

# Automatic Calibration of Spinning Actuated Lidar Internal Parameters

Hatem Alismaï\*  
National Robotics Engineering Center  
Robotics Institute  
Carnegie Mellon University  
Pittsburgh, PA 15201  
[halismaï@cs.cmu.edu](mailto:halismaï@cs.cmu.edu)

Brett Browning  
National Robotics Engineering Center  
Robotics Institute  
Carnegie Mellon University  
Pittsburgh, PA 15201  
[brettb@cs.cmu.edu](mailto:brettb@cs.cmu.edu)

## Abstract

*Actuated lidar, where a scanning lidar is combined with an actuation mechanism to scan a 3D volume rather than a single line, has been used heavily on a wide variety of field robotics applications. Common examples of actuated lidar include spinning/rolling and nodding/pitching configurations. Due to construction of actuated lidar, the center of rotation of the lidar mirror may not coincide with the center of rotation of the actuation mechanism. In order to triangulate a precise point cloud representation of the environment, the centers of rotation must be brought into alignment using a suitable calibration procedure. We refer to this problem as estimating the internal parameters of actuated lidar. In this work, we focus on spinning/rolling lidar and present a fully automated algorithm for calibration using generic scenes without the need for specialized calibration targets. The algorithm is evaluated on a range of real and synthetic data and is shown to be robust, accurate and has a large basin of convergence.*

## 1. Introduction

Lidar<sup>1</sup> has proven to be one of the most useful sensors for field robotics. It has found broad application to tasks including SLAM, obstacle avoidance, object detection/recognition, and ground surface estimation among others. This is evident in its early adoption in robotics research as a primary 3D sensor (Hebert and Krotkov, 1992; Singh and West, 1991) and its sustained use to date (Mertz et al., 2013). This is not surprising due to the accuracy of lidar measurements at long-range

as well as the ease and flexibility of its construction.

In comparison to other range sensors, lidar’s accuracy is on the order of a centimeter even at long-range (Wong et al., 2011). Construction is relatively simple, relying on a motor, gear train, and an encoder (Wulf and Wagner, 2003; Hebert and Krotkov, 1992). Moreover, the approach is flexible and allows for a wide range of customization for the application at hand. The two most common actuation schemes are spinning/rolling, and nodding/pitching, although others such as a two-axis wobbling lidar are possible. Spinners rotate the lidar about an axis parallel to the viewing direction and are most useful for tunnels or corridor-like environments (Thrun et al., 2003; Fairfield, 2009). Noddors, on the other hand, rotate the sensor around an axis in the scanning plane orthogonal to the viewing direction and are commonly used for general field robotics applications, where a higher density of points on the ground plane is desired (Stentz et al., 2007).

The operating principles of actuated lidar are simple. A single line scanning lidar, such as a SICK or a Hokuyo, is mounted on an actuation platform. By actuating the lidar’s scanning plane, we can sweep a 3D volume. Due to the mechanical construction of the sensor, the center of actuation of the motor may not coincide with the center of rotation of the lidar spinning mirror. Hence, in order to obtain precise 3D measurements, the transformation between the two centers of rotation must be estimated. Once this transformation has been estimated, a point cloud can be triangulated from a single effective scanning point. We refer to this problem as estimating the geometric *internal parameters* of the sensing assembly<sup>2</sup> as illustrated in Fig. 1. Once these parameters have been identified and esti-

\*Corresponding author <http://www.cs.cmu.edu/~halismaï>

<sup>1</sup>Also known in the literature as LADAR and single line Laser Range Finder (LRF).

<sup>2</sup>The problem is also known as estimating the parameters of the sensing assembly kinematic chain.

mated, the sensing assembly is geometrically calibrated.

This estimation of internal parameters — calibration in this work — is performed offline as a prerequisite for many robotic applications (Fairfield, 2009; Kelly et al., 2011; Stentz et al., 2007). While the calibration of the sensor is an important step in these applications, the geometric calibration problem has been largely neglected. Prior work has focused on two main problems involving lidar. One is sensor performance characterization in different operational settings (Kneip et al., 2009; Okubo et al., 2009; Alwan et al., 2005). The other is estimating the calibration with respect to rigidly mounted sensors on the robot, such as cameras (Unnikrishnan and Hebert, 2005; Alismail et al., 2012).

In recent years, multi-beam sensors (*e.g.* Velodyne) have proven popular in robotics research, especially in the context of autonomous driving (Urmson et al., 2008). This is evident with the surge in research work on calibration methods for multi-beam lidar. Methods for Velodyne calibration can be categorized based on whether they require a calibration target (Muhammad and Lacroix, 2010; Mirzaei et al., 2012; Levinson and Thrun, 2014). Targetless calibration methods are highly desirable as they greatly simplify the calibration process. Aside from a targetless calibration for Velodyne (Levinson and Thrun, 2014), Sheehan *et al.* (Sheehan et al., 2012) propose a method to calibrate three rigidly coupled lidar units. The three units are mounted on a rotating plate to achieve a 360° view of the environment. The algorithm is based on maximizing the quality (or crispness) of the resulting 3D point cloud via entropy optimization. Entropy-based methods have been shown to be robust and accurate for point cloud registration (Tsin, 2003; Jian and Vemuri, 2011), but they are sensitive to the sampling uniformity of the point clouds. Hence, entropy-based methods may not be suitable for sensors producing highly nonuniform sampling such as actuated lidar considered in this work.

Also related to our work is the recently proposed calibration method for a two-axis scanner by Dong *et al.* (Dong et al., 2013). The authors rely on forming an image of the scene via lidar intensity returns. Having obtained this image, the authors propose to model the lidar calibration problem as lens distortion estimation (under spherical projection) and solve it with Bundle Adjustment (Triggs et al., 2000). Other examples of using intensity images can be found in the work by (Pradeep et al., 2010), who jointly calibrate a nodding/tilting Hokuyo to other sensors on the robot.

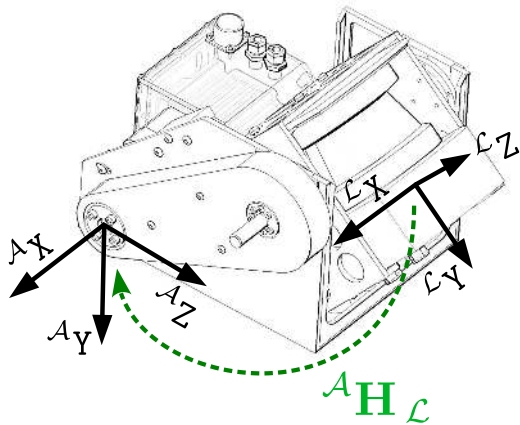
While the aforementioned sensors require an actuation step to obtain 3D point clouds, their calibration methods do not apply to actuated lidar sensors discussed in this work due to the different geometry of the

sensing mechanism. Notwithstanding, actuated lidar remains an important 3D sensor for robotic applications. For example, in contrast to a Velodyne, the field of view of actuated lidar is highly customizable to fit the application at hand. It is also cheaper and more compact for deployment on various robotic platforms.

Here, we tackle the problem of calibrating a spinning/rolling actuated lidar. A spinning lidar is constructed by rigidly mounting a laser scanner on a continuously spinning motor. Since it is not physically possible to mount the laser at the center of rotation of the motor, some mechanical offsets exist and must be estimated via a suitable calibration procedure. This calibration problem is performed offline prior to using the data for 3D perception purposes. It is normally performed once and may sometimes be repeated if evidence indicates loss of calibration. The problem amounts to estimating a (possibly constrained) rigid body transformation accounting for the offsets between the center of rotation of the lidar’s spinning mirror, and the center of rotation of the actuating motor.

In practice, calibration is often performed manually. Either by measuring the offsets on the sensor using a tape-measure, relying on CAD models, or a combination of both (Weingarten, 2006; Scherer et al., 2012). Assessment of the calibration quality and accuracy is usually performed by visual verification. Visual, or manual verification, is an important part of quality control, but it does not provide the ability to estimate a confidence measure associated with the estimated parameters. While a careful manual calibration can be done with reasonable accuracy, it is time consuming and error prone. It also does not provide the ability to verify the calibration by relying on repeatability properties of an automated algorithm. Hence, an automated algorithm is highly desirable and can have greater benefits than its manual counterparts.

In this work, we develop an algorithm to estimate the internal parameters for a spinning actuated lidar without requiring special calibration targets. Our approach is accurate and convenient. We make the weak assumption that the environment consists of (mostly) locally smooth surfaces at the scale of a local neighborhood of the input lidar point cloud. That is, over some small subset of neighboring lidar points, the surface can be well approximated by a plane, as is often implicitly assumed for point cloud registration (*e.g.* with point-plane ICP). The automation of the algorithm makes calibrating the internal parameters of the actuated lidar an easy and fast affair. The method presented here is based on the work of (Fairfield, 2009), where the author designs an automated algorithm tailored for tunnel-like environments.



**Figure 1:** Overview of the calibration problem. We seek to find a rigid body transformation  ${}^A\mathbf{H}_L$  between the lidar’s frame  $\mathcal{L}$  and the actuator’s frame  $\mathcal{A}$ . Note that the coordinate system is arbitrary and illustrated here for concreteness. Our calibration algorithm and implementation are not tied to a specific coordinate system convention.

This work is an extension of our previous work (Alismail et al., 2012), where we contribute the following enhancements: (i) increased robustness by filtering correspondences and automatic estimation of the neighborhood size for normal extraction, (ii) a method to estimate a covariance associated with the estimated parameters, and (iii) experiments on synthetic and real data assessing the quality and repeatability of the algorithm under various circumstances. We also address the applicability of the method for different lidar actuation mechanisms. Finally, we make the calibration code available for the research community. To the best of our knowledge, this is the only work in the literature that provides a generic, convenient and accurate method to recover the internal calibration parameters for a spinning actuated lidar.

In the next section, we describe the basic principles of actuated lidar for 3D perception to set the stage for our automated calibration approach.

## 2. Actuated Lidar for 3D Perception

Laser range finders, discussed in this work, operate by the time-of-flight principle. A narrow laser beam of an appropriate wavelength is shot through a spinning mirror. The spinning mirror transforms the 1D beam of light into a 2D scanning plane. Laser beams forming the scanning plane hit objects in the scene and return to the sensor. Given the time it takes for each beam to return, combined with the known mirror angle and the speed of light in the medium, we can obtain range

measurements relative to the center of rotation of the spinning mirror. Not all beams return to the lidar. Some never hit a reflective object in the scene, while others might return multiple times.

Depending on the lidar and the application at hand, multiple returns could be a useful phenomenon (Renslow et al., 2000). In other applications, however, multiple returns could obstruct important information and must be filtered out (Meng et al., 2010). In this work, multiple returns are not a cause for concern. We only work with valid returns (the ones that do return), and deal with multiple-returns by simply using the first. Thus, we assume calibration in an environment without significant dust or precipitation, which we find to be reasonable in practice. Extending to use robust cost functions to be able to reject reasonable multi-echo responses (up to some limit) would be relatively straightforward but likely is not useful in practice.

A simple solution to obtaining 3D scans is to actuate the lidar about an axis nonparallel to its scanning mirror such that we sweep the scanning plane across a volume in space. For this approach to work, we need to know two pieces of information. One is the pose of the actuator. This is typically a rotation about some known axis with a known measured angle, and is readily available by reading off the encoder value attached to the motor. The other is the offset between the center of rotation of the lidar’s mirror and that of the actuation motor. This is what we refer to as the calibration of *internal parameters*; the estimation of which is the subject of this work.

Estimating the internal parameters is often performed manually, either by manual measurements or using a CAD model for guidance. When carefully performed, manual calibration can provide acceptable results. Nonetheless, it is laborious, error prone and not repeatable. This is notably true when the CAD models for the sensor may not be available or when manufacturing tolerances do not adhere to the required accuracy. Hence, it is necessary to be able to estimate these parameters using an accurate, robust and repeatable algorithm with no user intervention.

