2024 IEEE International Conference on Robotics and Automation (ICRA 2024)
May 13-17, 2024. Yokohama, Japan

LCCRAFT: LiDAR and Camera Calibration Using Recurrent All-Pairs Field Transforms Without Precise Initial Guess

Yu-Chen Lee, and Kuan-Wen Chen*

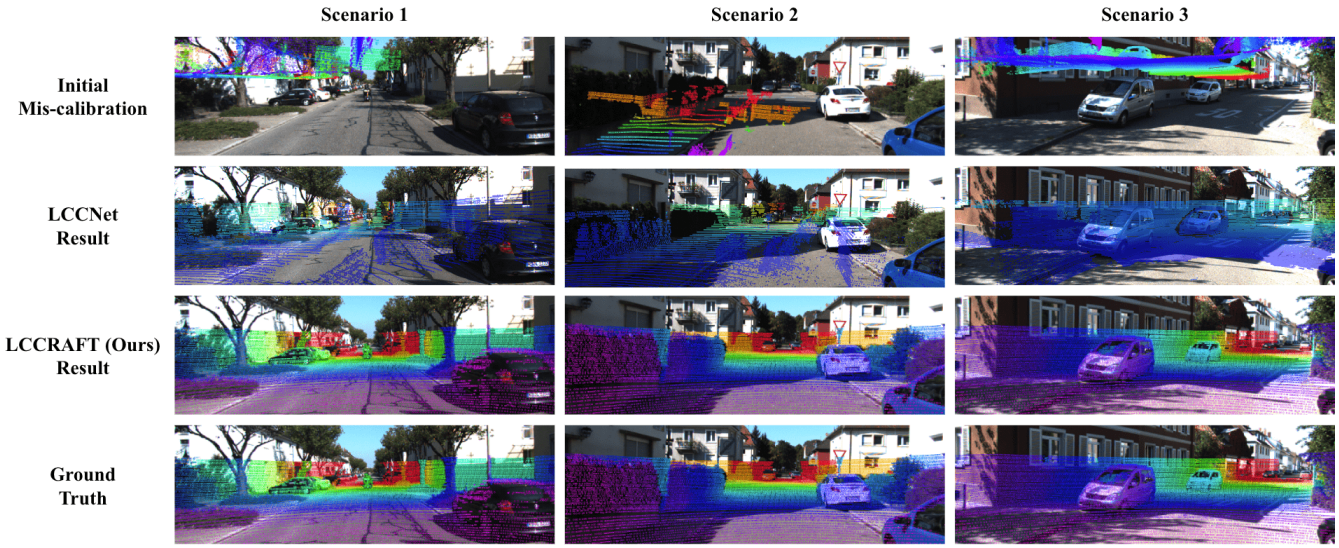

Fig. 1: Comparsion of the proposed LCCRAFT with LCCNet [1] to perform calibration without a precise initial guess.

***Abstract*— LiDAR-camera fusion plays a pivotal role in 3D reconstruction for self-driving applications. A fundamental prerequisite for effective fusion is the precise calibration between LiDAR and camera systems. Many existing calibration methods are constrained by predefined mis-calibration ranges in the training data, essentially tying the network to a specific data distribution. However, if the range of evaluation data differs from what the network has been trained on, the resulting estimates may not meet expectations. Moreover, most methods require a precise initial guess for calibration to succeed. In this paper, we introduce LCCRAFT, an online calibration network designed for LiDAR and camera systems. Leveraging the 4D correlation volume and correlation lookup techniques inherited from RAFT, we apply them to correlate RGB images and depth maps derived from the projection of point clouds. Through weight sharing between update iterations and by enabling the update operator to learn from data with varying degrees of error, LCCRAFT demonstrates adaptability to diverse mis-calibration scenarios. This includes cases where the initial mis- calibration is even more severe than what the system encountered during training, demonstrating the robustness of the model. The calibration process executes in 93ms on a single GPU, meeting real-time requirements. Despite the modest 9M model parameters, LCCRAFT achieves competitive performance as compared to the state-of-the-art method, which entails 69M parameters.**

I. Introduction

Over the past few years, there has been a growing focus on the field of self-driving, where the LiDAR–camera fusion technology plays a significantly crucial role. As autonomous driving scenarios can be highly dynamic and complex, relying solely on either LiDAR or a camera for stable perception can be challenging. With the fusion of the data from LiDAR and the camera, the aforementioned issue can be mitigated, providing more comprehensive and accurate environmental information for 3D reconstruction. To ensure effective fusion, accurate calibration between LiDAR and the camera is a fundamental requirement. However, RGB images and LiDAR point clouds are different modalities: images are 2D data with rich texture information, while point clouds are 3D data with precise structural details. Accurately identifying the 2D–3D correspondence for calibration is a challenging task because of their cross-modality nature.

