Overall response:

We thank all reviewers and editors for your time and effort put in reviewing our manuscript and the overall positive reviews you have given. Your detailed remarks and constructive comments indeed help us improve the quality of this manuscript. Following your suggestions, we have made some major changes summarized as below:

1> We have made a thorough revision of the writing (refined the language, improved the grammar and fixed all the offending typos) to improve the readability and facilitate understanding. Moreover, the writing is refined to be more concise and the content is better organized.
2> We have included more explanation to the problems raised by reviewers.
3> We have made significant improvement on the experimental section to make our paper more convincing.
In the following, we have attempted to address the concerns raised by the reviewers.

Review471019 (Reviewer21):

This paper proposes EasyColor, a framework for generating RGB-colored point clouds by aligning camera images with a LiDAR reflectivity map, without requiring precise time synchronization or extrinsic calibration. The key idea is to exploit reflectivity images derived from LiDAR intensity to bridge the modality gap between LiDAR and camera, enabling cross-domain image matching. The system combines reflectivity image alignment with epipolar visual constraints to jointly optimize camera poses and produce a photometrically consistent colorized map.

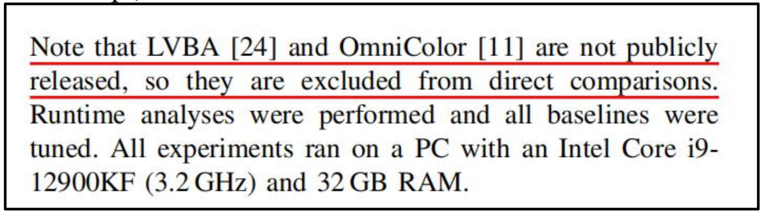
>>Response:

We thank reviewer for the overall positive comments and valuable suggestions. The following is our point-by-point answer to reviewer's concerns.

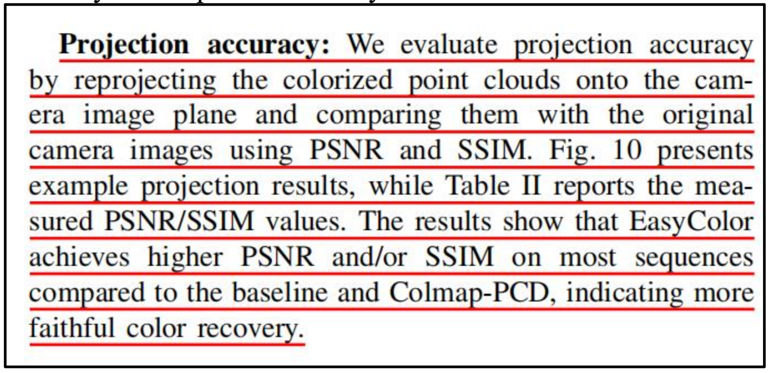
1> In my opinion since the objective of the paper is coloring pointcloud, just using pose estimation as accuracy is not enough. A comparison is PSNR and SSIM of reprojected images with Omnicolor or mutual information based methods is required to validate the claims.

>>Response:

Thank you so much for your constructive comments. In accordance with your suggestions, we have added experiments on the reprojected images. Fig. 10 presents the qualitative results of the reprojected images, and Table 2 reports the quantitative evaluation in terms of PSNR and SSIM. Unfortunately, we are unable to compare EasyColor with methods such as OmniColor or LVBA, because these methods are not open-sourced. Although we contacted the authors for assistance, we did not receive a response. We have added this clarification to the manuscript, as shown below:


Note that LVBA [24] and OmniColor [11] are not publicly released, so they are excluded from direct comparisons. Runtime analyses were performed and all baselines were tuned. All experiments ran on a PC with an Intel Core i9-12900KF (3.2 GHz) and 32 GB RAM.

Therefore, we can only compare against Colmap-PCD (the baseline used in LVBA) and our self-implemented baseline. Koide's mutual-information-based method is designed for extrinsic calibration between sensors, where the inputs are a single image and a point cloud. It cannot be applied to the task of colorizing a point cloud map from a sequence of images. In addition to the figures and tables, we have also added a discussion on projection accuracy in the qualitative analysis section.


Projection accuracy: We evaluate projection accuracy by reprojecting the colorized point clouds onto the camera image plane and comparing them with the original camera images using PSNR and SSIM. Fig. 10 presents example projection results, while Table II reports the measured PSNR/SSIM values. The results show that EasyColor achieves higher PSNR and/or SSIM on most sequences compared to the baseline and Colmap-PCD, indicating more faithful color recovery.

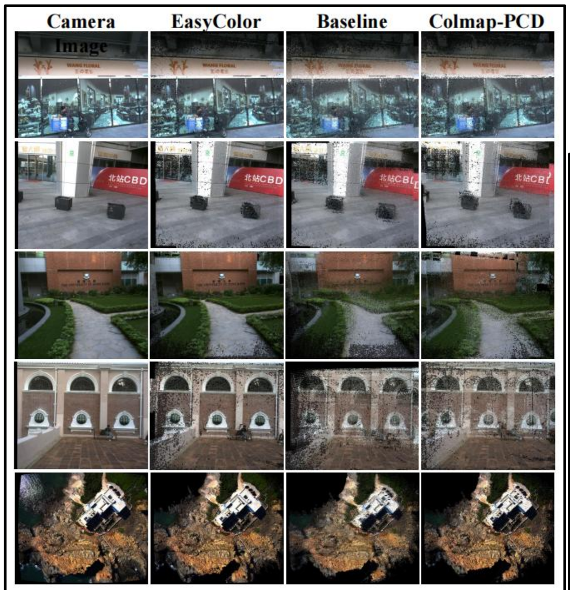


Fig. 10. Results of projecting colored point cloud generated by different methods on various public datasets onto image plane. Baseline result is obtained by coloring the point cloud using FastLIO2 [7] and official extrinsic file. A manual 0.1 s time offset was added to ensure a fair comparison.

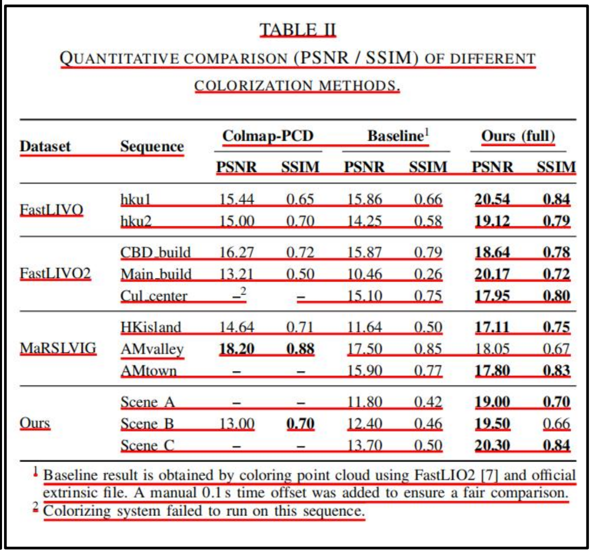


TABLE II

QUANTITATIVE COMPARISON (PSNR / SSIM) OF DIFFERENT COLORIZATION METHODS.

Dataset	Sequence	Colmap-PCD		Baseline[1]		Ours (full)	
		PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
FastLIVO	hku1	15.44	0.65	15.86	0.66	**20.54**	**0.84**
	hku2	15.00	0.70	14.25	0.58	**19.12**	**0.79**
FastLIVO2	CBD_build	16.27	0.72	15.87	0.79	**18.64**	**0.78**
	Main_build	13.21	0.50	10.46	0.26	**20.17**	**0.72**
	Cul_center	–[2]	–	15.10	0.75	**17.95**	**0.80**
MaRSLVIG	HKisland	14.64	0.71	11.64	0.50	**17.11**	**0.75**
	AMvalley	**18.20**	**0.88**	17.50	0.85	18.05	0.67
	AMtown	–	–	15.90	0.77	**17.80**	**0.83**
Ours	Scene A	–	–	11.80	0.42	**19.00**	**0.70**
	Scene B	13.00	**0.70**	12.40	0.46	**19.50**	0.66
	Scene C	–	–	13.70	0.50	**20.30**	**0.84**

[1] Baseline result is obtained by coloring point cloud using FastLIO2 [7] and official extrinsic file. A manual 0.1 s time offset was added to ensure a fair comparison.
[2] Colorizing system failed to run on this sequence.

