Finding Timestamp Offsets for a Multi-Sensor System Using Sensor Observations*

Raphael Voges, Christian S. Wieghardt, and Bernardo Wagner

Abstract

Multi-sensor systems are widely used for robotics applications. While additional sensors can increase the accuracy and robustness of the solution, it is inevitable to synchronize them in order to rely on the results. For our multi-sensor system consisting of an actuated laser scanner, its motor and a camera, we assume that the timestamps are only delayed by a constant offset. We propose two different approaches to calculate timestamp offsets from laser scanner to motor, one of which is additionally capable of determining the timestamp offset between laser scanner and camera. Both approaches use parts of a SLAM algorithm but apply different criteria to find an appropriate solution. Our experiments show that we are able to determine timestamp offsets with a reasonable accuracy. Furthermore, our experiments exhibit the significance of a proper synchronization for a multi-sensor system.

Introduction

For many applications a sensor system consisting of a laser scanner and camera is used to solve the SLAM problem (Droeschel *et al.*, 2014; J. Zhang and Singh, 2015). While in the past 2D laser scanners were sufficient for the navigation of mobile robots in planar environments, recent SLAM approaches deal with 3D data to avoid obstacles at all heights and simultaneously acquire a dense 3D point cloud of the environment (Nuechter *et al.*, 2007; Bosse and Zlot, 2009; Bosse, Zlot, and Flick, 2012; J. Zhang and Singh, 2014). However, 3D laser scanners that provide high resolution and long ranges are expensive. Therefore, cheaper 2D laser scanners that are usually only capable of acquiring scan points in a plane are actuated by a servo-drive to gather 3D data (Wulf and Wagner, 2003).

To transform the measurement points into three-dimensional space, it is required to know the appropriate encoder values of the servo drive for every set of scan points. Because of that, there exist two different approaches to determine these encoder values. The first and most simple possibility is using a motor that stops at discrete steps to let the laser scanner capture measurement points (Mandow *et al.*, 2010). This solution, however, leads to a lower data rate since the laser scanner and the motor have to wait for each other before performing a measurement or a rotation. Another possibility is to continuously monitor and control the motion of the motor while acquiring measurement points with a high scan frequency (Wulf and Wagner, 2003; Yoshida *et al.*, 2010). However, this can lead to a constant offset between the timestamps of the laser scanner and the motor due to the latency and transmission lags of sensors and computers. Therefore, it is essential to achieve a proper synchronization between the timestamps of the laser scanner and its rotating motor as it is already mentioned by Hebert and Krotkov (1992). If no synchronization is present, the offset for the corresponding encoder values for each set of scan points can lead to a large distortion in the resulting point cloud that is constructed by a SLAM (Simultaneous Localization And Mapping) approach (Wulf and Wagner, 2003).

Furthermore, if the images taken by a camera are fused with the measurements of the laser scanner, a proper synchronization between these two sensors is indispensable as well. Otherwise, the point clouds and images that may have been assigned with the same timestamp do not correspond to the same actual moment of time, and thus cannot be fused to generate a consistent representation of the environment.

Thus, our aim is to correct distortion in point clouds acquired by fusing the measurements of a rotating laser scanner and a camera that arises due to erroneous timestamp offsets between the three devices, i.e., namely the actuated laser scanner, its corresponding motor, and the camera. For this purpose, we assume that the timestamp offsets are constant throughout our measurement periods.

We present two different approaches to determine the timestamp offsets. The first, stationary approach can be used to synchronize the laser scanner and motor before using it for an online algorithm that requires correctly transformed 3D data. For this, it is necessary to not move the system for a short period of time and wait for the calibration to finish. The second, motion-based approach makes it possible to determine the offset between laser scanner, motor, and camera after the acquisition of a large dataset. Thereby, it becomes possible to use the dataset for offline computations although the initial synchronization of the multi-sensor system is not optimal. To verify the synchronization results between laser scanner and motor, we compare the offsets computed by both approaches.

Related work in the field of synchronizing actuated laser scanners is presented by Morales *et al.* (2011). Within this work the authors describe the design and development of a mechanical system that is used to rotate the laser scanner. Furthermore, a motion controller that is responsible for the synchronization between the mechanical system and the laser scanner is presented. The distinction is that our method solely focuses on the time synchronization, and thus can be applied to arbitrary motor and laser scanner combinations.

Similar to our approach, Sheehan *et al.* (2010) attempt to design a 3D laser scanner that is capable of automatic self-calibration. Their system consists of an arbitrary number of 2D laser scanners that are mounted on a rotating plate. In addition to other extrinsic parameters they also deal with the estimation of the clock skews between their devices. For this,

Leibniz University of Hannover - Institute of Systems Engineering - The Real Time Systems Group (voges@rts.uni-hannover.de).

*This paper is an extended version of work published in (Voges *et al.*, 2017). We extend our previous work by incorporating a camera as an additional sensor whose timestamp offset to the laser scanner needs to be determined.

Photogrammetric Engineering & Remote Sensing
Vol. 84, No.6, June 2018, pp. 357–366.
0099-1112/18/357–366

doi: 10.14358/PERS.84.6.357

they initially try to learn the offsets between the clocks of all devices. Afterwards, they use a second method to determine the transport delays. For this purpose they evaluate the "crispness" of resulting point clouds using different criteria than us. A further distinction is that our approach focuses on one laser scanner only, and thus it is not required to employ two separate algorithms to detect the offset.

Rehder *et al.* (2016) present a general approach to achieve a spatiotemporal calibration in multi-sensor systems. Their method employs a continuous-time batch estimation that does not rely on certain properties of specific sensors. Thereby, they are able to estimate the temporal offset between various sensor combinations. The authors show the usefulness of their approach by determining the timestamp offset between camera, IMU, and laser scanner. Similar to our approach, they assume that the clocks of the sensors are delayed by constant timestamp offsets that arise due to effects such as transmission delays or signal integration. However, they propose a fundamentally different method which uses different criteria and is computationally expensive. Furthermore, their approach cannot be used to estimate temporal offsets between an actuated laser scanner and its corresponding motor.

The remainder of this paper is organized as follows. First, the sensor system we used to evaluate the methods presented within this paper is introduced followed by our approach to calculate the timestamp offset between the laser scanner, motor, and camera. Since we developed two independent methods, this section is divided into two subsections: one for the stationary approach that calculates the offset prior to data acquisition and one for the approach that deals with large datasets. Experiments demonstrating both methods are then presented for a number of datasets of different characteristics. Subsequently, we analyze and discuss the results of our experiments, and finally, give a conclusion and future work.

System Overview

The idea of this paper is validated on a sensor system consisting of a Hokuyo UTM-30LX laser scanner that is actuated by a Dynamixel MX-64R motor, and a Microsoft Kinect v2. The laser scanner can provide a 2D scan with a field of view of 270° and an angular resolution of 0.25°. However, for our experiments the field of view is limited to 180° to avoid detection of the frame the laser scanner is attached to. Every measurement of the laser scanner takes 25 ms which leads to a scan frequency of 40 lines/sec. Furthermore, the laser scanner has a maximum detection range of 30 m and a minimum detection range of 0.1 m.

