Multical: Spatiotemporal Calibration for Multiple IMUs, Cameras and LiDARs

Xiangyang Zhi[†1], Jiawei Hou[†2], Yiren Lu, Laurent Kneip and Sören Schwertfeger[3]

***Abstract*— Spatiotemporal calibration of sensors, especially of those who do not share their fields of view, is becoming increasingly important in the fields of autonomous driving and robotics. This paper presents a general sensor calibration method, named *Multical*, that makes use of multiple planar calibration targets whose poses will be estimated alongside spatiotemporal calibration. Multical exploits continuous-time curves to represent the state of the sensor platform during data collection, and thus is a general framework to calibrate different kinds of sensors and deal with both spatial as well as temporal offsets. Multical includes algorithms to estimate the initial guesses of spatial transformations between sensors, and also the relative poses between calibration targets. Users do not need to provide any extrinsic priors. We apply the proposed calibration approach to both simulated and real-world experiments, and the results demonstrate the high fidelity of the proposed method.**

I. Introduction

Multi-sensor fusion has received increasing attention from various areas, *e.g.* localization and perception, and spatial calibration is essential if one wants to fuse data from different sensors into one reference frame. However, temporal calibration is also required and has been given much less attention. Hardware synchronization of sensors is often not available, and sensor data is often simply assigned the timestamp of the host's reception of the data. Due to data processing and transfer, these timestamps include unknown delays. Therefore, temporal calibration is required for highly accurate sensor fusion.

Cameras, LiDARs and Inertial Measurement Units (IMU) are widely used in mobile robotics and autonomous driving, the latter often being equipped with multiple cameras and LiDARs to increase the overall situational awareness of the system. The calibration of these sensors is cumbersome for two reasons: a) Existing calibration frameworks typically employ only a single calibration target. It is thus hard to calibrate all sensors simultaneously, especially if the sensors have non-overlapping FoVs. Hand-eye calibration [1], [2], [3] is a general paradigm to calibrate non-overlapping (correspondence-less) sensors, which usually consists of two stages, 1) estimating sensors' odometry and 2) solving equation $AX = XB$. However, the errors of odometry estimation will propagate to the solution of $AX = XB$, so we claim that hand-eye calibration is less accurate than methods utilizing direct feature correspondences. b) Existing calibration frameworks provide LiDAR-camera, LiDAR-IMU and camera-IMU calibration, but very few frameworks can jointly calibrate multiple LiDARs, cameras and IMUs. We claim that pair-wise calibration is inefficient and may not be accurate enough, even if followed by post-processing, *e.g.* pose graph optimization [4]. Therefore, an approach which can calibrate multiple cameras, LiDARs and IMUs simultaneously will greatly benefit the robotics community.

We propose to utilize multiple calibration targets to cover the FoVs of all sensors at the same time. In this paper, we utilize AprilTags [5] to distinguish the calibration targets. Along with a multi-target calibration system, we propose a method which jointly calibrates the target poses in the world frame and the extrinsics of multiple sensors, regardless of the question whether or not their FoVs are overlapping.

Conventional state estimation and sensor calibration techniques usually represent the optimized, time-varying variables in discrete time, *i.e.* at each measurement time. However, some types of sensors have very high data-sampling rates. For example, IMU data is usually captured hundreds of times per second and modern laser range finders (LRFs) sample points at a rate of 10kHz or more. Accommodating these high-rate measurements would imply a very high computation cost, because we would have to estimate the state at each measurement time, which sharply increases the number of variables. Additionally, the measurement times may continuously change when time offsets are updated, which makes an assignment between optimization variables and measurements very difficult. In recent years, continuous-time functions became more widely used to represent the state of robots and sensor platforms for applications like calibration [6], [7] and SLAM [8], [9]. By parameterizing the state with temporal basis functions, the pose at any exact time can be estimated while keeping the number of variables at a reasonable level, which is very convenient for incorporating high-rate sampling sensors. Furthermore, it also naturally supports temporal calibration [6], because the states are represented as functions of time.

We also exploit temporal basis functions to express the sensor setup trajectory. The work most similar to ours is Kalibr [10] which also calibrates IMUs, cameras and LRFs using the framework of [11], although LRFs calibration is not open-source. They use only one target, we extend

This work was supported by the School of Information Science and Technology at ShanghaiTech University.

†The two first authors have equal contributions to this paper.

[1]X. Zhi is with NIO Inc. (e-mail: xiy.zhi@gmail.com).

[2]J. Hou is with the School of Information Science and Technology, ShanghaiTech University, Shanghai 201210, China; the University of Chinese Academy of Sciences, Beijing 100049, China; the Shanghai Institute of Microsystem and Information Technology, Chinese Academy of Sciences, Shanghai 200050, China (e-mail: houjw@shanghaitech.edu.cn).

[3]These authors are with the School of Information Science Technology of ShanghaiTech University, China (e-mail: {luyr, lkneip, soerensch}@shanghaitech.edu.cn).

that to multiple targets to support the calibration of non-overlapping cameras and LiDARs. In addition, Kalibr detects several planes from point clouds to build a map for LiDAR calibration, but we explicitly limit the estimation to the planes of the calibration targets, which removes any special requirement on the environment. Furthermore, our method constructs direct constraints between cameras and LiDARs using the targets, which benefits the calibration.

Our method is easy-to-use: a) All data is collected at the same time, no pairwise sensor data collection is needed. b) Calibration does not require extra sensors (such as FARO) or prior knowledge of the poses of the targets. c) The user does not need to give any initial guess for the calibration parameters, because we use our novel targets and approaches to initialize spatial extrinsics between sensors automatically.

In summary, the main contributions of this paper are:

- We propose an algorithm which supports spatiotemporal calibration of camera-LiDAR-IMU setups, without providing initial guesses for spatial extrinsics. Our method uses multiple distinguishable planar targets and jointly calibrates the target poses as well as all sensor poses.
- We make the calibration library public[1]. To the best of our knowledge, this is the first open-source library that supports joint IMU-camera-LiDAR calibration.
- We release our dataset, including the package for simulated environment and sensors, with motion generation of the sensor platform, and the real world datasets.

After reviewing related work (Section II) we provide implementation details (Section III) and details about the experiments (Section IV). The paper concludes in Section V.

II. Related Work

As the calibration targets provide additional constraints, target-based calibration approaches presented here are regarded to be more robust and precise than targetless methods.

Calibration pattern-based calibration: The popular paper [12] proposes to exploit a planar pattern to calibrate the intrinsic parameters of a camera. Checkerboard-like patterns are popular as the corners can be detected robustly, however, one cannot distinguish them if observing part of the checkerboard or multiple checkerboards. 2D bar code markers such as AprilTags and ArUcos [13] solve these problems [14].