2> *Assumptions of the prebuild point cloud map is not clear. It is unclear how the quality of this map affects performance e.g., what happens with sparse or noisy maps? How is the point cloud map generated? What are the expectations on its accuracy? I am assuming FastLIO2 just aligns the pointclouds based on odometry.*

>>Response:

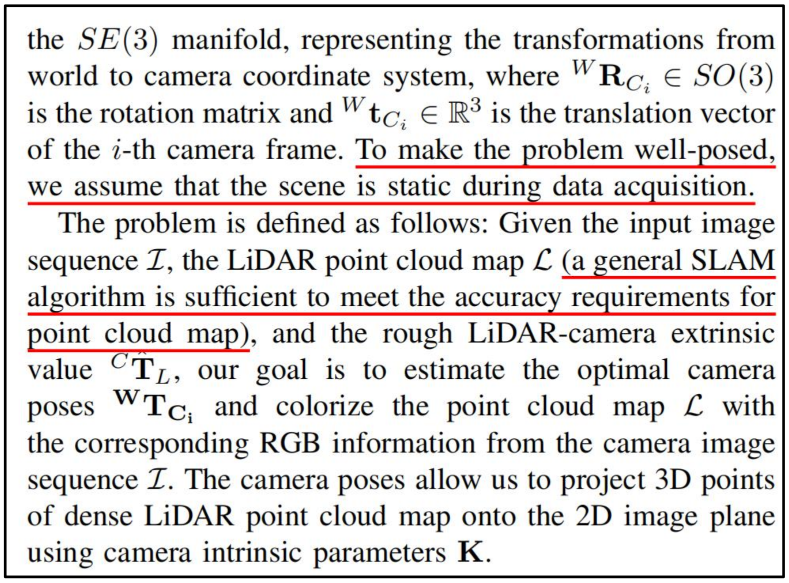
Thank you very much for your valuable comments. Below are our point-by-point responses to your questions:

(1) Robustness to sparse or noisy maps.
EasyColor relies on LiDAR scans with stable and reliable reflectivity measurements. Therefore, the LiDAR used to generate the pre-built map should operate in strongest-return mode. The density and noise characteristics of the map primarily depend on the LiDAR hardware. Through extensive experiments, we verified that EasyColor is compatible with a broad variety of sensors, including solid-state LiDARs (e.g., Livox AVIA as used in FastLIO2), mechanical LiDARs (e.g., Velodyne HDL-64E in KITTI), and our custom rotating LiDAR setup. The performance of EasyColor is not overly sensitive to map sparsity or noise. This robustness comes from the image-processing pipeline used to generate reflectivity images, where interpolation, contrast enhancement, and other refinement techniques effectively compensate for variations in point density and measurement noise. As a result, EasyColor remains stable across datasets of different qualities.

(2) How the pre-built map is generated.
In our experiments, the dense point-cloud map is produced by an upstream SLAM or mobile mapping system, which accumulates multiple LiDAR scans into a consistent global map. EasyColor is designed as a lightweight post-processing module that directly consumes such pre-built maps together with their corresponding image sequences.

(3) Expectations on map accuracy.
While higher-accuracy maps (e.g., those generated by terrestrial laser scanners) can further improve colorization quality, EasyColor does not require such high-precision inputs. The method is intended to provide a general, simple, and efficient colorization solution that works with any reasonably dense and geometrically consistent LiDAR map. Algorithms like FastLIO2, which serve as general-purpose mapping baselines, already provide sufficient accuracy to meet EasyColor's data requirements.

We have integrated these clarifications into the revised manuscript. We have added an explanation of the requirements for the point-cloud map in the Problem Definition section. The revised text is excerpted below:

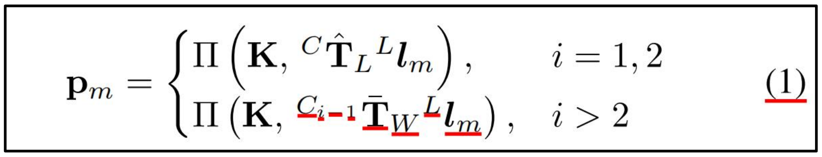

the $SE(3)$ manifold, representing the transformations from world to camera coordinate system, where ${}^{W}\mathbf{R}_{C_i} \in SO(3)$ is the rotation matrix and ${}^{W}\mathbf{t}_{C_i} \in \mathbb{R}^3$ is the translation vector of the i-th camera frame. To make the problem well-posed, we assume that the scene is static during data acquisition.

The problem is defined as follows: Given the input image sequence $\mathcal{I}$, the LiDAR point cloud map $\mathcal{L}$ (a general SLAM algorithm is sufficient to meet the accuracy requirements for point cloud map), and the rough LiDAR-camera extrinsic value ${}^{C}\hat{\mathbf{T}}_L$, our goal is to estimate the optimal camera poses ${}^{\mathbf{W}}\mathbf{T}_{\mathbf{C_i}}$ and colorize the point cloud map $\mathcal{L}$ with the corresponding RGB information from the camera image sequence $\mathcal{I}$. The camera poses allow us to project 3D points of dense LiDAR point cloud map onto the 2D image plane using camera intrinsic parameters $\mathbf{K}$.

3> *On equation 1, it is not clear if every point is converted to image frame. For i>2, does it not project to the world frame. Please clarify.*

>>Response:

Thank you so much for your reminder. In principle, every LiDAR point should be projected onto its corresponding image frame, and we fully agree with this requirement. We also acknowledge that Equation (1) contained a serious typographical error, which could lead to confusion. This typo has now been corrected in the revised manuscript.



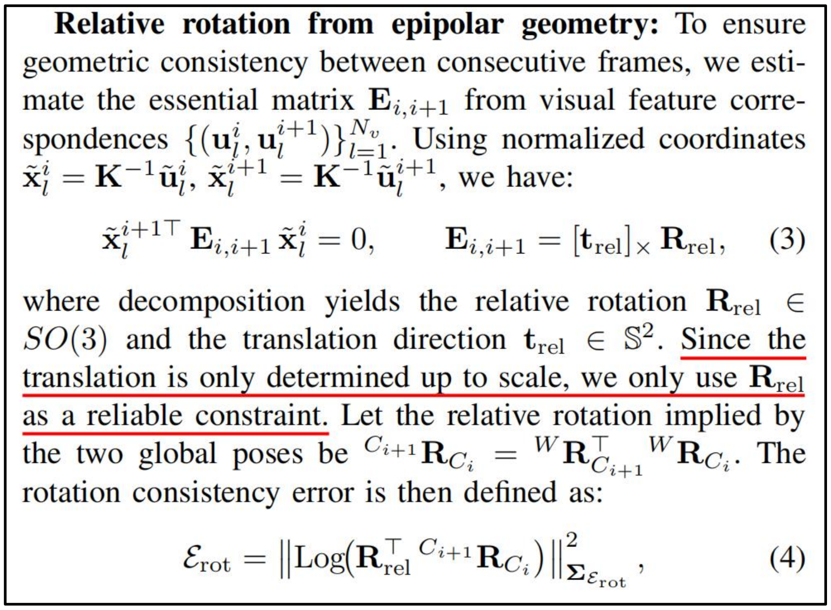
$$\mathbf{p}_m = \begin{cases} \Pi\left(\mathbf{K}, {}^{C}\hat{\mathbf{T}}_L {}^{L}\boldsymbol{l}_m\right), & i = 1, 2 \\ \Pi\left(\mathbf{K}, {}^{C_{i-1}}\bar{\mathbf{T}}_W {}^{L}\boldsymbol{l}_m\right), & i > 2 \end{cases} \tag{1}$$

In general multi-sensor SLAM setting, the pose of the camera at the first image frame is approximately aligned with the origin of the global point-cloud map, because the camera and LiDAR are rigidly mounted on the same platform. Therefore, for the first two image frames, we use the coarse extrinsic calibration to transform the LiDAR scan points from the world coordinate system into the corresponding camera coordinate systems of these initial frames. For the subsequent image frames, we rely on the optimized camera poses estimated from the previous frames. Specifically, the LiDAR points in the world frame are transformed into the camera frame of the preceding image using the refined pose estimates provided by the SLAM system. Although this procedure inevitably introduces pose inaccuracies, the inter-frame motion is moderate in our datasets, and EasyColor does not require strict time synchronization. As a result, the residual pose errors are generally tolerable and do not significantly affect the overall colorization quality.

4> *In epipolar constrainst why not use translation? The pose between two camera frames can obviously be estimated based on PnP which can easily constrain the translation. Am I missing something here?*

>>Response:

Thank you so much for your constructive comments. In fact, we addressed this point in the first version of the manuscript, and we have further strengthened the explanation in the revised version. Our consideration is that the epipolar constraint does not provide a true metric constraint on the translation vector, as the translation estimated from epipolar geometry is inherently defined up to an unknown scale. Although PnP can provide an estimated scale, this scale still needs to be treated as an additional optimization variable in the full system. Introducing such a scale parameter may increase the uncertainty of the optimization. For this reason, we chose not to rely on the translation component of the epipolar constraint, and instead use the available translation estimate directly. The revised text is excerpted below:



Relative rotation from epipolar geometry: To ensure geometric consistency between consecutive frames, we estimate the essential matrix $\mathbf{E}_{i,i+1}$ from visual feature correspondences $\{(\mathbf{u}_l^i, \mathbf{u}_l^{i+1})\}_{l=1}^{N_v}$. Using normalized coordinates $\tilde{\mathbf{x}}_l^i = \mathbf{K}^{-1}\tilde{\mathbf{u}}_l^i$, $\tilde{\mathbf{x}}_l^{i+1} = \mathbf{K}^{-1}\tilde{\mathbf{u}}_l^{i+1}$, we have:

$$\tilde{\mathbf{x}}_l^{i+1\top}\,\mathbf{E}_{i,i+1}\,\tilde{\mathbf{x}}_l^i = 0, \qquad \mathbf{E}_{i,i+1} = [\mathbf{t}_{\mathrm{rel}}]_\times\,\mathbf{R}_{\mathrm{rel}}, \tag{3}$$

where decomposition yields the relative rotation $\mathbf{R}_{\mathrm{rel}} \in SO(3)$ and the translation direction $\mathbf{t}_{\mathrm{rel}} \in \mathbb{S}^2$. Since the translation is only determined up to scale, we only use $\mathbf{R}_{\mathrm{rel}}$ as a reliable constraint. Let the relative rotation implied by the two global poses be ${}^{C_{i+1}}\mathbf{R}_{C_i} = {}^{W}\mathbf{R}_{C_{i+1}}^\top\,{}^{W}\mathbf{R}_{C_i}$. The rotation consistency error is then defined as:

$$\mathcal{E}_{\mathrm{rot}} = \left\|\mathrm{Log}\left(\mathbf{R}_{\mathrm{rel}}^\top\,{}^{C_{i+1}}\mathbf{R}_{C_i}\right)\right\|^2_{\mathbf{\Sigma}_{\mathcal{E}_{\mathrm{rot}}}}, \tag{4}$$

5> *Eq 5, I doubt we have photometric residuals?*

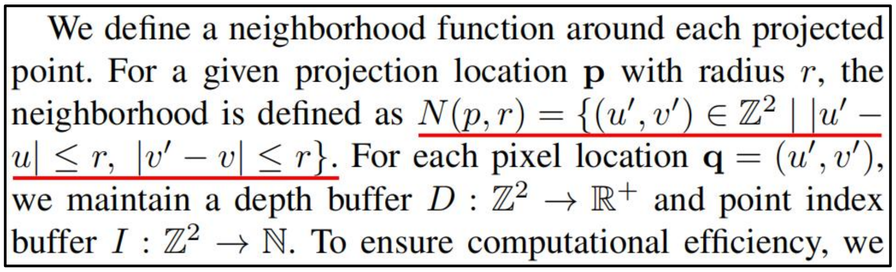
\>>Response:

Thank you so much for your valuable comments. We agree that photometric residuals could indeed be incorporated into Eq. 5. However, in this work, we intentionally simplify the solution of the joint optimization problem. Specifically, we first estimate the relative rotation between adjacent frames using the epipolar constraint, and then use this rotation estimate to further constrain the relative rotation obtained from the PnP solution. Therefore, although the inclusion of photometric residual is possible, it is not used in the overall optimization formulation. Moreover, the photometric residuals have already been used in the first step to optimize the epipolar constraint when estimating the relative rotation.

6> *the use of //・//∞ in Eq. 7’ s neighborhood definition) could be confusing —using L ∞ norm for pixel neighborhoods is unusual.*

\>>Response:

Thank you so much for your constructive comments. To avoid potential confusion, we removed the use of the L_∞norm in the neighborhood definition. The revised text is excerpted below:



We define a neighborhood function around each projected point. For a given projection location $\mathbf{p}$ with radius r, the neighborhood is defined as $N(p, r) = \{(u', v') \in \mathbb{Z}^2 \mid |u' - u| \le r,\ |v' - v| \le r\}$. For each pixel location $\mathbf{q} = (u', v')$, we maintain a depth buffer $D : \mathbb{Z}^2 \to \mathbb{R}^+$ and point index buffer $I : \mathbb{Z}^2 \to \mathbb{N}$. To ensure computational efficiency, we

7> *I am curious if this produces camera-LiDAR accurate calibration as byproduct or not. Have you tested that?*

\>>Response:

Thank you so much for your constructive comments. Although the inspiration for EasyColor originates from Koide’s reflectivity-based camera-LiDAR calibration method, our task is fundamentally different from extrinsic calibration. In EasyColor, the camera-LiDAR extrinsic parameters are implicitly absorbed into the camera poses expressed in the LiDAR map coordinate system. We have not attempted to explicitly decouple the extrinsic parameters from these poses, but this is indeed an interesting direction. We will include a discussion of this potential extension in the conclusion section of the manuscript.

Review 474229 (Reviewer27):

The overall framework is well structured and addresses a relevant problem in 3D perception and autonomous mapping. However, several key aspects require clarification and stronger experimental support. In particular, the core problem formulation, comparative validation, and evaluation strategy need refinement before the work can be considered for publication.

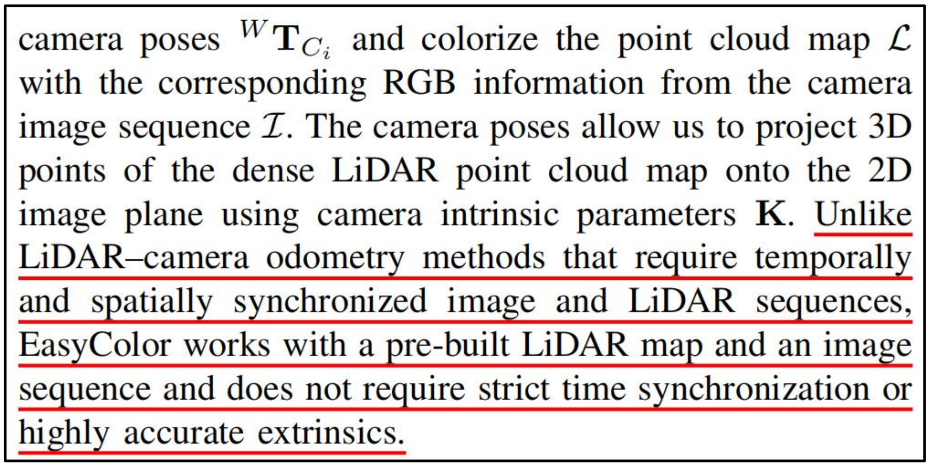
\>>Response:

We thank reviewer for the overall positive comments and valuable suggestions. The following is our point-by-point answer to reviewer's concerns.

1> *2.1 Task Framing and Problem Definition. Although the stated goal is dense point cloud colorization, the methodology effectively reduces the problem to estimating camera poses, resembling a visual odometry task aided by LiDAR intensity cues. The authors should clearly differentiate the objectives and outputs of EasyColor from standard LiDAR camera odometry or SLAM approaches:*

\>>Response:

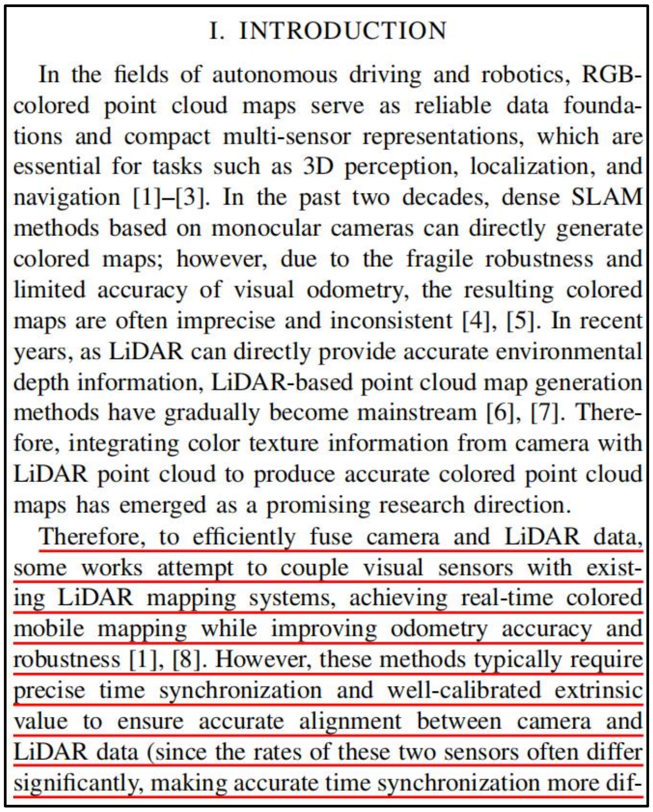
We thank you very much for your valuable suggestions. After extensive discussion and analysis within our team, we believe that EasyColor is fundamentally different from camera-LiDAR odometry. EasyColor requires a dense point cloud as input, whereas camera-LiDAR odometry typically takes a sequence of point cloud frames and image frames as input. Regarding the dense point cloud colorization task that EasyColor aims to solve, several existing works have indeed simplified this task into camera pose estimation, such as OminiColor (Liu, 2024) and ColmapPCD (Bai, 2024). This is because camera-based point cloud colorization has a clear physical interpretation: once an accurate camera pose is obtained, assigning colors from the camera to the point cloud follows a strict optical imaging relationship. We have added this explanation in the Problem Definition section. The revised text is excerpted below:


camera poses ${}^{W}\mathbf{T}_{C_i}$ and colorize the point cloud map $\mathcal{L}$ with the corresponding RGB information from the camera image sequence $\mathcal{I}$. The camera poses allow us to project 3D points of the dense LiDAR point cloud map onto the 2D image plane using camera intrinsic parameters $\mathbf{K}$. Unlike LiDAR–camera odometry methods that require temporally and spatially synchronized image and LiDAR sequences, EasyColor works with a pre-built LiDAR map and an image sequence and does not require strict time synchronization or highly accurate extrinsics.

