2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
October 1-5, 2023. Detroit, USA

Rollvox: Real-time and High-quality LiDAR Colorization with Rolling Shutter Camera

Sheng Hong[1], Chunran Zheng[2], Huan Yin[1†], Shaojie Shen[1†]

***Abstract*— In this study, we propose a novel system for real-time coloring LiDAR point clouds with a low-cost RS camera. The main challenges are dealing with the motion distortion of the RS camera and the multi-sensor time synchronization. To tackle these challenges, we carefully design a hardware synchronizer to ensure the strict alignment of the LiDAR, inertial measurement unit, and RS camera. With accurate timestamps, we first use LiDAR-inertial odometry (LIO) for pose estimation, and the poses of image line exposure are calculated by forward propagation based on a constant velocity motion model. Then, we propose our method based on the RS constraint for colorizing the LiDAR point cloud. For comparison, we colorize the LiDAR point cloud with conventional rolling shutter image undistortion. In the real-world tests, The results show that our proposed method produces more accurate and efficient colorization of point clouds. Besides, considering the situation of readout time not being provided, we propose a method to calibrate the readout time by minimizing the reprojection error of LIO's inter-frame pose and image optical flows. We release our code and self-collected datasets on Github[3] to benefit the community.**

I. Introduction

LiDAR mapping techniques, including LiDAR simultaneous localization and mapping (SLAM), have demonstrated significant success in providing high-precision and dense 3D point clouds for robotic applications. With the increasing interest in LiDAR colorization, which involves assigning RGB colors to 3D LiDAR points, colored LiDAR point clouds are being utilized in various applications, including autonomous driving [1], 3D scene reconstruction [2], virtual reality [3], and digital twins [4].

However, existing frameworks for LiDAR colorization rely on expensive global shutter (GS) cameras to capture images. For instance, widely-used LiDAR-inertial visual odometry (LIVO) systems such as LIC-fusion [5], LVI-SAM [6], CamVox [7], R3LIVE [8], and FAST-LIVO [9] necessitate GS cameras for colorization or state estimation. In R3LIVE [8], a tightly-coupled state estimation and mapping system that integrates LiDAR, inertial sensors, and a global shutter camera is used to render the colored LiDAR point cloud in real time. To colorize the point cloud, R3LIVE estimates the camera intrinsic and LiDAR-camera extrinsic parameters and the LiDAR-camera frame timestamp offset online. Similarly, Zheng et al. [9] presented FAST-LIVO, which reduces the number of state variables to be estimated by utilizing pre-calibrated camera intrinsic and LiDAR-camera extrinsic parameters and hardware-synchronized camera and LiDAR frames.

The reason for the widespread use of GS cameras in these multi-sensor colorization systems is due to their distortion-free image capturing. GS cameras, equipped with CCD sensors [10], are typically used in professional photography, cinematography, and machine vision applications since they expose all pixels on their sensors simultaneously, which eliminate motion artifacts and enable the camera to capture a clear image of a moving object. However, compared to rolling shutter (RS) cameras, GS cameras are typically more expensive and less sensitive to light.

On the other hand, RS cameras, equipped with CMOS sensors, offer better image quality and are a popular low-cost option for many applications, such as mobile phones, music players, and entry-level cameras [11]. Nevertheless, RS cameras have certain limitations, especially when capturing fast-moving objects. The sequential readout in RS cameras leads to slightly varying exposure times across rows, which can cause motion distortion and affect the accuracy of colorization. This presents a significant challenge for many applications, making RS cameras a less preferred choice for colorization.

In the field of visual odometry, there are also works on RS cameras, such as the work by Schubert et al. [12], which introduced a direct sparse odometry algorithm for RS cameras based on Meingast, M., Geyer, C., & Sastry, S. [13]. Geometric models of rolling-shutter cameras. The algorithm accurately locates the true projection position and is computationally intensive. Lang et al. [13] propose a probabilistic continuous-time visual-inertial odometry (VIO) for RS cameras. Li et al. [14] deployed the monocular VINS estimator [15] system on the mobile phone with an IMU and RS camera and made an augmented reality demonstration. Their experiments of visual odometry show accurate results with the consideration of the rolling shutter model.

To stabilize video captured by rolling shutter (RS) cameras, many well-established algorithms have been developed [16]–[19]. One representative work, proposed by Karpenko et al. [18], uses gyroscopes to stabilize images from RS cameras. This method relies on IMU-based pose estimation and online estimation of the camera and IMU delay and RS camera readout time. However, obtaining accurate translation motion from IMU can be difficult, so this method only considers the rotational motion. Furthermore, it

† Corresponding Author
[1] Department of Electronic Computer Engineering, The Hong Kong University of Science and Technology, Hong Kong SAR, China. shongap@connect.ust.hk, eehyin@ust.hk, eeshaojie@ust.hk.
[2] Department of Mechanical Engineering, The University of Hong Kong, Hong Kong SAR, China. zhengcr@connect.hku.hk
[3] https://github.com/sheng00125/Rollvox

2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) | 978-1-6654-9190-7/23/$31.00 ©2023 IEEE | DOI: 10.1109/IROS55552.2023.10342172

assumes that the depth of pixels in an image remains consistent across all rows. Despite these limitations, Karpenko et al.'s method effectively compensates for motion distortion caused by RS cameras and has been successfully deployed on an iPhone 4. Other researchers, such as Forssén and Ringaby [19], have attempted to include camera translation motion in their models but found that the results were inferior to those obtained using only a rotational model. Additionally, obtaining depth information using stereo or a feature-based structure from motion (SfM) algorithms is not always robust and often requires significant computational resources.