Calibration between a camera and an LRF or a LiDAR is a popular research topic. Vasconcelos *et al*. [15] provide a minimal closed-form solution for camera-LRF calibration. They utilize a checkerboard to estimate the plane poses in camera coordinates and depth readings in LRF frames. The disadvantage of their method is that the checkerboard lacks geometric features, so it is hard to find the plane with the measurements of an LRF. Often planar boards with certain geometric shapes or holes are used [16], [17], [18] to generate several direct correspondences between camera and LRF, which is considered accurate and robust. However, these custom-built targets are usually sophisticated and expensive. Previous works mainly focus on pairwise

[1]Available here: `https://github.com/zhixy/multical`

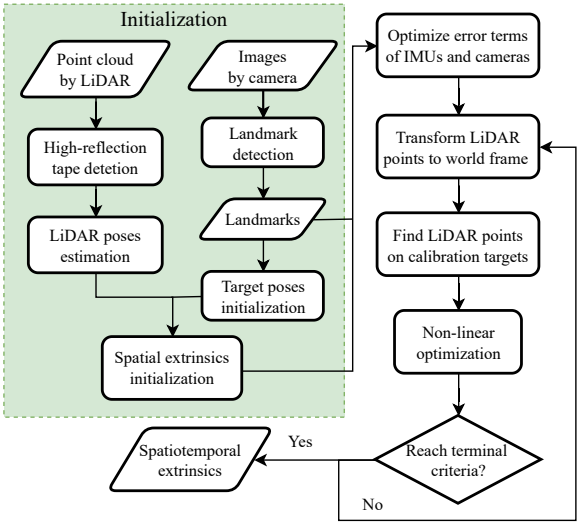

Fig. 1. Overview of the proposed method.

sensor calibration, but in practice, a robot usually is equipped with multiple cameras and multiple LiDARs with non-overlapping FoVs for which pairwise calibration is inefficient or even impractical. Some methods [19], [20] exploit an extra mirror to make the planar targets visible in cameras which cannot see the targets directly. These methods operate at the cost of estimating the mirror's pose. Xie *et al*. [21] setup an environment with multiple walls with bar codes on the walls. They acquire the 3D model of the environment with a high-definition 3D laser scanner. They then calibrate an autonomous vehicle equipped with two cameras and four LiDARs and show that their method is valid for non-overlapping sensors. In this paper, we also utilize multiple planar boards with AprilTags. Unlike [21], we calibrate the poses of the boards along with sensor calibration, which bypasses the requirement for capturing an accurate 3D model with an extra device. Furthermore, to estimate the initial guess of the LiDARs, we paste two high-reflection tapes on adjacent sides of the primary planar target, for which the details will be elaborated in Section III-C.

Continuous-time state representation and spatiotemporal calibration: In 2012, Furgale *et al*. [11] proposed a continuous-time-state, discrete-time-measurement batch estimation methodology. They utilize temporal basis B-splines to represent the time-varying optimization parameters and construct error terms with discrete measurements. Their approach is a very general framework for spatiotemporal calibration and thus inspired quite a few later papers, including ours. Specifically, Oth *et al*. [22] calibrate a rolling shutter camera under this framework. Rehder *et al*. [23] calibrate the extrinsics of multiple IMUs and the displacement of individual accelerometer axes, *i.e*. intrinsics, of low-cost IMUs. In [7], Nowicki deals with spatiotemporal calibration of a camera and 3d laser scanner with one planar target. Unlike other methods, B-splines are used to parameterize the trajectory of the plane determined by the calibration target, rather than the sensor platform. Lv *et al*. [24] propose a

LiDAR-IMU calibration method which does not utilize any target but a surflet map.

III. Methodology

Fig. 1 gives an overview of our method for calibrating cameras, LiDARs and IMUs. It is worth noting that our method is flexible and that the minimum configuration is a single calibration board, one camera and one IMU or LiDAR.

A. Calibration Targets

As mentioned before, a major difference between existing methods and ours is that we introduce more planar boards to calibrate non-overlapping sensors. We utilize AprilTags to distinguish the boards. Specifically, in this paper we extend the AprilGrid in Kalibr [23]. Fig. 2(a) shows our target with the AprilGrid, which contains 36 AprilTags. We extend the targets by using successive codes for following targets, so no two tags have the same code. The Cartesian coordinate of each board is defined as follows: the origin is at the bottom-left corner of the bottom-left tag, the x-axis points right, the y-axis points up and the z-axis is determined by the right-hand rule. The first board is regarded as the primary one, and the inertial world frame coincides with its coordinate.

For calibration data collection, we place several boards in the scenario, according to the distribution of the cameras, and then move the sensor platform in the environment while collecting sensor measurement data for about 2 minutes.

B. Variables and Notations

The variables to estimate in the calibration can be divided into time-varying variables and time-invariant variables. The pose of the body expressed in the inertial world frame is time-varying, and we utilize two split B-spline functions to represent the orientation and position, respectively. B-spline functions consist of basis functions which are non-zero only in finite intervals, which keeps the optimization system sparse with a limited number of control points, and improves the efficiency of the estimation. Furthermore, the B-spline function is polynomial, which is why the derivative is simple and analytical. Specifically, we implement B-spline functions of order 5 with control points $\boldsymbol{R}_1, \cdots, \boldsymbol{R}_N \in SO(3)$ and $\boldsymbol{t}_1, \cdots, \boldsymbol{t}_N \in \mathbb{R}^3$. The readers are referred to [25] for a thorough introduction about the $SO(3)$ B-spline and to [26] for a more efficient implementation for $SO(3)$ B-splines.

For the convenience of symbolic expression and discussion, we consider only one LiDAR and one IMU in addition to multiple cameras and targets in the rest of this section, although our method supports multiple LiDARs and IMUs. We use $\mathcal{C}_c (c = 1, 2, \cdots, C)$, $\mathcal{A}_a (a = 1, 2, \cdots, A)$, $\mathcal{L}$ and $\mathcal{I}$ to denote the coordinate frames defined by cameras, targets, LiDAR and IMU, respectively. We also define the inertial world frame $\mathcal{W}$ and the body frame $\mathcal{B}$. Based on that, we define $\boldsymbol{T}_{\cdot,\cdot} \in SE(3)$ as the transformation from one frame to another, *e.g.* $\boldsymbol{T}_{\mathcal{W},\mathcal{C}_1}$ represents the transformation from coordinate frame of camera 1 to the inertial world frame. In addition, we utilize $\boldsymbol{R}_{\cdot,\cdot} \in SO(3)$ and $\boldsymbol{t}_{\cdot,\cdot} \in \mathbb{R}^3$ to denote the rotation and translation of $\boldsymbol{T}_{\cdot,\cdot}$, respectively.

Without loss of generality, we specify the body frame to coincide with a sensor frame, which will not affect the calibration results. Given that only one camera is essential for the calibration pipeline, camera 1 is selected as the reference in this paper, *i.e.* coinciding with the body frame $\mathcal{B}$.