Traditionally [2]–[4], LiDAR-camera calibration often requires a controlled environment with artificial patterns or objects. This facilitates the precise alignment of the two sensors, as establishing 2D-3D correspondences becomes straightforward. However, generating such scenes demands a significant manual effort and is impractical for unknown real-world environments.

Recently, researchers have turned to deep learning methods for calibration tasks, departing from traditional approaches. Among these researchers, some are exploring the projection of 3D point clouds onto 2D depth maps to reduce the inherent disparity between 2D and 3D data. They then harness the robust fitting capabilities of convolutional neural networks (CNNs) to predict the 6 degrees of freedom (DoF)

Authors are with the Department of Computer Science, National Yang Ming Chiao Tung University, Hsinchu 300, Taiwan. (E-mail of corresponding author* Kuan-Wen Chen: kuanwen@cs.nycu.edu.tw)

extrinsic parameter transformation between the two sensors. However, achieving a precise estimation of the calibration matrix across all ranges of initial mis-calibration using a single network poses a challenge. Researchers often resort to iterative refinement. This involves deploying multiple networks designed for specific ranges of mis-calibration, albeit at the expense of increased network parameters. Furthermore, the network's performance may be compromised if the data's mis-calibration range differs from what the network has been trained on.

To address this, we developed a single network, LCCRAFT, capable of performing calibration without the need for iterative refinement. Taking inspiration from RAFT [5], LCCRAFT incorporates a shared weight update operator that refines the estimated calibration matrix with each iteration. By cascading the identical network twice, the update operator simultaneously learns from different ranges of mis-alibration, enhancing the network's robustness and generalization capabilities. Additionally, we calculate the point density in each area of the depth map to assign a confidence score. This enables the network to prioritize regions with richer information, reducing the noise introduced by areas with a lower point density.

Our experiments demonstrate that LCCRAFT achieves competitive results when compared with the state-of-the-art [1] model, despite having a significantly smaller model size of 9M parameters, as opposed to the 69M parameters of the state-of-the-art model. Furthermore, LCCRAFT demonstrates the capability to handle larger ranges of mis-calibration scenarios, as depicted in Fig. 1, even when these ranges had not been encountered during the training phase. Our contributions are as follows:

- We introduce LCCRAFT, a novel single calibration network specifically designed for LiDAR and camera systems. The purpose is to avoid the use of multiple iterative networks and enable the network to handle scenarios with different mis-calibration ranges, rather than training a whole network specifically for a particular range.
- We propose a correlation lookup for the depth map and implement a confidence score to give priority to areas with a higher point density. Additionally, we present a two-stage pipeline utilizing LCCRAFT for the calibration.
- LCCRAFT delivers competitive performance with a light model containing only 9M parameters, in contrast to the state-of-the-art [1] method that involves 69M parameters. Additionally, LCCRAFT achieves efficient calibration in just 93ms on a single GPU.

II. Related Work

A. Target-Based Calibration

In the context of LiDAR-camera external calibration, establishing correspondences between 2D and 3D features poses a significant challenge. To address this, methods commonly use specialized calibration targets such as checkerboards [2], V-shaped objects [3], or spherical targets [4] to achieve precise calibration results. However, these approaches may face limitations in real-time online environments because of the impracticality of obtaining these specific calibration targets promptly. Additionally, target-based calibration methods often entail longer calibration times, which may not align with the demands of real-time online calibration requirements.

B. Target-Less Calibration

In recent years, there has been notable research on target-free calibration methods. In contrast to target-based approaches, target-free methods directly extract feature points from the experimental scene without the need for manually placed targets. The main hurdle faced by these methods lies in reconciling the disparities between 2D and 3D data modalities, which poses a challenge in establishing correspondences between feature points in these two domains.