The Dynamixel MX-64R robot actuator is able to operate at an angle of 360° or at a continuous turn. Besides, the motor supports the measurement of its own position and speed. For this, it provides an angular resolution of 0.088°. To control the motion of the actuator we use the Dynamixel motor package that is available for the Robot Operating System (ROS).

The Microsoft Kinect v2 depth-sensing camera provides RGB (1920 × 1080 pixels, 30 fps), depth (512 × 424 pixels, 30 fps) and active IR images (512 × 424 pixels, 30 fps). Our camera's focal length is roughly 267 pixels, and it covers a field of view of 70° × 60°.

Unfortunately, Microsoft has not released information about the dimension of the RGB camera sensor. For this work we only use the RGB stream with a resolution of 480 × 270 pixels and 10 fps. The typical distance to objects that we use as features for our visual odometry is 2 to 20 meters.

We determine the camera's intrinsic parameters using the algorithm proposed by Z. Zhang (2000) and its implementation in the Open Source Computer Vision Library (OpenCV). We consider two parameters for the focal length, two parameters for the principal point and three parameters for the radial distortion. We do not determine the tangential distortion parameters since they show no effect for the Microsoft Kinect v2.

The laser scanner and the motor are connected to a Kontron KTQM87 based embedded PC which runs the motion controller for the actuator. Moreover, the embedded PC collects the measurement data from the laser scanner as well as the position data from the motor and assigns timestamps to them. Due to latency and transmission lags of sensors and the embedded PC, these timestamps may be delayed by a constant offset that needs to be determined.

It is important to note that the devices are attached to different ports of the embedded PC. While the motor is connected to the USB port using a USB to RS485 converter, the laser scanner is attached to the LAN port. Thus, our assumption about a constant offset remains valid since both devices do not interfere with the measurement data acquisition of each other due to the utilization of different ports. Additionally, our embedded PC does not operate at full computational load which further ensures a constant timestamp offset.

The Microsoft Kinect v2 is connected to Nvidia's Tegra-based Jetson TK1 embedded board to the USB. To synchronize the clocks of the Kontron PC and the Tegra board, we use the Network Time Protocol (NTP) (Mills, 1991). However, an offset between the laser scanner's and the camera's timestamps still remains, since there are transmission delays from each sensor to the system it is attached to. Furthermore, we do not expect NTP to work perfectly which introduces another source for possible timestamp offsets.

Our multi-sensor system can be seen in Figure 1. For our experiments we focused on the rolling scan method for which the laser scanner is rotated around its center. This gives the advantage of only one focus point in front of the laser scanner (Wulf and Wagner, 2003).

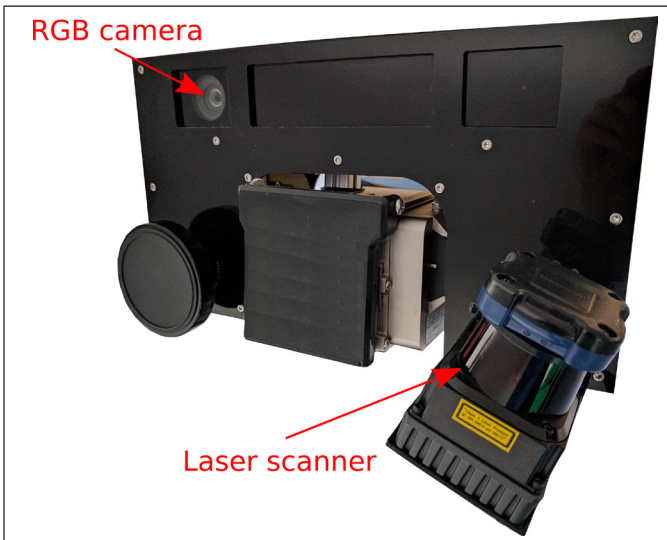

Figure 1. Our multi-sensor system consists of a Microsoft Kinect v2 and a Hokuyo UTM-30LX scanning laser rangefinder that is rotated by a Dynamixel MX-64R robot actuator.

The motor is set to control the laser scanner such that a sweep lasts 0.5 s, where a sweep is the rotation from −90° to +90° or in the inverse direction with the horizontal orientation as 0°. This yields a rotation frequency of 1 Hz since a sweep is half a full rotation. The frequency at which a complete 3D scan is acquired amounts to 2 Hz.

Estimating Timestamp Offsets Between Laser Scanner, Motor, and Camera

Within this section we will present two different approaches to compute the offsets between the timestamps of a laser scanner and its corresponding motor. The stationary approach can be used to determine the offset for a system that needs to be adequately synchronized for following online computations. The motion-based approach calculates the offset for a large dataset. This enables offline computations for the dataset although the offset between the timestamps of the laser scanner and motor was not known during data acquisition. Additionally, the motion-based approach can be used to determine the timestamp offset between a laser scanner and a camera that are fused for a SLAM algorithm. Before we start explaining our methods we give an overview of the SLAM approach that is proposed by J. Zhang and Singh (2015).

Overview of the Utilized SLAM Approach

For our methods we use the SLAM approach that is proposed by J. Zhang and Singh (2015). It is divided into two parts: visual odometry (J. Zhang, Kaess, *et al.*, 2014) and lidar odometry (J. Zhang and Singh, 2014). At first, the visual odometry calculates a motion estimation that is then further refined by the lidar odometry part.

Visual Odometry

The visual odometry method (J. Zhang, Kaess, *et al.*, 2014) uses visual features extracted from grayscale images to estimate the motion between consecutive frames. Since depth information is required to solve the odometry problem, the algorithm maintains a depth map consisting of point clouds that are measured using the laser scanner and transformed using the previously estimated motion. This depth map is projected onto the last image frame such that it is possible to find the depth of 2D visual features by interpolating between the three closest points from the depth map. If it is not possible to assign depth values to visual features using this method, the depth can be triangulated using the previously estimated motion or the feature will be used without depth.

To estimate the six-DOF (degree of freedom) parameters, the motion is modeled as rigid body transformation. The equations are set up such that the six-DOF parameters applied to the coordinates of a visual feature of the previous frame should yield the same coordinates as the corresponding visual feature of the consecutive frame. Using this procedure, features with depth provide two equations while features without depth provide one equation. Now, all equations are stacked to a nonlinear system that is solved using a nonlinear optimization algorithm.

We use ORB (Oriented FAST and Rotated BRIEF) (Rublee *et al.*, 2011) to extract and match the visual features between consecutive frames. An example of features that were extracted in a lecture room can be seen in Figure 2. Furthermore, we refrain from triangulating the depth for visual features since we only need the visual odometry as an initial guess for the lidar odometry. This reduces the computational cost.