### 2.1. Coordinate Conventions

For concreteness and clarity of presentation, we will adopt the common camera coordinate system conventions used in Computer Vision. The right-handed coordinate system is shown in Fig. 1, where the Z-axis points forward along the viewing ray, the X-axis points to the right, and the Y-axis points downward. Our algorithm and implementation do not require a specific coordinate system.

## 2.2. Calibration Parameters for a Spinning Lidar

Actuated lidar calibration amounts to estimating a rigid body transform between the instantaneous center of rotation of the actuator ( ${}^{\mathcal{A}}ICR$ ) and that of the lidar’s spinning mirror ( ${}^{\mathcal{L}}ICR$ ), where  $\mathcal{A}$  and  $\mathcal{L}$  denote the actuator and lidar frame respectively.

Consider a *spinning* lidar assembly with the coordinate system shown in Fig. 1. A 2D point in the lidar frame is obtained by converting a range measurement  $\rho$  from polar to Cartesian coordinates using the current position of the lidar rotating mirror  $\theta$ . This in-plane point in homogeneous coordinates is given by

$$\mathcal{L}\mathbf{x}(\theta_i, \rho_i) = \begin{pmatrix} x_i \\ 0 \\ z_i \\ 1 \end{pmatrix} = \begin{pmatrix} \cos \theta_i \\ 0 \\ \sin \theta_i \\ 1 \end{pmatrix} \rho_i. \quad (1)$$

In order to obtain a triangulated point in space we apply the actuator’s rotation associated with the in-plane point. In our example, this is a rotation about the Z-axis. However, for the triangulation to be performed correctly, the in-plane point must be transformed from the lidar’s frame to the actuator’s frame via a suitable calibration. Hence, the point in the actuator frame is given by

$${}^{\mathcal{A}}\mathbf{x}_{ij} = {}^{\mathcal{A}}\mathbf{R}^z(\phi_j) {}^{\mathcal{A}}\mathbf{H}_{\mathcal{L}} \mathcal{L}\mathbf{x}(\theta_i, \rho_i), \quad (2)$$

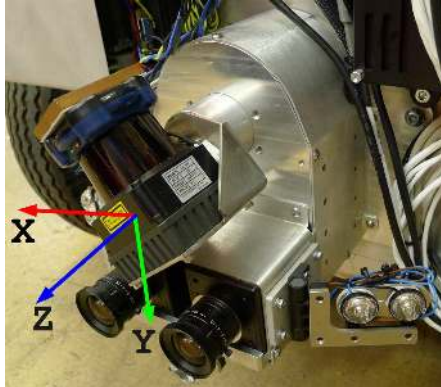
where  ${}^{\mathcal{F}}\mathbf{R}^a(\phi)$  denotes a rotation matrix with angle  $\phi$  about the unit axis  $\mathbf{a}$  in coordinate frame  $\mathcal{F}$ ,

$$\mathbf{R}^z(\phi) = \begin{pmatrix} \cos \phi & -\sin \phi & 0 & 0 \\ \sin \phi & \cos \phi & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}. \quad (3)$$

Finally, the calibration transform is the  $4 \times 4$  matrix  ${}^{\mathcal{A}}\mathbf{H}_{\mathcal{L}}$  that maps points from the lidar’s frame  $\mathcal{L}$  to the actuator’s frame  $\mathcal{A}$ . This is the unknown quantity we seek to estimate.

## 2.3. Degrees of Freedom (DOF)

In the case of an actuated lidar, the calibration is not a general 6DOF rigid-body transform due to the restricted motion pattern of the mechanism. In our example of a spinning lidar, the model is at most 5DOF corresponding to three rotational degrees of freedom and two translations. The translations are along the X-axis (left-right sliding) and the Y-axis (up-down). Translation along the rotation-axis, the Z-axis, is unobservable. We may simplify the calibration model further if the scanner’s pointing direction is aligned with the motor’s spinning axis. In this case, a rotation about the Z-axis is under-constrained. Table 1 summarizes the degrees of freedom for commonly used actuated lidar sensors.



**Figure 2:** Hokuyo scanner mounted on a spinning motor. The motor spins the lidar about the Z-axis (rolling).

## 3. Algorithm

The algorithm exploits the periodicity of the actuation mechanism to partition a stream of data from a stationary sensor into spatially overlapping sets. We assume that a measurement of the same surface patch in each set is performed with *different* joint angles (the actuation angles and the lidar mirror angle). This assumption works for spinning lidars and some two-axis systems. It will not in general work for nodding systems, which are unable to measure the same spot from two different joint angles, and is therefore a limitation to the approach that is discussed later.

Consider a spinning lidar, where the lidar rotates about its viewing axis (*c.f.* Fig. 2). Without loss of generality, let the first scanned point be at motor position 0.0 rad and let a *full-scan* be the one with motor position at  $2\pi$  rad. The periodicity of such actuation mechanism causes a symmetry for every *half-scan*. That is, we can split the full scan into two scans such that both sample the same surfaces.<sup>3</sup> Namely, we have two point sets as a function of the motor angles

$${}^{\mathcal{A}}\mathcal{X}(\phi_j) = \{ {}^{\mathcal{A}}\mathbf{x}_j = {}^{\mathcal{A}}\mathbf{R}^z(\phi_j) {}^{\mathcal{A}}\mathbf{H}_{\mathcal{L}} \mathcal{L}\mathbf{x} \mid \phi_j \leq \pi \}; \quad (4)$$

$${}^{\mathcal{A}}\mathcal{X}'(\phi_k) = \{ {}^{\mathcal{A}}\mathbf{x}'_k = {}^{\mathcal{A}}\mathbf{R}^z(\phi_k) {}^{\mathcal{A}}\mathbf{H}_{\mathcal{L}} \mathcal{L}\mathbf{x} \mid \phi_k > \pi \}. \quad (5)$$

When the sensor is stationary, these two point sets sample the same surfaces in space but the joint angles associated with the measurements will be unique for the same physical spot. Hence, the calibrating transform is such that both point sets are aligned,

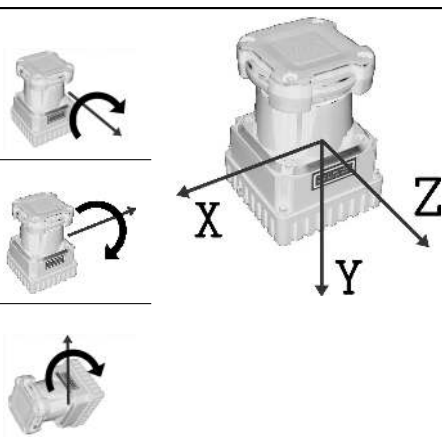
$${}^{\mathcal{A}}\mathbf{H}_{\mathcal{L}}^* = \underset{{}^{\mathcal{A}}\mathbf{H}_{\mathcal{L}}}{\operatorname{argmin}} \{ f({}^{\mathcal{A}}\mathcal{X}, {}^{\mathcal{A}}\mathcal{X}'; {}^{\mathcal{A}}\mathbf{H}_{\mathcal{L}}) \}, \quad (6)$$

<sup>3</sup>A video illustration is available online at <http://youtu.be/11JSPfZs30A>.



**Table 1:** Overview of common actuated lidar sensors and calibration degrees of freedom (DOF). Quantities in brackets denote under-constrained parameters.

Mechanism	Degrees of Freedom
Spinning	$r_x, r_y, [r_z], t_x, t_y$
Nodding	$[r_x], r_y, r_z, t_y, t_z$
Yawing	$r_x, [r_y], r_z, t_x, t_z$



where  $f$  is a function of the dissimilarity between the two point sets. This transform does not depend on the specific details of the actuation mechanism, such as the acceleration rate of the motor, or the type of motion, provided that we can split the full scan into two halves.

In practice, while the two half-scans sample the same surfaces, there are no strict point-point correspondences. This is due to the continuous nature of the actuation motion and sampling artifacts. Here, we use a point-plane distance as the dissimilarity function (Chen and Medioni, 1992). Our optimization objective, therefore, takes the following form:

$$f(\mathcal{X}, \mathcal{X}', \mathbf{H}) = \sum_{\mathbf{x}_i \in \mathcal{X}} w_i \|\mathbf{n}_i^\top (\mathbf{x}_i - \mathbf{H}\mathbf{x}'_{\varphi(i)})\|^2, \quad (7)$$

where  $\mathbf{n}_i = (n_i^x, n_i^y, n_i^z, 0)^\top$  is a unit normal at point  $\mathbf{x}_i$  from the first half-scan,  $w_i \in [0, 1]$  is a weight indicating our confidence of the estimated normal/correspondences,  $\mathbf{x}'_{\varphi(i)}$  is the corresponding point to  $\mathbf{x}_i$  from the other half-scan, where  $\varphi(i) : |\mathcal{X}| \rightarrow |\mathcal{X}'|$  is a function that returns the index of the corresponding point to  $\mathbf{x}_i$  from the set  $\mathcal{X}'$ . This correspondence function is based on the closest point as commonly performed in registration problems (Fitzgibbon, 2003). To increase efficiency and robustness, we choose to use bijective correspondences with ties broken based on distance. In the expression above, we dropped the explicit specification of the coordinate frame for clarity.

This nonlinear weighted least squares optimization problem can be solved efficiently with standard methods given a reasonable starting point. In the following we discuss the details of our implementation.

### 3.1. Rotation Parametrization

Rotation estimation and parameterization is a well-studied problem in Robotics and Estimation theory. The main complication is the nonlinearity of the configuration space, the special orthogonal Lie matrix group  $SO(3)$ . An excellent discussion of the various representations and associated distance metrics on the manifold can be found in the works of (Kuffner, 2004; Hartley et al., 2013).

In this work, since rotational offsets are expected to be small and we would like to constrain the estimated parameters, we opt to use the representation with the most intuitive physical explanation, Euler angles. Euler angles allow us to easily constrain the optimization in a straightforward manner. They are also relatively easy for derivative computations (Kelly, 1994). Extending our approach to other rotation representations is straightforward.

Euler angles, however, are susceptible to the well-known singularity: Gimbal lock (when two of the gimballs lie on the same plane). Nonetheless, we are almost guaranteed to avoid this singularity because the orientation offsets are small for this type of calibration problem. If the orientation misalignment is large, a rough initialization may be required. Alternatively, one could use more robust rotational representations.<sup>4</sup>

### 3.2. Optimization

The cost function (Eq. (7)) is a nonlinear function of the parameters. While it is possible to linearize the rotations explicitly, we choose to directly solve for

<sup>4</sup>We developed an implementation using quaternions but saw no empirical benefits and thus do not discuss it further here.

the parameters using standard nonlinear least squares methods such as Levenberg-Marquardt (LM) (Levenberg, 1944; Marquardt, 1963). For most cases, the offsets are small and an initialization with the identity ( ${}^A\mathbf{H}_{\mathcal{L}} = \mathbf{I}_4$ ) is close to the local minima. Hence, the user is alleviated from any manual measurements to initialize the optimization.