DeepI2P [6] leverages ResNet [7] for extracting image features and uses PointNet++ [8] for extracting features from point clouds. Following this, it uses a decoder similar to PointNet [9] to reframe the 2D-3D calibration challenge as a classification task. In this task, points are categorized on the basis of their presence within the camera's frustum. These categorized points are subsequently processed by an innovative inverse camera projection solver, which enables the estimation of the relative pose.

ATOP [10] has devised a cross-modal matching network. Initially, it conducts object detection to identify cars and pedes- trians in both 2D and 3D data. Subsequently, it leverages these refined features to locate the fields of view (FOV) for both detectors and establish correspondences. Finally, it uses particle swarm optimization (PSO) algorithms to estimate the external parameters. However, note that gathering and annotating data for object detection can be an exceedingly time-consuming and labor-intensive task. Additionally, in real-world scenarios, the presence of people or vehicles is not always guaranteed. Therefore, the practical applicability of this approach warrants further investigation.

RegNet [11] serves as an innovative approach, commencing by projecting point clouds onto a 2D plane. This facilitates the application of CNNs to the deduction of the 6-DoF external calibration between LiDAR and the camera. CMRNet [12] and LCCNet [1] take inspiration from RegNet, projecting point clouds onto depth maps. They subsequently compute a correlation volume of features utilizing RGB images and directly feed this volume into fully connected layers for estimating rotation quaternions and translation vectors. These methods use iterative refinement to attain more precise estimation outcomes. However, achieving this necessitates training multiple distinct networks to accommodate varying mis-calibration ranges. This not only leads to a substantial increase in the number of parameters, but also implies that each network is specifically suited for data with particular error distributions. Consequently, achieving generalization across different distributions of test data becomes a challenge.

Drawing inspiration from LCCNet [1], we endeavored to address the calibration challenge through an optical flow

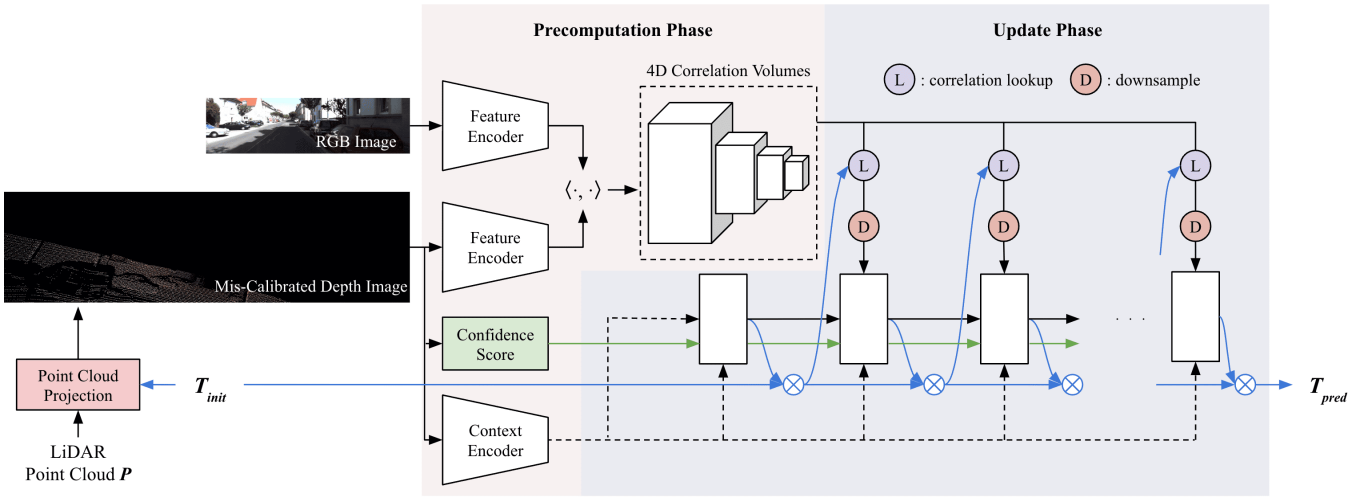

Fig. 2: The overview of LCCRAFT. We add color to the components we designed.