Lidar Odometry

The lidar odometry method (J. Zhang and Singh, 2014) consists of two algorithms that run separately. The first algorithm (sweep to sweep refinement) takes the motion estimation calculated by the visual odometry as an initial guess that is susceptible to drift. We get motion estimates with a frequency of 10 Hz from our visual odometry algorithm and apply them to project our scan points do the beginning of the sweep. The remaining drift that cannot be determined using the visual odometry, is considered as slow motion during a sweep;where a sweep is the rotation from −90° to + 90° or in the inverse direction with the horizontal orientation as 0°; and thus, can be modeled with constant velocity. Now, the aim is to determine this drift and use it to correct distortion in the point clouds that arises due to the motion of the laser scanner during a sweep.

To extract scan features, the curvature of every 2D scan point with respect to its neighbors is calculated as:

$$c(i) = \frac{1}{2 \cdot N \cdot \|\mathbf{X}_i\|} \cdot \left\| \sum_{j=1}^{N} \left(\mathbf{X}_i - \mathbf{X}_{i-j}\right) + \left(\mathbf{X}_i - \mathbf{X}_{i+j}\right) \right\|$$

where $\mathbf{X}_i$ are the 2D-coordinates of a scan point i, N is the number of 2D scan points that are considered for the curvature calculation to both sides of i in the ordered 2D scan and $\|\mathbf{X}_i\|$ is the euclidean norm for $\mathbf{X}_i$. Because of the scanner's resolution, two successive scan points are 0.25° apart. For our experiments we set $N = 5$, which showed to be robust amongst different datasets.

The 2D scan points are distributed into four subregions and sorted based on the curvature values. From every subregion the two points with the largest curvature that also exceed a threshold are chosen as edge points and the four points with the smallest curvature that also do not exceed a threshold are chosen as planar points. We chose a threshold of 0.8 for edge points and 0.45 for planar points, which showed to be robust amongst all datasets as well. An example for the feature extraction can be seen in Figure 3.

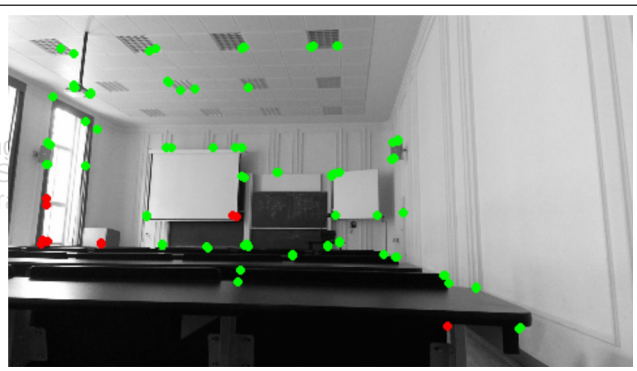

Figure 2. Image of a lecture room with ORB features marked as dots. Features whose depth was found are marked in green while red dots correspond to features with unknown depth.

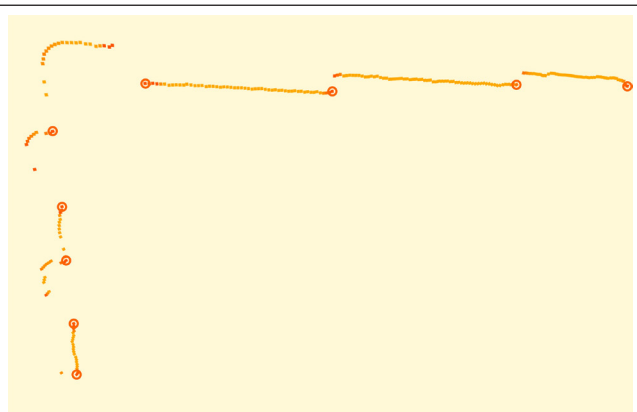

Figure 3. An example of 2D laser scan points for which the curvature is calculated. Dark orange points correspond to scan points with large curvature values while light orange points indicate a small curvature value. Circled points are extracted as edge points. Since too many planar points were extracted they are not depicted in this figure.

Afterwards, the sweep to sweep refinement determines the motion of the laser scanner by matching extracted edge points to edges and planar points to planar patches from the previous sweep. Subsequently, an optimization algorithm is used to minimize the distance between the correspondences.

The second algorithm (sweep to map registration) is responsible for matching the edge points and planar points from the last sweep onto the global map. Thereby, the second algorithm is able to correct for drift over time. An example for a map can be seen in Figure 4.

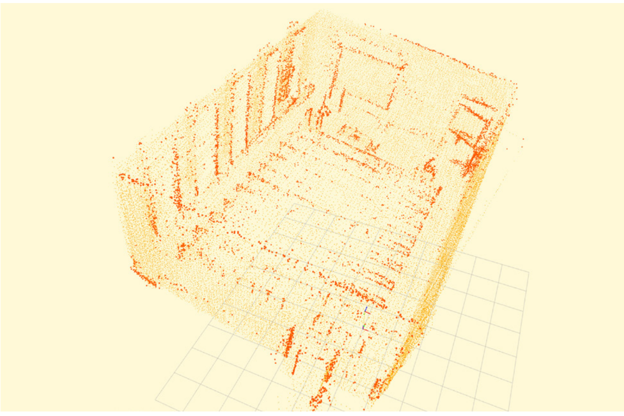

Figure 4. An example of a map that is used within the sweep to map registration. Dark orange points were extracted as edge points while light orange points were extracted as planar points.

To register an edge point from the most recent sweep onto the map, the algorithm finds edge points within a certain region around the newly extracted edge point in the map and fits an edge through them. In a similar way, the corresponding planar patch for a planar point is determined. Afterwards, both feature types are combined in an optimization algorithm to minimize the distance from edge points to corresponding edges and from planar points to corresponding planar patches.

For the lidar odometry algorithm to work, it is essential to detect enough edges and planar patches. However, if these feature points are missing for a short period of time only, the visual odometry motion estimation that is used as an input for the lidar odometry algorithm is capable of compensating the absent laser feature points. Similarly, if no visual features are available to estimate the motion in the visual odometry algorithm, the lidar odometry algorithm can still work. However, if the visual odometry fails while performing rapid movements (especially rotations) the lidar odometry algorithm will not be able to recover. Although visual and laser scan features were missing for short periods of time for all our datasets, the motion estimation algorithm was able to perform well. To obtain a reasonable motion estimation (and therefore a reliable timestamp offset) we advise to obtain data in environments which allow the laser scanner to observe at least two perpendicular edges and one planar patch at all times. Additionally, it is important for the visual odometry algorithm to find features that are distributed across the entire image.

Stationary Approach Prior to Data Acquisition for Online Computations

To compute the offset between the timestamps of the laser scanner and the motor, the system is set up as follows. The motor is set to rotate the laser scanner at a constant angular velocity around the center of the scanner. Subsequently, the devices are brought into a fixed position and required to remain in place until a short dataset is recorded.

The idea is to determine the offset that leads to the smallest movement calculated by a SLAM approach that incorporates the desired time offset between the timestamps of the laser scanner and the motor. Since the system remains stationary for the computation, the movement calculated by a SLAM approach should be zero. However, due to transformation errors caused by a false timestamp offset the 3D data points do not match perfectly from one sweep to another and lead to erroneously computed movements by the SLAM algorithm. This is because consecutive sweeps are acquired in opposite directions. We use parts of the lidar odometry algorithm proposed by J. Zhang and Singh (2014) that was introduced in the previous subsection.