Inspired by previous GS-based colorization systems and RS-based visual localization approaches, we propose a novel and effective method for colorizing LiDAR point clouds using a RS camera in this study. The proposed method takes advantage of the precise positioning of the LIO system and the accurate measurement of LiDAR for 3D measurement, utilizing the rolling shutter constraint method to accurately iteratively locate the true projection position of LiDAR points in the rolling shutter camera for colorization. Our method solves the problem of color misregistration on the point cloud caused by the motion distortion of the RS camera, improves the accuracy of the colorization algorithm, and provides a new and effective approach for rendering LiDAR point clouds.

Overall, the primary contributions of this study can be summarized as follows:

- Development of a novel colorization approach that enhances the accuracy and efficiency of the algorithm specifically for RS cameras, surpassing conventional undistortion techniques.
- Introduction of a hardware synchronization strategy that ensures precise alignment of the RS camera, IMU, and LiDAR sensors to achieve a more precise sensor timestamp.
- Proposal of a readout time calibration technique that further improves the accuracy of the colorization algorithm by minimizing the reprojection error of inter-frame pose interpolation and the LK optical flow of two frames.

To our knowledge, this study is the first to propose a method for colorizing LiDAR point clouds using a RS camera. Our novel algorithm accurately locates the true projection position of LiDAR points in RS camera, allowing for precise and efficient colorization of LiDAR point clouds. We have extensively evaluated and validated our proposed method through experimentation, and our results demonstrate its superior performance compared to traditional RS camera distortion correction methods. Furthermore, we validate the proposed RS camera colorization method in the FAST-LIO framework [20] [21], demonstrating its effectiveness in a real-world scenario.

II. System Overview

The Rollvox system overview is depicted in Fig. 2, which comprises two primary modules: the LiDAR point clouds colorization module and the RS camera readout time calibration module.

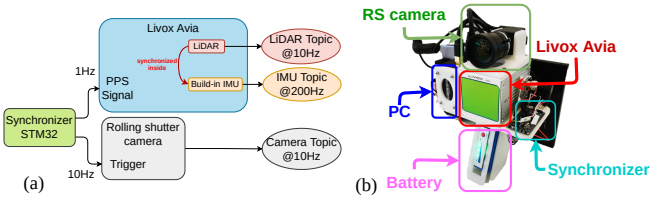

Fig. 1. Hardware system of Rollvox. (a) the diagram of our hardware system (b) our hardware platform for data acquisition.

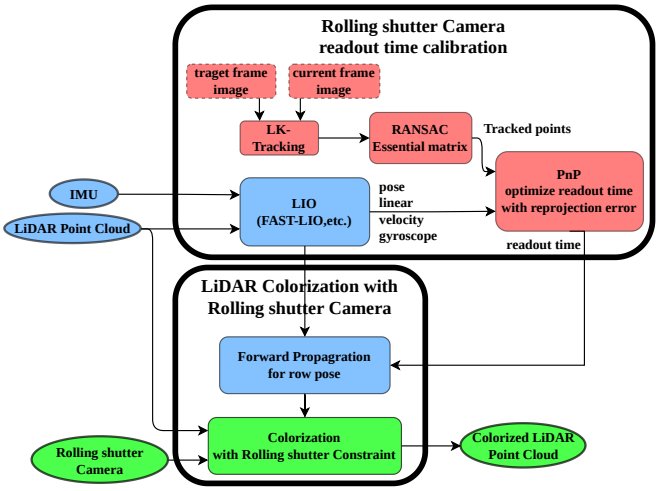

Fig. 2. The software system overview of Rollvox.

Estimating the readout time is a critical factor for forward propagation, which involves calculating the pose of each image row. However, In practical applications, the readout time depends on several factors, including the camera's binning, decimation, Image ROI, etc. camera manufacturers only sometimes provide readout time for users. Therefore, It is necessary to design a readout time calibration module that optimizes the camera's readout time with the reprojection error of LIO pose and LK tracking of inter-frames.

Once the readout time is calibrated, then forward propagation can be used to calculate each pose of the line exposure of the RS camera. The forward propagation process uses a constant velocity motion model to interpolate the camera's pose at each line exposure time based on the calibrated readout time. Only with the camera's pose of each line, the point clouds can be correctly colorized based on the camera's motion. More specifically, the actual projection position on the image plane of each LiDAR point can be accurately computed based on the rolling shutter constraint principle to colorize the LiDAR point clouds.