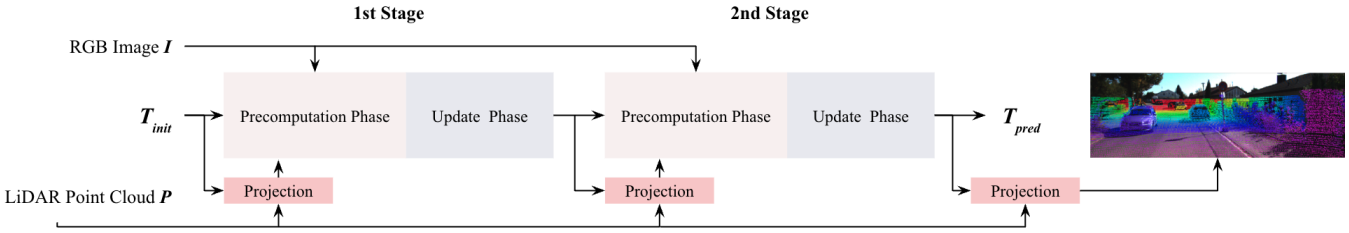

Fig. 3: The pipeline of our calibration method.

methodology. To accomplish this, we integrated RAFT [5] as the backbone of our network, with the goal of refining calibration within a unified framework. We also implemented shared-weight update operators to enhance the robustness and adaptability of the network. This approach empowers the network to effectively manage a range of mis-calibration scenarios.

III. Methodology

In this section, we begin by presenting an overview of LCCRAFT, highlighting its similarities with RAFT [5] in terms of the general process. We then delve into the unique components of LCCRAFT and the rationale behind their design. Next, we introduce a two-stage pipeline utilizing LCCRAFT for calibration. Finally, we provide detailed explanations of each component within LCCRAFT.

For an in-depth understanding of RAFT, please refer to [5]. In this paper, we provide a concise overview of the RAFT architecture, with a primary focus on our proposed design.

A. Overview

Our network architecture is primarily inspired by RAFT [5], as illustrated in Fig. 2. A depth map is first transformed from a LiDAR point cloud by using an initial mis-calibration matrix T_{init}. We extract features from the RGB image and the depth map by using an identical feature encoder. Additionally, a context encoder is applied to extract context features from the depth map. The 4D correlation volume is constructed by computing the similarity between all pairs of features. A lookup operation is defined to query correlation features from the 4D correlation volume. In each update iteration, we query correlation features by using pixel correspondence of the depth map. These features are then combined with the other information and passed to the update operator, which estimates the current calibration matrix. The correspondence map is generated through computation by using the estimated matrix from the previous iteration. After several iterations of updates, LCCRAFT will output the final estimated calibration matrix. Note that the computations in the precomputation phase only need to be executed once.

B. Our Designs

Unfortunately, RAFT [5] cannot be directly applied to the calibration task. That is because the challenges of calibration are different from those of optical flow. Optical flow typically deals with relatively small variations in perspective. However, mis-calibration can involve substantial rotations or shifts, resulting in significant disparities between the initial depth map and the RGB image. To address this, we extend the depth map to twice the size of the RGB image. This is done with the aim of capturing as many projection points as possible to retain crucial information.

Additionally, unlike flow, where correspondence for each pixel can be calculated, in calibration, only about 5% of the pixels on the depth map have projection points. As a result, the majority of pixels cannot have their correspondence estimated. To tackle this, we devised a new correlation lookup method and introduced the concept of a confidence

score. This allows the update operator to focus solely on regions containing projection points, while disregarding noise introduced by pixels lacking useful information.

Furthermore, in the update operator, we replaced the originally estimated flow in RAFT [5] with a 3D flow. This 3D flow is then passed through several fully connected layers to estimate the calibration matrix.

C. Pipeline

In order to enhance the robustness of LCCRAFT and enable learning from different mis-calibration ranges simultaneously, we have devised a two-stage pipeline to accomplish the calibration task. As depicted in Fig. 3, we feed the initial mis-calibrated matrix, RGB image, and the projected depth map into the first stage of LCCRAFT. The output is then projected onto the point cloud to generate a new depth map, which, along with the output from the first stage and the RGB image, is fed into the second stage of LCCRAFT. Note that the two-stage LCCRAFT share the exact same weights, allowing for a minimal parameter footprint.