2> *What unique challenges arise in colorization that are not addressed by odometry?*

\>>Response:

We thank you so much for your comment. The limitations of LiDAR–camera odometry in colorization tasks mainly stem from the difficulty of achieving high-precision time synchronization and extrinsic calibration between LiDAR and cameras, which prevents accurate alignment between point cloud and color information. We have highlighted these limitations in the Introduction section. The relevant revised text is provided below:



I. INTRODUCTION

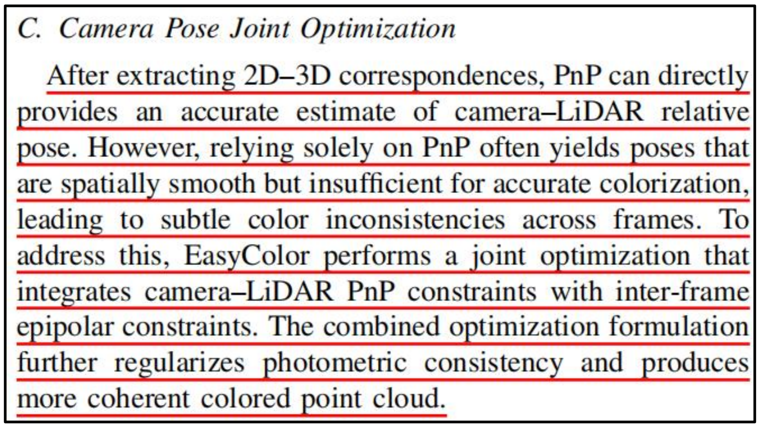
In the fields of autonomous driving and robotics, RGB-colored point cloud maps serve as reliable data foundations and compact multi-sensor representations, which are essential for tasks such as 3D perception, localization, and navigation [1]–[3]. In the past two decades, dense SLAM methods based on monocular cameras can directly generate colored maps; however, due to the fragile robustness and limited accuracy of visual odometry, the resulting colored maps are often imprecise and inconsistent [4], [5]. In recent years, as LiDAR can directly provide accurate environmental depth information, LiDAR-based point cloud map generation methods have gradually become mainstream [6], [7]. Therefore, integrating color texture information from camera with LiDAR point cloud to produce accurate colored point cloud maps has emerged as a promising research direction.

Therefore, to efficiently fuse camera and LiDAR data, some works attempt to couple visual sensors with existing LiDAR mapping systems, achieving real-time colored mobile mapping while improving odometry accuracy and robustness [1], [8]. However, these methods typically require precise time synchronization and well-calibrated extrinsic value to ensure accurate alignment between camera and LiDAR data (since the rates of these two sensors often differ significantly, making accurate time synchronization more dif-

3> *How does the optimization directly contribute to improved color consistency rather than just pose refinement? Clarifying this conceptual distinction is essential for evaluating the novelty of the work.*

>>Response:

We thank you so much for your valuable comments. In the overall optimization of EasyColor, the residuals $\mathcal{E}_{\mathrm{rot}}$ constructed from the epipolar constraint effectively enhance color consistency. Specifically, when the camera poses are optimized solely through PnP, the resulting poses are generally spatially smooth. However, directly using these poses for point cloud colorization often leads to slight color inconsistencies or misalignments. By incorporating inter-frame epipolar constraints, we further regularize the photometric differences between consecutive frames on top of the PnP pose constraints, which significantly improves color consistency. We have added clarifications in the revised manuscript, and the relevant updated text is provided below:



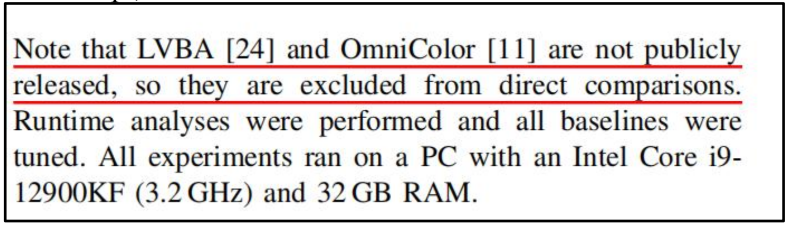
C. Camera Pose Joint Optimization

After extracting 2D–3D correspondences, PnP can directly provides an accurate estimate of camera–LiDAR relative pose. However, relying solely on PnP often yields poses that are spatially smooth but insufficient for accurate colorization, leading to subtle color inconsistencies across frames. To address this, EasyColor performs a joint optimization that integrates camera–LiDAR PnP constraints with inter-frame epipolar constraints. The combined optimization formulation further regularizes photometric consistency and produces more coherent colored point cloud.

4> *2.2 Lack of Comparative Evaluation A significant limitation is the absence of direct comparisons with other frameworks that perform the same task. The Related Work section lists prior methods such as OmniColor, LVBA, and Koide's approach, yet none are quantitatively compared. Including baseline comparisons on shared datasets (for example, KITTI-360 or SemanticKITTI) would significantly strengthen the contribution and demonstrate EasyColor's relative performance and applicability.*

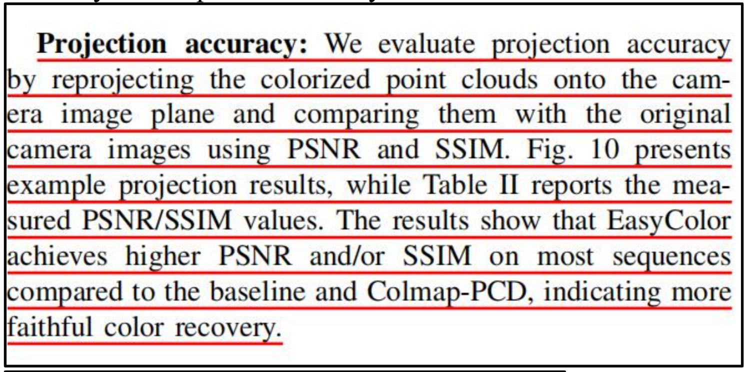
>>Response:

Thank you so much for your constructive comments. In accordance with your suggestions, we have added experiments on the reprojected images. Fig. 10 presents the qualitative results of the reprojected images, and Table 2 reports the quantitative evaluation in terms of PSNR and SSIM. Unfortunately, we are unable to compare EasyColor with methods such as OmniColor or LVBA, because these methods are not open-sourced. Although we contacted the authors for assistance, we did not receive any response. We have added this clarification to the manuscript, as shown below:



Note that LVBA [24] and OmniColor [11] are not publicly released, so they are excluded from direct comparisons. Runtime analyses were performed and all baselines were tuned. All experiments ran on a PC with an Intel Core i9-12900KF (3.2 GHz) and 32 GB RAM.

Therefore, we can only compare against Colmap-PCD (the baseline used in LVBA) and our self-implemented baseline. Koide's mutual-information-based method is designed for extrinsic calibration between sensors, where the inputs are a single image and a point cloud. It cannot be applied to the task of colorizing a point cloud map from a sequence of images. In addition to the figures and tables, we have also added a discussion on projection accuracy in the qualitative analysis section.



Projection accuracy: We evaluate projection accuracy by reprojecting the colorized point clouds onto the camera image plane and comparing them with the original camera images using PSNR and SSIM. Fig. 10 presents example projection results, while Table II reports the measured PSNR/SSIM values. The results show that EasyColor achieves higher PSNR and/or SSIM on most sequences compared to the baseline and Colmap-PCD, indicating more faithful color recovery.

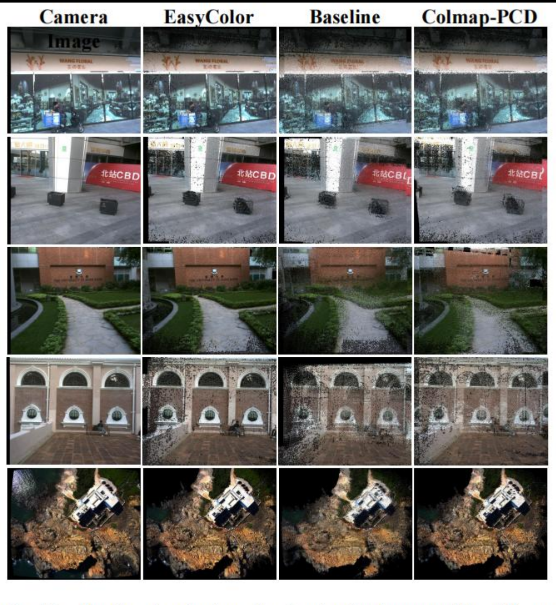


Fig. 10. Results of projecting colored point cloud generated by different methods on various public datasets onto image plane. Baseline result is obtained by coloring the point cloud using FastLIO2 [7] and official extrinsic file. A manual 0.1 s time offset was added to ensure a fair comparison.