In addition to obtaining a solution for the estimated parameters, we would like to estimate the covariance associated with these parameters. Let the vector of parameters be  $\boldsymbol{\theta}$ , then the Jacobian is the matrix of partial derivatives of the error function  $\mathbf{e}_i$  with respect to  $\boldsymbol{\theta}$  and is given by

$$\mathbf{J}_i := -\frac{\partial \mathbf{e}_i(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}} \quad (8)$$

$$\mathbf{J} := (\mathbf{J}_i \ \cdots \ \mathbf{J}_n)^\top, \quad (9)$$

where  $\mathbf{e}(\boldsymbol{\theta})$  is the error function defined to be the difference between the measurement vector  $\mathbf{z}$  and the predicted parameters  $\mathbf{f}(\boldsymbol{\theta})$ ,

$$\mathbf{e}(\boldsymbol{\theta}) := \mathbf{z} \ominus \mathbf{f}(\boldsymbol{\theta}), \quad (10)$$

where  $\ominus$  indicates the difference in vector space. Let the covariance of each measurement be  $\boldsymbol{\Sigma}_i$ , and let  $\boldsymbol{\Sigma}$  be a block-diagonal concatenation of the measurement covariance matrices, *i.e.*:

$$\boldsymbol{\Sigma} = \begin{pmatrix} \boldsymbol{\Sigma}_i & & \\ & \ddots & \\ & & \boldsymbol{\Sigma}_n \end{pmatrix}. \quad (11)$$

Then, upon convergence of LM we arrive at the Fisher information matrix:

$$\mathbf{F} := \mathbf{J}^\top \boldsymbol{\Sigma}^{-1} \mathbf{J}. \quad (12)$$

To obtain a covariance associated with the estimated parameters, we simply invert the information matrix. This is the Cramér-Rao lower bound (van de Geer, 2005; Haralick, 2000).

The estimate of the covariance is rather crude and may overestimate the true covariance (Censi, 2007). Nonetheless, it is useful to assess the accuracy and the reliability of the solution by analyzing the shape of the local extrema as will be demonstrated in Section 4.

### 3.3. Normal Extraction

Our optimization function (Eq. (7)) depends on the quality of the estimated normals. A common approach to determining a local tangent plane normal on unorganized point sets is based on the Eigenvalue decomposition of the covariance matrix in a specified neighborhood. Let  $\boldsymbol{\Sigma}(\mathbf{x}; r)$  be the (weighted) covariance matrix

estimated in a neighborhood of radius  $r$ ,

$$\boldsymbol{\Sigma}(\mathbf{x}; r) = \sum_{\mathbf{x}_i \in \mathbf{N}(\mathbf{x}; r)} w(\mathbf{x}, \mathbf{x}_i; r) (\mathbf{x}_i - \boldsymbol{\mu})(\mathbf{x}_i - \boldsymbol{\mu})^\top, \quad (13)$$

where  $\mathbf{N}(\mathbf{x}; r)$  denotes the neighborhood of radius  $r$  for the point  $\mathbf{x}$ ,  $\boldsymbol{\mu}$  is a weighted centroid estimated within the same neighborhood

$$\boldsymbol{\mu} = \sum_{\mathbf{x}_i \in \mathbf{N}(\mathbf{x}; r)} w(\mathbf{x}, \mathbf{x}_i; r) \mathbf{x}_i, \quad (14)$$

and  $w : \mathbb{R} \rightarrow [0, 1]$  is a monotonically decreasing function of distance such as the Gaussian, or the squared exponential:

$$w(\mathbf{x}_i, \mathbf{x}; r) = \frac{1}{\xi} \exp\left(-\frac{\|\mathbf{x}_i - \mathbf{x}\|^2}{r^2}\right). \quad (15)$$

This weighting function is normalized with:

$$\xi = \sum_{\mathbf{x}_j \in \mathbf{N}(\mathbf{x}; r)} \exp\left(-\frac{\|\mathbf{x}_j - \mathbf{x}\|^2}{r^2}\right). \quad (16)$$

The tangent plane normal can be then estimated as the Eigenvector corresponding to the minimum Eigenvalue of the covariance matrix  $\boldsymbol{\Sigma}(\mathbf{x}; r)$ . Such method to compute surface normals is common in Computer Graphics and surface reconstruction (Amenta and Kil, 2004). Further, an estimate of the goodness of planarity can be obtained by analyzing the Eigenvalues (Bosse and Zlot, 2009; Bosse et al., 2012). Let the three Eigenvalues of  $\boldsymbol{\Sigma}(\mathbf{x}; r)$  be  $\lambda_3 \geq \lambda_2 \geq \lambda_1$ , then

$$p_c = \frac{2(\lambda_2 - \lambda_1)}{\text{trace}(\boldsymbol{\Sigma}(\mathbf{x}; r))} \in [0, 1] \quad (17)$$

provides a confidence estimate of the degree of planarity, where  $p_c \rightarrow 1$  indicates higher confidence in the estimated normal.

### 3.4. Adaptive Neighborhood Size

Quality and accuracy of normal estimation depends on the extent of the neighborhood used to compute the covariance (Eq. (13)). Too large a radius increases the chance of including points that do not belong to the local plane; consequently smoothing out the estimating normals. In contrast, too small a radius increases the likelihood of capturing the noise in the data rather than surface details; therefore roughening the estimated normals.

There exists specialized methods for estimating the scale for normal extraction, or other surface features. However, such methods might have high computational

requirements. For example, the algorithm described in (Unnikrishnan et al., 2010), while producing excellent results, requires a large amount of memory for the computation of geodesic distances.

In this work, we choose a simple method for scale selection. We select the neighborhood size based on the largest distance to the  $k^{\text{th}}$  neighbor. The intuition is that distance to the  $k^{\text{th}}$  neighbor is an indication of the sampling density. Normal vectors estimated at points with higher sampling densities will be given a smaller neighborhood size. These points are usually close to the lidar where more details can be detected. At longer range, or oblique scanning angles, the sampling density is lower. Hence, a larger radius is assigned, thereby increasing the robustness of normal estimation.

The implementation of this strategy is straight forward. For every 3D point we need two nearest neighbors computations: the first is to select the scale/search radius, and the second is to select points within the radius. Nearest neighbor search is performed via a KD-tree (Muja and Lowe, 2014)<sup>5</sup> to keep the computation efficient.

This strategy was found to produce normals with better quality than using a fixed radius or a fixed number of neighbors in qualitative comparisons outweighing the cost of an additional KD-tree lookup per point. In this work, we found  $k = 50$  to be a good value.

### 3.5. Summary

A summary of the algorithm is shown in Alg. 1. Starting from an initialization  $\mathbf{H}^{(0)}$ , we compute weighted normals and point correspondences based on the closest distance. These plane normals and correspondences are held fixed in the inner loop to produce a calibration  $\mathbf{H}^{(k+1)}$ . This process continues until convergence. The process has to be repeated in this fashion because the shape of the point cloud depends on the value of the calibration. Each estimate of the calibration will produce a different point cloud and consequently, different normals. Convergence is determined if a maximum number of iterations is reached or when the change in the estimated parameters is too small.

## 4. Experiments & Results

### 4.1. Synthetic Data

We use synthetic data to obtain a quantitative assessment of the algorithm’s accuracy as there is no readily available and reliable method for obtaining ground truth on a real sensor. Synthetic data also allows us to vary the actuation mechanism and noise magnitude. For

ease of visualization and analysis, we use a cuboid as the simulation environment (shown in Fig. 3). The cube’s side length is 10 meters with the sensor assembly located at its center.

#### 4.1.1 Synthetic spinner

A spinner is an actuated lidar with a rolling motion about the pointing direction of the lidar. Fig. 3 shows the result of our calibration in a simple noise-free scenario, in which case the algorithm estimates error-free calibration parameters (up to floating point precision). It is instructive to look at the resulting scanning pattern of the calibrated sensor (right column of Fig. 3). Note the gap centered in the middle of the plane. This is the plane orthogonal to the spinning axis and facing directly in front of the lidar. As expected, this is the *blind spot* of the spinning assembly. The blind spot is inevitable and is caused by offsets between the center of rotation of the lidar’s mirror and the center of rotation of the actuator (this blind spot can also be observed in real data as shown in Fig. 9d). The extent of this blind spot is proportional to the mechanical offsets between the center of rotation of the lidar’s mirror and the center of rotation of the actuator’s. In contrast, assuming that the centers of rotation align (left column of Fig. 3), a blind spot is lacking and the geometry is clearly incorrect.

We evaluate the algorithm’s accuracy using the synthetic data (shown in Fig. 3) by performing the calibration 50 times with different noise levels  $\sigma_r$ . The data is generated from a full revolution of the lidar with motor resolution of  $1.618^\circ$  yielding approximately 120 000 points per half-scan. Zero-mean Gaussian noise is added to the returns where the noise uncertainty is varied as a function of the angle of incidence on the surface and the dependence is calibrated from Velodyne data as in (Browning et al., 2012). We also vary the ground truth calibration parameters for every run according to a normal distribution with  $t_x$  and  $t_y$  translational offsets distributed with a mean of 5 cm and a standard deviation of 1.618 cm. The synthetic calibration values are large in comparison to real systems. In contrast to our real sensor, the values are more than five times along the X-axis (left-right sliding) and more than two times along the Y-axis (up-down sliding). Results are summarized in Fig. 4. The results show that the algorithm is accurate and repeatable under various magnitudes of noise. In fact, the largest translation error across all experiments is 0.78 mm, while the largest rotation error is  $0.03^\circ$ .

Fig. 6 shows an example of the input point cloud to the calibration algorithm to illustrate the magnitude

<sup>5</sup>We use nanoflann implementation <https://code.google.com/p/nanoflann/>

---

**Algorithm 1** Overview of the automated calibration algorithm
 

---

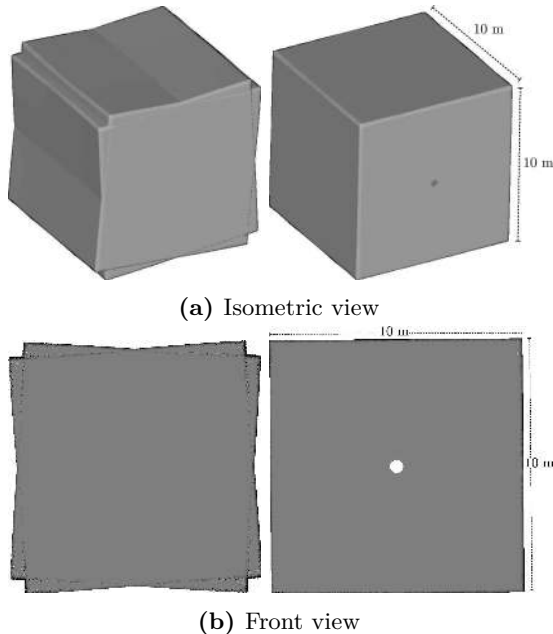
```

1: function CALIBRATION( $\mathcal{X}, \mathcal{X}'; \mathbf{H}^{(0)}$ )
2:   repeat
3:      $\{\mathcal{X}, \mathcal{X}'\} = \text{ApplyCalibration}(\mathcal{X}, \mathcal{X}', \mathbf{H}^{(k)})$ 
4:      $\{\mathbf{n}_i, w_i\} = \text{ComputeWeightedNormals}(\mathcal{X}; \mathbf{H}^{(k)})$ 
5:      $\{\mathbf{x}_i, \mathbf{x}'_i\} = \text{FindNeighbors}(\mathcal{X}, \mathcal{X}'; \mathbf{H}^{(k)})$ 
6:      $\mathbf{H}^{(k+1)} \leftarrow \underset{\mathbf{H}^{(k)}}{\text{argmin}} \left\{ \sum_i w_i \|\mathbf{n}_i^\top (\mathbf{x}_i - \mathbf{H}^{(k)} \mathbf{x}'_i)\|^2 \right\}$ 
7:   until convergence
8: end function
  
```

▷ Shape of point cloud depends on current calibration

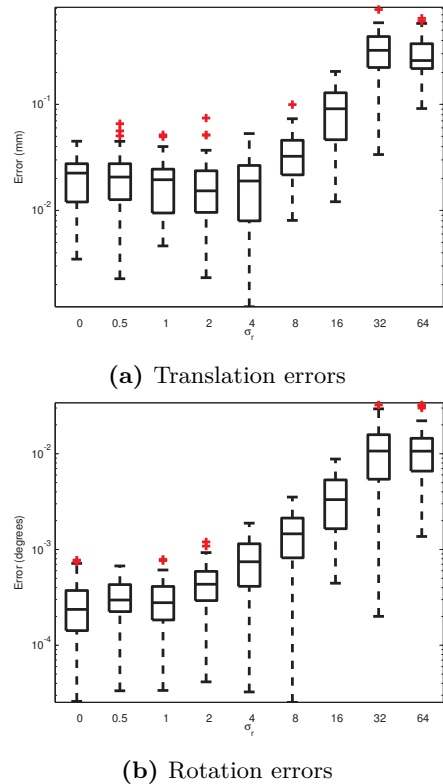
▷ Optimize for  $\mathbf{H}$  with fixed normals and neighbors

---



**Figure 3:** Synthetic data with a spinning sensor. The left column shows the point cloud prior to calibration. Notice the large distortions, which what otherwise would be straight planes are now bent. The calibration recovers the exact parameters (right). The calibration offset used for this figure is 50 mm along the X- and Y-axes. To accentuate the geometry, points are rendered with the estimated normals.

