m)}\mathbf{d}_i || a_m\left({}^{(m)}\mathbf{d}_i, \mathcal{E}\right)\right]\right) \tag{2}$$

where $\mathcal{E} \in \{\mathcal{E}_{self}, \mathcal{E}_{cross}\}$, $[.||.]$ denotes the concatenation, ϕ_m is modeled with an MLP, and $a_m\left({}^{(m)}\mathbf{d}_i; \mathcal{E}\right)$ is the Multi-Head Attention from [15] applied to the set of edges $\mathcal{E}$, which is calculated as follows:

$$a_m({}^{(m)}\mathbf{d}_i, \mathcal{E}) = \sum_{j:(i,j)\in\mathcal{E}} \alpha_{ij}\mathbf{v}_j \tag{3}$$

where $\alpha_{ij} = \text{Softmax}_j\left(\mathbf{q}_i^T\mathbf{k}_j/\sqrt{D/h}\right)$ is the attention score, query $\mathbf{q}_i$ key $\mathbf{k}_j$, and value $\mathbf{v}_j$ are the linear projections of descriptors $\mathbf{d}_i$ and $\mathbf{d}_j$, and h is the number of heads. In self-attention $\mathbf{k}_j$ and $\mathbf{v}_j$ come from the same descriptor set with $\mathbf{q}_i$ (either points or lines), whereas in cross-attention, if $\mathbf{q}_i$ comes from the points set, $\mathbf{k}_j$ and $\mathbf{v}_j$ will be calculated using the line descriptors set, and vice versa.

3) Mapping Regressors: Finally, we use two different MLP networks to regress the 3D coordinates of the points and lines. The models input the *fine descriptors* resulting from the attentional module as follows:

$$\hat{\mathbf{P}}_i = \phi_p\left({}^{(m)}\mathbf{d}^p\right) \tag{4}$$

$$\hat{\mathbf{L}}_i = \phi_l\left({}^{(m)}\mathbf{d}^l\right) \tag{5}$$

where $\phi_p : \mathbb{R}^D \rightarrow \mathbb{R}^4$ and $\phi_l : \mathbb{R}^D \rightarrow \mathbb{R}^7$ are shared by all descriptors in the same set (points or lines). Because the number of triangulated 3D points and lines is different from the 2D points and lines detected from every image, we extend one more dimension for reliability prediction of $\hat{\mathbf{P}}_i$ and $\hat{\mathbf{L}}_i$, which is similarly defined in [28], as follows:

$$\hat{r} = \frac{1}{1+|\beta z|} \in (0, 1], \tag{6}$$

where z is the last value of $\hat{\mathbf{P}}_i = (\hat{\mathbf{P}}_i^{'}, z)^T$ or $\hat{\mathbf{L}}_i = (\hat{\mathbf{L}}_i^{'}, z)^T$, β is a scale factor chosen to make the expected reliability r easy to reach a small value when input descriptors have no 3D coordinates.

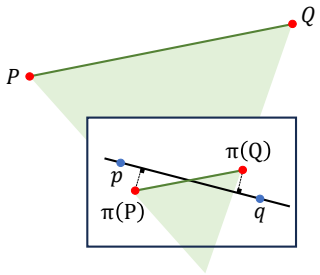


Fig. 4: **Line reprojection loss**. Given two 2D endpoints p and q, and their predictions of 3D endpoints P and Q, we minimize the reprojection distance of $\pi(P)$ and $\pi(Q)$ to the 2D segment pq on the image plane. This allows the length of PQ in 3D space independent with 2D segment pq length, which can also solve the occlusion problem in the camera view.