The main reasons for the above design are twofold. Firstly, as mentioned earlier, calibration often involves substantial rotations or translations, resulting in an initial depth map that is frequently distorted and challenging to interpret. Therefore, by adjusting the depth map in the first stage to closely align its perspective with the RGB image, we can conduct a more precise calibration in the second stage. Secondly, by utilizing two initial matrices with different error ranges, we enable the update operator to concurrently converge towards data from various ranges. This grants it greater versatility, allowing it to learn not just for specific mis-calibration ranges but also across a broader spectrum.

D. Feature Extraction

The feature encoder is jointly employed for processing both I and D, producing features $F^I \in R^{H' \times W' \times C}$ and $F^D \in R^{2H' \times 2W' \times C}$, where $H' = \frac{H}{8}$, $W' = \frac{W}{8}$, $C = 256$.

The context encoder mirrors the structure of the feature encoder, but it does not share weights. In our research, we apply the context encoder specifically to extract context features from the depth map D.

E. Visual Similarity Computation

1) 4D Correlation Volume: With the image features F^I and depth map features F^D at hand, we create a 4D correlation volume by calculating the similarity between all pairs of features using inner products. The mathematical representation of the correlation volume is as follows:

$$\begin{aligned} Corr(F^D, F^I) &\in R^{H' \times W' \times 2H' \times 2W'}, \\ Corr_{ijkl} &= \sum_h F^D_{ijh} \cdot F^I_{klh} \end{aligned} \tag{1}$$

Subsequently, we perform pooling on the last two dimensions of the correlation volume to construct a correlation pyramid denoted as $\{C_1, C_2, C_3, C_4\}$, where $C_k \in R^{H \times W \times 2H/2^k \times 2W/2^k}$. The kernel sizes for pooling are set as 1, 2, 4, and 8.

2) Correlation Lookup: Given a point $p \in P$ and a projection matrix $Proj$, we can acquire the initial pixel $x = Proj \cdot T_0 \cdot p = (u, v)$ in D. The correspondence of x can be mapped to $x' = Proj \cdot T_t \cdot p = (u' - o, v' - o)$ in I, where $o = (o_u, o_v)$ represents the offset between D and I. For pixels with no projected points, it becomes impossible to ascertain their correspondence. Hence, we assign a specific value $x'_{invalid} = (-1e^5, -1e^5)$, which indirectly leads to the indexed correlation feature being set to zero in the subsequent steps.

Following this, we proceed to create a grid of x''s neighbors:

$$N(x') = \{x' + dx \mid dx \in Z^2, \|dx\|_1 \leq r\} \tag{2}$$

where r represents the radius of the grid. We consider $N(x')$ as the correspondence map for the depth map. We downsample $N(x')$ to $1/8$ of its original resolution, and the correspondences are also divided by 8 to align with the resolution of the 4D correlation volume. Subsequently, $N(x')$ is used to index into the correlation volume, and we apply bilinear sampling to handle real number values.

3) Confidence Score: Because of the uneven distribution of points on the depth map D, with many areas lacking points, the indexed correlation features in these regions might contain unnecessary noise. Consequently, we cannot assign equal importance to all of the areas.

To address this, we use a mask α to indicate whether each pixel has a projected point or not, as depicted below:

$$\alpha(x) = \begin{cases} 1, & \text{if pixel } x \text{ has an projected point}, \forall x \in D \\ 0, & \text{otherwise} \end{cases} \tag{3}$$

Subsequently, we downsample α along with the previously mentioned correspondence map to generate a density map. By calculating the density in each area, we can explicitly assign a confidence score to each correlation feature. This enables the network to prioritize areas with high point density, which contain richer information. The obtained score is then concatenated with the correlation features and fed into a recurrent update operator.

F. Iterative Updates

The update operator is a recurrent ConvGRU unit, as illustrated in Fig. 4. The inputs consist of the concatenation of correlation features, correspondence map, context features, and confidence scores, as mentioned above. Furthermore, it incorporates a hidden state h_{t-1} from the preceding iteration.