Since our system will not move within the map, we exclusively use the sweep to sweep refinement which has the important characteristic that it matches consecutive sweeps acquired in opposite directions (for one sweep the laser scanner is rotated from −90° to 90° and for the following sweep the scanner is rotated from 90° to −90° or vice versa). As a result, an offset between the timestamps of the laser scanner and the motor leads to an offset in the consecutive point clouds which in turn induces a nonzero motion calculated by the sweep to sweep refinement.

To determine the motion calculated by the sweep to sweep refinement, it is necessary to first compute the translational and rotational movement separately. The translation can be computed as:

$$t = \sqrt{t_x^2 + t_y^2 + t_z^2}$$

where t_x, t_y and t_z are translations along the x-, y- and z- axes. Similarly, the magnitude of the rotation can be computed as:

$$\theta = \sqrt{\theta_x^2 + \theta_y^2 + \theta_z^2}$$

where $(\theta_x, \theta_y, \theta_z)^{\mathrm{T}}$ is a vector representing the rotation axis while simultaneously matching the magnitude of the rotation by its length. Both the translational and rotational movement can be combined in the following equation:

$$d = \sqrt{t^2 + c \cdot \theta^2}$$

where $c \geq 0$ is a weighting factor. For our experiments we set $c=1$. Now, we determine d_i for every sweep i during our short dataset and aim to minimize the average motion $\frac{1}{k}\sum_{i=1}^{k} d_i$ that is calculated by the sweep to sweep refinement over all k sweeps.

To determine the timestamp offset for a large dataset that is already recorded, we again use the SLAM approach that is proposed by J. Zhang and Singh (2015). As opposed to the previous subsection, the system is now allowed to move which makes it impossible to use the same strategy as before. Instead, we use both the visual odometry method and the lidar odometry method and try to find the timestamp offset that induces the greatest clarity in the resulting point cloud of the environment. For this purpose, it is crucial to define adequate criteria that are viable to evaluate the clarity of a point cloud. Using these criteria, it is then possible to determine both the timestamp offset from laser scanner to motor and from laser scanner to camera.

Motion-based approach after data acquisition for offline computations

The loop closure error cannot be used as a criterion since it can be small although the SLAM algorithm performed poorly between start and end pose. That is because the algorithm may be able to close the loop by matching feature points

detected at the end pose to feature points from the map that were added at the start pose. Nevertheless, the computed trajectory between start and end pose may still deviate from reality. On this account, we refrain from specifying the loop closure error for our algorithm since it can be misleading.

The first suitable criterion is the amount of matches *n* that are registered onto the map in the sweep to map registration. This is a meaningful criterion since more matches indicate that more features could be integrated into the map and thus the clarity is greater. However, if the amount of matches is low, it follows that the extracted features were not close to their corresponding edge lines or planar patches, which suggests that the clarity of the resulting point cloud is low. Thus, the aim is to maximize this criterion.

Before we can compute *n*, we need to define the sets $D_{\mathcal{E}}^k$ and $D_{\mathcal{H}}^k$ that contain all matched edge points and planar points for sweep *k* as:

$$D_{\mathcal{E}}^k := \{ i \in \mathcal{E}_{k+1} \mid d_{\mathcal{E}}(i,j,l) < \delta_{\mathcal{E}};\ j,l \in \mathcal{Q}_k \}$$

where $\mathcal{E}_{k+1}$ is the set of edge points that were extracted for sweep $k+1$, $\mathcal{Q}_k$ is the set of all points that were integrated into the global map until sweep k, $d_{\mathcal{E}}(i,j,l)$ is the distance between an edge point i and its corresponding edge line that is represented by two points j,l in the global map $l \in \mathcal{Q}_k$ (cf. equation (2) by J. Zhang and Singh (2014)) and $\delta_{\mathcal{E}}$ is the maximum distance between an edge point and its corresponding edge line to consider them as a match. $D_{\mathcal{H}}^k$ is defined as:

$$D_{\mathcal{H}}^k := \{ i \in \mathcal{H}_{k+1} \mid d_{\mathcal{H}}(i,j,l,m) < \delta_{\mathcal{H}};\ j,l,m \in \mathcal{Q}_k \}$$

where $\mathcal{H}_{k+1}$ is the set of planar points that were extracted for sweep $k+1$, $d_{\mathcal{H}}(i,j,l,m)$ is the distance between an planar point i and its corresponding planar patch that is represented by three points (j,l,m) in the global map $m \in \mathcal{Q}_k$ (cf. equation (3) by J. Zhang and Singh (2014)) and $\delta_{\mathcal{H}}$ is the maximum distance between an planar point and its corresponding planar patch to consider them as a match. Using both sets we can derive an equation for the total number of matches *n* as:

$$n = \sum_{k \in S} \left(\left| D_{\mathcal{E}}^k \right| + \left| D_{\mathcal{H}}^k \right| \right),$$

where S is the set of all sweeps that were acquired during the measurement and $|A|$ is the cardinality of the set A.

The second criterion is the average error *e* for each match that is registered onto the map. The lower the error the closer the matches are to their corresponding edge lines or planar patches which in turn suggests a greater clarity. Thus, the aim is to minimize this criterion. The average error *e* can be computed as:

$$e = \frac{\sum_{k \in S} \left(\sum_{i \in D_{\mathcal{E}}^k} d_{\mathcal{E}}(i,j,l) + \sum_{i \in D_{\mathcal{H}}^k} d_{\mathcal{H}}(i,j,l,m) \right)}{n}$$

where $d_{\mathcal{E}}(i,j,l)$ and $d_{\mathcal{H}}(i,j,l,m)$ are the distances between feature correspondences as depicted in the equations.

Now, with these two criteria it is possible to determine appropriate timestamp offsets from laser scanner to motor and from laser scanner to camera. The reasoning for this is that a more fitting offset induces a point cloud that shows a greater clarity. This follows from the fact that scan features transformed with the correct encoder values can be matched more straightforward than those that are not correctly transformed. Similarly, if the initial motion estimation from the visual odometry algorithm is correct, the lidar odometry is able to find more matches. In contrast, if the timestamp offsets do not fit, features from consecutive sweeps do not align and thus cannot be integrated into the global map.

To further confirm the relevance of our criteria, we carried out an additional subjective test. For this, we compared the two criteria to the authors' perceived clarity of the point clouds that resulted from running the SLAM approach using different timestamp offsets. To enable the reader to judge the clarity of different point clouds as well, we show the results of our experiments in the following section.