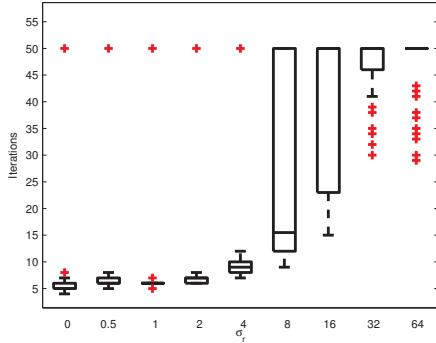
of noise. This is shown for two values of  $\sigma_r$ : 4 mm and 64 mm. Typical noise values for lidar are on the order of 10 mm to 15 mm at most, which is under our maximum noise test level of 64 mm (note this is the standard deviation, not the variance). These large values were included to stress-test the algorithm’s performance. The algorithm is accurate and can handle a large amount of Gaussian noise, even with a standard deviation close to 7 cm. The consistency of the results can be also viewed in terms of the number of (outer) iterations needed for convergence. Fig. 5 shows this number as a function noise. The maximum number of iterations was capped at 50. For small to moderate levels of noise the



**Figure 4:** Synthetic spinner results on the cube environment with various levels of noise (shown in millimeters) and different calibration parameters. The error is shown in *log* scale on a standard box plot. The median translation error, across all levels of noise, is 0.023 mm and the maximum is 0.78 mm. Median rotation error is  $6.5 \times 10^{-4}$  degrees with a maximum of  $0.03^\circ$ .

algorithm reaches a solution quickly. For small noise magnitudes, we expect the algorithm to converge in less than 10 iterations. Otherwise, some error might have happened and further investigation might be in order. Yet, an usually large number of iterations is not necessarily an indication of a bad quality calibration as shown in Fig. 4. In real scenarios, however, we expect to observe less variability in the number of iterations





**Figure 5:** Number of outer iterations of the algorithm as a function of noise levels. The number of iterations here correspond to the results shown in Fig. 3. The maximum number of iterations allowed was capped to 50. Noise,  $\sigma_r$  is show in millimeters.

as noise in real data is less than our simulation.

The number of inner iterations needed for the optimizer to reach a minima is consistently less than 10 despite the magnitude of noise. A possible conclusion of this behavior is that for a given point configuration and an appropriate calibration scene, the optimization problem is well constrained with a sharp extrema.

#### 4.1.2 Basin of convergence

We use the simulated cube data to study the basin of convergence of the algorithm. This is achieved by a generating a synthetic calibration with the most common calibration parameters. We simulate translations along the X- and Y-axes over a grid of values  $\{(t_x, t_y)\} = \{(0.5, 25.0) \times (0.5, 25.0)\}$  centimeters and a resolution of 0.5 cm. All experiments were initialized with the identity. Results are shown in Fig. 8. The algorithm is able to recover the correct parameters up to a translation vector of  $\approx 20$  cm on each of the X- and Y-axes. Beyond 20 cm, the shape of the point cloud becomes severely distorted, *e.g.* see Fig. 22. Nonetheless, 20 cm on both axes is a very large calibration offset in comparison to values found on real actuated lidar sensors. It is significantly less than typical manufacturing errors commonly observed in real sensors.

## 4.2. Real Data: Hokuyo Spinner

For real data from a spinner, we use a Hokuyo UTM-30LX-EW laser scanner mounted on a spinning motor (*c.f.* Fig. 2). The Hokuyo is configured to have 270° field of view, with angular resolution of  $1/4$  of a degree, producing 1081 measurements per scanline.

Calibrating a spinning lidar is challenging as the motor’s Home position is arbitrary. Every spinning sensor

potentially has a different homing position. Hence, it is not an easy task to integrate scene constraints into the optimization. For example, it would be difficult to include a ground plane constraint such as the one used by (Underwood et al., 2010) into the calibration framework without incorporating additional information or assumptions about the scene.

In the following, we perform various experiments for real-life situations that could impact calibration results using data from the Hokuyo Spinner.

## 4.3. Effect of Missing & Sparse Data

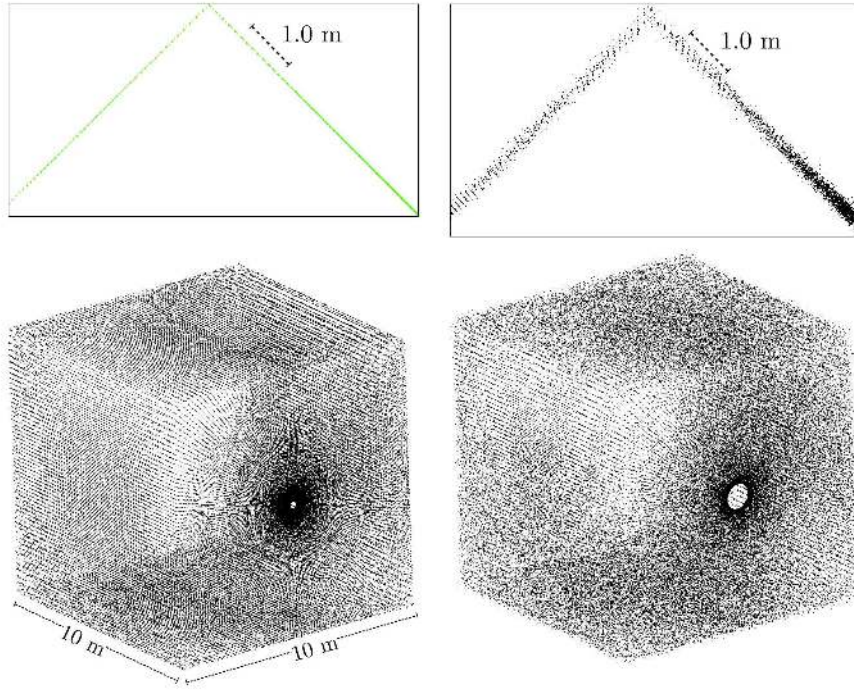
Here, we address the question of missing correspondences and show that the distance to tangent plane metric is not severely affected by lack of precise point correspondences. In Fig. 9 we show our calibration results using an indoor office environment. We colorize the points with reflectance values from the Hokuyo for better visualization. The automated calibration results show a big improvement as can be seen by the geometry (orthogonality of planes constituting corners) and the sharpness of the visualized reflectance values.

Using the same office data (shown in Fig. 9), we evaluate the algorithm’s repeatability and robustness in face of missing point correspondences. We randomly remove a percentage of points from the original dataset, run the calibration algorithm and report the variance of the estimated parameters. For every percentage of removed points we repeat this process 50 times, each with a different random subset of points removed. Results are summarized in Table 2, which indicate very small variations in estimated parameters due to missing correspondences.

## 4.4. Effect of Actuation Speed

Actuation speed of the motor has a major influence on the sampling density. A lower actuation speed results in a higher sampling density, and consequently more accurate normals. Nonetheless, since the calibration parameters do not depend on the speed of actuation we desire an algorithm that is independent of the actuation speed as well. This is convenient as data collection for calibration purposes need not to have stringent requirements.

Table 4 summarizes the results for various actuation speeds for approximately the same number of revolutions. Data is collected from our Hokuyo Spinner. Results are consistent given enough points can be associated between the two half scans and normals are extracted reliably. At 30 RPM, however, the sampling is sparse, and we do not have enough overlapping segments. Normal estimation is also inaccurate as many surfaces are sampled as lines due to high speed of ac-



**Figure 6:** Illustration of the magnitude of noise used in the simulation experiments. The left column shows the cube environment with  $\sigma_r = 4$  mm, and on the right with  $\sigma_r = 64$  mm. The top row shows a closeup slice of top left corner of the box for the different noise levels. In Fig. 7 both point sets are overlaid for better visualization. The size of the blind spot differs between the two views because of the random ground truth calibration.

**Table 2:** Summary statistics for a 4DOF calibration on the same data with varying percentages of randomly deleted points. The initial point set size is  $\approx 400\,000$  points. Mean ( $\mu$ ) and standard deviation ( $\sigma$ ) of each parameter is reported,  $r_x$  and  $r_y$  are rotation angles about the X- and Y-axes in **degrees**,  $t_x$  and  $t_y$  are the translation parameters in **millimeters**. Variations in estimates due to missing point-point correspondences are practically negligible.

		Parameter Statistics							
		$\mu_{r_x}$	$\sigma_{r_x}$	$\mu_{r_y}$	$\sigma_{r_y}$	$\mu_{t_x}$	$\sigma_{t_x}$	$\mu_{t_y}$	$\sigma_{t_y}$
% removed points	10%	0.425	$0.76 \times 10^{-3}$	0.834	$0.78 \times 10^{-3}$	0.519	0.019	-26.07	0.035
	20%	0.424	0.001	0.837	0.001	0.505	0.027	-26.14	0.052
	30%	0.423	0.001	0.840	0.001	0.452	0.028	-26.15	0.061
	40%	0.425	0.002	0.841	0.002	0.431	0.046	-26.21	0.070
	50%	0.427	0.003	0.842	0.002	0.435	0.045	-26.27	0.090
	60%	0.429	0.003	0.842	0.002	0.432	0.059	-26.37	0.109
	70%	0.430	0.004	0.843	0.003	0.442	0.075	-26.42	0.142
	80%	0.430	0.006	0.843	0.004	0.453	0.097	-26.38	0.181
Average		0.427	0.003	0.840	0.002	0.459	0.050	-26.248	0.092

tuation. Notably, point-plane optimization combined with the scene structure is able to recover the rotation estimates consistently across the different sampling rates. This is because a small number of normals is enough to constrain the orientation. Nonetheless, for

an accurate estimate of translation we require better correspondences, which were not possible at high RPM.

**Table 3:** Summary statistic for a 4DOF calibration on the same data with varying number of randomly selected points. Angles are reported in degrees, and translations are reported in millimeters. Results are consistent until the number of points drops below 12000.

		Parameter Statistics							
		$\mu_{r_x}$	$\sigma_{r_x}$	$\mu_{r_y}$	$\sigma_{r_y}$	$\mu_{t_x}$	$\sigma_{t_x}$	$\mu_{t_y}$	$\sigma_{t_y}$
# points $\times$ 1000	200	0.434	0.003	0.853	0.002	0.667	0.057	-26.733	0.110
	100	0.438	0.006	0.854	0.005	0.688	0.101	-26.789	0.206
	50	0.437	0.012	0.855	0.009	0.693	0.216	-26.758	0.374
	25	0.434	0.026	0.854	0.019	0.615	0.373	-26.525	0.707
	12	0.412	0.044	0.856	0.033	0.582	0.641	-26.524	1.257
	6	0.380	0.111	0.847	0.081	0.295	1.426	-26.651	2.482
	3	0.267	0.249	0.834	0.172	0.164	2.963	-26.031	4.671
	1	0.200	0.548	0.750	0.450	-1.316	8.307	-26.016	12.209

**Table 4:** Calibration results for the same sensor and the same environment with various actuation speeds. Actuation speed is measured as revolutions per minute (RPM). Rotations are reported in *degrees*, and translations are in *millimeters*. Estimation of rotation parameters is consistent across various speeds (different data densities). However, estimation of the translation parameters becomes unstable at the highest speed. This can be explained by the low density cloud leading to poor point-plane correspondence as show in Fig. 10. The last column shows the determinant of the estimated covariance matrix associated with the parameters, which indicates a degradation of calibration certainty with reduced sampling rate.