The output of the ConvGRU, denoted as h_t, is then subjected to two convolutional layer to estimate a 3D-flow f_t. Then, f_t is fed to the fully connected layers to generate the estimated rotation quaternion Δq_t and translation vector Δt_t. Subsequently, we convert the quaternion into a rotation matrix ΔR_t. The estimated calibration matrix T_t can be computed as follows:

$$T_t = \begin{bmatrix} \Delta R_t & \Delta t_t \\ 0 & 1 \end{bmatrix} \cdot T_{t-1} \tag{4}$$

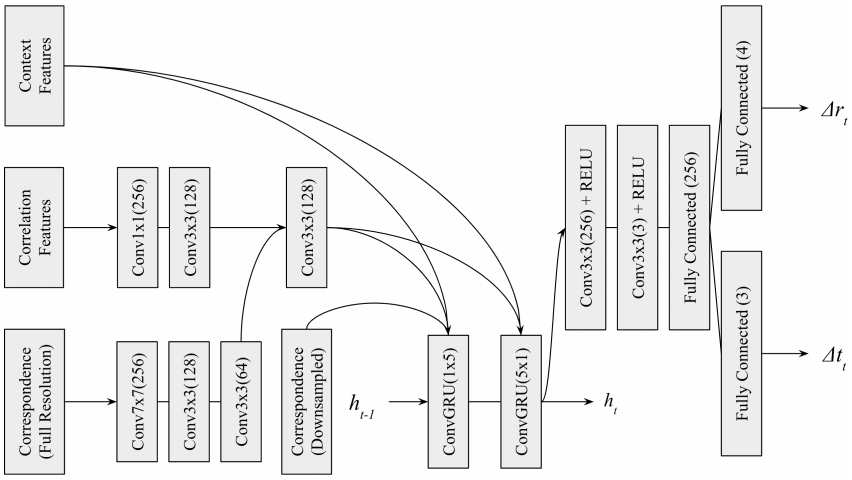

Fig. 4: The architecture of update operator. It takes the correspondence map, correlation features and context features as input, and updates the hidden state h_t, which is used to estimate the calibration matrix. Three fully connected layer is applied in order to extract features from a broader, more global perspective.

G. Loss Function

For every update iteration, we calculate a loss L_t and aggregate them after the final iteration:

$$\begin{aligned} L_t &= \omega_D L_D + \omega_R L_R + \omega_T L_T \\ L &= \sum_{t=1}^{M} L_t \end{aligned} \tag{5}$$

where M represents the number of iterations, L_D denotes the point cloud distance loss, L_R indicates the rotation loss, L_T refers to the translation loss, and ωs stands for the respective loss weights.

1) Point Cloud Distance Loss: Given a point cloud P, the point cloud distance loss can be calculated using the following formula:

$$L_D = \frac{1}{N} \sum_{i=1}^{N} \| T_{gt} \cdot P - T_t \cdot P \|_2 \tag{6}$$

where N represents the number of point clouds, and $\| \cdot \|_2$ denotes the L2 Normalization.

2) Translation Loss and Rotation Loss: For the translation loss L_T, we use the smooth L1 loss. In the case of the rotation loss L_T, we use the angular distance in order to compute differences between quaternions.

IV. Experiments

A. Implementation Details

Our work is implemented using the PyTorch library [14]. The proposed network, LCCRAFT, is trained from scratch for 500 epochs with a batch size of 20, utilizing the AdamW [15] optimizer. The initial learning rate is set to $8e^{-4}$. Gradients are also clipped within the range of [-1, 1], following the approach employed in RAFT [5]. The training process is conducted using four GeForce RTX 3090 Ti GPUs, while testing is performed using one GPU.

B. Dataset

Our experiments are conducted on the KITTI Odometry Dataset [16], which consists of 21 sequences captured in various scenarios. Ground truth data for the extrinsic calibration parameters, specifically between the LiDAR and the RGB cameras (both left and right), are provided by the dataset. For training and validation, we utilize sequences 00-08, which consist of 40,818 frames. Sequences 09-10, containing 5,584 frames, are reserved for testing purposes.

To create initial poses T_{init}, we introduce a random transformation T_{rand} applied to the ground truth poses $T_{init} = T_{rand} \cdot T_{gt}$. Our objective is to estimate this added random transformation, which introduces mis-calibration between the LiDAR and the camera systems. We adopt the largest translation and rotation mis-calibration range from LCCNet [1], specifically $(\pm 1.5m, \pm 20^\circ)$. This range is applied consistently for both the training and the testing phases. We randomly crop and resize the RGB images to a size of 160×512 in order to prevent the depth maps from becoming excessively large. This approach effectively mitigates a substantial increase in the computational cost associated with the 4D correlation volume.