In summary, the variables to estimate are as follows:

$$\begin{aligned}\hat{\boldsymbol{x}} := [&\boldsymbol{P}_R^T, \boldsymbol{P}_t^T, \boldsymbol{T}_{\mathcal{C},\mathcal{B}}^T, \boldsymbol{T}_{\mathcal{W},\mathcal{A}}^T, \boldsymbol{T}_{\mathcal{L},\mathcal{B}}^T, \boldsymbol{T}_{\mathcal{I},\mathcal{B}}^T, \\ &d_{\mathcal{C}}^T, d_{\mathcal{L}}, d_{\mathcal{I}}, {}_{\mathcal{W}}\boldsymbol{g}^T, \boldsymbol{b}_a^T, \boldsymbol{b}_\omega^T]^T,\end{aligned} \tag{1}$$

where

- $\boldsymbol{P}_R^T := [\boldsymbol{P}_{R_1}^T, \cdots, \boldsymbol{P}_{R_N}^T]^T$ are the control points of the B-spline representing the orientation of the body.
- $\boldsymbol{P}_t^T := [\boldsymbol{P}_{t_1}^T, \cdots, \boldsymbol{P}_{t_N}^T]^T$ are the control points of the B-spline representing the position of the body.
- $\boldsymbol{T}_{\mathcal{C},\mathcal{B}}^T := [\boldsymbol{T}_{\mathcal{C}_2,\mathcal{B}}^T, \cdots, \boldsymbol{T}_{\mathcal{C}_C,\mathcal{B}}^T]^T$ denote the transformation from the body to the $C-1$ cameras.
- $\boldsymbol{T}_{\mathcal{W},\mathcal{A}}^T := [\boldsymbol{T}_{\mathcal{W},\mathcal{A}_1}^T, \cdots, \boldsymbol{T}_{\mathcal{W},\mathcal{A}_A}^T]^T$ denote the transformation from AprilGrid targets to $\mathcal{W}$.
- $d_{\mathcal{C}}^T := [d_{\mathcal{C}_2}, \cdots, d_{\mathcal{C}_C}]^T$ are the time offsets of image timestamps of the $C-1$ cameras.
- $d_{\mathcal{L}}$ is the time offset of point timestamps of the LiDAR.
- $d_{\mathcal{I}}$ is the time offset of the IMU.
- ${}_{\mathcal{W}}\boldsymbol{g}$ is the gravity vector expressed in $\mathcal{W}$.
- $\boldsymbol{b}_a$ is the accelerometer bias.
- $\boldsymbol{b}_\omega$ is the gyroscope bias.

Since the time interval of data collection is short (typically 1-2 minutes), and the IMU biases usually drift very slowly, we estimate the IMU biases with time-invariant variables here, as some recent approaches do [26], [24].

C. Initial Calibration using Graph Optimization

Initial guesses of variables are critical for non-linear optimization tasks like calibration. One main contribution of this work is a series of robust and thorough initial calibration algorithms, which will be elaborated in the following. The initial calibration consists of two steps, the target poses will be estimated first, and transformations between sensors will be estimated later using the initialized targets.

We assume that cameras' intrinsic parameters are pre-calibrated and the geometry information of the targets is known, so we can estimate the transformation from the coordinate frame of the camera to that of the target observed by the camera [12]. When the camera is observing two targets, we estimate the transformation between the two targets by

$$\tilde{\boldsymbol{T}}_{\mathcal{A}_i,\mathcal{A}_j} := \tilde{\boldsymbol{T}}_{\mathcal{A}_i,\mathcal{C}_k}(t)\tilde{\boldsymbol{T}}_{\mathcal{A}_j,\mathcal{C}_k}(t)^{-1}, \tag{2}$$

where t is the timestamp of the image captured by camera c_k. Note that as we utilize AprilTags, the images in which only part of the targets are observed may also contribute constrains. On the other hand, the transformation is also defined as

$$\boldsymbol{T}_{\mathcal{A}_i,\mathcal{A}_j} := \boldsymbol{T}_{\mathcal{W},\mathcal{A}_i}^{-1}\boldsymbol{T}_{\mathcal{W},\mathcal{A}_j}. \tag{3}$$

Therefore, we can evaluate the residual as the difference between $\tilde{\boldsymbol{T}}_{\mathcal{A}_i,\mathcal{A}_j}$ and $\boldsymbol{T}_{\mathcal{A}_i,\mathcal{A}_j}$:

$$\boldsymbol{r}_t := [\log(\tilde{\boldsymbol{R}}_{\mathcal{A}_i,\mathcal{A}_j}^T\boldsymbol{R}_{\mathcal{A}_i,\mathcal{A}_j})^T, \lambda(\tilde{\boldsymbol{t}}_{\mathcal{A}_i,\mathcal{A}_j} - \boldsymbol{t}_{\mathcal{A}_i,\mathcal{A}_j})^T]^T, \tag{4}$$

where $\boldsymbol{R}$ and $\boldsymbol{t}$ denote the rotation matrix and translation vector of the transformation, $\log(\cdot)$ is an operation that maps $SO(3)$ to its corresponding rotation vector, and λ is a scale factor to balance rotation and translation errors. We gather all available measurements and minimize the overall residuals:

$$[\hat{\boldsymbol{T}}_{\mathcal{W},\mathcal{A}_1}, \cdots, \hat{\boldsymbol{T}}_{\mathcal{W},\mathcal{A}_A}]^T = \arg\, min\{\sum \|\boldsymbol{r}_t\|^2\}. \quad (5)$$

We adopt the Levenberg-Marquardt algorithm to solve (5) and estimate the poses of the targets. Note that this is a graph optimization problem, so we need to arrange the targets in a reasonable way and capture enough images to ensure the graph is connected. This will ensure that a unique solution of the problem always exists.

After estimating the initial guesses of the target poses, we initialize the transformation between sensors and reference frame. With the initial guesses of all target poses, the camera poses expressed in the world frame can be estimated no matter which target is observed. Afterwards, we initialize the body pose spline with the estimated poses of camera 1. We then formulate the initial calibration as an optimization problem by constructing error terms associated with the gyroscope measurements and the estimated poses of LiDAR and cameras. If at time t, the camera i observes target $\mathcal{A}_j$, we estimate the transformation as

$$\tilde{\boldsymbol{T}}_{\mathcal{C}_i,\mathcal{A}_j}(t) := \boldsymbol{T}_{\mathcal{C}_i,\mathcal{B}}\boldsymbol{T}^{-1}_{\mathcal{W},\mathcal{B}}(t)\boldsymbol{T}_{\mathcal{W},\mathcal{A}_j}, \quad (6)$$

and also calculate $\boldsymbol{T}_{\mathcal{C}_i,\mathcal{A}_j}(t)$ by the image observation. An error term relative to the extrinsic parameter of camera i is built like in (4).

The residual associated with the gyroscope is given by:

$$\boldsymbol{r}_\omega := \tilde{\boldsymbol{\omega}} - \boldsymbol{R}_{\mathcal{I},\mathcal{B}}\boldsymbol{R}^T_{\mathcal{W},\mathcal{B}}(t)\,{}_{\mathcal{W}}\boldsymbol{\omega}_{\mathcal{B}}(t) + \boldsymbol{b}_\omega \quad (7)$$

where $\tilde{\boldsymbol{\omega}}$ is the gyroscope measurement and ${}_{\mathcal{W}}\boldsymbol{\omega}_{\mathcal{B}}(t)$ is the angular velocity of the body expressed in $\mathcal{W}$. The accelerometer measurements will not be used during initial calibration, because they are coupled with gravity, which is unknown in the world frame defined by the targets. As a result, the initial guess of the translation between IMU and body frame can not be estimated because of a lack of constraints. However, our experiments show that the translation will converge to the right displacement even without initialization. Also, we do not estimate the time offsets during initialization and regard them as 0 unless specified by users.