Finding the Appropriate Timestamp Offsets

To find an appropriate offset, we use a brute-force approach that takes an initial offset, a required accuracy for the final offset and a maximum number of steps. Thus, it is necessary to manually guess an initial range that encloses the optimal offset. This range can be arbitrarily large for the laser scanner to camera synchronization. However, for the laser scanner to motor synchronization it must not exceed the duration of one motor rotation to avoid multiple minimums that may arise from the cyclic properties of the problem. The algorithm then iterates over possible offsets starting from the initial offset and taking steps in both directions in the size of the required accuracy. For every offset, the required parts of the SLAM approach are executed and its criteria (magnitude of motion or total number of matches and average error per match) are determined as depicted in the previous subsections. Finally, the offset for which the appropriate criteria approached their optimum is returned.

Experiments

In this section we present the results of our experiments using the system outlined in a previous section. At first, we display the results of our stationary method discussed above. Afterwards, the datasets are introduced before finally presenting the results of our motion-based approach.

Stationary Approach Prior to Data Acquisition

Before recording large datasets, we acquired six datasets of roughly four seconds each. For these datasets the system was kept in a fixed position to avoid motion during the measurement. In order to minimize error influences due to measurement noise and observed scenes, we chose to use six different positions in the same room. The offset between the timestamps of the laser scanner and the motor is determined using the previously described method. Initially, we ran the algorithm with a desired accuracy of 1 ms in the range from 19 ms to 29 ms to determine a first approximation of the solution. We worked with this initial range since our algorithm did not succeeded with offsets outside this range. These experiments revealed that the final solution lies within the range of 22 ms to 26 ms which prompted us to carry out an additional experiment with an accuracy of 0.1 ms within this range. Further tests were not required since the final experiment with an accuracy of 0.1 ms yielded results that did not improve by a large margin from one step to another.

The results for all six experiments can be seen in Table 1. It can be seen that all calculated timestamp offsets vary around 24 ms while the magnitude of motion ranges from 0.0037 to 0.0314. This can be explained by measurement noise that influences the laser scanner. However, the calculated timestamp offsets still coincide which shows that the approach is prone to some measurement noise. The average offset over all six datasets amounts to 24.0 ms.

Additionally, in Figure 5a and 5b the resulting graphs for experiment number one and four, respectively, are depicted. These are exemplary for all six experiments. It can be seen that the magnitude of motion increases sharply if the observed timestamp offset differs a few milliseconds from the optimum.

Table 1. Results of the stationary offset computation prior to data acquisition.

No.	Offset [ms]	Magnitude of motion d
1	24.3	0.0037
2	23.4	0.0314
3	23.9	0.0203
4	24.1	0.0070
5	23.8	0.0220
6	24.6	0.0194

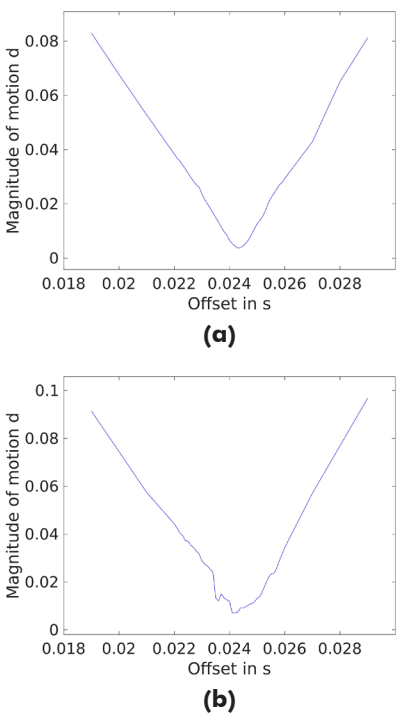

Figure 5. Results generated by the stationary approach to calculate timestamp offsets prior to data acquisition. The measurement graphs belong to experiments whose results are depicted in Table 1: (a) Experiment No. 1, and (b) Experiment No. 4.

Datasets

For our experiments we acquired four datasets of different characteristics. Before we introduce and discuss the results that our motion-based approach yields, we want to present those datasets to give the reader an impression. To create a representation of the datasets, we used the previously discussed SLAM approach that is presented by J. Zhang and Singh (2015). Note that no georeferencing was used and we carried the sensor setup along for all data recordings.

The first dataset contains data that was recorded in an abandoned metro station. In a period of roughly 159 s we cover a distance of around 117 m with a maximum velocity of 1.3 m/s and a maximum angular rate of 0.44 rad/s. Moreover, we finish the measurement in the same spot that we started it in. The final map can be seen in Figure 6a.

For the second dataset we moved across a parking area. In a period of roughly 208 s we cover a distance of around 155 m with a maximum velocity of 1.5 m/s and a maximum angular rate of 0.61 rad/s. Again, we finish in the same spot that we started in. In Figure 6b the corresponding map can be seen.

The third dataset was recorded on a cemetery. In a period of roughly 287 s we cover a distance of around 236 m with a maximum velocity of 1.4 m/s and a maximum angular rate of 0.74 rad/s. Once more, we finish in the same spot that we started in. The final map can be seen in Figure 6c.

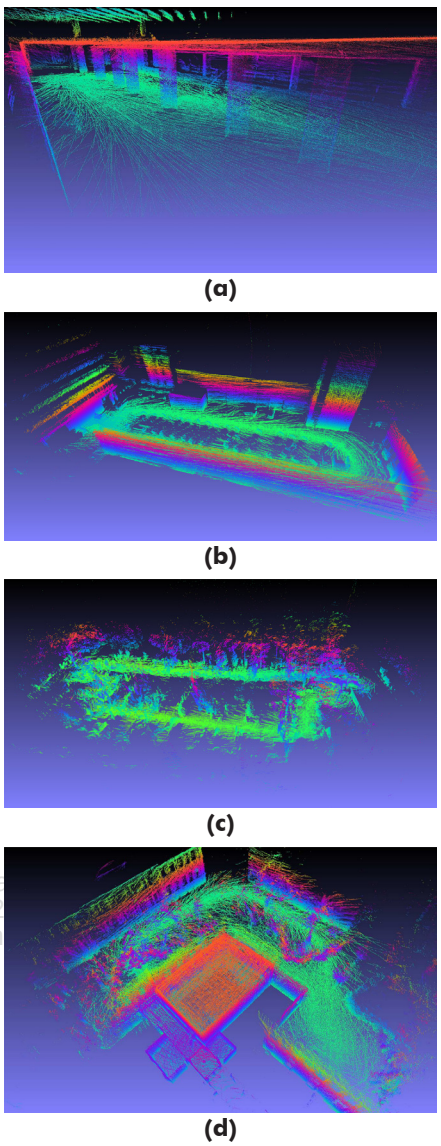

Figure 6. Maps generated using the discussed SLAM approach. Point clouds are color coded by elevation to generate 3D perception: (a) Metro station, (b) Parking area, (c) Cemetery, and (d) Lecture room

For the fourth dataset we choose an environment of several different characteristics. We started our measurement in an empty lecture room and finished it in an outside area in which trees and building facades were present. In a period of roughly 274 s we cover a distance of around 184 m with a maximum velocity of 1.9 m/s and a maximum angular rate of 0.91 rad/s. This time we do not finish in the same spot that we started in. Figure 6d depicts the final map for this dataset.