Calibration parameters					
RPM	$r_x$	$r_y$	$t_x$	$t_y$	$\det(\Sigma_{\theta})$
5	0.260	0.786	1.547	-29.941	$3.161 \times 10^{-17}$
10	0.253	0.874	1.421	-29.310	$3.253 \times 10^{-15}$
15	0.381	0.762	-3.462	-32.601	$6.890 \times 10^{-14}$
20	0.442	0.873	5.202	-35.326	$8.701 \times 10^{-13}$
25	0.383	0.764	-5.960	-17.527	$3.862 \times 10^{-12}$
30	0.445	0.858	1.730	-15.001	$1.928 \times 10^{-11}$

#### 4.5. Effect of Field of View (FOV)

The field of view (FOV) of the lidar determines the how much of the scene the lidar can observe. In some situations, it is not possible to use the full FOV for the lidar due to its placement on the sensor. For example, if the lidar was mounted on the grill of an autonomous car, then the maximum FOV would be 180°.

Here, we experiment with different FOV from the Hokuyo sensor. We use the same dataset for calibration, but with different lidar FOV. The different FOV’s are obtained by post-processing the data. The data used in this experiment are shown in Fig. 11.

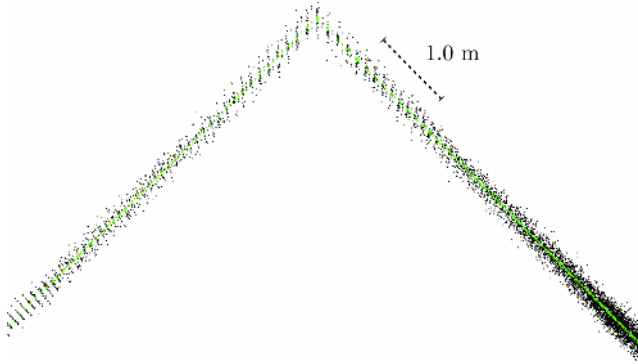
To asses the effect of the FOV, we let the calibration estimated with the full FOV of the lidar be the baseline comparison. For all other FOV’s we report the calibration signed difference against the baseline, *i.e.*:

$$\delta \mathbf{r} = \mathbf{r}_{270^\circ} - \mathbf{r}_{f^\circ} \quad (18)$$

$$\delta \mathbf{t} = \mathbf{t}_{270^\circ} - \mathbf{t}_{f^\circ}, \quad (19)$$

for  $f \in \{200, 180, 90, 45\}$ . Differences are computed individually for each of the five estimated parameters. Namely, x-axis rotation  $\delta r_x$  and translation  $\delta t_x$ , and y-axis rotation and translation  $\delta t_y$ . Results are reported in table 5. We observe an acceptable estimate of rotation for all reduced FOV’s. This is due to the small rotational offsets of the uncalibrated sensing assembly. Furthermore, we observe an acceptable performance up to a 180° FOV. For smaller FOV’s, translation estimates become unreliable due to lack of structure in calibration scene as can be observed in Fig. 11.

Degradation of the calibration quality due to limited FOV can also be seen by examining the uncertainty estimates obtained from the optimization procedure (Eq. (12)). Taking the determinant of the covariance matrix, averaged over multiple trials, as a single number indication of uncertainty, we can see a significant increase in uncertainty with reduction of the FOV as shown in table 6.



**Figure 7:** Illustration of the synthetic noise. The figure shows a cross section of the box geometry around a corner facing the viewing ray of the sensor. The black/dark points are generated with noise magnitude of  $\sigma_r = 64$  mm, while the green/light points are generated with  $\sigma_r = 4$  mm. Ground truth would be a sharp  $90^\circ$  composed of two straight lines, not shown in the figure. Both noise levels are in fact an exaggeration of expected sensor noise. They are included to stress-test the algorithm’s performance. The corner of the box is at approx. 14.14 m from the location of the sensor.

**Table 5:** Calibration results from a reduced FOV lidar compared against calibration results from the full FOV. Angles are reported in degrees and translations are reported in centimeters.

FOV	$\delta r_x$	$\delta r_y$	$\delta t_x$	$\delta t_y$
200	0.010	0.003	-0.001	-0.038
180	0.017	0.005	-0.001	-0.070
90	0.102	0.000	-0.052	-0.550
45	0.901	-0.025	-0.626	-9.006

#### 4.6. Effect of the Environment

Structure of the calibration environment plays a role in the accuracy of the results. A good calibration algorithm, if possible, should not have stringent requirements on the scene structure. Clearly, one should identify and avoid *degenerate geometries* that cannot constrain the degrees of freedom.

The ability to use the algorithm in various environments is especially important for robotic applications where it is not uncommon to require an in-field recalibration. Rough terrain, environmental conditions, and transporting the vehicle to the field might compromise the integrity of a previous calibration.

Here, we show that the algorithm is repeatable when used in different environments provided that the geometry constrains the calibration parameters. In particular, we do not require planar man-made environments of certain dimensions; local planarity is sufficient. Further,

we will show that our algorithm is not strongly affected by the maximum range returns values used to calibrate. This is important because for the same sensor we would like to obtain the same calibration parameters whether the calibration was performed in a small room, or out in the open.

Fig. 12 and Table 7 show the calibration results from the same sensor with data collected from different indoor scenes with different range variations. Estimates of the calibration parameters are consistent across different datasets. The exception is the case of the long narrow corridor. As expected, the structure of the narrow corridor does not provide enough constraints on the calibration parameters. This situation can be detected by examining the covariance of the estimated parameters.

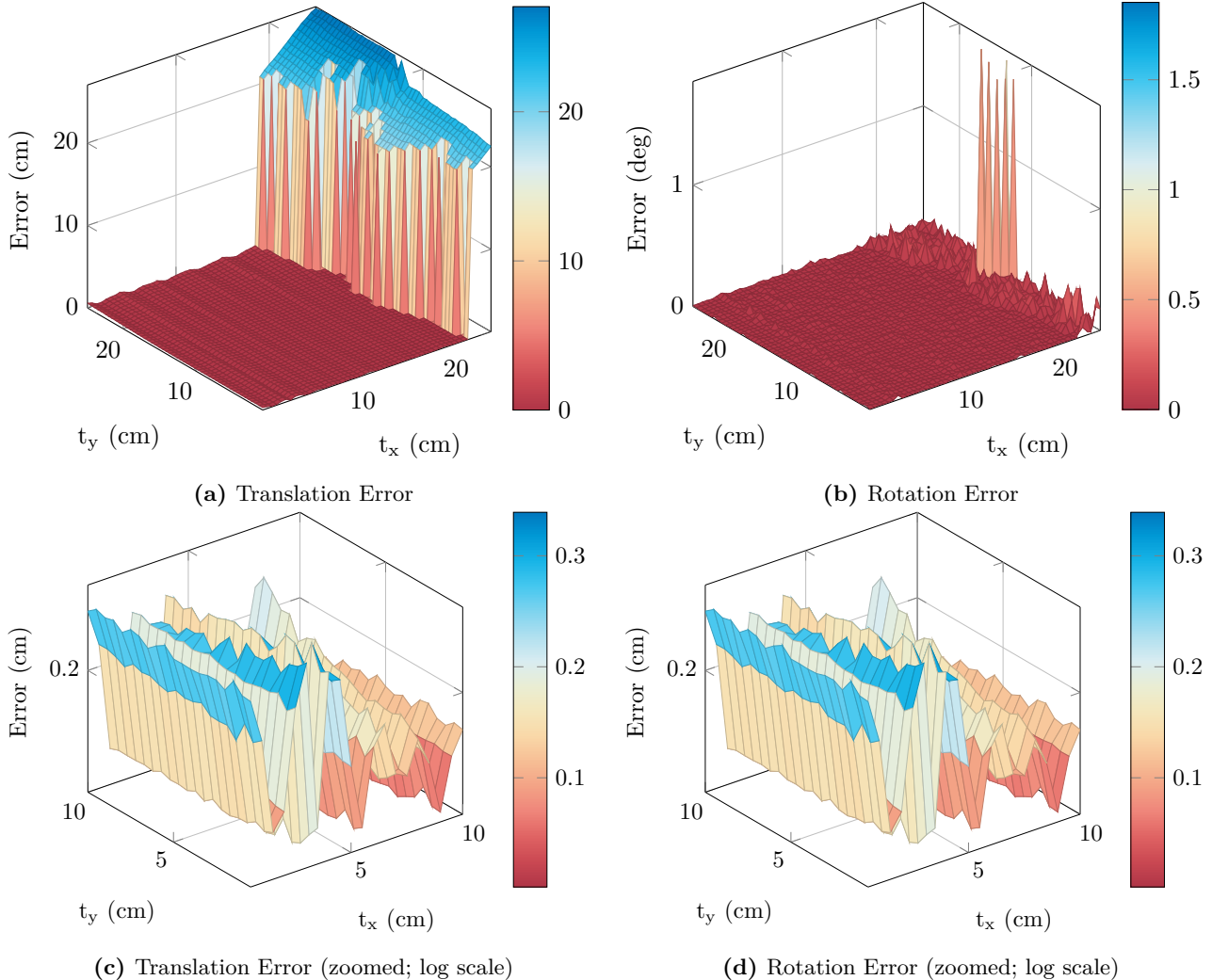
Finally, we apply the algorithm on data collected from an outdoor scene. Results on outdoor data are shown Fig. 14 and verify that the algorithm is suitable for use with outdoor data.

#### 4.7. Application to 3D SLAM

We demonstrate the quality of the calibration for a 3D SLAM application. We use the spinning sensor shown in Fig. 2. The assembly consists of a Hokuyo range scanner mounted on a spinning platform and rigidly coupled with a 1 mega-pixel stereo camera. The sensor assembly performs a full revolution every 4 s (*i.e.* point cloud output is 1/4 Hz). The Hokuyo has a field of view of  $270^\circ$  with angular resolution of  $0.25^\circ$  (*i.e.* 1081 range measurements per scanline). Stereo visual odometry (VO) (Alismail et al., 2010) is used to account for the motion of the vehicle while scanning. This is possible via another calibration procedure between the camera and the lidar assembly (Alismail et al., 2012). We receive pose updates from VO at the rate of 10 Hz. The poses are interpolated per range measurement to allow us to reconstruct the point cloud from the moving sensor. In case of VO dropouts — *e.g.* due to sudden changes in lighting or motion blur — we drop the VO frame and the associated lidar points. VO accumulates drift over time. To reduce this drift, we apply point-point ICP (Besl and McKay, 1992) for every 2 s worth of lidar data. This is shown to reduce local VO drift without the need for a Bundle Adjustment procedure that our embedded computing platform cannot perform in real time. While it is possible to use SLAM methods designed to capture the continuous nature of the scanning process (Bosse and Zlot, 2009; Alismail et al., 2014; Tong et al., 2013), we opt for this registration-based SLAM approach due to its popularity and the impact of internal calibration on performance.