Similarly, if we can find the transformation between a LiDAR and an arbitrary target at a given moment in time, error terms can be constructed for initial LiDAR extrinsic calibration. The LiDARs have to hit the boards with enough samples for the board detection to work. To this end, as illustrated in Fig. 2, we put two perpendicular high-reflection tape stripes at the left and bottom edges of the *primary* target board. Note that no stripes are taped on the other boards. The tape fully determines a coordinate system, say $\mathcal{R}$: the origin point locates at the intersection of the two tape stripes, the x-axis and y-axis extend along the tapes at the bottom and left side. In addition, since we make $\mathcal{W}$ coincide with $\mathcal{A}_1$,

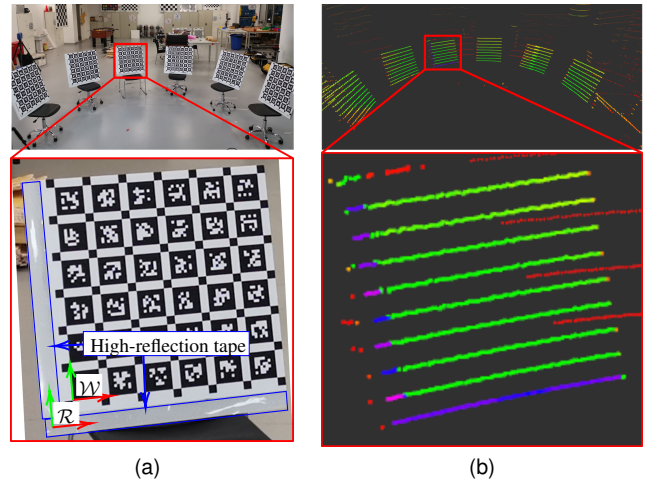

Fig. 2. (a): two high-reflection tapes which are pasted on the two sides of the primary target board. (b): the corresponding point cloud collected by a LiDAR, where the hue of points represents the intensity of laser reflection.

and $\boldsymbol{T}_{\mathcal{W},\mathcal{R}}$ is prior-known, we can estimate $\tilde{\boldsymbol{T}}_{\mathcal{L}_l,\mathcal{R}}(t)$ by the extrinsics of the LiDAR, *i.e.*

$$\tilde{\boldsymbol{T}}_{\mathcal{L}_l,\mathcal{R}}(t) := \boldsymbol{T}_{\mathcal{L}_l,\mathcal{B}}\boldsymbol{T}^{-1}_{\mathcal{W},\mathcal{B}}(t)\boldsymbol{T}_{\mathcal{W},\mathcal{R}}. \quad (8)$$

The rest of this sub-section focuses on estimating $\boldsymbol{T}_{\mathcal{R},\mathcal{L}_l}(t_k)$ from the measurements, *i.e.* point clouds, of LiDARs.

The two tape stripes in each point cloud are with conspicuous physical and geometric properties: 1) They will show with high intensity, compared to other points in the point cloud; 2) They are line-shaped elements. Therefore, it is trivial to extract them from point cloud. And we then fit them with two straight lines. The intersection of the two fitted lines is regarded as the position of the tape. However, the two 3D lines we extract may not be exactly coplanar due to noise, thus they will not intersect with each other. In that case, we approximate the origin of $\mathcal{R}$ with the middle of the shortest line segment which joins the two lines. A simple method is utilized to obtain the timestamp: we find the point nearest to the intersection from the point cloud frame and associate its timestamp to the estimated pose.

Lastly, we estimate the orientation of the tape stripes. We define the vector from the intersection to the barycenter of point inliers on the tape, and the line direction which is close to the vector. This is acceptable given that we know that the LiDAR cannot observe the tape stripes from the back side. Therefore, the x- and y-axes are simply distinguished by the right-hand rule, and the z-axis is calculated using the cross product. Because of noise, the estimated x- and y-axes may be not perpendicular, so we adjust the y-axis to the cross product of z- and x-axes, ensuring the axes' orthogonality.

Although there are some approximations above, we find it is good enough to estimate initial guesses of the spatial extrinsics of LiDARs. Moreover, the optimization introduced in section III-D will not involve the tape stripes at all, so these approximations have no impact on the final results.

D. Optimization

After the initialization, we conduct non-linear optimization to estimate all variables in (1) by solving the following least-squares problem:

$$\begin{aligned}\hat{\boldsymbol{x}} = \arg min\{ &\sum_{k=1}^{K} \left\|\boldsymbol{r}_{\omega}^{k}\right\|_{\sum_{\omega}}^{2} + \sum_{k=1}^{K} \left\|\boldsymbol{r}_{a}^{k}\right\|_{\sum_{a}}^{2} \\ &+ \sum_{c=1}^{C}\sum_{a=1}^{A}\sum_{j=1}^{J}\sum_{m=1}^{M} \left\|\boldsymbol{r}_{\mathcal{C}_c}^{ajm}\right\|_{\sum_{\mathcal{C}_c}}^{2} \\ &+ \sum_{a=1}^{A}\sum_{i=1}^{I} \left\|\boldsymbol{r}_{\mathcal{L}}^{ai}\right\|_{\sum_{\mathcal{L}}}^{2}\},\end{aligned} \tag{9}$$

where $\boldsymbol{r}_{\omega}^{k}$, $\boldsymbol{r}_{a}^{k}$, $\boldsymbol{r}_{\mathcal{C}_c}^{j}$ and $\boldsymbol{r}_{\mathcal{L}}^{i}$ are the error terms defined as the difference between the expected and the direct measurements of gyroscope, accelerometer, cameras and LiDAR, respectively, and $\sum_{\omega}$, $\sum_{a}$, $\sum_{\mathcal{C}_c}$ and $\sum_{\mathcal{L}}$ are the corresponding covariances of the measurement noise. Note that these covariances can be obtained directly from each sensor's datasheet, no extra calculation is necessary.

$\boldsymbol{r}_{\omega}^{k}$ is defined similar to (7), except that an additional offset is taken into consideration:

$$\boldsymbol{r}_{\omega}^{k} := \tilde{\boldsymbol{\omega}}^{k} - \boldsymbol{R}_{\mathcal{I},\mathcal{B}}\boldsymbol{R}_{\mathcal{W},\mathcal{B}}^{T}(t_k+d_{\mathcal{I}})\,{}_{\mathcal{W}}\boldsymbol{\omega}_{\mathcal{B}}(t_k+d_{\mathcal{I}})+\boldsymbol{b}_{\omega}. \tag{10}$$

Before illustrating $\boldsymbol{r}_{a}^{k}$, we first derive the acceleration of the IMU given the continuous pose of the body and the rigid transformation between IMU and body. By (7/4) of [27], the linear acceleration of IMU expressed in $\mathcal{W}$ is

$$\begin{aligned}{}_{\mathcal{W}}\boldsymbol{a}_{\mathcal{I}}(t) = {}_{\mathcal{W}}\boldsymbol{a}_{\mathcal{B}}(t) + {}_{\mathcal{W}}\boldsymbol{\alpha}_{\mathcal{B}}(t) \times {}_{\mathcal{W}}\boldsymbol{t}_{\mathcal{B},\mathcal{I}}(t) + \\ {}_{\mathcal{W}}\boldsymbol{\omega}_{\mathcal{B}}(t) \times ({}_{\mathcal{W}}\boldsymbol{\omega}_{\mathcal{B}}(t) \times {}_{\mathcal{W}}\boldsymbol{t}_{\mathcal{B},\mathcal{I}}(t)),\end{aligned} \tag{11}$$