C. Loss Function

The predicted $\hat{\mathbf{P}}_i$ and $\hat{\mathbf{L}}_i$ are optimized using their pseudo ground truths $\mathbf{P}_i$ and $\mathbf{L}_i$ from 3D models for each image as follows:

$$\mathcal{L}_m = \sum_{i=1}^{N} r_i^p \|\mathbf{P}_i - \hat{\mathbf{P}}_i^{'}\|_\gamma + \sum_{i=1}^{M} r_i^l \|\mathbf{L}_i - \hat{\mathbf{L}}_i^{'}\|_\gamma, \tag{7}$$

where $r \in \{0, 1\}$, γ is a robust norm. Because the loss Eq. 7 only focuses on regressing valid points and lines, we optimize the reliability prediction for non-robust descriptors as follows:

$$\mathcal{L}_r = \sum_{i=1}^{N} \|r_i^p - \hat{r}_i^p\|_\gamma + \sum_{i=1}^{M} \|r_i^l - \hat{r}_i^l\|_\gamma. \tag{8}$$

where $\hat{r}_i$ is calculated using Eq. 6 for both points and lines. In Fig. 5, we show an example of reliability prediction results. Furthermore, we optimize the model using the available camera poses by reprojecting the predicted 3D points and lines onto the image plane:

$$\begin{aligned}\mathcal{L}_\pi = &\sum_{i=1}^{N} v_i^p r_i^p \left\|\pi(\mathbf{T}, \mathbf{P}_i) - \mathbf{u}_i^p\right\|_\gamma \\ &+ \sum_{i=1}^{M} v_i^l r_i^l \psi\left(\pi(\mathbf{T}, \mathbf{L}_i), \mathbf{u}_i^l\right),\end{aligned} \tag{9}$$

where $\mathbf{T}$ is the ground truth pose, $\pi(.)$ is the reprojection function, $\mathbf{u}_i^p \in \mathbb{R}^2$ and $\mathbf{u}_i^l \in \mathbb{R}^4$ are the 2D positions of the point and line endpoints on the image, $\psi(.)$ is a function that calculates the distance between reprojected 3D lines and its ground truth coordinates $\mathbf{u}_i^l$, as illustrated in Fig. 4, and $v_i \in \{0, 1\}$, $v_i = 1$ when a prediction is between 10 cm and 1000 m in front of the camera, and has projection error lower than 1000 px, and otherwise $v_i = 0$.

However, the reprojection loss, as illustrated in Eq. 9, is still highly non-convex and difficult to train at the beginning stage. Thus, we adapt the robust projection error defined in [11] as follows:

$$\mathcal{L}_{\pi}^{robust} = \tau(t) tanh\left(\frac{\mathcal{L}_{\pi}}{\tau(t)}\right), \tag{10}$$

where $\tau(.)$ is the threshold used to dynamically rescale $tanh(.)$, and varies throughout the training:

$$\tau(t) = \omega(t)\tau_{max} + \tau_{min}, \text{ with } \omega(t) = \sqrt{1 - t^2}, \tag{11}$$

where $t \in (0, 1)$ denotes the relative training progress. This forces the threshold τ to have a circular schedule that remains close τ_{max} at the beginning and reaches τ_{min} at the end of the training.

Finally, we integrate all loss functions to optimize the surrogate model as follows:

$$\mathcal{L} = \delta_m \mathcal{L}_m + \delta_r \mathcal{L}_r + \delta_{\pi} \mathcal{L}_{\pi}^{robust}, \tag{12}$$

where δ is the hyperparameter coefficient used to balance three loss functions.

IV. EXPERIMENTS

Network Setting. We implemented our approach in Pytorch [38], with the following settings for the network architecture. The graph attention consists of a single line-transformer model for all lines that receive $T = 12$ point descriptors including two endpoints, and five graph attention layers of (*self, cross, self, cross, self*). We used the same number of attention heads for both the transformer-encoder and self-cross-attention modules. For the final mapping layers, two different MLPs (D, 512, 1024, 512, 4) and (D, 512, 1024, 512, 7) are used to regress the 3D points and lines.

Hyperparameters Choices. We chose $\beta = 100$ and initialized the soft threshold $\tau_{max} = 50$px for training with the indoor dataset, $\tau_{max} = 100$px for the outdoor dataset, and $\tau_{min} = 1$px. The hyperparameters δ in Eq. 12 were selected as one for all three sub-losses. We optimized our method using Adam [39] with a start learning rate of $3 \cdot 10^{-4}$ and shrank it seven times with a decay parameter of 0.5, for the indoor dataset. For outdoors, we use a smaller learning rate of $5 \cdot 10^{-5}$ and shrank it ten times. We then trained all the environments with 2.5M iterations.