The sensor is mounted on a small-sized mobile robot





**Figure 8:** Basin of convergence. Translation error (left) and rotation error (right) for a grid of translation along the X- and Y-axes starting from 0.5 cm to 25 cm. The algorithm is able to estimate the correct calibration parameters up to the boundary of 20 cm. The last row shows a zoomed in view of the error surface for a translation magnitude up to 10 cm, where the maximum translation error is 0.34 cm and the maximum rotation error is 0.045°.

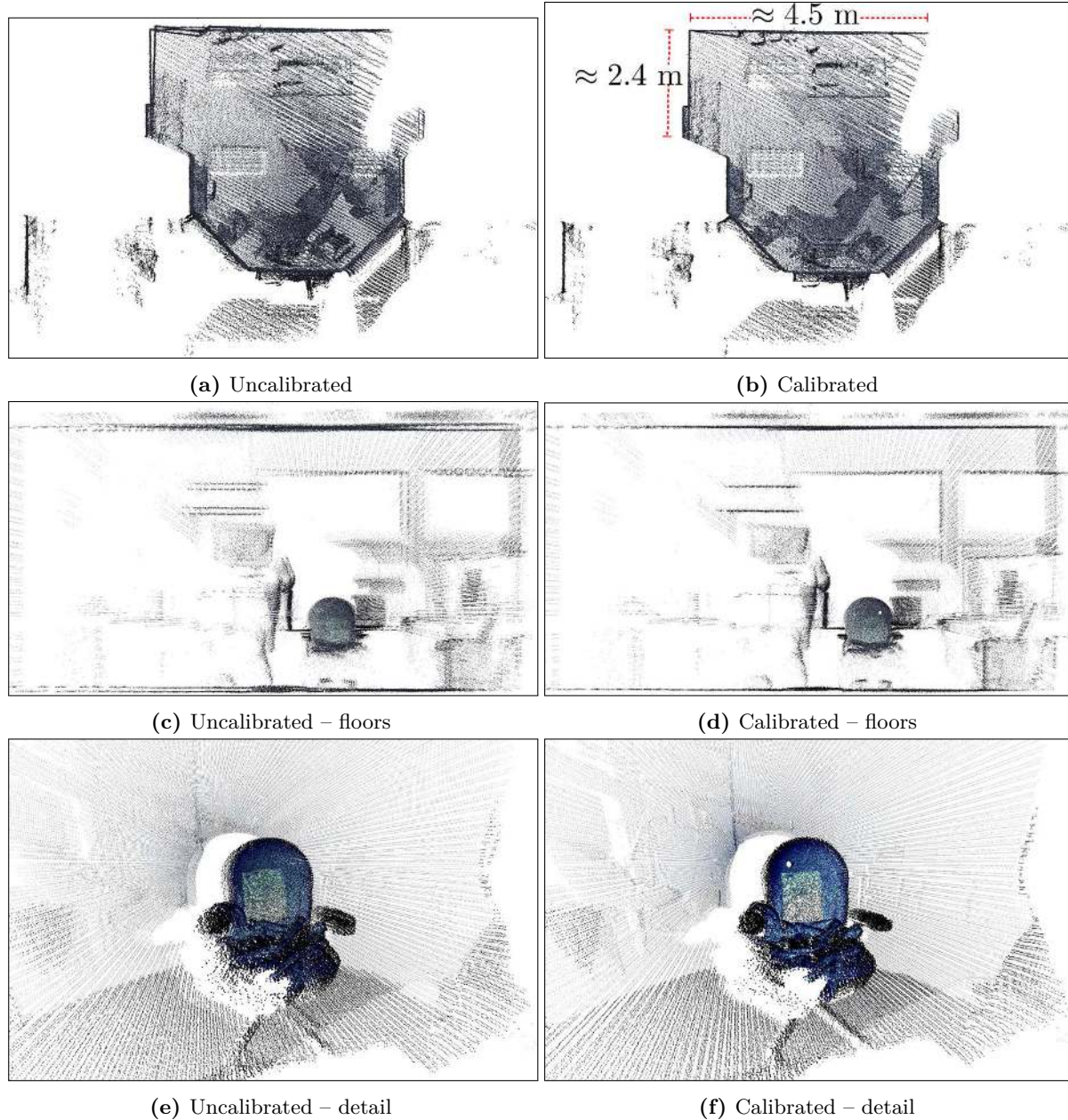
**Table 6:** Determinant of the covariance matrix associated with the estimated parameters for the different FOV’s.

FOV	270	200	180	90	45
$\det(\Sigma) \times 10^8$	0.0054	0.0068	0.0092	0.0527	1.0814

platform (Packbot from iRobot) with the primary application for 3D mapping of underground mines. The system, however, is not restricted to underground environments. We show examples of our mapping results to illustrate the quality of the calibration. As with all complex systems, calibration plays a small, but important, role in the overall system performance. If our calibration accuracy was not satisfactory, we expect to see visible errors in the resulting scene structure. In this case, such

errors are expected to manifest as “double-walls”, or ghosting artifacts in the resulting scene reconstruction. We illustrate this effect in Section 4.8.

Fig. 15 shows some examples of our system’s performance in an underground mine. The closeup views show crisp point clouds and the lack of any visible artifacts. In all of the results shown here, a loop closure procedure was not performed as to not hide any calibration errors. A longer run is shown in Fig. 16 where

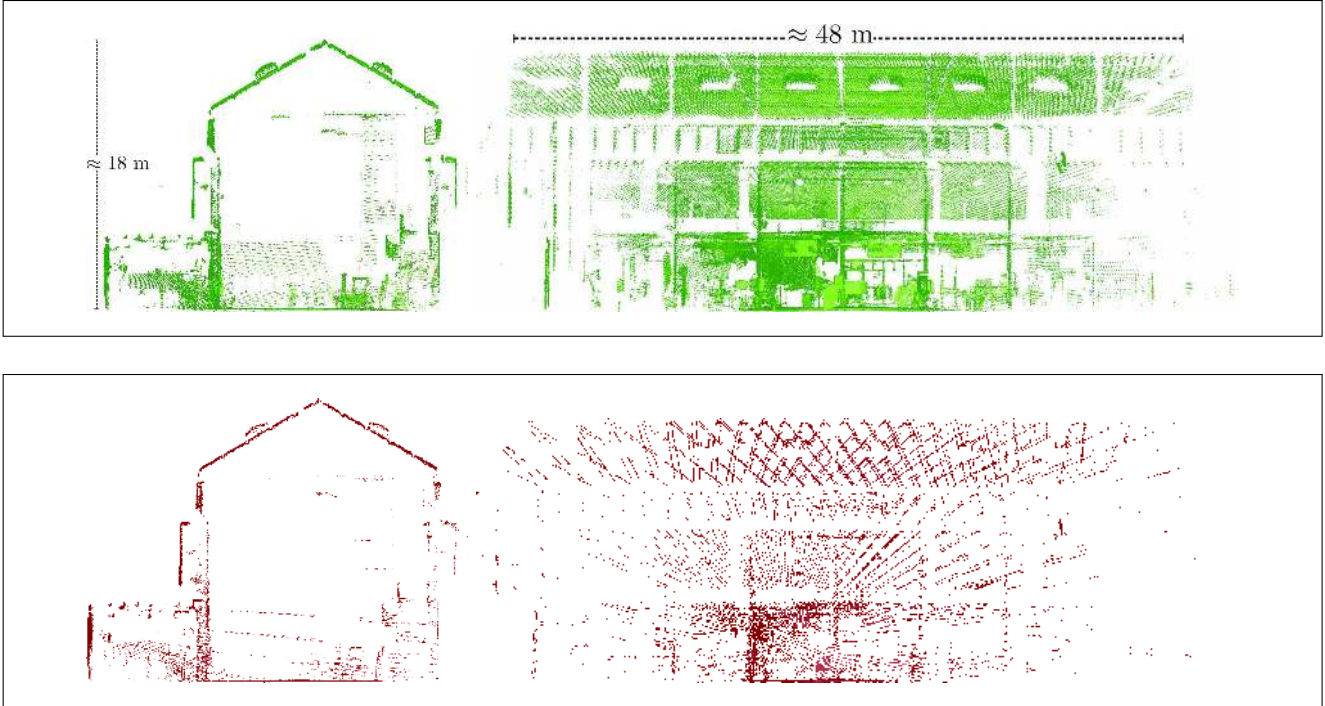


**Figure 9:** Real data inside an office environment. The figure shows the ceiling and walls of the room. Points are colored with the lidar reflectance returns for better visualization. Our algorithm does not make use of reflectance data. Notice the corrected errors in the corners of the room and better point cloud crispness as indicated by reflectance visualization. The bottom row shows a closeup view of a chair directly placed in front of the sensor. A sheet of white paper is placed on the chair to enhance visualization. Prior to calibration (Figs. 9c and 9e) we can see clear ghosting artifacts on the chair as well as double-floors. These effects are eliminated post calibration (Figs. 9d and 9f).

the longest corridor in the mine is approximately 150 m without chances for loop closure.

Perhaps a better illustration of the calibration quality is by inspecting the reconstruction of indoor environments where it is easy to verify the correctness of scanned man-made environments. Fig. 17 shows the

results of our system in an indoor industrial complex collected at the second floor of our facility. The original point cloud has more than 40 million points. Fig. 18 shows some detailed views of the mapped building.



**Figure 10:** Calibration results from two representative RPM settings. Top row shows data collected with 5 RPM, bottom row shows 30 RPM. Data was collected in our test facility. The height of the building measured from the ground to top of the ceiling (left column) is approx. 18 m, while the length (right column) is approx. 48 m. While at a first glance, both calibration results produce similar point clouds, closer inspection shows some errors using data collected with 30RPM. We used the same number of neighbors to compute the adaptive neighborhood for each of the datasets.

**Table 7:** Using different data to calibrate the same sensor. Calibration results shown in Fig. 12 for a 4 DOF model  $(r_y, r_x, t_x, t_y)$ , with rotations in *degrees* and translations in *millimeters*. Results are consistent among the Corner and the Mailroom data. Nonetheless, we can observe an underestimated translation in the Corridor data due to the under-constrained geometry of the scene. This can be detected by observing the covariance matrix where the uncertainty of the estimated translation is orders of magnitude higher for the Corridor data compared to the other datasets.

Dataset	Parameters			
	$r_x$	$r_y$	$t_x$	$t_y$
Corner	-0.594	0.666	2.598	-25.420
Mailroom	-0.602	0.652	2.597	-25.592
Corridor	-0.671	0.625	0.660	-22.012

#### 4.8. Effect of Calibration on 3D SLAM

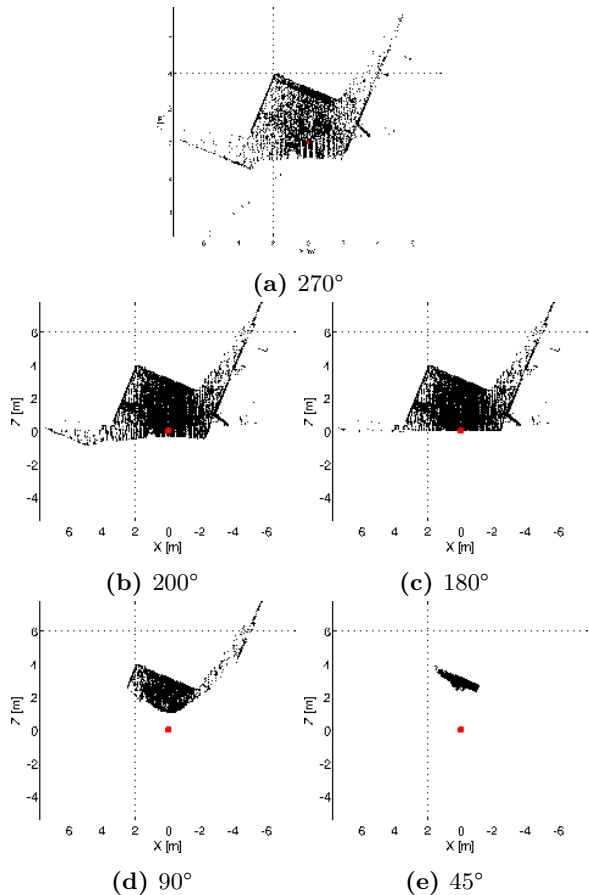
In this section, we show the effect of calibration on SLAM results. Fig. 19 shows SLAM results using the same sensor and SLAM algorithm described in the previous section. The left panel shows the result without applying the calibration. While we do not have ground truth, we can see that SLAM results with our calibration method are more accurate. The resulting map is sharper and the indoor environment is more defined. The structure on the ground plane is also straight after

calibration in comparison to a bent structure prior to calibration.