Motion-Based Approach After Data Acquisition

As explained in the previous section, our motion-based approach allows us to estimate both the timestamp offset between laser scanner and motor, and the timestamp offset between laser scanner and camera. To exclude the possibility of an unknown correlation between both timestamp offsets, we evaluated our criteria for different combinations of these two parameters and came to the conclusion that these parameters are indeed uncorrelated.

Table 2. Results of the offset computation between laser scanner and motor for all four datasets and for both criteria that were introduced.

Dataset	Offset using the total number of matches n [ms]	Offset using the average error per match e [ms]
Metro station	23.9	24.1
Parking area	23.8	23.7
Cemetery	23.6	23.3
Lecture room	23.6	23.8

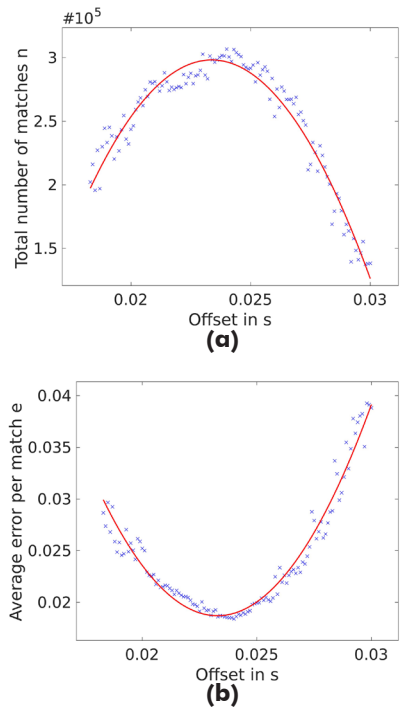

Figure 7. Measurement graphs for the timestamp offset between laser scanner and motor for the parking area dataset: (a) Total number of matches *n*, and (b) Average error per match *e*.

Table 3. Results of the offset computation between laser scanner and camera for all four datasets and for both criteria that were introduced.

Dataset	Offset using the total number of matches n [ms]	Offset using the average error per match e [ms]
Metro station	140	137
Parking area	90	83
Cemetery	76	59
Lecture room	100	96

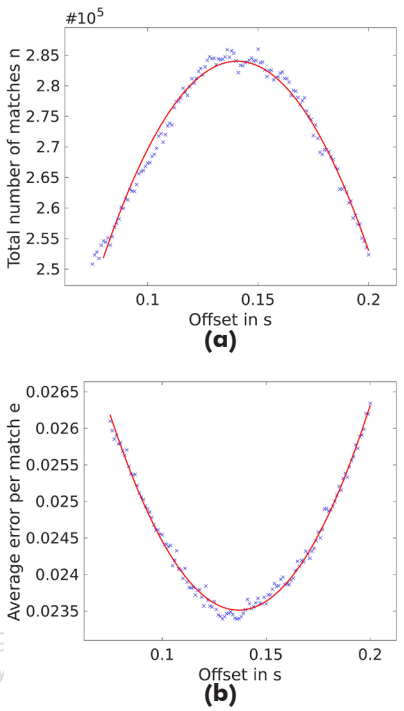

Figure 8. Measurement graphs for the timestamp offset between laser scanner and camera for the metro station dataset: (a) Total number of matches *n*, and (b) Average error per match *e*.

Laser Scanner to Motor

For all introduced datasets, we executed our approach to determine the offset between the timestamps of the laser scanner and the motor that was previously presented. The results are depicted in Table 2. We strove for the same accuracy as shown in the previous subsection, and thus used the ranges for the timestamp offset as depicted there.

The offsets presented are determined by evaluating the two criteria previously introduced, namely the total number of matches *n* and the average error per match *e* in the sweep to map registration. After examining the results, we decided to fit a second-degree polynomial to our data and determine its extremum. It can be seen that both criteria lead to similar results for all four datasets. Similar to our stationary method that determines the timestamp offset prior to data acquisition, the average offset over all four datasets amounts to 23.7 ms for both criteria.

Figure 7 displays the measurement graphs for both criteria evaluated on the parking area dataset. In Figure 7a it becomes evident that the second-degree polynomial fitted to the total number of matches *n* approaches its maximum at 23.8 ms. Similarly, in Figure 7b it can be seen that the second-degree polynomial fitted to the average error per match has its minimum at 23.7 ms.

Those two graphs are exemplary for all four datasets, and thus it can be concluded that the timestamp offset has a great effect on the quality of results produced by the presented SLAM approach. This can be observed by both the major drop of the total number of matches *e* and the large growth of the average error per match if the offset deviates by more than 3 ms from the extremum.

Laser Scanner to Camera

Similarly, we utilized our motion-based approach to determine the timestamp offset between laser scanner and camera. After examining the data points computed by our approach, we again decided to fit a second-degree polynomial to our data. The relevant extrema are depicted in Table 3. For this sensor combination we depict results with an accuracy of 1 ms since we did not see noteworthy improvements of both criteria with an accuracy of 0.1 ms.

Figure 8 displays the measurement graphs for both criteria evaluated on the metro station dataset. In Figure 8a it becomes evident that the second-degree polynomial fitted to the total number of matches *n* approaches its maximum at 140 ms. Similarly, in Figure 8b, it can be seen that the second-degree polynomial fitted to the average error per match *e* has its minimum at 137 ms. Again, these two graphs are exemplary for all four datasets, and thus show the relevance of an appropriate timestamp offset between laser scanner and camera.

Discussion

Laser Scanner to Motor

As can be seen in the previous section the calculated timestamp offset ranges from 23.4 ms to 24.6 ms for our stationary method and from 23.3 ms to 24.1 ms for our motion-based approach. In sum, both approaches yield the same average of approximately 24 ms over all experiments. Thus, it can be concluded that the calculated timestamp offsets coincide for both methods within an accuracy of 1 ms. This means that both approaches are convenient to determine the timestamp offset between laser scanner and motor. Furthermore, this shows that the results reflect reality with a high probability since both methods operate independently on different data and with different criteria while still obtaining the same results.

Moreover, it is evident, considering especially the results for our motion-based method using those large datasets, that an accuracy of 1 ms is sufficient for our purpose. As can be seen from Figure 7, the criteria of our experiments stay around the same level for offsets between 23.5 ms and 24.5 ms which makes it sufficient to choose an offset within this range for appropriate results of the SLAM algorithm. Thus, it can be stated that a timestamp offset within an accuracy of 1 ms is adequate.

Another important finding is that we get similar results for the polynomial's extrema irrespective of whether we use intervals of 0.1 ms or 1 ms to evaluate our criteria as previously described. This suggests that even if we strive for an accuracy of 0.1 ms we can reduce the number of iterations by fitting an appropriate polynomial to our data points that are gathered with an accuracy of 1 ms. We did not use a curve-fitting algorithm for our stationary approach since the number of iterations is not as important as for our motion-based approach. This is because the datasets for our stationary approach last only a few seconds, and thus an iteration does not take long.