We applied data augmentation to all the experiments. Specifically, we applied random adjustments to the brightness $\pm 15\%$ and contrast $\pm 10\%$ of the input image. We further randomly rotated the image $\pm 30^{\circ}$ and re-scaled the images within 66 and 150 percent. For each applied augmentation, we adjusted the camera poses and focal lengths according to the changes in the images. All experiments were performed using an Nvidia GeForce GTX 1080ti GPU and Intel Core i9-9900 CPU.

Camera Pose Estimation. Given the predicted points and line maps, we estimated the camera poses using two different settings, one using points only, and another using both points and lines for visual localization. For points, we directly used RANSAC PnP implemented by Hloc [6]. For points and lines-assisted, we used the same mechanism as in Limap [2].

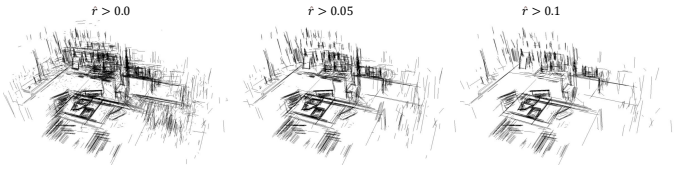


Fig. 5: **Reliable Line-Map Prediction Results.** We show predicted line-map filtering with a different threshold $\hat{r}$ in RedKitchen scene from 7scenes [16]

A. Datasets and 3D Ground Truth Models

We conducted our experiments on two standard camera re-localization datasets, both indoor and outdoor:

7Scenes [16]: An RGB-D dataset consists of seven different small environments, scaled up to $18m^3$. For each scene, the authors provided several thousand images for training and testing. This dataset features challenges such as repeating structures of the Stair scene or motion blurs that occur throughout the dataset. Images were recorded using KinectFusion [40] which also provides ground-truth poses and depth channels.

Cambridge Landmarks [17]: An RGB outdoor dataset scaling from $875m^2$ to $8000m^2$. The dataset consists of five scenes, including several hundred frames for the training and test sets, split by the authors. Ground-truth camera poses were obtained by reconstructing the SfM models.

SfM ground truth models. The main purpose of our method is to represent multiple 3D sparse models as a neural network for later localization with high-accuracy constraints and low storage demand. We leveraged two SfM tools, point-based SfM Hloc [6], [7] and line-based SfM Limap [2] to obtain these models. Hloc and Limap are currently the most robust localization methods that utilize points and line maps. However, they are burdened by large memory requirements and complex feature-matching processes, posing challenges for real-time applications and lower-end consumer devices. This creates a large gap between real-time and low-grade consumer application. The proposed method addresses this burden. For all experiments, we used Superpoint [35] and DeepLSD [37] to extract the 2D points and lines.