The hardware setup is depicted in fig. 1, which comprises an onboard computer DJI manifold-2C (with an Intel i7-8550u CPU) and 8 GB RAM, a Hik-vision industrial camera (MV-CE120-10UC), and a Livox Avia LiDAR. An embedded MCU is adapted to generate a 2-channel PWM. A 1Hz & 10Hz PWM is generated with a rising edge aligned to synchronize the LiDAR and camera.

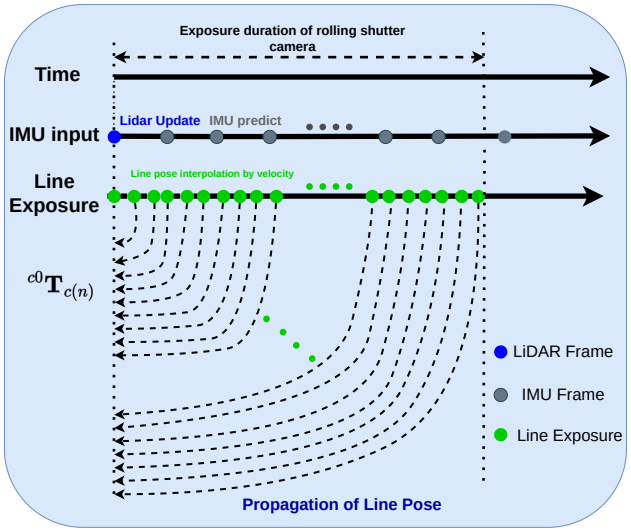

Fig. 3. Scheme of forward propagation for the camera pose at line exposure time

III. METHODOLOGY

The previous section introduced the software modules and hardware setup of the system. In this section, we provide a detailed explanation of the algorithms used in each module. Firstly, we describe the forward propagation algorithm, based on the constant velocity motion model, that calculates precise poses for each row. We then highlight issues with traditional rolling shutter motion rectification methods for point cloud colorization, even with precise row poses. Next, we explain how to use the rolling shutter constraint iterative approach to precisely project and colorize the LiDAR point clouds. Finally, we present a readout time calibration method to address inaccuracies in the readout time.

A. Forward Propagation

The forward propagation algorithm is employed to calculate the camera pose at each exposure time for every frame captured by the RS camera, based on the posture provided by LIO and the readout time. As illustrated in Fig. 3, when a LiDAR frame is received, a LiDAR update is performed in the LIO system, and a predicted operation is performed for each received IMU frame. The LiDAR points undergo "motion undistortion" [20] by the posture of interpolation of IMU and are transformed into the world frame. To obtain motion information of the RS camera at each row exposure time, a "forward propagation" process is adopted to calculate the camera's pose at each line exposure time relative to the first row $^{c_0}\mathbf{T}_{c(n)}$.

The detailed processes of the forward propagation are as below:

1) Calculate the actual exposure time of the current row n based on the camera timestamp and readout time: $t_{(n)} = nt_{rd} + t_0$. Obtain the closest IMU motion information based on the exposure time of the current row, including its collection time t_{I_j}, linear velocity $\mathbf{v}_j$, angular velocity $\mathbf{w}_j$, and the predicted LIO pose $^{w}\mathbf{T}_{I_j}$.

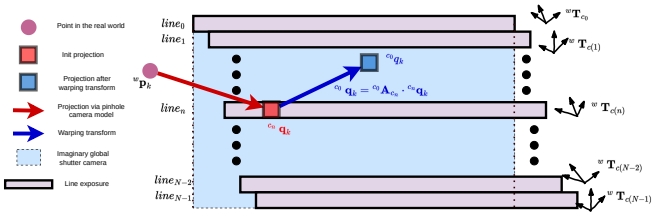

Fig. 4. Scheme of image rectification by warp transformation

2) Calculate the time difference between the current row and the closest IMU information:

$$\delta t = t_{(n)} - t_{I_j} \tag{1}$$

3) Based on the constant velocity motion model, calculate the relative pose update between the current row and the latest IMU prediction using its linear velocity $\mathbf{v}_j$ and angular velocity $\mathbf{w}_j$:

$${}^{c}\mathbf{T}_{w}(n) = {}^{I}\mathbf{T}_{C}{}^{-1} \cdot \begin{bmatrix} e^{\delta t \cdot \mathbf{w}_j} & \delta t \cdot \mathbf{v}_j \\ 0 & 1 \end{bmatrix} \cdot {}^{I_j}\mathbf{T}_{w} \tag{2}$$

4) Transform the relative pose of the current row to the pose relative to the 0th row:

$${}^{c_0}\mathbf{T}_{c(n)} = {}^{c_0}\mathbf{T}_{w} \cdot {}^{c(n)}\mathbf{T}_{w}{}^{-1} \tag{3}$$

Within the exposure time of a camera frame, the camera's motion pose of each line exposure time relative to the first row can be obtained by forward propagation based on the predicted IMU pose, using angular and linear velocities. The first row's pose relative to the world frame $^{c_0}\mathbf{T}_{w}$ is already known, so the pose of each row relative to the first row $^{c_0}\mathbf{T}_{c(n)}$ also can be determined.