where ${}_{\mathcal{W}}\boldsymbol{a}_{\mathcal{B}}(t)$ is the linear acceleration of the body expressed in $\mathcal{W}$, ${}_{\mathcal{W}}\boldsymbol{\alpha}_{\mathcal{B}}(t)$ is the angular acceleration of the body expressed in $\mathcal{W}$ and ${}_{\mathcal{W}}\boldsymbol{t}_{\mathcal{B},\mathcal{I}}(t)$ is the position of $\mathcal{I}$ with respect to $\mathcal{B}$, expressed in $\mathcal{W}$. Moreover, ${}_{\mathcal{W}}\boldsymbol{t}_{\mathcal{B},\mathcal{I}}(t)$ is calculated by variables in (1) as follows:

$${}_{\mathcal{W}}\boldsymbol{t}_{\mathcal{B},\mathcal{I}}(t) = \boldsymbol{R}_{\mathcal{W},\mathcal{B}}(t)\boldsymbol{t}_{\mathcal{B},\mathcal{I}} = -\boldsymbol{R}_{\mathcal{W},\mathcal{B}}(t)\boldsymbol{R}_{\mathcal{I},\mathcal{B}}^{T}\boldsymbol{t}_{\mathcal{I},\mathcal{B}} \tag{12}$$

Then the residuals associated to the accelerometer are

$$\boldsymbol{r}_{a}^{k} := \tilde{\boldsymbol{a}}^{k} - \boldsymbol{R}_{\mathcal{I},\mathcal{B}}\boldsymbol{R}_{\mathcal{W},\mathcal{B}}^{T}(t_k+d_{\mathcal{I}})({}_{\mathcal{W}}\boldsymbol{a}_{\mathcal{I}}(t_k+d_{\mathcal{I}})+{}_{\mathcal{W}}\boldsymbol{g})+\boldsymbol{b}_{a}. \tag{13}$$

Suppose that every target $\mathcal{A}_a$ includes J landmarks on it, *i.e.* $\{{}_{\mathcal{A}_a}\boldsymbol{p}_j | a = 1, \cdots, A; j = 1, \cdots, J\}$. If ${}_{\mathcal{A}_a}\boldsymbol{p}_j$ is observed by cth camera $\mathcal{C}_c$ at time t_m, and the observation locates at $\tilde{\boldsymbol{y}}_{cmj}$ of the image, we define the residual in (9) associated to the camera as

$$\boldsymbol{r}_{\mathcal{C}_c}^{ajm} := \tilde{\boldsymbol{y}}_{cmj} - \boldsymbol{h}_{\mathcal{C}_c}(\boldsymbol{T}_{\mathcal{C}_c,\mathcal{B}}\boldsymbol{T}_{\mathcal{W},\mathcal{B}}^{-1}(t_m+d_{\mathcal{C}_c})\boldsymbol{T}_{\mathcal{W},\mathcal{A}_a}\,{}_{\mathcal{A}_a}\boldsymbol{p}_j), \tag{14}$$

where $\boldsymbol{h}_{\mathcal{C}_c}(\cdot)$ denotes the non-linear camera model.

Finally, we introduce the probability measurement model and relative error terms associated to LiDARs. In this work, only LiDAR points which hit on any calibration target plane will be utilized to create error terms, because we believe that the multiple targets in use construct enough constraints for calibration. Inspired by [10], we exploit the "line-of-sight" distance [28] to model the LiDAR measurement, *i.e.,* the errors associated with the LiDAR are the differences between the estimated and measured ranges of valid points. Fig. 3 illustrates the geometric relations of frames and variables with the following mathematical derivation:

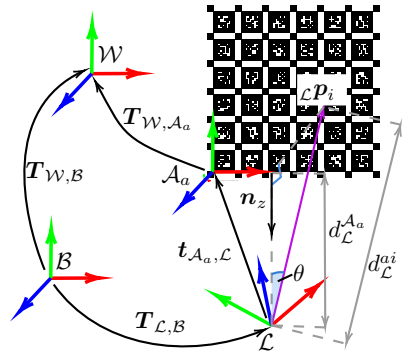

Fig. 3. The range estimation of LiDAR points hitting on the target.

$$\begin{aligned}d_{\mathcal{L}}^{ai} &:= \frac{d_{\mathcal{L}}^{\mathcal{A}_a}}{\cos\theta} \\ &= -\frac{\boldsymbol{n}_z^{T}(\boldsymbol{T}_{\mathcal{W},\mathcal{A}_a}^{-1}\boldsymbol{T}_{\mathcal{W},\mathcal{B}}(t_i+d_{\mathcal{L}})(-\boldsymbol{R}_{\mathcal{L},\mathcal{B}}^{T}\boldsymbol{t}_{\mathcal{L},\mathcal{B}}))}{\boldsymbol{n}_z^{T}\boldsymbol{R}_{\mathcal{W},\mathcal{A}_a}^{T}\boldsymbol{R}_{\mathcal{W},\mathcal{B}}(t_i+d_{\mathcal{L}})\boldsymbol{R}_{\mathcal{L},\mathcal{B}}^{T}\,{}_{\mathcal{L}}\boldsymbol{p}_i/\left\|{}_{\mathcal{L}}\boldsymbol{p}_i\right\|_2},\end{aligned} \tag{15}$$

where $d_{\mathcal{L}}^{\mathcal{A}_a}$ is the distance from the origin of a LiDAR frame to a target plane, ${}_{\mathcal{L}}\boldsymbol{p}_i$ is the point vector expressed in the LiDAR frame, $\boldsymbol{n}_z^{T} := [0, 0, 1]^{T}$ is the normal vector of the target plane expressed in the target frame, and θ represents the angle between ${}_{\mathcal{L}}\boldsymbol{p}_i$ and $\boldsymbol{n}_z$. Note that we assign each point an independent timestamp, rather than the same timestamp for points belonging to one LiDAR frame. The advantage is, that we do not need to deal specially with the motion distortion of LiDAR scans; it is handled naturally by the independent timestamp. The error term associated to the LiDAR is defined as the difference of the measurement range, $\left\|{}_{\mathcal{L}}\boldsymbol{p}_i\right\|_2$, and the estimated ranges as defined in (15):

$$\boldsymbol{r}_{\mathcal{L}}^{ai} := \left\|{}_{\mathcal{L}}\boldsymbol{p}_i\right\|_2 - d_{\mathcal{L}}^{ai}. \tag{16}$$

Unlike [10], which assumes that the plane is defined by the known normal and distance to origin expressed in the world frame, we are calibrating the targets' poses $\boldsymbol{T}_{\mathcal{W},\mathcal{A}_a}$ simultaneously with the extrinsic parameters of the LiDAR.

E. Plane Points Filtering and Refinement

Before adding error terms of the LiDAR, we perform one non-linear optimization with error terms of the IMU and cameras, which aims to make the body trajectory precise enough to filter out outlier points in the LiDAR scans which do not lie on a calibration target plane. The method hence uses points on target planes only, and is introduced here.