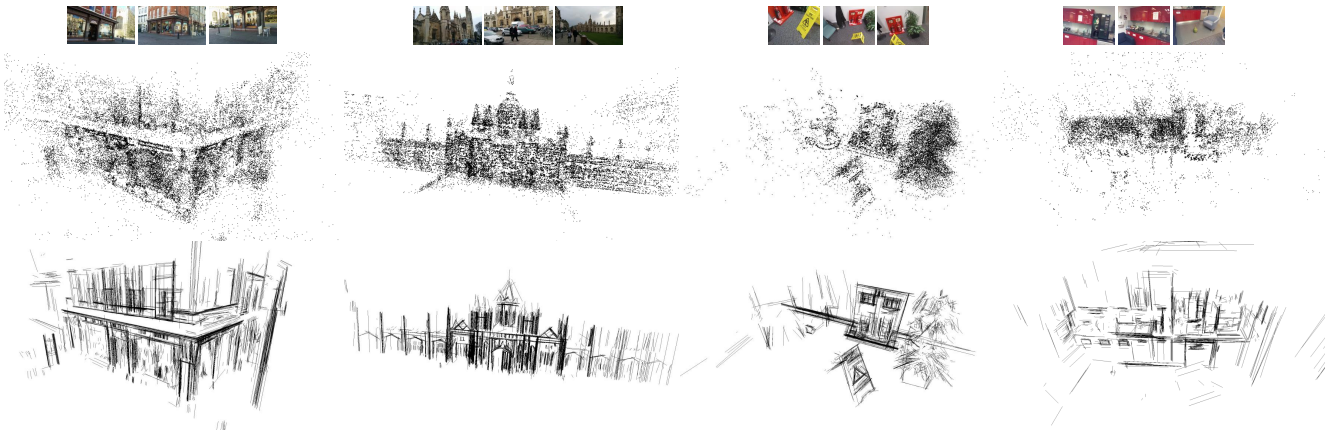
B. Indoor Re-Mapping and Localization

In this section, we report our localization results for learning indoor sparse point and line maps representation. We compare our results with those obtained from state-of-the-art baseline methods, as detailed in Table I. In Fig. 6, we show some example results of 3D points and lines mapped by PL2Map.

We compare our indoor localization results against three established baselines: Hloc [6], PtLine [24], and Limap [2]. These were chosen because of their reliance on similar mapping components, specifically, points and lines. Regarding

TABLE I: **Localization results on 7scenes [16]**. We report the median translation and rotation errors in cm and degrees and pose accuracy (%) at 5 cm / 5 deg. threshold of different relocalization methods using points and lines on the *7scenes* dataset. The methods marked with ⋆ are FM-based or database-based methods. The results in red are the best and blue indicates the second best.

	⋆Hloc $^{\text{point}}$ [6], [7]	⋆PtLine $^{\text{point \& line}}$ [24]	⋆Limap $^{\text{point \& line}}$ [2]	Pl2Map $^{\text{point}}$ **(ours)**	Pl2Map $^{\text{point \& line}}$ **(ours)**
Chess	2.4 / 0.84 / 93.0	2.4 / 0.85 / 92.7	2.5 / 0.85 / 92.3	2.0 / 0.65 / 95.5	1.9 / 0.63 / 96.0
Fire	2.3 / 0.89 / 88.9	2.3 / 0.91 / 87.9	2.1 / 0.84 / 95.5	2.0 / 0.81 / 93.3	1.9 / 0.80 / 94.0
Heads	1.1 / 0.75 / 95.9	1.2 / 0.81 / 95.2	1.1 / 0.76 / 95.9	1.2 / 0.74 / 97.8	1.1 / 0.71 / 98.2
Office	3.1 / 0.91 / 77.0	3.2 / 0.96 / 74.5	3.0 / 0.89 / 78.4	2.8 / 0.78 / 82.3	2.7 / 0.74 / 84.3
Pumpkin	5.0 / 1.32 / 50.4	5.1 / 1.35 / 49.0	4.7 / 1.23 / 52.9	3.5 / 0.96 / 63.1	3.4 / 0.93 / 64.1
RedKitchen	4.2 / 1.39 / 58.9	4.3 / 1.42 / 58.0	4.1 / 1.39 / 60.2	3.8 / 1.13 / 66.7	3.7 / 1.10 / 68.9
Stairs	5.2 / 1.46 / 46.8	4.8 / 1.33 / 51.9	3.7 / 1.02 / 71.1	8.5 / 2.4 / 27.8	7.6 / 2.0 / 33.3



a) ShopFacade b) KingsCollege c) Fire d) Pumpkin

Fig. 6: Qualitative results on both outdoor and indoor scenes (a, b from the Cambridge dataset [17] and c, d from the 7scenes dataset [16]). Three different views are shown on the top. The second and third rows are our prediction results of 3D point and line maps respectively, using a random number of 20 test images.