B. Warping transfromation for undistortion

Numerous motion compensation algorithms have been proposed for correcting motion distortion in RS cameras during video stabilization. Two conventional methods exist for stabilizing video based on motion. The first method employs structure from motion (SfM) [19] to estimate the motion, which is computationally intensive. The second method obtains the pose by integrating gyroscope data from the IMU [18]. Regardless of the method used to obtain the camera pose, whether through SfM or IMU integration, a warp transform operation is used to correct the image.

This study presents the warp transformation matrix procedure for image correction, which follows the subsequent steps. First, a virtual global exposure camera is assumed to capture an image at the first line exposure time with the pose $^{w}\mathbf{T}c_0$. Given a point $^{w}\mathbf{p}_k$ in the world frame, it is projected onto the RS camera's image plane of a pixel as $^{c_n}\mathbf{q}_k$ with the row number row n, and the camera pose of this line is $^{w}\mathbf{T}c_{(n)}$. It should be noted that the camera pose of row n, $^{w}\mathbf{T}c_{(n)}$, can be obtained using the forward propagation algorithm introduced earlier with the imaginary global shutter camera pose of the first row $^{w}\mathbf{T}c_0$.

Second, the pixel of $^{c_n}\mathbf{q}_k$ captured by the RS camera at row N can be transformed to the imaginary global shutter camera at the first row as follows:

$${}^{c_0}\mathbf{q}_k = \frac{z_{c_n}}{z_{c_0}}\mathbf{K} \cdot {}^{c_0}\mathbf{T}_{c(n)} \cdot \mathbf{K}^{-1} \cdot {}^{c_n}\mathbf{q}_k \tag{4}$$

Usually, it is assumed that $z_{c_0} = z_{c_n}$ [18], which simplifies equation (4) as follows:

$$^{c_0}\mathbf{q}_k = \mathbf{K}\,^{c_0}\mathbf{T}_{c(n)}\mathbf{K}^{-1} \cdot {}^{c_n}\mathbf{q}_k \tag{5}$$

This can be seen as a type of warp operation:

$$^{c_0}\mathbf{A}_{c(n)} = \mathbf{K}\,^{c_0}\mathbf{T}_{c(n)}\mathbf{K}^{-1} \tag{6}$$

and the equation (5) can be written as:

$$^{c_0}\mathbf{q}_k = {}^{c_0}\mathbf{A}_{c(n)} \cdot {}^{c_n}\mathbf{q}_k \tag{7}$$

where,the $^{c_0}\mathbf{A}_{c(n)}$ denotes the warp matrix projecting the pixel from the $c_{(n)}$(exposure at time of line n) to the c_0(exposure at time of line 0).

However, the assumption that $z_{c_0} = z_{c_n}$ is a strong assumption. Although this method is effective for correcting distortion in RS cameras and stabilizing videos, it is not sufficiently precise for colorizing LiDAR point clouds, particularly for scenes containing depth-discontinuous objects. This approach can lead to significant color misregistration in LiDAR point clouds.

C. Colorization using rolling shutter constraint

In this subsection, we introduce a method for colorizing LiDAR data that leverages the rolling shutter constraints, since the conventional approach outlined earlier is not suitable for this application. The projection location of LiDAR point clouds cannot be determined analytically [13], and hence, we resort to an iterative procedure that utilizes the rolling shutter constraint formula.

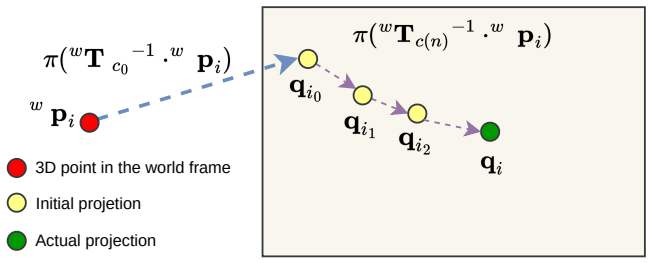

Fig. 5. Scheme of the process of solving the actual projection on the image plane by rolling shutter constraints

As previously described, the rolling shutter constraints enable the computation of the actual exposure time of a point:

$$t^\star = t(\mathbf{p}'(t^\star)) \tag{8}$$

where

$$\mathbf{p}'_k = \pi_c(^{i}\mathbf{T}_j \cdot \pi_c^{-1}(\mathbf{p}_k, d_p)) \tag{9}$$

and $^{i}\mathbf{T}_j$ represents the camera's relative pose between two exposure lines. The aforementioned process entails significant computational costs due to the need to calculate the relative pose between lines in every iteration. Hence, it is not ideal for processing a large amount of LiDAR points for colorization purposes.

To address this limitation, we simplify our algorithm as follows.

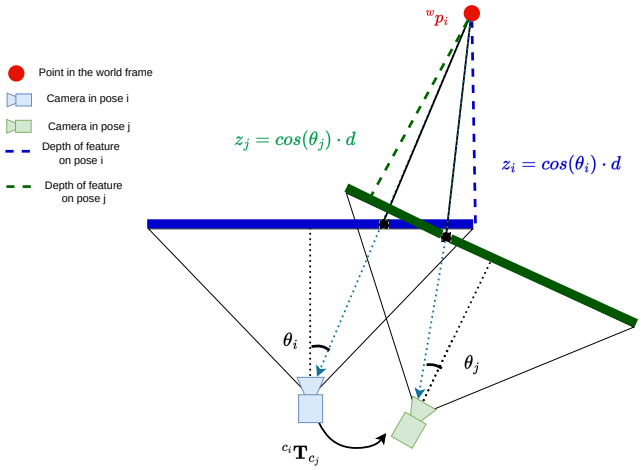

Fig. 6. Assumption of distance consistency.