## 5. Discussion

In this work, we did not make use of lidar intensity returns, which could contribute useful constraints on the calibration. Nonetheless, forming this image is not always possible. For some sensors, intensity returns may not be available, or may not have enough sampling density to construct an image suitable for calibration.





**Figure 11:** Data used for calibration from different FOV's. The sensor is located at the origin (indicated with a square).

In such cases, interpolation artifacts might severely impact the accuracy of corner localization. Consider, for example, a spinning lidar observing a calibration target at long-range. Regardless of the spinning resolution of the actuator, a clean intensity image is difficult to obtain (*c.f.* Fig. 9).

In addition, intensity-based methods rely on calibration targets. This has some drawbacks, namely the need to correctly identify and extract the calibration pattern from a large amount of data. This also restricts the calibration to close-range, where the calibration target can be accurately localized in the intensity image and may induce range-dependent bias in the results.

We note that it is possible to use an actuated lidar in SLAM applications without a specialized calibration step as describe in the work by (Bosse and Zlot, 2009). The algorithm proposed by (Bosse and Zlot, 2009) solves the SLAM problem by optimizing a continuous trajectory that aligns half-scans from a spinning lidar assembly. However, calibration remains important and useful. During their recent work, (Zlot and Bosse,

2014) indicate the use of their SLAM algorithm to calibrate a spinning lidar from stationary data prior to using the sensor for SLAM. Similar to our results, (Zlot and Bosse, 2014) observe an improved SLAM system when the internal parameters of the sensor have been calibrated.

Our algorithm is accurate, automated and generic in the sense that it does not enforce restrictive assumptions on the sensor configuration. The only requirement on the actuation mechanism is that points from each set (half-scans for spinner) are produced from distinct joint angles (lidar mirror and actuation angles). Otherwise, both sets will be identical and do not provide enough information to allow for an automated calibration.

The canonical example for this case is a nodding/pitching sensor as illustrated in Fig. 21. On the left column of Figs. 21a and 21b we show data from a calibrated nodder (using CAD measurements). To the right is an illustration of how the data would appear if the calibration offsets were not accounted for. We have used a large offset of one meter to stress the effect visually. The datasets contain several nods of the scene. In Fig. 21c both, calibration and uncalibrated, are overlaid to show the sever distortion. Not only that distance measurements are incorrect, but also we observe non-rigid shearing and stretching. Solving the nodder would require making more restrictive assumptions on the environment (e.g. it consists of large, readily identifiable planes such as the ground surface), or integrating motion which would couple the pose estimation problem.

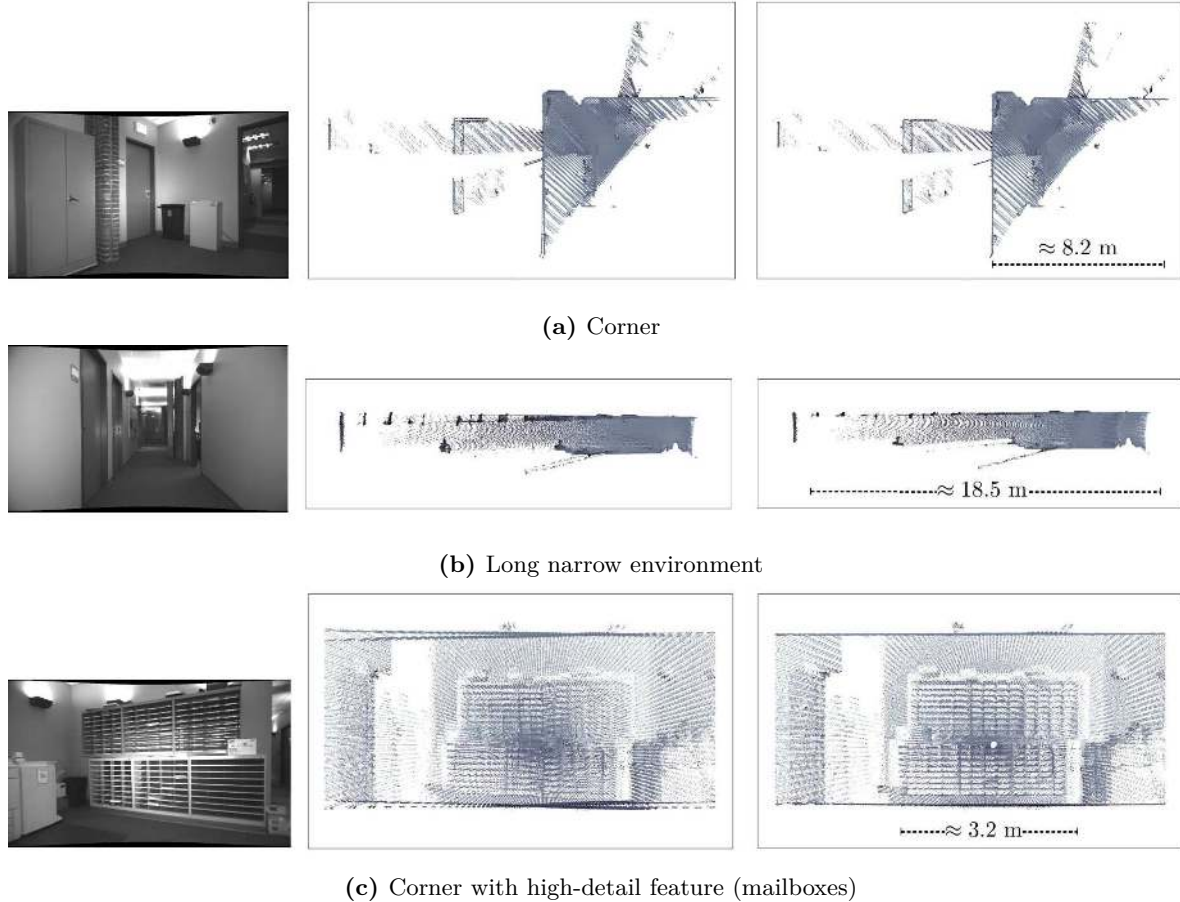
As can be seen, data collected while the sensor is nodding upwards is identical to the data collected while the sensor is nodding downwards. This is because the lidar beam sampling the scene is identical in both nodding directions. In the spinning lidar case, the second half-scan switches the beams sampling the scene and we obtain enough information for automated calibration.

## 5.1. Failure Cases

Although we have observed excellent empirical performance on spinning and wobbling (two-axis) lidar systems, the algorithm is not fail-proof. It is expected to fail in the following scenarios:

- Insufficiently constrained geometry. For example, a single plane in the scene will not be able to constrain the full degrees of freedom of the calibration model. To avoid this situation, we recommend collecting data with at least three visible plane orientations. Room-like structures, with several orthogonal plane orientations are ideal.
- Lack of surface correspondences. We have shown that the algorithm is resilient to lack of precise





**Figure 12:** Example calibration results for the same sensor using data from different indoor environments. On the left we show an image of the environment for reference. The center column shows the environment from an uncalibrated sensors. Calibration results are shown on the right. A closeup view of the details in the Mailroom data is shown in Fig. 13.



**Figure 13:** Detailed top view of the mail room scene shown in Fig. 12c. The top figure shows an uncalibrated sensor, while the bottom shows the calibration results. Protrusions going “inside” the wall are the mail boxes. Notice the expected more regular structure of the boxes post calibration.

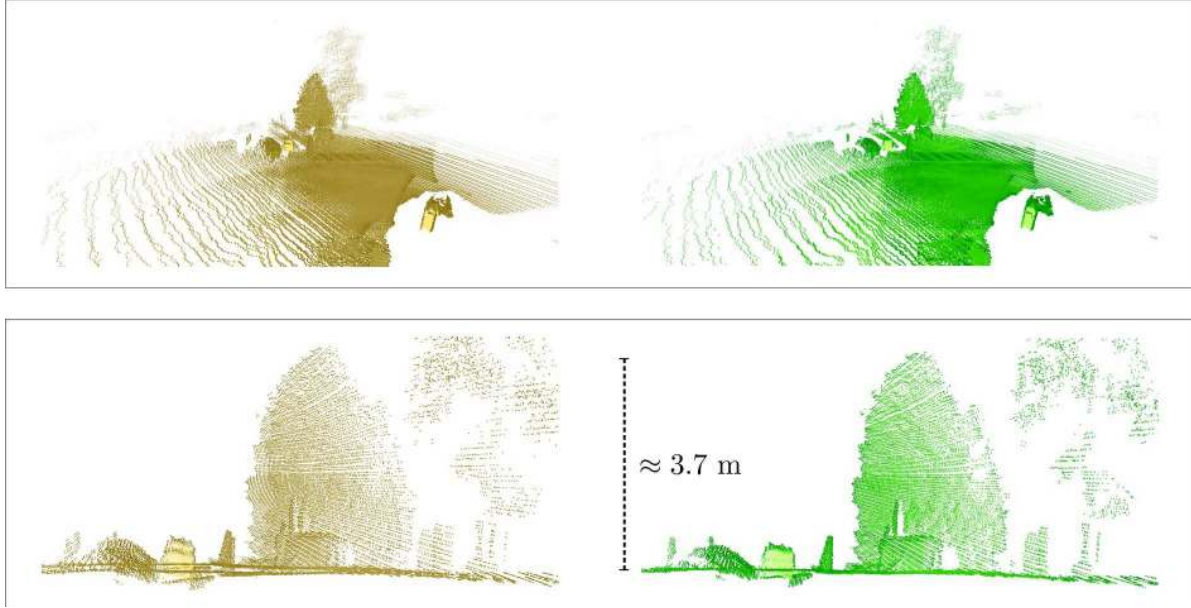
point-point correspondences. However, for extremely sparse data the probability of correct surface correspondences drops and might affect the accuracy of the algorithm. Such situations might happen with the actuator’s speed is very fast and the majority of the scene structure lies at a far distance (i.e. the point sampling density is large

with respect to the surface geometry roughness). This is easily avoided by adjusting the actuation speed and selecting the proper environment for calibration.

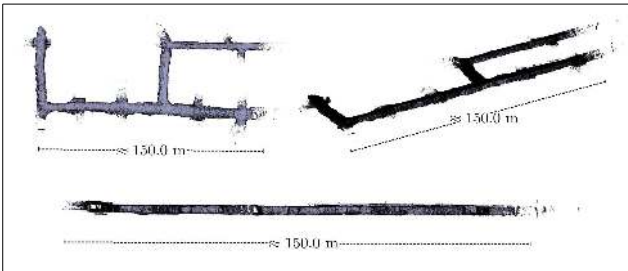
- Very large calibration offsets with lack of a good initialization. Our problem here differs significantly from standard registration problems. The calibration values directly affect the resulting point clouds. This is better visualized in Fig. 22. The figure shows a large offset that causes severe distortion of the data that we cannot recover from without the aid of an initialization. The initialization need *not* be very accurate, it is merely needed to reduces the distortion in the data.