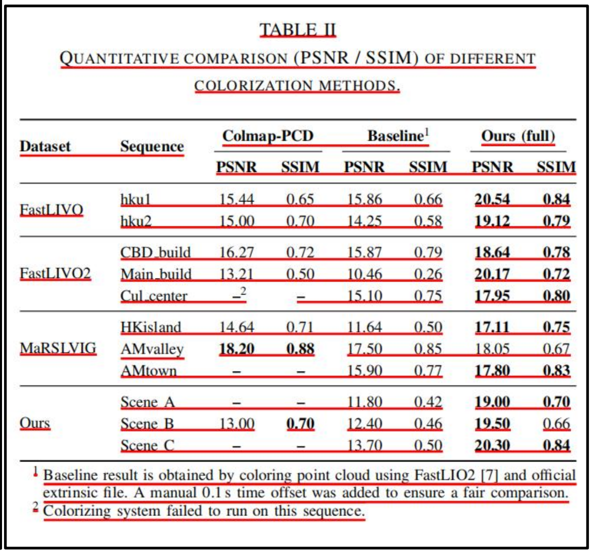


TABLE II

QUANTITATIVE COMPARISON (PSNR / SSIM) OF DIFFERENT COLORIZATION METHODS.

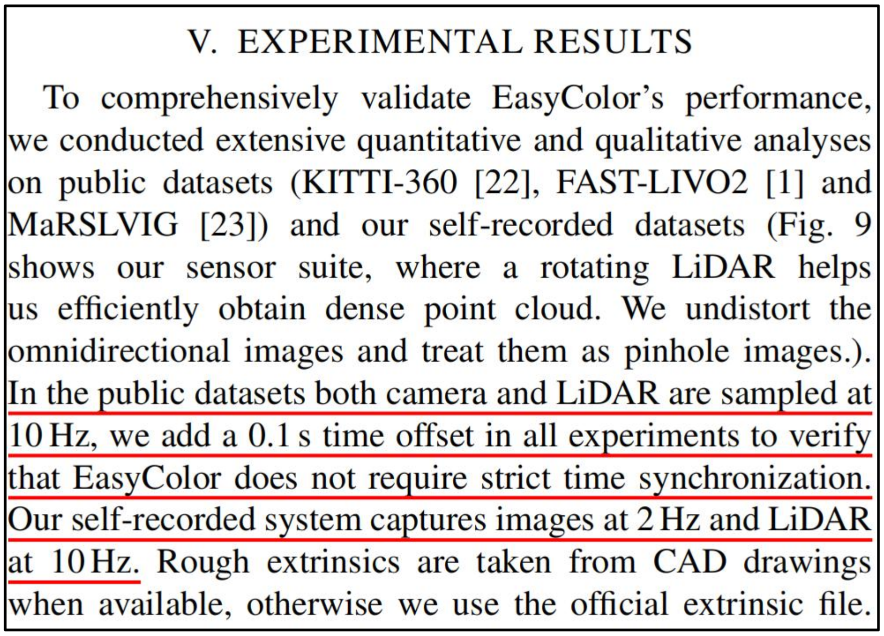
Dataset	Sequence	Colmap-PCD		Baseline[1]		Ours (full)	
		PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
FastLIVO	hku1	15.44	0.65	15.86	0.66	**20.54**	**0.84**
	hku2	15.00	0.70	14.25	0.58	**19.12**	**0.79**
FastLIVO2	CBD_build	16.27	0.72	15.87	0.79	**18.64**	**0.78**
	Main_build	13.21	0.50	10.46	0.26	**20.17**	**0.72**
	Cul_center	–[2]	–	15.10	0.75	**17.95**	**0.80**
MaRSLVIG	HKisland	14.64	0.71	11.64	0.50	**17.11**	**0.75**
	AMvalley	**18.20**	**0.88**	17.50	0.85	18.05	0.67
	AMtown	–	–	15.90	0.77	**17.80**	**0.83**
Ours	Scene A	–	–	11.80	0.42	**19.00**	**0.70**
	Scene B	13.00	**0.70**	12.40	0.46	**19.50**	0.66
	Scene C	–	–	13.70	0.50	**20.30**	**0.84**

[1] Baseline result is obtained by coloring point cloud using FastLIO2 [7] and official extrinsic file. A manual 0.1 s time offset was added to ensure a fair comparison.
[2] Colorizing system failed to run on this sequence.

5> *The paper claims that EasyColor does not require time synchronization between the LiDAR point cloud and image sequence. However, this is only discussed qualitatively. The authors should provide the frame rate or sampling frequency used in the experiments.*

>>Response:

We thank you so much for your valuable suggestions. As requested, we have added an explanation of the frame rate of the data acquisition devices at the beginning of the experimental section. The revised text is excerpted below:



V. EXPERIMENTAL RESULTS

To comprehensively validate EasyColor's performance, we conducted extensive quantitative and qualitative analyses on public datasets (KITTI-360 [22], FAST-LIVO2 [1] and MaRSLVIG [23]) and our self-recorded datasets (Fig. 9 shows our sensor suite, where a rotating LiDAR helps us efficiently obtain dense point cloud. We undistort the omnidirectional images and treat them as pinhole images.). In the public datasets both camera and LiDAR are sampled at 10 Hz, we add a 0.1 s time offset in all experiments to verify that EasyColor does not require strict time synchronization. Our self-recorded system captures images at 2 Hz and LiDAR at 10 Hz. Rough extrinsics are taken from CAD drawings when available, otherwise we use the official extrinsic file.

6> Describe the maximum tolerable time offset or frame gap beyond which performance deteriorates.

>>Response:

Thank you very much for your valuable suggestion. We have added a dedicated section on potential failure cases to clarify the conditions under which EasyColor may experience performance deterioration. As illustrated in Fig. 13, the essential factor that leads to system failure is the breakdown of cross-domain image matching. Therefore, any factor that may cause image-matching errors, such as large inter-frame motion, significant time offsets, or an inaccurate rough extrinsic that reduces the co-visible region between the reflectivity map and the camera image, can indirectly degrade the performance of EasyColor. These risks are difficult to quantify precisely. In practice, EasyColor only requires that adjacent frames maintain a sufficient co-visible region. Based on our experiments, as long as the time offset between the camera and LiDAR is within approximately 5 seconds and the data acquisition platform moves smoothly, the system can operate reliably.

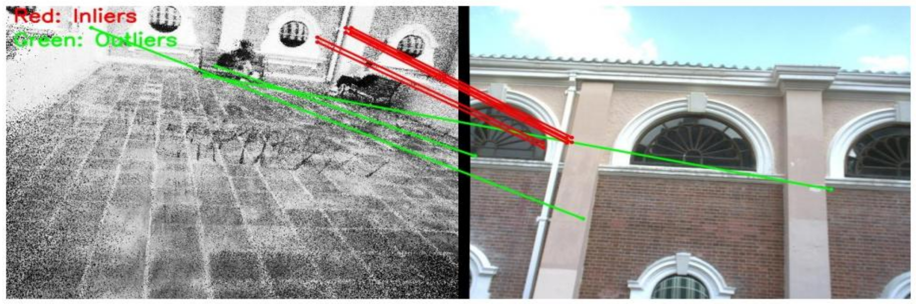
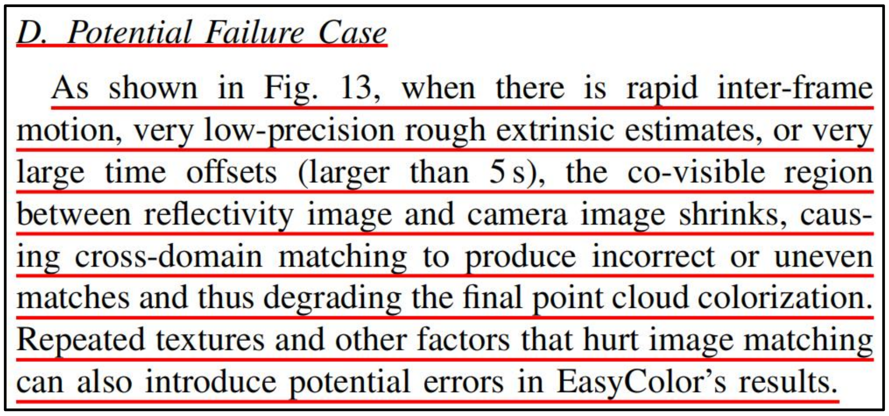


Fig. 13. A potential failure case of EasyColor.(red: inliers, green: outliers)

In addition to the incorrect matching example shown in Fig. 13, we have also added a separate section to explain the potential limitations of EasyColor, as follows:



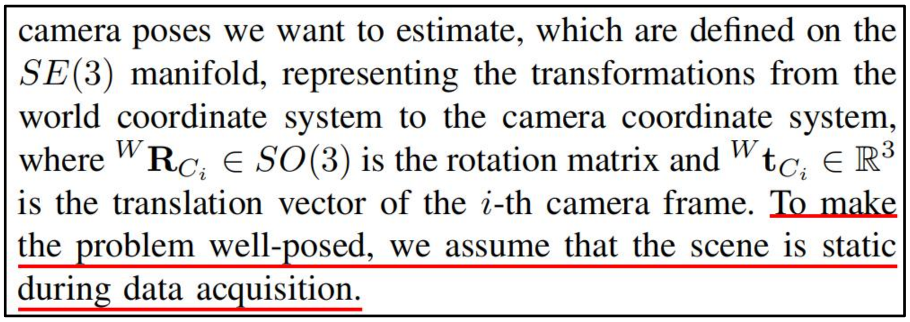
D. Potential Failure Case

As shown in Fig. 13, when there is rapid inter-frame motion, very low-precision rough extrinsic estimates, or very large time offsets (larger than 5 s), the co-visible region between reflectivity image and camera image shrinks, causing cross-domain matching to produce incorrect or uneven matches and thus degrading the final point cloud colorization. Repeated textures and other factors that hurt image matching can also introduce potential errors in EasyColor's results.