Firstly, we look up the table of camera poses at the exposure time and determine the pose of camera exposure at row v_i, represented as $^{c}\mathbf{T}_{w(v_i)}$. Next, given a point $^{w}\mathbf{p}_i$ in the world coordinate system, we project it onto the image plane based on the initial camera pose of the 0th row, denoted as $^{c}\mathbf{T}_{w(0)}$. then, we use the pose of line $v_i = 0$ to calculate the initial projection pixel.

$$\mathbf{q}'_i = \begin{bmatrix} u'_i \\ v'_i \end{bmatrix} = \pi_c(^{c}\mathbf{T}_{w(v)} \cdot {}^{w}\mathbf{p}_i)) \tag{10}$$

Secondly, an initial projected position $\mathbf{q}'_i$ is obtained as $\begin{bmatrix} u'_i & v'_i \end{bmatrix}$, and the corresponding row number v'_i is extracted. The convergence of the projection position of the LiDAR point cloud is checked by verifying whether $|v_i - v'_i| = 0$. If the projection position has converged, the point can be colored based on the pixel value of the current projection position. To expedite the process, our proposed method employs lookup tables for each line posture calculation and multi-threading, thereby enabling real-time and precise colorization for LiDAR point clouds captured using a rolling shutter camera.

D. Readout time calibration for RS camera

In this subsection, we present a method for initializing the readout time parameter for RS cameras in conjunction with LIO, aiming to tackle the challenge of practical readout time estimation for RS cameras.

Although pixels in an image may have varying depths in the z-axis, feature points between two frames tracked by optical flow have the same distance to the optical center, neglecting the translation, as shown in Fig. 6. Specifically, for a feature point $^{w}\mathbf{p}_i$ in the camera frame $^{c}\mathbf{p}_i$ with a depth of z_i, the depth ratio of feature points between frames can be expressed as below, with a scaling factor of $cos\theta$:

$$\frac{z_i}{z_j} = \frac{d \cdot cos\theta_i}{d \cdot cos\theta_j} \tag{11}$$

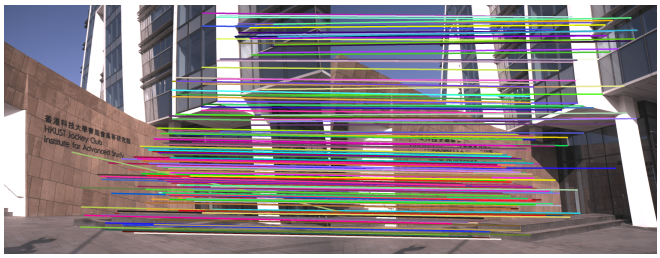

Fig. 7. Optical flow-based LK algorithm for tracking current frame (right) to its previous frame (left).

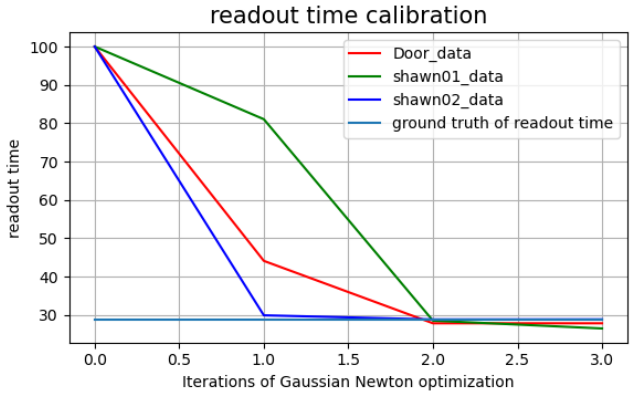

Fig. 8. Readout time calibration results. The readout time is converged around the ground truth using our proposed calibration method.

in which $cos\theta_i$ can be calculated as follows:

$$cos\theta_i = \frac{\begin{bmatrix} u_i & v_i & 1 \end{bmatrix} \cdot \begin{bmatrix} 0 & 0 & 1 \end{bmatrix}^T}{\| \begin{bmatrix} u_i & v_i & 1 \end{bmatrix} \| \| \begin{bmatrix} 0 & 0 & 1 \end{bmatrix} \|} \tag{12}$$

In the context of an RS camera's motion in the $SE(3)$ space, we utilize LK optical flow to perform feature point matching between two frames of the moving camera images. Optical flow is employed to track 2D points in the i-th frame with those in the j-th frame, while also considering the assumption that the points' distance to the optical center remains relatively constant between the two frames. We construct a loss function that incorporates readout time to estimate the readout time, based on this assumption. The loss function is expressed as follows:

$$F_{(t_{rd})} = \| \mathbf{K} \cdot^{c_j} \mathbf{T}_{c_i} \cdot \mathbf{K}^{-1} \frac{cos\theta_i}{cos\theta_j} \mathbf{q}_i - \mathbf{q}_j \|^2 \tag{13}$$

To minimize the reprojection error, we can employ the Gauss-Newton method to optimize the readout time of the rolling shutter camera.