C. Evaluation

Table I compares the calibration performance of different methods. It can be observed that LCCRAFT achieves competitive results comparable to LCCNet [1], with the advantage of faster processing time. It is important to mention that LCCNet has a total of 69 million parameters, whereas LCCRAFT requires only 9 million to achieve similar performance. Additionally, while ATOP [10] shows outstanding results, it comes with significant drawbacks. Their approach considers all 360 degrees of point clouds and requires additional algorithm-based optimization on top of object detection, resulting in a very high computational cost and impracticality for real-time applications. LCCRAFT, on

TABLE I: Comparison results with other calibration methods.

Method	**Initial Mis-calibration**	$tx(cm)$	$ty(cm)$	$tz(cm)$	$pitch(°)$	$roll(°)$	$yaw(°)$	**running time(s)**
RegNet [11]	$(\pm 1.5m, \pm 20°)$	7	7	4	0.25	0.36	0.24	**0.037**
CalibNet [13]	$(\pm 0.2m, \pm 10°)$	4.2	1.6	7.22	0.9	0.18	0.15	-
LCCNet [1]	$(\pm 1.5m, \pm 20°)$	0.243	0.380	0.459	**0.019**	0.030	0.040	0.12
ATOP [10]	w/o	**0.010**	**0.002**	**0.040**	0.04	0.28	0.27	13.59
LCCRAFT	$(\pm 1.5m, \pm 20°)$	0.821	0.771	1.631	0.117	**0.029**	0.041	0.093

TABLE II: Evaluation Using Large Mis-calibration Ranges.

Initial Mis-calibration	**Method**	$tx(cm)$	$ty(cm)$	$tz(cm)$	$pitch(°)$	$roll(°)$	$yaw(°)$
$(\pm 3m, \pm 40°)$	LCCNet [1]	18.03	9.95	11.82	10.088	1.488	1.678
	LCCRAFT (Ours)	**13.13**	**3.82**	**5.82**	**0.146**	**0.703**	**0.175**
$(\pm 6m, \pm 20°)$	LCCNet [1]	95.03	54.98	72.41	2.347	2.684	1.458
	LCCRAFT (Ours)	**65.91**	**35.50**	**30.36**	**0.981**	**1.234**	**0.397**
$(\pm 10m, \pm 10°)$	LCCNet [1]	328.74	244.48	285.49	8.127	4.849	4.320
	LCCRAFT (Ours)	**59.85**	**22.54**	**40.20**	**1.247**	**0.572**	**0.493**

TABLE III: Calibrating evaluation after Nth stage

after Nth stage	$tx(cm)$	$ty(cm)$	$tz(cm)$	$pitch(°)$	$roll(°)$	$yaw(°)$	**running time(s)**
1st (6 iterations)	31.38	12.87	16.17	1.296	0.419	0.467	0.037
2nd (12 iterations)	0.821	**0.771**	1.631	**0.117**	**0.029**	0.041	0.093
3rd (12 iterations)	**0.665**	1.031	**1.384**	0.235	0.032	**0.038**	0.144

the other hand, strikes a balance between performance and efficiency with a remarkably small parameter count.

To demonstrate the robustness of LCCRAFT, we conducted experiments with more severe initial mis-calibration, as shown in Table II. Unlike LCCNet [1], which struggles with effective calibration in cases of significant differences due to training with multiple networks tailored to specific ranges, LCCRAFT maintains a certain level of performance even when the mis-calibration range is greater than the $(\pm 1.5m, \pm 20°)$ used in training. In fact, it achieves excellent alignment in some test data, as depicted in Fig. 1, surpassing LCCNet's overall performance by a significant margin.

In our ablation study about Nth-stage calibration evaluation shown in Table III, the first stage exhibits a sizable estimation error due to challenges aligning the initial depth map with the RGB image, resulting in limited convergence. In the second stage, leveraging first-stage estimations, we compute image similarity for a more effective calibration matrix update, leading to a significant performance improvement. Despite experimenting with a third stage, the results fell short of expectations, showing only slight improvement, and, in some cases, increased errors. This is likely because errors have already converged to a certain extent in the second stage, and the upper limit of convergence depends on image completeness. Due to computational costs, images were cropped to 160×512, potentially limiting convergence. Considering both costs and performance, we opted for the balanced two-stage pipeline for implementing LiDAR-camera calibration.