7> *Clarify whether the algorithm assumes continuous motion or static scenes.*

>>Response:

We thank you very much for your valuable suggestions. EasyColor assumes a static scene, ensuring that LiDAR SLAM does not produce point-cloud ghosting caused by dynamic objects during map construction. We have added this clarification in the Problem Definition section of the manuscript. The revised text is excerpted below:



camera poses we want to estimate, which are defined on the $SE(3)$ manifold, representing the transformations from the world coordinate system to the camera coordinate system, where ${}^{W}\mathbf{R}_{C_i} \in SO(3)$ is the rotation matrix and ${}^{W}\mathbf{t}_{C_i} \in \mathbb{R}^3$ is the translation vector of the i-th camera frame. To make the problem well-posed, we assume that the scene is static during data acquisition.

8> *2.4 Quantitative Evaluation of Colorization The authors state that assessing the colorization quality is difficult due to the absence of ground truth and dismiss existing 2D-based metrics as unsuitable. However, a 2D projection-based metric could still offer useful quantitative insights. It is strongly recommended to include at least one approximate evaluation metric, such as: Projection-based PSNR or SSIM between reprojected colorized views and camera images, or A pixel-wise color consistency error across adjacent frames. Even if imperfect, these would provide a reproducible measure of colorization performance.*

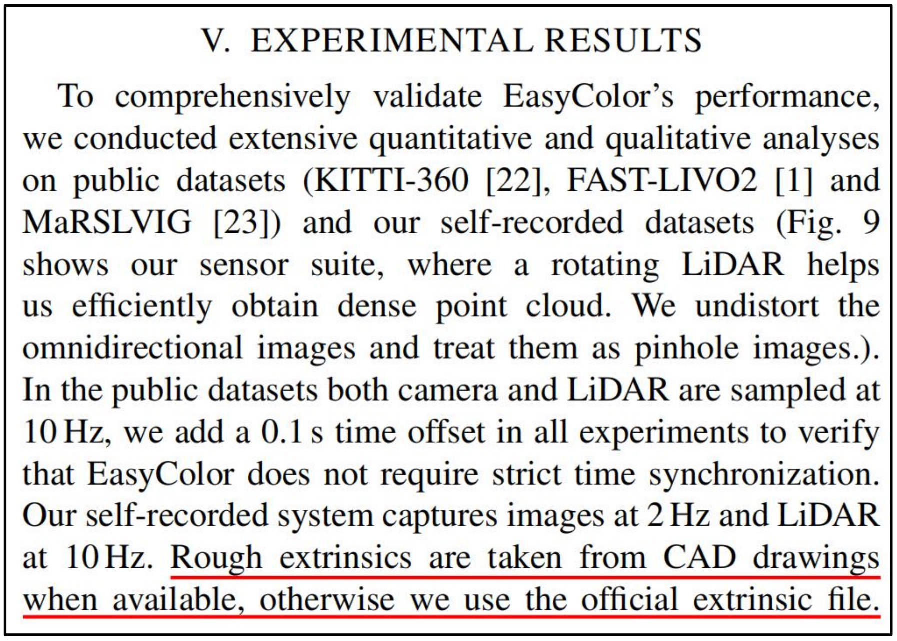
>>Response:

Thank you so much for your constructive comments. In accordance with your suggestions, we have added experiments on the reprojected images. Fig. 10 presents the qualitative results of the reprojected images, and Table 2 reports the quantitative evaluation in terms of PSNR and SSIM. Unfortunately, we are unable to compare EasyColor with methods such as OmniColor or LVBA, because these methods are not open-sourced. Although we contacted the authors for assistance, we did not receive a response. Therefore, we can only compare against Colmap-PCD (the baseline used in LVBA) and our self-implemented baseline. Koide's mutual-information-based method is designed for extrinsic calibration between sensors, where the inputs are a single image and a point cloud. It cannot be applied to the task of colorizing a point cloud map from a sequence of images.

9> *Provide more detail on how rough extrinsic parameters are initialized (manual estimation or default sensor setup).*

>>Response:

We thank you so much for your valuable suggestions. As requested, we have added the relevant technical details at the beginning of the experimental section. The revised text is excerpted below:



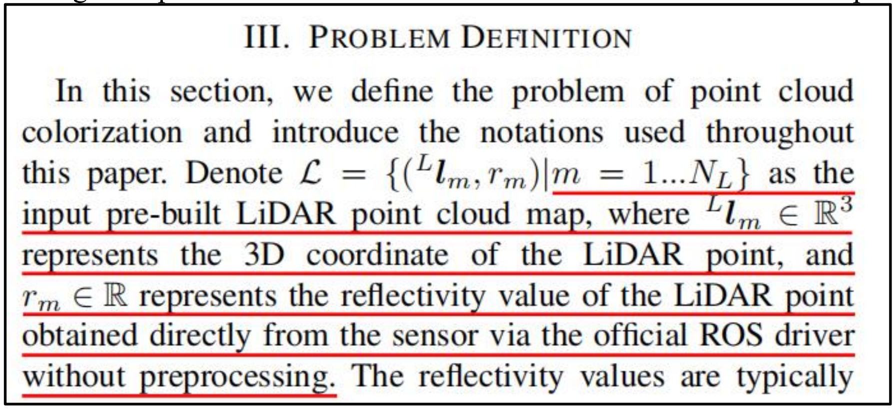
V. EXPERIMENTAL RESULTS

To comprehensively validate EasyColor's performance, we conducted extensive quantitative and qualitative analyses on public datasets (KITTI-360 [22], FAST-LIVO2 [1] and MaRSLVIG [23]) and our self-recorded datasets (Fig. 9 shows our sensor suite, where a rotating LiDAR helps us efficiently obtain dense point cloud. We undistort the omnidirectional images and treat them as pinhole images.). In the public datasets both camera and LiDAR are sampled at 10 Hz, we add a 0.1 s time offset in all experiments to verify that EasyColor does not require strict time synchronization. Our self-recorded system captures images at 2 Hz and LiDAR at 10 Hz. Rough extrinsics are taken from CAD drawings when available, otherwise we use the official extrinsic file.

10> Clarify whether LiDAR reflectivity intensity is preprocessed (normalized, filtered, or range-compensated).

>>Response:

We thank you very much for your valuable suggestion. In our implementation, we directly use the reflectivity intensity values provided by the official LiDAR driver, without applying any additional normalization, filtering, or range compensation. We have added this clarification to the manuscript. The revised text is excerpted below:



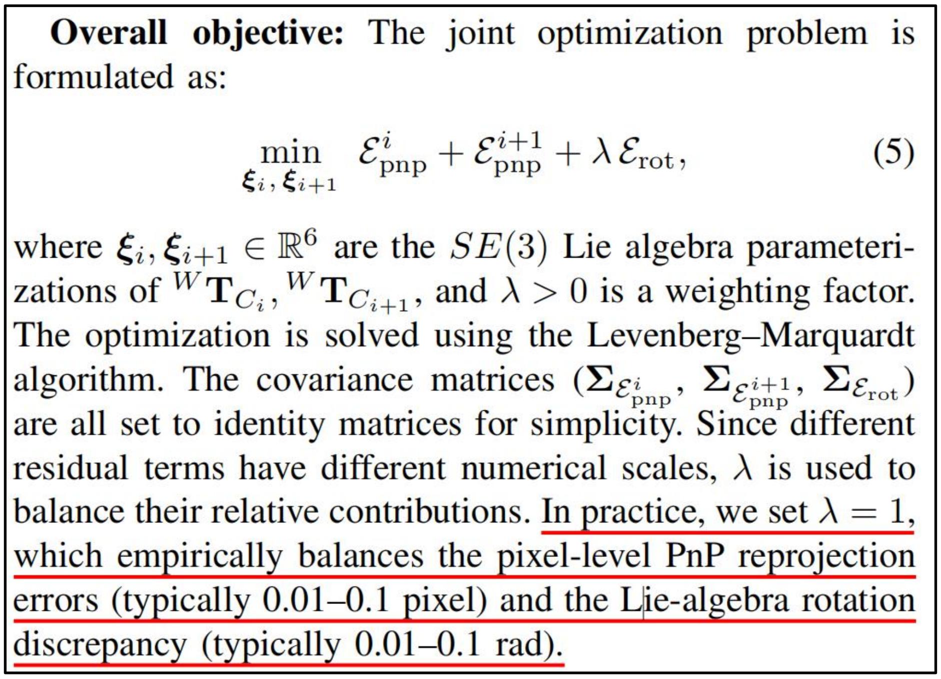
III. PROBLEM DEFINITION

In this section, we define the problem of point cloud colorization and introduce the notations used throughout this paper. Denote $\mathcal{L} = \{({}^L\boldsymbol{l}_m, r_m) | m = 1...N_L\}$ as the input pre-built LiDAR point cloud map, where ${}^L\boldsymbol{l}_m \in \mathbb{R}^3$ represents the 3D coordinate of the LiDAR point, and $r_m \in \mathbb{R}$ represents the reflectivity value of the LiDAR point obtained directly from the sensor via the official ROS driver without preprocessing. The reflectivity values are typically

11> Indicate how the optimization weights were chosen (λ parameters in Eq. 5).