IV. Experiments

We first present the readout time calibration results and then colored LiDAR point clouds with the RS constraints. Finally, an online colorization LIO system is presented to demonstrate the effectiveness of the proposed Rollvox.

A. Evaluation on readout time calibration

We begin by presenting a case study that utilizes our method for image matching, as illustrated in Fig. 7. The optical flow tracking results of the two frames are shown, and we employ RANSAC (using the Essential matrix) to eliminate outliers and establish a robust matching relationship.

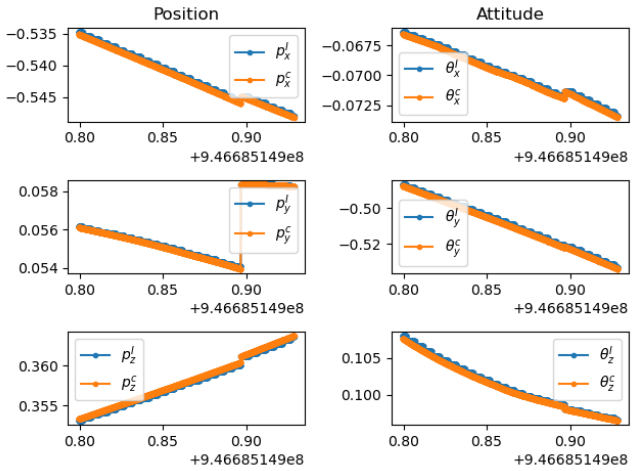

Fig. 9. The orange line is the forward propagation of the camera pose, and the blue line is the prediction of the IMU pose.

As depicted in the system block diagram in Fig. 2, the calibration module initially detects the gyroscope and linear velocity obtained from the LIO system and then selects a pair of frames with suitable activation for readout time calibration. To evaluate the performance of the readout time calibration algorithm, we conducted experiments on our self-collected datasets, as demonstrated in Fig. 8. In the experiments, we set the readout time at a deviation of 100us from the ground truth. The results of the readout time for each dataset eventually converge around the true value of 28.8us, with an error within 2us after 3-4 iteration steps.

B. Colorization with rectified image and RS constraints

As illustrated in Fig. 9, the camera's pose at each line exposure time is interpolated based on the constant velocity model explained in section III-A. This interpolation allows for motion compensation during colorization.

The RS camera's top-to-bottom shutter movement and rapid left-to-right motion result in motion distortion, as depicted in Fig. 11 (a). The blue square in Fig. 11 indicates significant motion distortion around the door compared to the static image shown in Fig. 11 (e). Consequently, color misregistration occurs, with the color shifting towards the left on the door, as demonstrated in Fig. 11 (b).

In section III-B, we described the use of warp transform to rectify the RS camera's captured image and stabilize it for colorization of the LiDAR point clouds. Fig. 11 (c) displays the rectified images obtained through the warp transform. Here, the pixels are warped to the left to effectively cancel out the RS effect, resulting in a more stable image. Using this stabilized image for colorization, we achieve the outcomes illustrated in Fig. 11 (d). However, we note color misregistration, particularly at depth discontinuity edges.

Our algorithm's colorization results are shown in Fig. 11

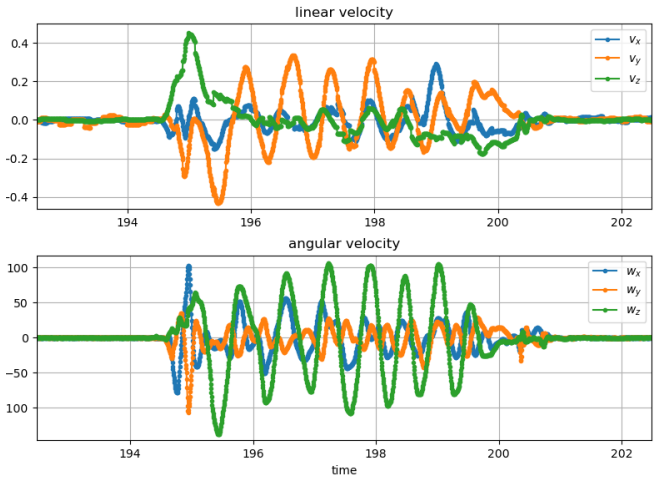

Fig. 10. The angular velocity and linear velocity of our dataset shawn02.bag

(f). As the actual projection on the image plane is derived from the rolling shutter constraints, our algorithm's colorization works well, as evident from the proper coloring of the depth discontinuity edges (yellow and red squares in Fig. 11).

C. Validation with FAST-LIO2

Before presenting the online tests, we first evaluated the computational efficiency of our colorization algorithm. The evaluation was conducted on a desktop PC equipped with an Intel Core i7-9750H CPU, a 6-core processor operating at 2.60 GHz. Our algorithm's processing time averaged at 4.018 ms for an average of 24,000 points per LiDAR frame, specifically from the Livox AVIA.