The most straightforward solution is given by plane fitting, but we argue that it is non-trivial to fit planes for filtering points on targets. First, the scenario could probably contain other planes, *e.g.* ground and walls, which would confuse the selection. In addition, even if the fitted planes are correct, the

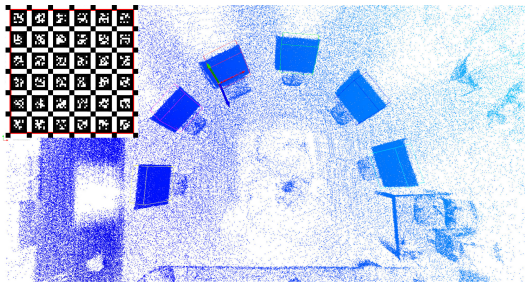

Fig. 4. Finding LiDAR points on the targets: All points are transformed to the world frame. For each board, we utilize a bounding box to illustrate the filtering, *i.e.* points inside the bounding boxes are regarded as the inliers on the corresponding target boards. At the top-left of the figure, the rectangle colored red marks the projection of the 3D bounding box on the AprilGrid.

estimated plane parameters may be inaccurate due to noise and inaccurate extrinsic parameters, because points that are not on the target may be erroneously included.

In this work, we propose a simple but efficient and robust method to filter target points. Since all variables have been initialized, we can transform all points to each target frame:

$$ {}_{\mathcal{A}_a}\boldsymbol{p}_i = \boldsymbol{T}^{-1}_{\mathcal{W},\mathcal{A}_a}\boldsymbol{T}_{\mathcal{W},\mathcal{B}}(t_i + d_{\mathcal{L}})\boldsymbol{T}^{-1}_{\mathcal{L},\mathcal{B}}\,{}_{\mathcal{L}}\boldsymbol{p}_i, \tag{17} $$

where ${}_{\mathcal{L}}\boldsymbol{p}_i$ and ${}_{\mathcal{A}_a}\boldsymbol{p}_i$ are the point positions expressed in the LiDAR frame $\mathcal{L}$ and $\mathcal{A}_a$, respectively. The points on the target board are filtered out according to ${}_{\mathcal{A}_a}\boldsymbol{p}_i$. Ideally, the z-axis component of ${}_{\mathcal{A}_a}\boldsymbol{p}_i$ is 0 if the point is on the target. But because of measurement noise and estimation errors of the variables, some inliers do not satisfy this condition, therefore we relax the z-axis range to $[-r_z, r_z]$. We define a 3D bounding box to filter points for each target, see Fig. 4.

Obviously, point filtering and optimization benefit from each other. More inliers and less outliers lead to more accurate optimization and the more accurate the estimation of the variables is, the more accurate the point cloud can be expressed in the world frame, which in turn makes it easier to distinguish inliers from outliers. To improve the calibration, we therefore alternatively improve the calibration, we alternatively perform filtering and optimization, as shown in Figure 1. r_z is set to 30cm at first, and is tightened with each iteration, which is useful to reject outliers. Note that not only the variables related to the LiDAR, but all variables in (1) are always estimated during the optimization.

IV. EXPERIMENTS

A. Simulation Experiments

For our simulation experiments we built two different multi-sensor platforms, which we denote the *IMU-camera platform* and the *ICL platform*. The experiments are defined in the Gazebo environment with AprilGrid boards.

For comparison experiments we built the *IMU-camera platform* described in [29], while the *ICL platform* features an IMU, four cameras (same as in Fig. 5) and two LiDAR sensors. The two Velodyne VLP-16 LiDARS are tilted at 15 degrees. The ICL platform is used to show our calibration performance on sensors with partial or no FoV overlap. During the data collection, the trajectories of the platforms include single-axis and random 6-DoF motions. The linear acceleration and the angular velocity are limited within the range of $\pm 3.0\ m/s^2$ and $\pm 0.3\ rad/s$, respectively. We ensure each target is observed with another target simultaneously by the same camera at least once. We record 30 ROS bags for each sensor platform, each with about 90s of data.

TABLE I

CALIBRATION ERRORS WITH ONE CAMERA, COMPARED TO KALIBR

30 bags	t errors (mm)		R errors (rad)		Time errors (ms)	
	Mean	σ	Mean	σ	Mean	σ
Kalibr	1.8724	0.4215	8.4e-04	1.6e-04		
+3ms	1.9009	0.3352	8.5e-04	1.6e-04	0.0017	0.0189
Ours	1.9702	0.2450	6.7e-04	1.2e-04		
+3ms	2.0151	0.2360	6.9e-04	1.1e-04	-0.0026	0.0272

TABLE II

CALIBRATION ERRORS FOR SIMULATED *ICL Platform* W.R.T. IMU

30 bags	t errors (mm)		R errors (rad)		Time errors (ms)	
	Mean	σ	Mean	σ	Mean	σ
Cam1	1.82	0.39	3.6e-04	9.2e-05		
Cam2	1.73	0.40	3.2e-04	9.5e-05		
Cam3	1.66	0.44	3.7e-04	8.0e-05		
Cam4	1.80	0.38	3.4e-04	7.5e-05		
C1 -3ms	1.84	0.41	3.6e-04	9.4e-05	0.0015	0.0700
C2 -3ms	1.75	0.42	3.1e-04	9.7e-05	0.0016	0.0708
C3 -3ms	1.71	0.43	3.7e-04	8.5e-05	0.0002	0.0672
C4 -3ms	1.80	0.43	3.4e-04	7.1e-05	0.0023	0.0730
LiDAR$_l$	2.6056	0.4838	4.7e-04	1.8e-04		
LiDAR$_r$	2.4806	0.5687	5.6e-04	1.8e-04		
L$_l$ +3ms	2.6045	0.4833	4.7e-04	1.8e-04	-0.0027	0.0298
L$_r$ +0ms	2.4802	0.5677	5.6e-04	1.8e-04	5.0e-04	0.0083

We evaluate the spatial and temporal calibration on both sensor platforms with the following criteria:

1) Temporal Calibration: First we test the temporal calibration capability of our approach by introducing noise to the frame's timestamps. We modify the timestamps by adding Δd_{gt} to the original timestamps, and re-estimate the time offset between the sensors. The difference between the new time-offset estimations $\hat{d}_n$ and the original ones $\hat{d}_o$, denoted as $\hat{\Delta d} = \hat{d}_n - \hat{d}_o$, is then compared to the given Δd_{gt}:

$$ e_{\Delta d} = |\Delta d_{gt} - \hat{\Delta d}|, \tag{18} $$

where $e_{\Delta d}$ is the time offset error. Note that $\hat{d}_o$ is not zero, because there exist systematic time offsets between different sensors simulated by Gazebo.

2) Spatial Calibration: We compare the estimated transformations against ground truth to compute the translation error and the rotation error as follows:

$$ e_t = \|\boldsymbol{t}_{gt} - \hat{\boldsymbol{t}}\|_2, \tag{19} $$

$$ e_R = \arccos \frac{trace(\boldsymbol{R}_{gt}^T \hat{\boldsymbol{R}}) - 1}{2} \tag{20} $$

where the rotation error is computed according to Rodrigues' rotation formula [30].