>>Response:

We thank you very much for your valuable suggestions. In practice, we set (λ = 1), which balances the pixel-level PnP reprojection errors (measured in pixels, typically 0.01–0.1 pixel) and the rotation discrepancy in the Lie algebra (measured in radians, typically 0.01–0.1). This value also provides robust performance under varying texture and illumination conditions. We have added clarifications in the corresponding sections of the manuscript. The revised text is excerpted below:



Overall objective: The joint optimization problem is formulated as:

$$\min_{\boldsymbol{\xi}_i,\, \boldsymbol{\xi}_{i+1}} \quad \mathcal{E}_{\text{pnp}}^{i} + \mathcal{E}_{\text{pnp}}^{i+1} + \lambda\, \mathcal{E}_{\text{rot}}, \tag{5}$$

where $\boldsymbol{\xi}_i, \boldsymbol{\xi}_{i+1} \in \mathbb{R}^6$ are the $SE(3)$ Lie algebra parameterizations of ${}^{W}\mathbf{T}_{C_i}, {}^{W}\mathbf{T}_{C_{i+1}}$, and $\lambda > 0$ is a weighting factor. The optimization is solved using the Levenberg–Marquardt algorithm. The covariance matrices ($\boldsymbol{\Sigma}_{\mathcal{E}_{\text{pnp}}^{i}}$, $\boldsymbol{\Sigma}_{\mathcal{E}_{\text{pnp}}^{i+1}}$, $\boldsymbol{\Sigma}_{\mathcal{E}_{\text{rot}}}$) are all set to identity matrices for simplicity. Since different residual terms have different numerical scales, λ is used to balance their relative contributions. In practice, we set $\lambda = 1$, which empirically balances the pixel-level PnP reprojection errors (typically 0.01–0.1 pixel) and the Lie-algebra rotation discrepancy (typically 0.01–0.1 rad).

12> Discuss potential failure cases, such as dynamic objects or non-overlapping fields of view.

>>Response:

Thank you very much for your valuable suggestion. We have added a dedicated section on potential failure cases to clarify the conditions under which EasyColor may experience performance deterioration. As illustrated in Fig. 13, the essential factor that leads to system failure is the breakdown of cross-domain image matching. Therefore, any factor that may cause image-matching errors, such as large inter-frame motion, significant time offsets, or an inaccurate rough extrinsic that reduces the co-visible region between the reflectivity map and the camera image, can indirectly degrade the performance of EasyColor. These risks are difficult to quantify precisely. In practice, EasyColor only requires that adjacent frames maintain a sufficient co-visible region. Based on our experiments, as long as the time offset between the camera and LiDAR is within approximately 5 seconds and the data acquisition platform moves smoothly, the system can operate reliably.

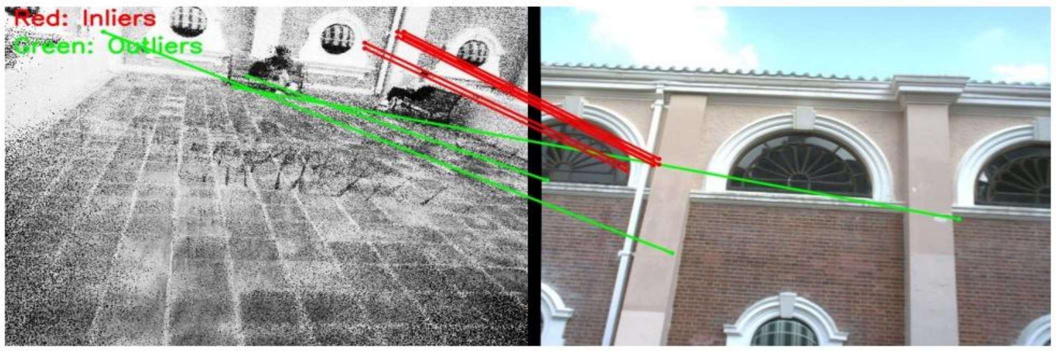
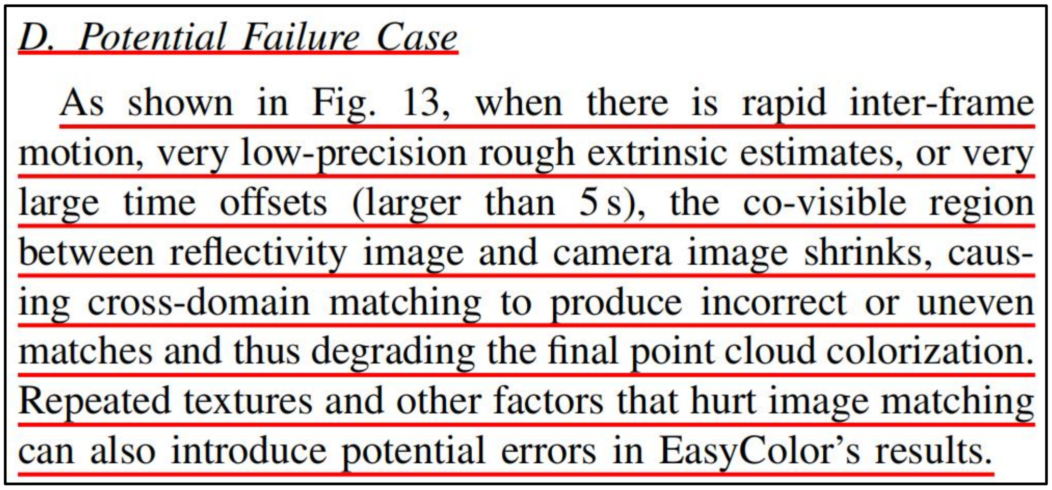


Fig. 13. A potential failure case of EasyColor.(red: inliers, green: outliers)

In addition to the incorrect matching example shown in Fig. 13, we have also added a separate section to explain the potential limitations of EasyColor, as follows:



D. Potential Failure Case

As shown in Fig. 13, when there is rapid inter-frame motion, very low-precision rough extrinsic estimates, or very large time offsets (larger than 5 s), the co-visible region between reflectivity image and camera image shrinks, causing cross-domain matching to produce incorrect or uneven matches and thus degrading the final point cloud colorization. Repeated textures and other factors that hurt image matching can also introduce potential errors in EasyColor's results.

13> Figures are clear but somewhat overloaded; simplifying Figures 2, 6, and 10 could improve readability.

>>Response:

Thank you very much for your valuable suggestions. Due to time constraints, simplifying and redesigning the entire algorithm flowchart is challenging for us at the current stage. Instead, we have adjusted the layout of Fig. 11 to make it more compact and clearer. We will redesign Fig. 2 and Fig. 6 in the final camera-ready version.

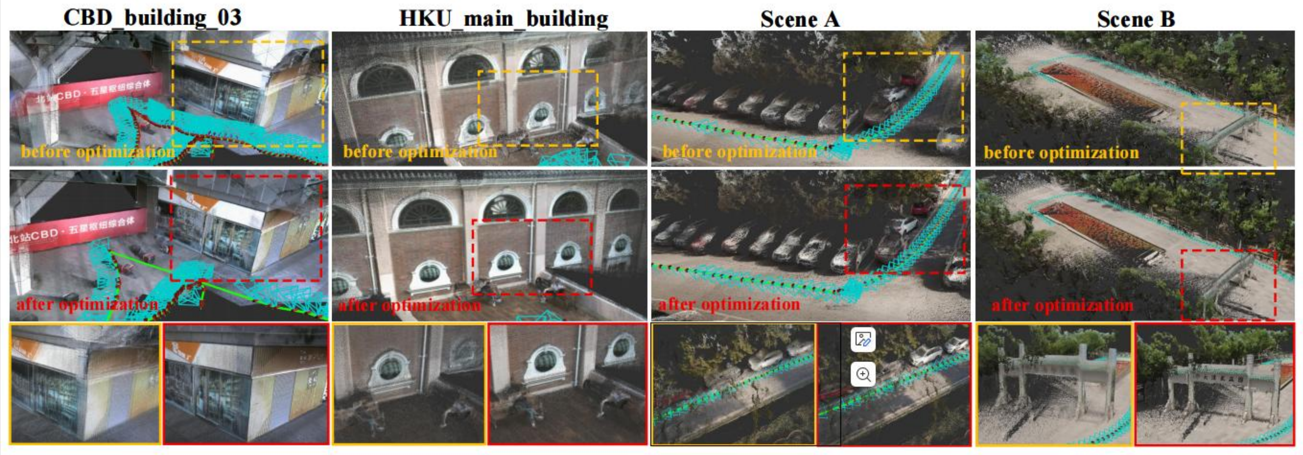


Fig. 11. Qualitative comparison of point cloud colorization results on FastLIVO2 dataset (CBD and main building sequence) and our self-recorded dataset (Scene A and Scene B). The reflectivity point cloud map for FastLIVO2 dataset is built using FastLIO2 [6], while the map for our dataset is generated with FastLIO2 [6] using our sensor suite (as shown in Fig. 9). Yellow boxes highlight unoptimized details. Red boxes highlight optimized details.