To assess our algorithm's effectiveness in colorizing point clouds obtained from the LiDAR-inertial system, we deployed our Rollvox system with FAST-LIO2 and collected data under intense shaking conditions. Fig. 10 illustrates the angular and linear velocities of the aggressive motion for dataset club01.bag. Despite the significant angular velocity, which reached up to $140.09°/s$, the RS camera images remained sharp and unblurred. Furthermore, our proposed colorization algorithm for LiDAR point clouds demonstrated its capability, as evidenced by the clear and accurate colorization results obtained by overlaying multiple frames in Fig. 12 (b).

V. Conclusion

In this study, we present a novel method for efficient and accurate colorization of LiDAR point clouds using an RS camera. To ensure precise synchronization of the camera, LiDAR, and IMU timestamps, we design and implement a hardware synchronization solution. Next, accurate camera motion poses are obtained by forward propagation based on a constant velocity model. We employ the rolling shutter constraint method to determine the true projection positions of the LiDAR point cloud on the RS camera imaging plane. To improve the efficiency of row pose calculation during the RS constraint process, we adopt a search table approach. Experimental results show that our algorithm outperforms

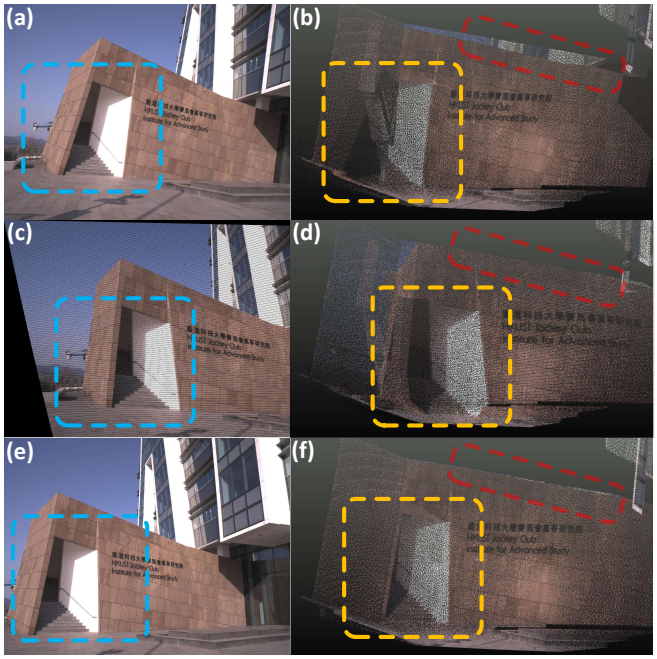

Fig. 11. The image captured by RS camera with original motion distortion (a) and with motion rectification (c). Their colorization for LiDAR point clouds are shown in (b) and (d) respectively. The image captured on the static is shown in (e). The colorized LiDAR point clouds with our algorithm are shown in (f)

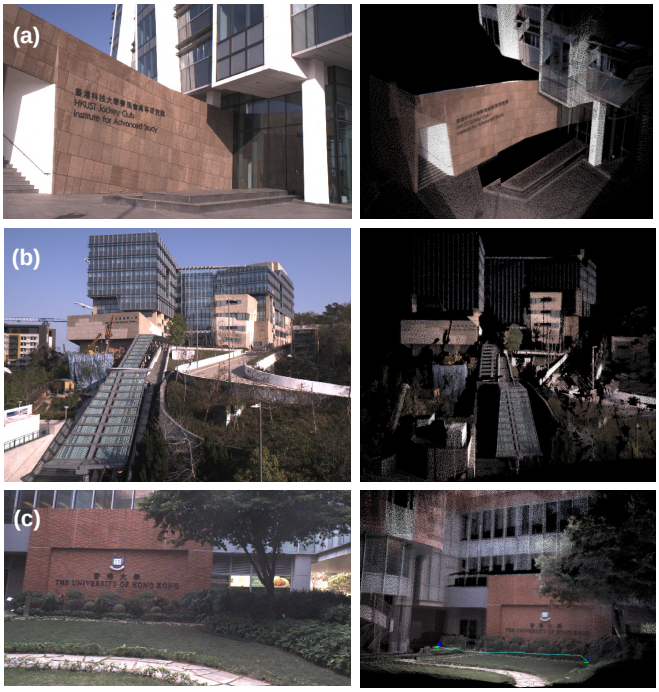

Fig. 12. Colorization results using our Rollvox with FAST-LIO2. The left side is the scenes captured with the RS camera, while the right is the colored LiDAR point clouds. (a). club01.bag (maximum angular velocity $140.09°/s$) (b). shawn02.bag (maximum angular velocity $105.93°/s$) (c). hku03.bag (maximum angular velocity $104.34°/s$)

traditional RS image stabilization algorithms, particularly in coloring non-continuous edges. We also propose a method for calibrating the readout time by minimizing reprojection errors, which addresses the challenge of obtaining readout

time for RS cameras. Finally, our algorithm is deployed on FAST-LIO2 and achieves real-time, high-quality colorization of LiDAR point clouds using an RS camera.