In summary, the results indicate that both presented approaches are able to achieve the goal of determining the timestamp offset between laser scanner and motor. The decision as to which method should be used depends on the available data. If the dataset, the offset should be calculated for, is already available, the motion-based method can be used to avoid setting up the system again. However, if the timestamp offset is required for online calculations, it is inevitable to run the stationary method before starting those calculations.

To show the influence of the timestamp offset between laser scanner and motor on the final result, we depict two point clouds that are obtained using different timestamp offsets. While all other parameters remain unchanged, the timestamp offset is set to 24 ms for Figure 9 and to 19 ms for Figure 10. Both figures show point clouds that originate from the metro station dataset in top view (cf. Figure 6a).

The greatest difference is recognizable for the pillar in the center of both figures. While for Figure 9 the pillar is easily observable in the shape of a hexagon, it is not obvious for Figure 10. Likewise, the stairs that can be seen on the left and right side for both point clouds are more distinct for Figure 9. Thus, it can be stated that the point cloud in Figure 9 indicates a greater clarity. Furthermore, it becomes evident again that the timestamp offset between laser scanner and motor has a great effect on the resulting point cloud as an adjustment of merely 5 ms leads to a lower perceived clarity for our experiments.

Laser Scanner to Camera

Again, both criteria induce similar results for all four datasets, and thus are equally appropriate to use. Nevertheless, we were not able to find offsets with the same accuracy as for the laser scanner to motor synchronization which can be observed by the fact that the plateau of the polynomial in Figure 8a is

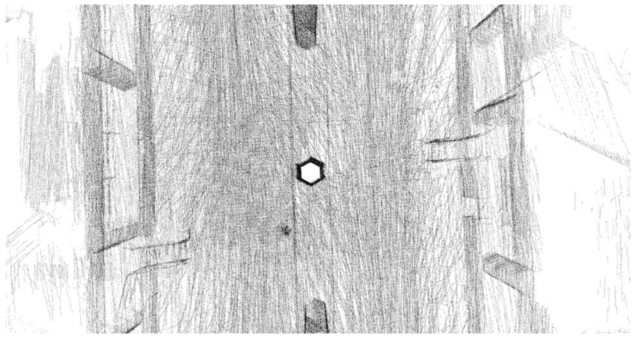

Figure 9. Map section generated by the SLAM approach for the metro station dataset using an appropriate timestamp offset between laser scanner and motor of 24 ms.

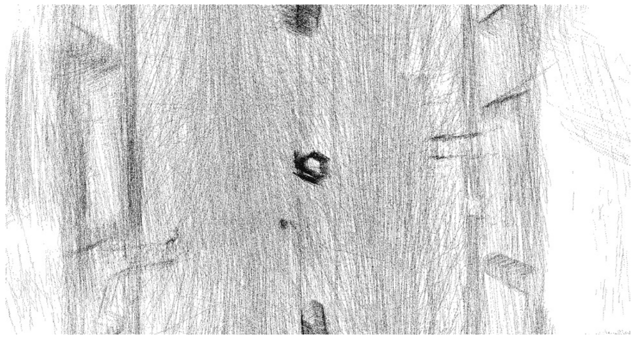

Figure 10. Map section generated by the SLAM approach for the metro station dataset using an inappropriate timestamp offset between laser scanner and motor of 19 ms.

much wider than the plateau in Figure 7a. However, since the images of the camera are only used as an initial guess for the motion estimation, these results were to be expected. Recall that the lidar odometry algorithm uses the motion estimate from the visual odometry to project the laser scan points to the beginning of the sweep. Afterwards, the lidar odometry algorithm determines the remaining drift for an entire sweep that cannot be determined using the visual odometry. This means, that the lidar odometry algorithm can compensate for small timestamp offset errors in the optimization step that is supposed to find the drift for a sweep. The laser scanner's measurements, however, are directly linked to both our criteria since an incorrect timestamp offset between laser scanner and motor leads to erroneously transformed 3D points, which in turn lead to a worse performance of the SLAM algorithm.

Again, the number of iterations needed to find the timestamp offset can be reduced by selecting larger intervals for the determination of data points and fitting a second-degree polynomial to these data points. For our datasets an interval of 10 ms between data points is sufficient to obtain similar results as for an interval of 1 ms.

The results for the timestamp offset between laser scanner and camera range from 59 ms to 140 ms, and thus show a much wider gap than our results for the laser scanner to motor synchronization. Additionally, Figure 8 shows that an offset of 59 ms (that is optimal for the cemetery dataset in terms of the average error per match) is far from the optimum for the metro station dataset and leads to 14 percent less matches for this dataset. This suggests that we cannot find an optimal timestamp offset between laser scanner and camera that is valid for all four datasets, and thus we refrain from specifying an average over those four datasets. A possible reason might be the synchronization of the computer's clocks using NTP. NTP updates the synchronization parameters between both

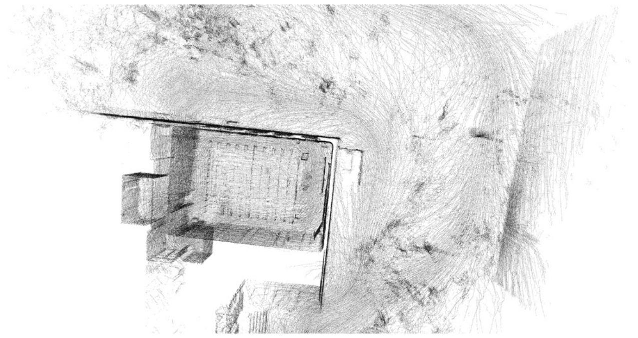

Figure 11. Map section generated by the SLAM approach for the lecture room dataset using an appropriate timestamp offset between laser scanner and camera of 100 ms.

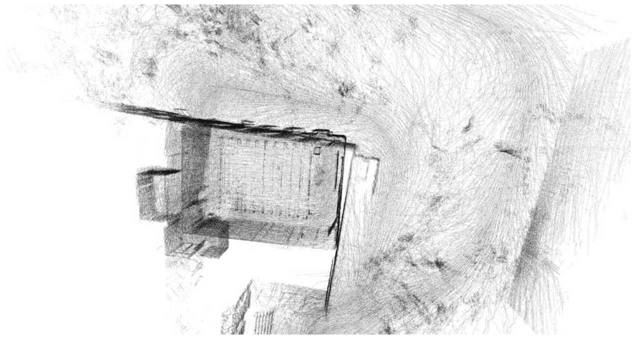

Figure 12. Map section generated by the SLAM approach for the lecture room dataset using an inappropriate timestamp offset between laser scanner and camera of 180 ms.

computers continuously, and thus needs a long lead time until the synchronization is accurate. We were not able to record all datasets in a row, but rather had to shut down both the Kontron PC and the Tegra board in between our measurements. Thus, upon restart the clocks had to be synchronized again using NTP which may have led to different timestamp offsets between both systems, and, consequently, between both sensors. We do not face this problem for the laser scanner to motor synchronization since both sensors are attached to the same computer. To obtain a constant timestamp offset between laser scanner and camera, it would be advisable to connect both sensors to a common computer with enough processing power to handle both sensor's datastreams.