TABLE II: **Localization results on 7scenes [16] with depth map labels.** We report the localization results on 7scenes when depth is available to refine the SfM models. The results are presented in cm, degree, and % accuracy as in Table I.

		points	points+lines
without depth	Stairs	8.5 / 2.4 / 27.8	7.6 / 2.0 / 33.3
	Avg. all scenes	3.4 / 1.07 / 75.4	3.2 / 0.99 / 77.0
with depth	Stairs	8.4 / 2.5 / 35.7	4.7 / 1.24 / 53.0
	Avg. all scenes	3.2 / 1.05 / 79.6	2.6 / 0.85 / 83.4

the use of only point maps, our approach demonstrated superior performance over Hloc in six out of seven evaluated scenes. For instance, within the Pumpkin scene, our method achieved localization errors of 3.5 cm in translation, 0.96° in rotation, and attained an accuracy of 63.1 % (using a threshold of 5 cm / 5 deg). In contrast, Hloc exhibited less favorable results, with localization errors of 5.0 cm in translation, 1.32° of localization errors, and 50.4% accuracy. These findings underscore the efficacy of our surrogate mapping model, which yields a performance improvement of 12.7% over that of Hloc's database-based mapping approach.

Line-assisted localization. In Table I, we present our localization results using both the predicted points and line map. As can be seen, we achieved the best results when localizing with lines-assisted. These results confirm the efficiency of lines combined with points for visual localization, where an improvement can be observed for all seven scenes. Interestingly, our method learned using points and lines SfM models produced by Hloc and Limap, but the localization results obtained by our re-mapping method can even reach a large improvement margin.

Additionally, we provide localization results with available depth to refine the SfM models in both point and line maps. The results are listed in Table II. The original PL2Map still struggles with the repeated structure of the Stairs scene, but with SfM models refined using depth data, PL2Map shows a significant improvement in Stairs scene accuracy from 33.3% to 53.0%. The overall improvement was also observed for all scenes, evidenced in average median errors. Note that depth is used solely for training and inference relies solely on RGB images.

C. Outdoor Re-Mapping and Localization

In this section, we present the localization results of the proposed method using the Cambridge outdoor dataset [17]. Unlike small-scale indoor environments, learning and SCR-based methods [10], [28], [11] still struggle to close the

TABLE III: **Localization results on the Cambridge Landmarks dataset [17].** We present the median translation and rotation errors in cm and degrees and pose accuracy (%) at 5 cm / 5 deg. threshold of different relocalization methods using points and lines. The results in red are the best and blue represents the second best in the same category of learning-based mapping method. The best overall results are in **bold**.

		Great Court	King's College	Old Hospital	Shop Facade	St. Mary's Church
FM-based	Hloc $^{\text{point}}$ [6], [7]	**9.5 / 0.05 / 20.4**	6.4 / **0.10** / 37.0	12.5 / 0.23 / 22.5	2.9 / 0.14 / 78.6	**3.7** / 0.13 / 61.7
	PtLine $^{\text{point \& line}}$ [24]	11.2 / 0.07 / 17.8	6.5 / **0.10** / 37.0	12.7 / 0.24 / 20.9	**2.7 / 0.12** / 79.6	4.1 / 0.13 / 62.3
	Limap $^{\text{point \& line}}$ [2]	9.6 / 0.05 / 20.3	**6.2 / 0.10 / 39.4**	**11.3 / 0.22 / 25.4**	**2.7** / 0.13 / **81.6**	**3.7 / 0.12 / 63.8**
Learning-based	SANet $^{\text{dense}}$ [12]	328 / 2.0 / -	32 / 0.5 / -	32 / 0.5 / -	10 / 0.5 / -	16 / 0.6 / -
	DSAC* $^{\text{dense}}$ [10]	49 / 0.3 / -	15 / 0.3 / -	21 / 0.4 / -	5 / 0.3 / -	13 / 0.4 / -
	ACE $^{\text{dense}}$ [11]	43 / 0.2 / -	28 / 0.4 / -	31 / 0.6 / -	5 / 0.3 / -	18 / 0.6 / -
	D2S $^{\text{point}}$ [28]	38 / 0.18 / -	15 / 0.24 / -	21 / 0.40 / -	6 / 0.32 / -	16 / 0.50 / -
	PL2Map $^{\text{point}}$ (**ours**)	33.0 / 0.16 / 2.51	7.3 / 0.13 / 29.4	16.8 / 0.30 / 8.34	4.17 / 0.23 / 62.1	12.2 / 0.39 / 9.06
	PL2Map $^{\text{point \& line}}$ (**ours**)	33.8 / 0.16 / 2.64	7.1 / 0.13 / 33.5	15.4 / 0.28 / 9.34	3.75 / 0.21 / 68.9	13.3 / 0.43 / 9.06