REFERENCES

[1] K. El Madawi, H. Rashed, A. El Sallab, O. Nasr, H. Kamel, and S. Yogamani, "Rgb and lidar fusion based 3d semantic segmentation for autonomous driving," in *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*. IEEE, 2019, pp. 7–12.

[2] Y. Zhong, S. Wang, S. Xie, Z. Cao, K. Jiang, and D. Yang, "3d scene reconstruction with sparse lidar data and monocular image in single frame," *SAE International Journal of Passenger Cars-Electronic and Electrical Systems*, vol. 11, no. 07-11-01-0005, pp. 48–56, 2017.

[3] M. Giorgini and J. Aleotti, "Visualization of agv in virtual reality and collision detection with large scale point clouds," in *2018 IEEE 16th International Conference on Industrial Informatics (INDIN)*. IEEE, 2018, pp. 905–910.

[4] Y. Pan, A. Braun, I. Brilakis, and A. Borrmann, "Enriching geometric digital twins of buildings with small objects by fusing laser scanning and ai-based image recognition," *Automation in Construction*, vol. 140, p. 104375, 2022.

[5] X. Zuo, P. Geneva, W. Lee, Y. Liu, and G. Huang, "Lic-fusion: Lidar-inertial-camera odometry," in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2019, pp. 5848–5854.

[6] T. Shan, B. Englot, C. Ratti, and D. Rus, "Lvi-sam: Tightly-coupled lidar-visual-inertial odometry via smoothing and mapping," in *2021 IEEE international conference on robotics and automation (ICRA)*. IEEE, 2021, pp. 5692–5698.

[7] Y. Zhu, C. Zheng, C. Yuan, X. Huang, and X. Hong, "Camvox: A low-cost and accurate lidar-assisted visual slam system," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2021, pp. 5049–5055.

[8] J. Lin and F. Zhang, "R3live: A robust, real-time, rgb-colored, lidar-inertial-visual tightly-coupled state estimation and mapping package," in *2022 International Conference on Robotics and Automation (ICRA)*. IEEE, 2022, pp. 10 672–10 678.

[9] C. Zheng, Q. Zhu, W. Xu, X. Liu, Q. Guo, and F. Zhang, "Fast-livo: Fast and tightly-coupled sparse-direct lidar-inertial-visual odometry," in *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2022, pp. 4003–4009.

[10] J. Vautherin, S. Rutishauser, K. Schneider-Zapp, H. F. Choi, V. Chovancova, A. Glass, and C. Strecha, "Photogrammetric accuracy and modeling of rolling shutter cameras." *ISPRS Annals of Photogrammetry, Remote Sensing & Spatial Information Sciences*, vol. 3, no. 3, 2016.

[11] A. El Gamal and H. Eltoukhy, "Cmos image sensors," *IEEE Circuits and Devices Magazine*, vol. 21, no. 3, pp. 6–20, 2005.

[12] D. Schubert, N. Demmel, V. Usenko, J. Stuckler, and D. Cremers, "Direct sparse odometry with rolling shutter," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 682–697.

[13] M. Meingast, C. Geyer, and S. Sastry, "Geometric models of rolling-shutter cameras," *arXiv preprint cs/0503076*, 2005.

[14] P. Li, T. Qin, B. Hu, F. Zhu, and S. Shen, "Monocular visual-inertial state estimation for mobile augmented reality," in *2017 IEEE international symposium on mixed and augmented reality (ISMAR)*. IEEE, 2017, pp. 11–21.

[15] T. Qin, P. Li, and S. Shen, "Vins-mono: A robust and versatile monocular visual-inertial state estimator," *IEEE Transactions on Robotics*, vol. 34, no. 4, pp. 1004–1020, 2018.

[16] W.-h. Cho and K.-S. Hong, "Affine motion based cmos distortion analysis and cmos digital image stabilization," *IEEE Transactions on Consumer Electronics*, vol. 53, no. 3, pp. 833–841, 2007.

[17] C.-K. Liang, L.-W. Chang, and H. H. Chen, "Analysis and compensation of rolling shutter effect," *IEEE Transactions on Image Processing*, vol. 17, no. 8, pp. 1323–1330, 2008.

[18] A. Karpenko, D. Jacobs, J. Baek, and M. Levoy, "Digital video stabilization and rolling shutter correction using gyroscopes."

[19] P.-E. Forssén and E. Ringaby, "Rectifying rolling shutter video from hand-held devices," in *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. IEEE, 2010, pp. 507–514.

[20] W. Xu and F. Zhang, "Fast-lio: A fast, robust lidar-inertial odometry package by tightly-coupled iterated kalman filter," *IEEE Robotics and Automation Letters*, vol. 6, no. 2, pp. 3317–3324, 2021.

[21] W. Xu, Y. Cai, D. He, J. Lin, and F. Zhang, "Fast-lio2: Fast direct lidar-inertial odometry," *IEEE Transactions on Robotics*, vol. 38, no. 4, pp. 2053–2073, 2022.