V. Conclusion

In this paper, we introduce a novel single calibration network, LCCRAFT, specifically designed for LiDAR and camera systems. The network aims to handle calibration tasks without resorting to iterative refinement, which typically requires the use of multiple networks and leads to a significant increase in model parameters. We have developed a correlation lookup tailored for depth maps, along with the utilization of confidence scores to mitigate noise in sparse areas of the depth map. Our two-stage pipeline design enables the update operator to learn from problems of varying difficulty levels and alleviates issues arising from highly distorted initial depth maps. Through experiments, we demonstrate the versatility and robustness of LCCRAFT. Finally, in scenarios where a precise initial guess is not available, LCCRAFT demonstrates superior performance to that of the state-of-the-art methods.

VI. Acknowledgement

This work was supported in part by the National Science and Technology Council, Taiwan (111-2628-E-A49-003-MY2 and 112-2634-F-A49-007-).

REFERENCES

[1] X. Lv, B. Wang, Z. Dou, D. Ye, and S. Wang, "Lccnet: Lidar and camera self-calibration using cost volume network," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognitiop*, pp. 2894–2901, 2021.

[2] Y. Park, S. Yun, C. S. Won, K. Cho, K. Um, and S. Sim, "Calibration between color camera and 3d lidar instruments with a polygonal planar board," *Sensors*, vol. 14, no. 3, pp. 5333–5353, 2014.

[3] S. Chen, J. Liu, X. Liang, S. Zhang, J. Hyyppä, and R. Chen, "A novel calibration method between a camera and a 3d lidar with infrared images," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 4963–4969, IEEE, 2020.

[4] T. Tóth, Z. Pusztai, and L. Hajder, "Automatic lidar-camera calibration of extrinsic parameters using a spherical target," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 8580–8586, IEEE, 2020.

[5] Z. Teed and J. Deng, "Raft: Recurrent all-pairs field transforms for optical flow," in *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part II 16*, pp. 402–419, Springer, 2020.

[6] J. Li and G. H. Lee, "Deepi2p: Image-to-point cloud registration via deep classification," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 15960–15969, 2021.

[7] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

[8] C. R. Qi, L. Yi, H. Su, and L. J. Guibas, "Pointnet++: Deep hierarchical feature learning on point sets in a metric space," *Advances in neural information processing systems*, vol. 30, 2017.

[9] C. R. Qi, H. Su, K. Mo, and L. J. Guibas, "Pointnet: Deep learning on point sets for 3d classification and segmentation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 652–660, 2017.

[10] Y. Sun, J. Li, Y. Wang, X. Xu, X. Yang, and Z. Sun, "Atop: An attention-to-optimization approach for automatic lidar-camera calibration via cross-modal object matching," *IEEE Transactions on Intelligent Vehicles*, vol. 8, no. 1, pp. 696–708, 2022.

[11] N. Schneider, F. Piewak, C. Stiller, and U. Franke, "Regnet: Multimodal sensor registration using deep neural networks," in *2017 IEEE intelligent vehicles symposium (IV)*, pp. 1803–1810, IEEE, 2017.

[12] D. Cattaneo, M. Vaghi, A. L. Ballardini, S. Fontana, D. G. Sorrenti, and W. Burgard, "Cmrnet: Camera to lidar-map registration," in *2019 IEEE intelligent transportation systems conference (ITSC)*, pp. 1283–1289, IEEE, 2019.

[13] G. Iyer, R. K. Ram, J. K. Murthy, and K. M. Krishna, "Calibnet: Geometrically supervised extrinsic calibration using 3d spatial transformer networks," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 1110–1117, IEEE, 2018.

[14] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, *et al.*, "Pytorch: An imperative style, high-performance deep learning library," *Advances in neural information processing systems*, vol. 32, 2019.

[15] I. Loshchilov and F. Hutter, "Decoupled weight decay regularization," *arXiv preprint arXiv:1711.05101*, 2017.

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