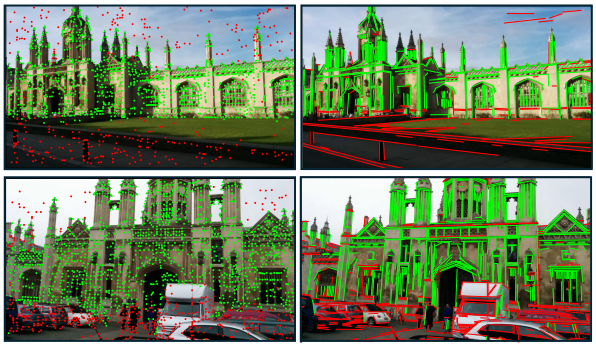


Fig. 7: **Reliability prediction results on both 2D points and lines.** We show some examples of reliability prediction on the King's College scene [17] with thresholds of $r^p > 0.6$ and $r^l > 0.05$.

gap with FM-based methods in large-scale outdoor scenarios. Thus, we compared our method with four additional SCR-based baselines: SANet [12], DSAC* [10], ACE [11], and D2S [28]. We present the results in Table III. Among the learning-based approaches, our localization results exhibited the lowest errors when utilizing only the predicted point maps. For instance, in the King's College scene, our method outperforms [28], which also employs keypoint descriptors, by achieving a 51% reduction in translation error. This substantial improvement narrows the accuracy gap with FM-based methods.

Line-assisted localization. Table III also includes the results of our method when localization is achieved through the integration of both points and line maps. Across all five scenes, PL2Map demonstrated an enhancement in localization accuracy compared with scenarios where only the predicted points map was utilized.

D. Systems Efficiency

Similar to [28], our method demonstrated the ability to disregard outlier features associated with dynamic elements or those unsuitable for localization through a single feedforward pass. This result is shown in Fig, 7, where the PnP RANSAC step can benefit from focusing on a high-quality set of correspondences [28].

TABLE IV: Comparison in requirements of localization pipeline using points and lines.

Requirements	Localization Method			
	Hloc [6]	PtLine [24]	Limap [2]	Pl2Map (ours)
Database	yes	yes	yes	no
Image-retrieval	yes	yes	yes	no
Points Matcher	yes	yes	yes	no
Lines Matcher	-	yes	yes	no

Table IV compares our approach with three primary baselines in terms of localization requirements, highlighting the efficiency of the proposed method by eliminating the need for a matching step and storing 3D maps as descriptors. Consequently, our approach requires significantly less memory, requiring approximately 25 MB for the network weight, in stark contrast to the several GBs required by Hloc [6] and Limap [2].

V. CONCLUSIONS

We propose the innovative PL2Map pipeline, designed to encapsulate sparse 3D points and lines within a unified model. After training with a designated scene, our pipeline efficiently generates 2D-3D correspondences for point and line features. In familiar settings, PL2Map not only serves as a cost-effective alternative to the conventional approach of storing and matching expensive descriptors but also shows robust re-mapping capabilities, which result in state-of-the-art camera relocalization.

Future efforts could expand this work to a larger scale and include scene-agnostic pre-training of the attentional module across diverse conditions. Such advancements aim to achieve a quicker and more robust re-mapping methodology.