14> Typographical corrections: "cosrresponding" → "corresponding"; pf unreliable" → "of unreliable."

>>Response:

We thank you so much for your valuable reminder. We first corrected the typo you pointed out, and we also reviewed the entire manuscript to ensure that no other typos were overlooked.

Review 474235 (Reviewer30):

The paper was well written and is considerated to be accepted in current form.

>>Response:

We thank reviewer for the positive comment and suggestion. We have made a thorough revision of the writing (refined the language, improved the grammar and fixed all the offending typos) to improve the readability and facilitate understanding. Moreover, the writing is refined to be more concise and the content is better organized.

Editor:

This paper introduces EasyColor, a method for colorizing dense LiDAR point clouds from image sequences without precise time synchronization or extrinsic calibration. It uses LiDAR reflectivity to generate intensity images, aligns them with RGB frames via cross-domain matching, and refines camera poses through joint optimization based on frame consistency and epipolar geometry. Overall, the paper is well-written, well-structured, and well-illustrated. It addresses a relevant theme for the RA-L community.

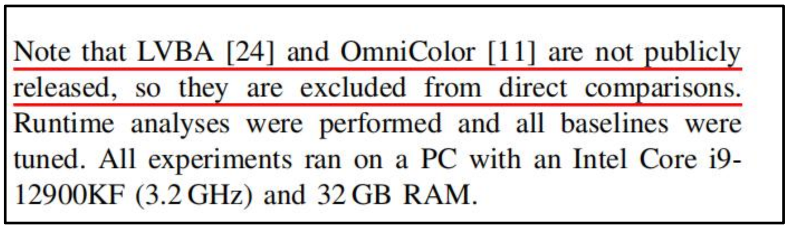
>>Response:

We sincerely thank the editor for the valuable suggestions and comments. We have carefully addressed all reviewers' feedback and enriched the revised manuscript with additional technical details and clarifications to further strengthen its clarity and credibility.

1> However, several key Aspects require more clarification (please see the review reports from reviewers 1 and 2). The work would benefit from more convincing experimental support using well-established datasets and benchmarks. Moreover, the proposed method should be compared to the state of the art. The evaluation protocol and metrics should also be clarified.

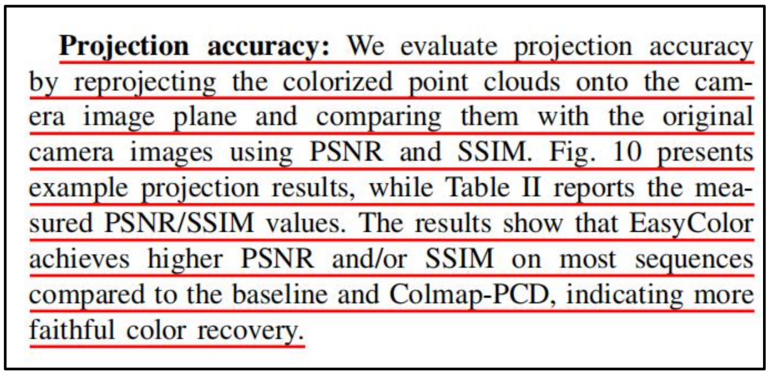
>>Response:

We thank you so much for your valuable suggestions. In accordance with your suggestions, we have added experiments on the reprojected images. Fig. 10 presents the qualitative results of the reprojected images, and Table 2 reports the quantitative evaluation in terms of PSNR and SSIM. Unfortunately, we are unable to compare EasyColor with methods such as OmniColor or LVBA, because these methods are not open-sourced. Although we contacted the authors for assistance, we did not receive a response. We have added this clarification to the manuscript, as shown below:



Note that LVBA [24] and OmniColor [11] are not publicly released, so they are excluded from direct comparisons. Runtime analyses were performed and all baselines were tuned. All experiments ran on a PC with an Intel Core i9-12900KF (3.2 GHz) and 32 GB RAM.

Therefore, we can only compare against Colmap-PCD (the baseline used in LVBA) and our self-implemented baseline. Koide's mutual-information-based method is designed for extrinsic calibration between sensors, where the inputs are a single image and a point cloud. It cannot be applied to the task of colorizing a point cloud map from a sequence of images. In addition to the figures and tables, we have also added a discussion on projection accuracy in the qualitative analysis section.



Projection accuracy: We evaluate projection accuracy by reprojecting the colorized point clouds onto the camera image plane and comparing them with the original camera images using PSNR and SSIM. Fig. 10 presents example projection results, while Table II reports the measured PSNR/SSIM values. The results show that EasyColor achieves higher PSNR and/or SSIM on most sequences compared to the baseline and Colmap-PCD, indicating more faithful color recovery.

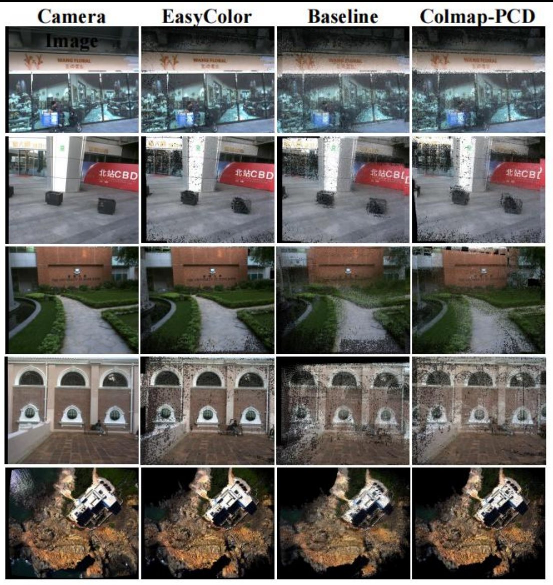


Fig. 10. Results of projecting colored point cloud generated by different methods on various public datasets onto image plane. Baseline result is obtained by coloring the point cloud using FastLIO2 [7] and official extrinsic file. A manual 0.1 s time offset was added to ensure a fair comparison.

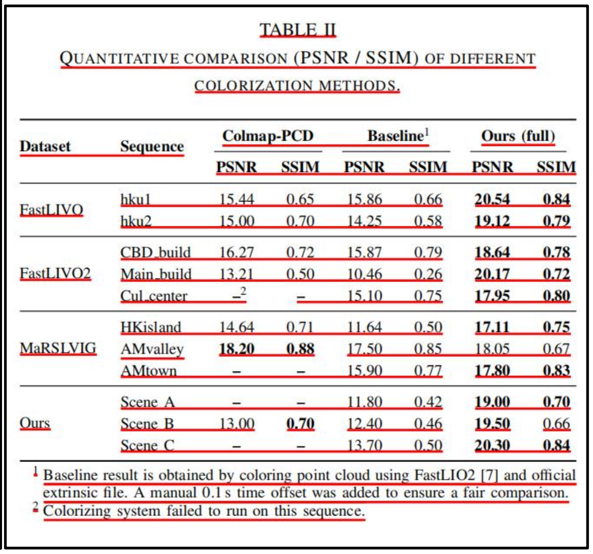


TABLE II

QUANTITATIVE COMPARISON (PSNR / SSIM) OF DIFFERENT COLORIZATION METHODS.

Dataset	Sequence	Colmap-PCD		Baseline[1]		Ours (full)	
		PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
FastLIVO	hku1	15.44	0.65	15.86	0.66	**20.54**	**0.84**
	hku2	15.00	0.70	14.25	0.58	**19.12**	**0.79**
FastLIVO2	CBD_build	16.27	0.72	15.87	0.79	**18.64**	**0.78**
	Main_build	13.21	0.50	10.46	0.26	**20.17**	**0.72**
	Cul_center	–[2]	–	15.10	0.75	**17.95**	**0.80**
MaRSLVIG	HKisland	14.64	0.71	11.64	0.50	**17.11**	**0.75**
	AMvalley	**18.20**	**0.88**	17.50	0.85	18.05	0.67
	AMtown	–	–	15.90	0.77	**17.80**	**0.83**
Ours	Scene A	–	–	11.80	0.42	**19.00**	**0.70**
	Scene B	13.00	**0.70**	12.40	0.46	**19.50**	0.66
	Scene C	–	–	13.70	0.50	**20.30**	**0.84**

[1] Baseline result is obtained by coloring point cloud using FastLIO2 [7] and official extrinsic file. A manual 0.1 s time offset was added to ensure a fair comparison.
[2] Colorizing system failed to run on this sequence.

2> Three reviewers with prior experience in computer vision and perception evaluated the manuscript. The reviewers comments and recommendations are attached. Based on the reviewers feedback (reviewer 3 is not taken into account because their evaluation is not substantial) and my own review, I suggest that the paper cannot be accepted for publication in RA-L.

>>Response:

We thank you very much for your valuable comments. We are confident that the revised manuscript has addressed the concerns raised by both the reviewers and the editor.