Our algorithm will degenerate into Kalibr when there is only one AprilGrid board for calibration, which is used as the baseline. We compare our algrithm with three boards against this baseline, and the results are shown in Table I. We add $\Delta d_{gt} = 3ms$ to the original IMU timestamps in the temporal

TABLE III

REAL DATA CALIBRATION RESULTS (FROM THE SPECIFIED SENSOR TO THE IMU)

33 bags	Estimated translation displacement (mm)						Estimated rotation displacement (rad)						Time errors (ms)	
Real	Mean			σ			Mean			σ			Mean	σ
Cam1	225.22	9.52	-226.69	1.17	1.07	0.87	3.1447	0.0098	1.5664	4.8e-04	5.7e-04	7.3e-04		
Cam2	132.84	8.03	-273.01	1.04	1.20	0.78	1.6033	0.0015	1.5787	5.6e-04	7.2e-04	9.6e-04		
Cam3	-206.81	6.54	-226.00	1.09	0.93	0.92	3.1415	0.0200	1.5683	0.0015	4.3e-04	6.7e-04		
Cam4	-122.60	8.97	-251.30	1.07	1.10	1.16	-1.5601	-0.0075	1.5457	5.2e-04	5.2e-04	5.1e-04		
C1 +0ms	225.22	9.52	-226.69	1.17	1.07	0.87	3.1447	0.0098	1.5664	4.8e-04	5.7e-04	7.3e-04	-0.0011	0.0063
C2 +3ms	132.84	8.04	-273.01	1.04	1.20	0.78	1.6033	0.0015	1.5787	5.6e-04	7.2e-04	9.6e-04	0.0019	0.0098
C3 +0ms	-206.81	6.54	-226.00	1.09	0.93	0.92	3.1415	0.0200	1.5683	0.0015	4.3e-04	6.7e-04	-9.5e-04	0.0053
C4 +0ms	-122.60	8.97	-251.30	1.08	1.10	1.16	-1.5601	-0.0075	1.5457	5.2e-04	5.2e-04	5.1e-04	-0.0011	0.0053
LiDAR	-104.65	0.54	-147.02	1.96	1.40	2.96	1.5863	-0.0019	0.0111	9.9e-04	0.0012	0.0014		
L +3ms	-104.63	0.50	-147.04	1.97	1.43	2.97	1.5863	-0.0019	0.0111	9.8e-04	0.0013	0.0014	0.0423	0.0933

TABLE IV

APRILGRID BOARD POSE ESTIMATION ERROR

	t errors (mm)		R errors (rad)	
Board ID	Mean	σ	Mean	σ
Sim 1	0.3322	0.0616	3.1e-04	5.6e-05
Sim 2	0.9893	0.1638	7.4e-04	1.2e-04
Sim 3	0.7171	0.0860	4.5e-04	6.6e-05
Sim 4	0.8194	0.1463	3.3e-04	6.2e-05
Sim 5	0.9309	0.1367	3.1e-04	1.2e-04
Sim 6	1.2959	0.1177	8.8e-04	1.3e-04
Real 1	19.9630	1.1932	0.0039	5.2e-04
Real 2	9.4352	0.3118	0.0024	4.9e-04
Real 3	11.2665	0.6590	0.0027	3.8e-04
Real 4	16.3319	0.5331	0.0023	3.0e-04
Real 5	22.9952	0.8487	0.0037	4.4e-04

calibration experiment. According to the result table, our algorithm performs similarly to Kalibr in both spatial and temporal calibration, showing that our algorithm successfully deals with both the single-board and multi-board scenarios.

For the *ICL platform*, Table II reports the error means and standard deviations of the estimated transformations. Compared to the results in Table I, we find that with more cameras our algorithm doesn't degrade and works even better. It's worth pointing out that though the resolution of the LiDAR is much lower than that of the cameras, the results of the LiDAR calibration are still highly accurate. The standard deviations of the translation errors are only in the range of $(0.37,\ 0.57)\ mm$. We then estimate the relative board poses and compute the estimation error. The results are reported in Table IV. The translation error means for board 1 to board 5 are below $1mm$, while the error for board 6 is $1.3mm$. A likely explanation for the reduced accuracy is given by the fact that this board placed further away from the primary target, so its position is the most affected by general spatial drift. The rotation estimation errors remain very low, in the order of 1e-04. These results again demonstrate accuracy and precision for our calibration method.

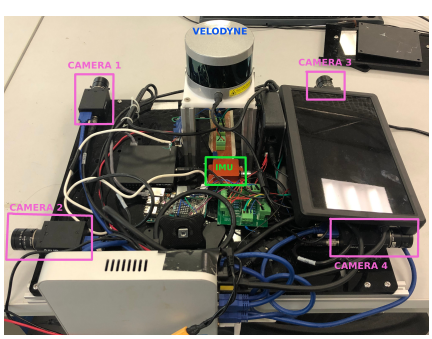

Fig. 5. The sensor platform with a LiDAR, an IMU and four cameras.

We conduct two groups of temporal calibration experiments for the *ICL platform*. In the first group we add $3ms$ to the original IMU timestamps, i.e., $\Delta t_{gt} = -3ms$. In the second we add $\Delta t_{gt} = 3ms$ to the LiDAR$_{left}$'s timestamp. Table II shows that the time offset error means are all smaller than $0.003ms$ with standard deviations lower than $0.073ms$. This experiment demonstrates the high accuracy and precision of our temporal calibration. Additionally, both Tables I and II show that the translation and rotation estimations are stable even if the time stamps of the sensors are changed.

B. Real-World Experiments

We setup a platform with one LiDAR (Velodyne VLP-16), one IMU (Xsens MTi-300) and four cameras (GS3-U3-51S5C-C), as shown in Fig. 5. The sampling rates of IMU and cameras are 200Hz and 10Hz, respectively. All sensors are hardware-synchronized using the approach introduced in [31]. Six AprilGrid boards are placed like in Fig. 2. They form a semicircle, which allows the back-to-back cameras to see boards at the same time. We record 33 ROS bags with about 90 seconds of data each.

The estimated transformations from the four cameras and the LiDAR, respectively, to IMU are listed in Table III. It is impossible to get the ground truth of the real sensor poses. We assessed the calibration precision through the standard deviation (σ). The σ for translation estimations are around $1mm$ for all axes of cameras, while that of the LiDAR sensor remains below $3mm$, due to its lower resolution. The standard deviations of the rotation estimation are below $0.0015\ (rad)$. These results show that our method has high precision also with real data. We test the temporal calibration by adding 3ms to the timestamps of *camera1* and the Velodyne and evaluate with (18). In Table III, we see that the mean of time offset estimation errors are below $0.002ms$ with σ smaller than $0.01ms$ for cameras, while that for the LiDAR is $0.04ms$ with $\sigma = 0.0933ms$.

We use a FARO Focus 3D laser scanner to create a high-accuracy 3D scan of the environment and to measure the relative pose between the AprilGrid boards as ground truth. From the computed rotation estimation errors of the board

poses in Table IV we see that the error means are below 0.004 and the standard deviations are all in the order of $1e-04$. One may notice that the biggest translation estimation error mean (the one from the primary target to target 5), is about $2.3cm$. This accuracy is acceptable, though it is worse than the results obtained in the simulation. It is mainly caused by the real-world sensor noise, which increases with the distance to the target. Nevertheless, the maximal standard deviation of the board translation estimation is below 1.1mm, which shows our outstanding spatial calibration performance.

V. Conclusions

This paper proposed an approach for spatiotemporal calibration of camera-LiDAR-IMU setups with multiple planar calibration targets. Utilizing multiple calibration targets allows us to ultimately enable the calibration of sensors which do not share FoVs. Our approach estimates the relative poses of these targets while calibrating the sensors, and the experiments indicate the excellent precision and robustness of the estimation. Furthermore, a novel and fully automatic initialization of the spatial extrinsics is proposed, especially for the LiDARs.