VI. ACKNOWLEDGMENT

The author from Ritsumeikan University who participates in this IROS conference is partially supported by the IROS Development and Promotion Fund.

REFERENCES

[1] K. Xu, Y. Hao, S. Yuan, C. Wang, and L. Xie, "Airvo: An illumination-robust point-line visual odometry," in *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2023, pp. 3429–3436.

[2] S. Liu, Y. Yu, R. Pautrat, M. Pollefeys, and V. Larsson, "3d line mapping revisited," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 21 445–21 455.

[3] Y. Zhang, P. Zhu, and W. Ren, "Pl-cvio: Point-line cooperative visual-inertial odometry," in *2023 IEEE Conference on Control Technology and Applications (CCTA)*. IEEE, 2023, pp. 859–865.

[4] R. Gomez-Ojeda, F.-A. Moreno, D. Zuniga-Noël, D. Scaramuzza, and J. Gonzalez-Jimenez, "Pl-slam: A stereo slam system through the combination of points and line segments," *IEEE Transactions on Robotics*, vol. 35, no. 3, pp. 734–746, 2019.

[5] F. Shu, J. Wang, A. Pagani, and D. Stricker, "Structure plp-slam: Efficient sparse mapping and localization using point, line and plane for monocular, rgb-d and stereo cameras," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2023, pp. 2105–2112.

[6] P.-E. Sarlin, C. Cadena, R. Siegwart, and M. Dymczyk, "From coarse to fine: Robust hierarchical localization at large scale," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 12 716–12 725.

[7] P.-E. Sarlin, D. DeTone, T. Malisiewicz, and A. Rabinovich, "Superglue: Learning feature matching with graph neural networks," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 4938–4947.

[8] S. Yoon and A. Kim, "Line as a visual sentence: context-aware line descriptor for visual localization," *IEEE Robotics and Automation Letters*, vol. 6, no. 4, pp. 8726–8733, 2021.

[9] R. Pautrat, J.-T. Lin, V. Larsson, M. R. Oswald, and M. Pollefeys, "Sold2: Self-supervised occlusion-aware line description and detection," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 11 368–11 378.

[10] E. Brachmann and C. Rother, "Visual camera re-localization from rgb and rgb-d images using dsac," *IEEE transactions on pattern analysis and machine intelligence*, vol. 44, no. 9, pp. 5847–5865, 2021.

[11] E. Brachmann, T. Cavallari, and V. A. Prisacariu, "Accelerated coordinate encoding: Learning to relocalize in minutes using rgb and poses," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 5044–5053.

[12] L. Yang, Z. Bai, C. Tang, H. Li, Y. Furukawa, and P. Tan, "Sanet: Scene agnostic network for camera localization," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2019, pp. 42–51.

[13] L. Zhou, Z. Luo, T. Shen, J. Zhang, M. Zhen, Y. Yao, T. Fang, and L. Quan, "Kfnet: Learning temporal camera relocalization using kalman filtering," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 4919–4928.

[14] P. Lindenberger, P.-E. Sarlin, and M. Pollefeys, "Lightglue: Local feature matching at light speed," *arXiv preprint arXiv:2306.13643*, 2023.

[15] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," *Advances in neural information processing systems*, vol. 30, 2017.

[16] J. Shotton, B. Glocker, C. Zach, S. Izadi, A. Criminisi, and A. Fitzgibbon, "Scene coordinate regression forests for camera relocalization in rgb-d images," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2013, pp. 2930–2937.

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

[18] R. Arandjelovic, P. Gronat, A. Torii, T. Pajdla, and J. Sivic, "Netvlad: Cnn architecture for weakly supervised place recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 5297–5307.

[19] T. Sattler, Q. Zhou, M. Pollefeys, and L. Leal-Taixe, "Understanding the limitations of cnn-based absolute camera pose regression," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2019, pp. 3302–3312.

[20] T. B. Bach, T. T. Dinh, and J.-H. Lee, "Featloc: Absolute pose regressor for indoor 2d sparse features with simplistic view synthesizing," *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 189, pp. 50–62, 2022.