Similarly as for the laser scanner to motor synchronization, we show the influence of an inappropriate timestamp offset between laser scanner and camera by comparing the same map section for two different offsets. Both figures show the lecture room in top view and some parts of its surroundings. To create those maps, we carried our multi-sensor system through the lecture room at first and then took a walk around the building. We chose an appropriate timestamp offset of 100 ms for Figure 11, and thus it can be seen that the inner and outer walls of the room align perfectly parallel. In contrast to that, the inner and outer walls in Figure 12 are not parallel since we chose an inappropriate timestamp offset of 180 ms.

However, the local clarity - which is the clarity for individual, small parts of the map (e.g., the clarity of the outer wall) - in Figure 12 is comparable to that in Figure 11. This is in contrast to the laser scanner to motor synchronization where the local clarity decreases for an inappropriate timestamp offset (cf. Figure 10). One possible explanation for these observations could be that the SLAM approach does not need an accurate initial motion estimation from the visual odometry to produce locally accurate results. For the global motion estimation, however, it needs these initial motion estimations to be correct in order to find the correct matches in the lidar odometry algorithm. In summary, it can be said that the timestamp offset between laser scanner and camera needs to be determined appropriately to generate accurate results.

Conclusions and Future Work

Incorrect synchronization for a multi-sensor system can lead to an erroneous motion estimation and distortion in the resulting point clouds when using it to solve the SLAM problem. To solve the problem of synchronization between an actuated laser scanner and its motor we presented two independent approaches to calculate the timestamp offset between these two devices. Both use different parts of a SLAM approach proposed by J. Zhang and Singh (2015) and distinct criteria to find an appropriate offset. Moreover, our motion-based approach can be further used to find the timestamp offset between a laser scanner and a camera.

Our experiments have shown that both approaches yield similar results within an accuracy of 1 ms for the laser scanner to motor synchronization. However, the experimental results also showed that an accuracy of 1 ms is sufficient. Thus, it can be stated that both methods are convenient to determine the desired offset. Furthermore, we managed to find an appropriate timestamp offset between laser scanner and camera using our motion-based approach, and thus think that the underlying idea can be used for arbitrary sensor combinations. However, it should be noted that different experimental setups might yield different accuracies for the timestamp offset if the distance to objects and the robot's movement characteristics vary from our experiments. Finally, we were able to demonstrate the negative effect an incorrect synchronization within our multi-sensor system can have on the resulting point clouds and the motion estimation.

Future work involves applying the motion-based approach to other sensor combinations that are fused in a similar SLAM approach. Additionally, an approach similar to our stationary method should be developed especially for the laser scanner to camera synchronization in order to provide a reference to results from the motion-based approach. Moreover, we plan to find out why the timestamp offset between laser scanner and camera was not constant across all datasets. Finally, we aim to make precise statements about our approach's accuracy by evaluating the slope of our fitted polynomials and examine which influence the distance to objects and the robot's movement characteristics have on the accuracy.

Acknowledgments

This work was supported by the German Research Foundation (DFG) as part of the Research Training Group i.c.sens [RTG 2159].

References

Bosse, M., and R. Zlot, 2009. Continuous 3D scan-matching with a spinning 2D laser, *Proceedings of IEEE International Conference on Robotics and Automation 2009*, May 2009, Kobe, Japan, pp. 4312–4319.

Bosse, M., R. Zlot, and P. Flick, 2012. Zebedee: Design of a Spring-Mounted 3D Range Sensor with Application to Mobile Mapping, *IEEE Transactions on Robotics*, 28(5):1104–1119.

Droeschel, D., J. Stuckler, and S. Behnke, 2014. Local multi-resolution representation for 6D motion estimation and mapping with a continuously rotating 3D laser scanner, *Proceedings of IEEE International Conference on Robotics and Automation 2014*, May 2009, Hong Kong, China, pp. 5221–5226.

Hebert, M., and E. Krotkov, 1992. 3D measurements from imaging laser radars: How good are they?, *Image and Vision Computing*, 10(3):170–178.

Mandow, A., J.L. Martí nez, A.J. Reina, and J. Morales, 2010. Fast range-independent spherical subsampling of 3D laser scanner points and data reduction performance evaluation for scene registration, *Pattern Recognition Letters*, 31(11):1239–1250.

Mills, D., 1991. Internet time synchronization: The network time protocol, *IEEE Transactions on Communications*, 39(10):1482–1493.

Morales, J., J.L. Martinez, A. Mandow, A. Pequeno-Boter, and A. Garcia-Cerezo, 2011. Design and development of a fast and precise low-cost 3D laser rangefinder, *Proceedings of IEEE International Conference on Mechatronics 2011*, April 2011, Istanbul, Turkey, pp. 621–626.

Nuechter, A., K. Lingemann, J. Hertzberg, and H. Surmann, 2007. 6D SLAM - 3D mapping outdoor environments, *Journal of Field Robotics*, 24(8-9):699–722.

Rehder, J., R. Siegwart, and P. Furgale, 2016. Spatio-temporal laser to visual/inertial calibration with applications to hand-held, large scale scanning, *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems 2014*, September 2014, Chicago, Illinois, pp. 459–465.

Rublee, E., V. Rabaud, K. Konolige, and G. Bradski, 2011. ORB: An efficient alternative to SIFT or SURF, *Proceedings of IEEE International Conference on Computer Vision 2011*, November 2011, Barcelona, Spain, pp. 2564–2571.

Sheehan, M., A. Harrison, and P. Newman, 2010. Automatic Self-Calibration of a Full Field-Of-View 3D n-Laser Scanner, *Experimental Robotics* (O. Khatib, V. Kumar, and G. Sukhatme editors), Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 165–178.

Voges, R., C.S. Wieghardt, and B. Wagner, 2017. Timestamp offset determination between an actuated laser scanner and its corresponding motor, *ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information Sciences*, IV-1/W1, pp. 99–106.

Wulf, O. and B. Wagner, 2003. Fast 3D scanning methods for laser measurement systems, *Proceedings of 14th International Conference on Control Systems and Computer Science (CSCS14)*, July 2003, Bucharest, Romania, pp. 2991–2996.

Yoshida, T., K. Irie, E. Koyanagi, and M. Tomono, 2010. *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems 2010*, October 2010, Taipei, Taiwan, pp. 1414–1420.

Zhang, J., M. Kaess, and S. Singh, 2014. Real-time depth enhanced monocular odometry, *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems 2014*, September 2014, Chicago, Illinois, pp. 4973–4980.

Zhang, J., and S. Singh, 2014. LOAM: Lidar odometry and mapping in real-time, *Proceedings of Robotics: Science and Systems Conference (RSS)*, July 2014, Berkeley, California.

Zhang, J., and S. Singh, 2015. Visual-lidar odometry and mapping: low-drift, robust, and fast, *Proceedings of IEEE International Conference on Robotics and Automation 2015*, May 2015, Seattle, Washington, pp. 2174–2181.

Zhang, Z., 2000. A flexible new technique for camera calibration, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(11):1330–1334.