In the future we will make our calibration framework even more general by including other types of sensors. One limitation of this paper is that we assume that the initialization graph is connected to ensure that the poses of all targets can be estimated. But in fact hand-eye, robot-world calibration methods [32] could be utilized to estimate poses of targets and extrinsics without the need to build a connected graph, which we may consider in the future.

References

[1] R. Y. Tsai and R. K. Lenz, "A new technique for fully autonomous and efficient 3d robotics hand/eye calibration," *IEEE Transactions on Robotics and Automation*, vol. 5, no. 3, pp. 345–358, 1989.

[2] K. Daniilidis, "Hand-eye calibration using dual quaternions," *The International Journal of Robotics Research*, vol. 18, no. 3, pp. 286–298, 1999.

[3] X. Zhi and S. Schwertfeger, "Simultaneous hand-eye calibration and reconstruction," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2017, pp. 1470–1477.

[4] H. Chen and S. Schwertfeger, "Heterogeneous multi-sensor calibration based on graph optimization," in *2019 IEEE International Conference on Real-time Computing and Robotics (RCAR)*. IEEE, 2019.

[5] E. Olson, "AprilTag: A robust and flexible visual fiducial system," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, May 2011, pp. 3400–3407.

[6] P. Furgale, J. Rehder, and R. Siegwart, "Unified temporal and spatial calibration for multi-sensor systems," in *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Nov. 2013, p. 1280.

[7] M. R. Nowicki, "Spatiotemporal calibration of camera and 3d laser scanner," *IEEE Robotics and Automation Letters*, vol. 5, no. 4, pp. 6451–6458, 2020.

[8] P. Furgale, C. H. Tong, T. D. Barfoot, and G. Sibley, "Continuous-time batch trajectory estimation using temporal basis functions," vol. 34, no. 14, pp. 1688–1710.

[9] J.-H. Kim, C. Cadena, and I. Reid, "Direct semi-dense SLAM for rolling shutter cameras," in *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1308–1315.

[10] J. Rehder, R. Siegwart, and P. Furgale, "A general approach to spatiotemporal calibration in multisensor systems," *IEEE Transactions on Robotics*, vol. 32, no. 2, pp. 383–398, 2016.

[11] P. Furgale, T. D. Barfoot, and G. Sibley, "Continuous-time batch estimation using temporal basis functions," in *2012 IEEE International Conference on Robotics and Automation*, May 2012, pp. 2088–2095.

[12] Z. Zhang, "A flexible new technique for camera calibration," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 11, pp. 1330–1334, Nov. 2000.

[13] S. Garrido-Jurado, R. Muoz-Salinas, F. J. Madrid-Cuevas, and M. J. Marn-Jimnez, "Automatic generation and detection of highly reliable fiducial markers under occlusion," *Pattern Recognition*, vol. 47, no. 6, pp. 2280–2292, 2014.

[14] A. Richardson, J. Strom, and E. Olson, "AprilCal: Assisted and repeatable camera calibration," in *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Nov. 2013, pp. 1814–1821.

[15] F. Vasconcelos, J. P. Barreto, and U. Nunes, "A Minimal Solution for the Extrinsic Calibration of a Camera and a Laser-Rangefinder," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 34, no. 11, pp. 2097–2107, Nov. 2012.

[16] A.-I. García-Moreno, J.-J. Gonzalez-Barbosa, F.-J. Ornelas-Rodriguez, J. B. Hurtado-Ramos, and M.-N. Primo-Fuentes, "Lidar and panoramic camera extrinsic calibration approach using a pattern plane," in *Mexican Conference on Pattern Recognition*. Springer, 2013, pp. 104–113.

[17] Y. Zhuang, F. Yan, and H. Hu, "Automatic extrinsic self-calibration for fusing data from monocular vision and 3-d laser scanner," *IEEE Transactions on Instrumentation and Measurement*, vol. 63, no. 7, pp. 1874–1876, 2014.

[18] V. Fremont, S. A. R. F., and P. Bonnifait, "Circular targets for 3d alignment of video and lidar sensors," *Advanced Robotics*, vol. 26, no. 18, pp. 2087–2113, 2012. [Online]. Available: https://doi.org/10.1080/01691864.2012.703235

[19] R. K. Kumar, A. Ilie, J.-M. Frahm, and M. Pollefeys, "Simple calibration of non-overlapping cameras with a mirror," in *2008 IEEE Conference on Computer Vision and Pattern Recognition*, 2008.

[20] G. Long, L. Kneip, X. Li, X. Zhang, and Q. Yu, "Simplified mirror-based camera pose computation via rotation averaging," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 1247–1255.

[21] Y. Xie, R. Shao, P. Guli, B. Li, and L. Wang, "Infrastructure Based Calibration of a Multi-Camera and Multi-LiDAR System Using Apriltags," *2018 IEEE Intelligent Vehicles Symposium (IV)*, 2018.

[22] L. Oth, P. Furgale, L. Kneip, and R. Siegwart, "Rolling shutter camera calibration," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2013, pp. 1360–1367.

[23] J. Rehder, J. Nikolic, T. Schneider, T. Hinzmann, and R. Siegwart, "Extending kalibr: Calibrating the extrinsics of multiple IMUs and of individual axes," in *2016 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2016, pp. 4304–4311.

[24] J. Lv, J. Xu, K. Hu, Y. Liu, and X. Zuo, "Targetless calibration of lidar-imu system based on continuous-time batch estimation," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2020, pp. 9968–9975.

[25] H. Sommer, J. R. Forbes, R. Siegwart, and P. Furgale, "Continuous-time estimation of attitude using b-splines on lie groups," *Journal of Guidance, Control, and Dynamics*, vol. 39, no. 2, pp. 242–261, 2016.

[26] C. Sommer, V. Usenko, D. Schubert, N. Demmel, and D. Cremers, "Efficient derivative computation for cumulative b-splines on lie groups," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 11 148–11 156.

[27] J. L. Meriam and L. G. Kraige, *Engineering mechanics: dynamics*. John Wiley & Sons, 2012, vol. 2.

[28] E. So and E. Menegatti, "A unified approach to extrinsic calibration between a camera and a laser range finder using point-plane constraints," in *Proceedings of the 1st International Workshop on Perception for Mobile Robots Autonomy*, 2012.

[29] P. Furgale, J. Rehder, and R. Siegwart, "Unified temporal and spatial calibration for multi-sensor systems," in *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2013, pp. 1280–1286.

[30] D. Q. Huynh, "Metrics for 3d rotations: Comparison and analysis," *Journal of Mathematical Imaging and Vision*, vol. 35, no. 2, pp. 155–164, 2009.

[31] H. Chen, Z. Yang, X. Zhao, G. Weng, H. Wan, J. Luo, X. Ye, Z. Zhao, Z. He, Y. Shen *et al.*, "Advanced mapping robot and high-resolution dataset," *Robotics and Autonomous Systems*, p. 103559, 2020.

[32] H. Zhuang, Z. Roth, and R. Sudhakar, "Simultaneous robot/world and tool/flange calibration by solving homogeneous transformation equations of the form AX=YB," *IEEE Transactions on Robotics and Automation*, vol. 10, no. 4, pp. 549–554, 1994.