[21] Q. Zhou, T. Sattler, M. Pollefeys, and L. Leal-Taixe, "To learn or not to learn: Visual localization from essential matrices," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2020, pp. 3319–3326.

[22] L. Svärm, O. Enqvist, F. Kahl, and M. Oskarsson, "City-scale localization for cameras with known vertical direction," *IEEE transactions on pattern analysis and machine intelligence*, vol. 39, no. 7, pp. 1455–1461, 2016.

[23] B. Zeisl, T. Sattler, and M. Pollefeys, "Camera pose voting for large-scale image-based localization," in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 2704–2712.

[24] S. Gao, J. Wan, Y. Ping, X. Zhang, S. Dong, Y. Yang, H. Ning, J. Li, and Y. Guo, "Pose refinement with joint optimization of visual points and lines," in *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2022, pp. 2888–2894.

[25] R. Pautrat, I. Suárez, Y. Yu, M. Pollefeys, and V. Larsson, "Gluestick: Robust image matching by sticking points and lines together," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023, pp. 9706–9716.

[26] T. B. Bui, D.-T. Tran, and J.-H. Lee, "Fast and lightweight scene regressor for camera relocalization," *arXiv preprint arXiv:2212.01830*, 2022.

[27] P. Speciale, J. L. Schonberger, S. B. Kang, S. N. Sinha, and M. Pollefeys, "Privacy preserving image-based localization," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 5493–5503.

[28] B.-T. Bui, D.-T. Tran, and J.-H. Lee, "D2s: Representing local descriptors and global scene coordinates for camera relocalization," *arXiv preprint arXiv:2307.15250*, 2023.

[29] T. Do, O. Miksik, J. DeGol, H. S. Park, and S. N. Sinha, "Learning to detect scene landmarks for camera localization," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 11 132–11 142.

[30] S. T. Nguyen, A. Fontan, M. Milford, and T. Fischer, "Focustune: Tuning visual localization through focus-guided sampling," in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 2024, pp. 3606–3615.

[31] F. Shu, Y. Xie, J. Rambach, A. Pagani, and D. Stricker, "Visual slam with graph-cut optimized multi-plane reconstruction," in *2021 IEEE International Symposium on Mixed and Augmented Reality Adjunct (ISMAR-Adjunct)*. IEEE, 2021, pp. 165–170.

[32] Y. Wu, Y. Zhang, D. Zhu, Z. Deng, W. Sun, X. Chen, and J. Zhang, "An object slam framework for association, mapping, and high-level tasks," *IEEE Transactions on Robotics*, 2023.

[33] Z. Liao, Y. Hu, J. Zhang, X. Qi, X. Zhang, and W. Wang, "So-slam: Semantic object slam with scale proportional and symmetrical texture constraints," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 4008–4015, 2022.

[34] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International journal of computer vision*, vol. 60, pp. 91–110, 2004.

[35] D. DeTone, T. Malisiewicz, and A. Rabinovich, "Superpoint: Self-supervised interest point detection and description," in *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, 2018, pp. 224–236.

[36] R. G. Von Gioi, J. Jakubowicz, J.-M. Morel, and G. Randall, "Lsd: A fast line segment detector with a false detection control," *IEEE transactions on pattern analysis and machine intelligence*, vol. 32, no. 4, pp. 722–732, 2008.

[37] R. Pautrat, D. Barath, V. Larsson, M. R. Oswald, and M. Pollefeys, "Deeplsd: Line segment detection and refinement with deep image gradients," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 17 327–17 336.

[38] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer, "Automatic differentiation in pytorch," 2017.

[39] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[40] S. Izadi, D. Kim, O. Hilliges, D. Molyneaux, R. Newcombe, P. Kohli, J. Shotton, S. Hodges, D. Freeman, A. Davison *et al.*, "Kinectfusion: real-time 3d reconstruction and interaction using a moving depth camera," in *Proceedings of the 24th annual ACM symposium on User interface software and technology*, 2011, pp. 559–568.