## 5.2. Accuracy Assessment

Calibration results will need to be inspected carefully. As was shown in Fig. 12, it is possible for the algorithm to generate a very visually convincing result



**Figure 14:** Calibration results on outdoor data. The left column shows a point cloud triangulated from an uncalibrated sensor with clear misalignment. On the right, the point cloud triangulated with the estimated calibration.



**Figure 15:** Mine mapping results. Top left figure shows a small section of the mine from top. The longest corridor is approximately 150 m. To the right is a 3D view of the scene. On the bottom is a profile view demonstrating the lack of vertical drift.

on under-constrained geometry. In addition to inspecting the covariance of the estimates, we recommend a visual inspection of the calibration results using a “test data” of different scenes. Yet, a better verification of the calibration accuracy is evaluation as a function of system’s performance, such as SLAM results compared to ground truth if available.

## 6. Conclusions & Future Work

In this work we presented an algorithm for fully automated targetless calibration of actuated spinning lidar internal parameters. These parameters are the mechanical offsets between the center of rotation of the lidar’s mirror and the center of rotation of the actuation

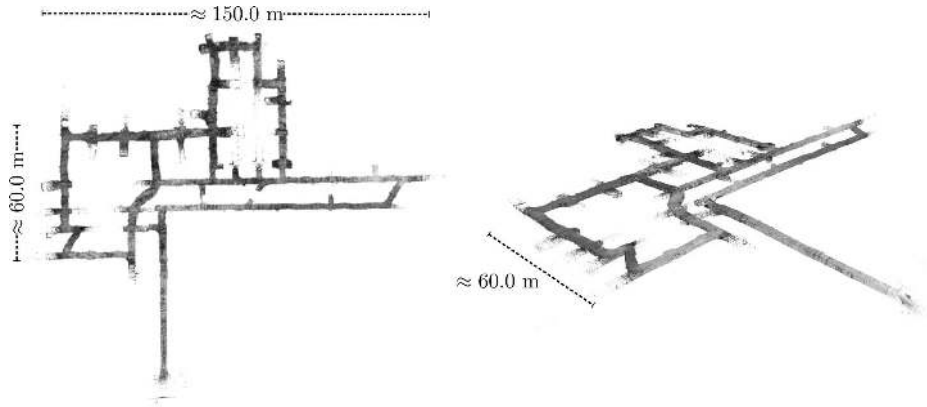
mechanism. The algorithm was evaluated on real and simulated data and was shown to be repeatable, robust and accurate.

Our algorithm does not rely on specific calibration targets and is readily applicable for any locally-planar environments. Robustness to errors in normal estimation was achieved by using an adaptive data-driven selection of the neighborhood radius. Furthermore, by automatically detecting unreliably estimated planes we can reduce their effect on calibration accuracy by down weighting their contributions to the error function.

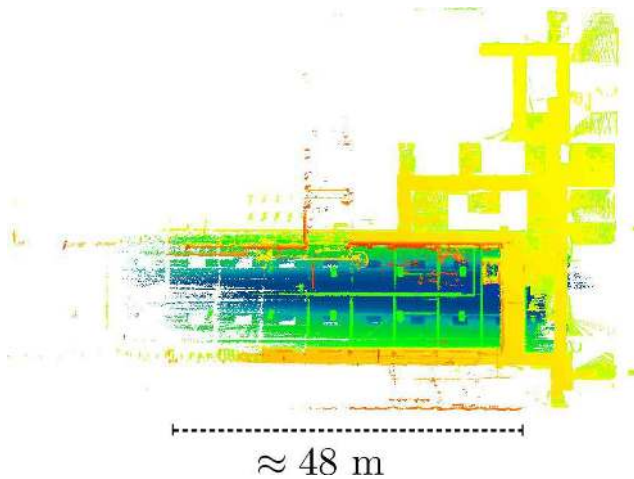
In this work, we did not address calibration problems pertaining to timing errors, range bias due to different object materials, or bias due to thermal fluctuations. These might be avenues of future work.

We also did not make use of lidar reflectance/intensity information in the calibration process. This is an advantage that makes the algorithm applicable to a variety of sensors. However, reflectance data contain important information that could be used to aid in the search for correspondences and even improve the cost function. We have not observed a need to include lidar reflectance in any of the steps. A more interesting direction of future work is designing an automated algorithm for nodder calibration.

Another venue of future work is studying the impact of calibration errors on the shape of the point clouds (Habib and Kersting, 2010), and the overall system performance. Better understanding of calibration error effects on system performance is important



**Figure 16:** 3D mapping results in a mine environment. We show a top down view and a 3D view reconstruction. Distance travelled by the robot is approximately 1.0 km. While we do not have precise ground truth, the results look visually good and there is a small amount of drift in the estimated robot trajectory. These results are obtained without loop closure.

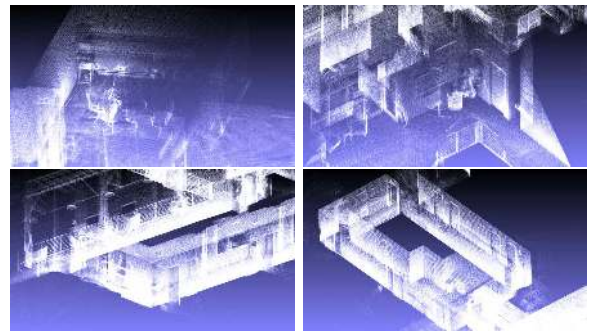


**Figure 17:** SLAM results in an indoor environment with several loops. The view is from top and colorized by height. This is the same building shown in Fig. 10. A highbay area is shown at the center of the figure. This is the area with the largest height variation and is surrounded by offices. The overhanging structures in the middle of the highbay are lighting fixtures.

to achieving high accuracy, identifying mis-calibration events and devising online calibration methods (Roy and Thrun, 1999; Hansen et al., 2012; Levinson and Thrun, 2014).

## Acknowledgements

We thank the anonymous reviewers for their feedback, which helped improve the work. We also acknowledge the following people for their help: Doug Baker, Jacqueline Libby, Michael McGraw, Eric Meyhofer, David Prasser, and Venkataramanan Rajagopalan.

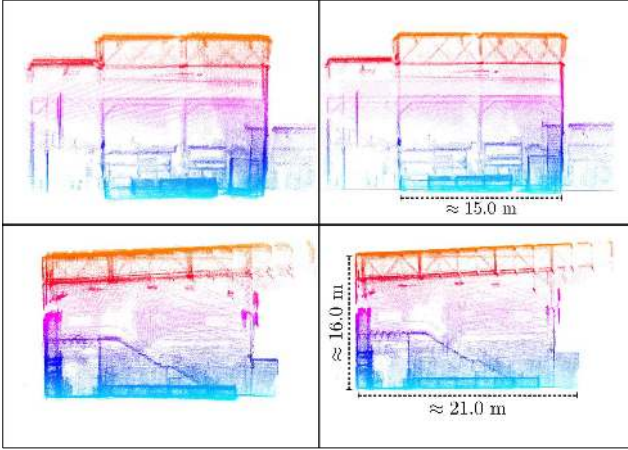


**Figure 18:** Closeup views of some parts of the building shown in Fig. 17. The robot visited the place more than once as part of the mapping process.

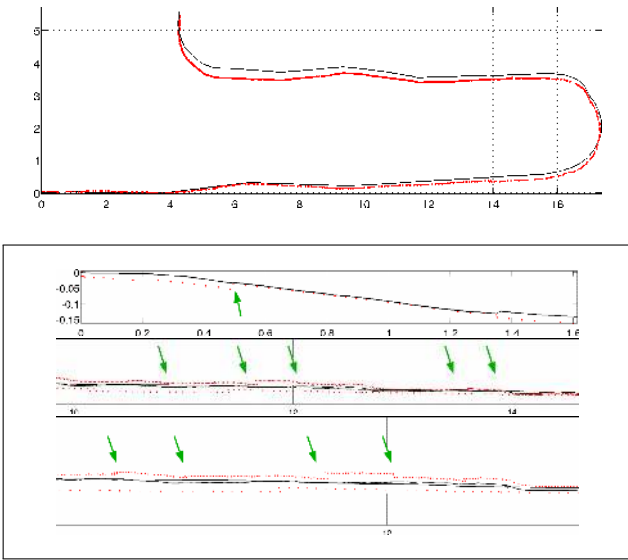
## References

- Alismail, H., Baker, L., and Browning, B. (2012). Automatic calibration of a range sensor and camera system. In *Second International Conference on 3D Imaging, Modeling, Processing, Visualization and Transmission (3DIMPVT)*, pages 286–292.
- Alismail, H., Baker, L., and Browning, B. (2014). Continuous trajectory estimation for 3D SLAM from actuated lidar. In *IEEE International Conference on Robotics and Automation (ICRA)*.
- Alismail, H., Browning, B., and Dias, M. B. (2010). Evaluating pose estimation methods for stereo visual odometry on robots. In *the 11th International Conference on Intelligent Autonomous Systems (IAS-11)*, pages 101–110.
- Alwan, M., Wagner, M., Wasson, G., and Sheth, P. (2005). Characterization of infrared range-finder PBS-03JN for 2-D mapping. In *the IEEE International Conference on Robotics and Automation (ICRA)*, pages 3936–3941.
- Amenta, N. and Kil, Y. J. (2004). Defining point-set surfaces.





**Figure 19:** Effect on SLAM. On the left we show SLAM results without applying calibration the offsets ( ${}^A\mathbf{H}_L = \mathbf{I}$ ). On the right is SLAM with our calibration applied. Note the sharper structure from the indoor environment once the calibration has been applied. Robot trajectory is shown in Fig. 20. Points are colorized by height.

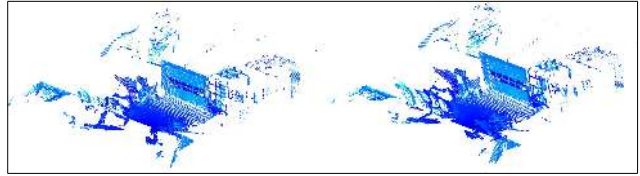


**Figure 20:** Effect on robot trajectory. Solid dark/black is the estimated robot trajectory from sensor data with calibration. Dotted red/light is the estimated trajectory without calibration. Units in meters. The trajectory correspondences to the scene shown in Fig. 19. Robot trajectory without calibration exhibits periodic staircase effects where ICP tries to align lidar half-scans (shown in the closeup views and highlighted with arrows). Jumps in the trajectory correspond to the start/end of a half-scan. The jumps are very close to the estimated calibration parameters  $\approx 3$  cm.

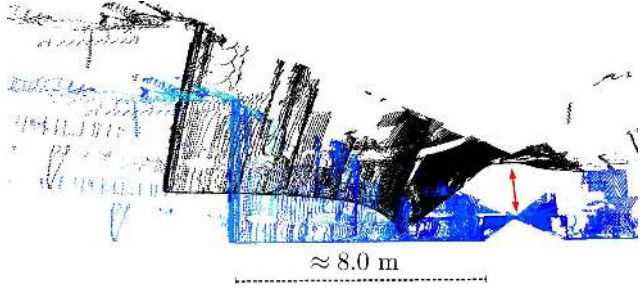
In *ACM Transactions on Graphics*, volume 23, pages 264–270. ACM.



(a) Front view.



(b) 3D view.



(c) Side view. The point cloud generated without calibration is colored in black/dark. The arrow indicates location of the motor's center of rotation in both datasets.

**Figure 21:** Typical problem with a nodding lidar where alignment of the lidar scanner mirror angle across half-scans does not produce information suitable for calibration. In Fig. 21a and Fig. 21b point clouds generated with a manually calibrated nodder and uncalibrated are shown on the left and right respectively. See text for details.

Besl, P. J. and McKay, H. D. (1992). A method for registration of 3-D shapes. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 14(2):239–256.

Bosse, M. and Zlot, R. (2009). Continuous 3d scan-matching with a spinning 2d laser. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 4312–4319.

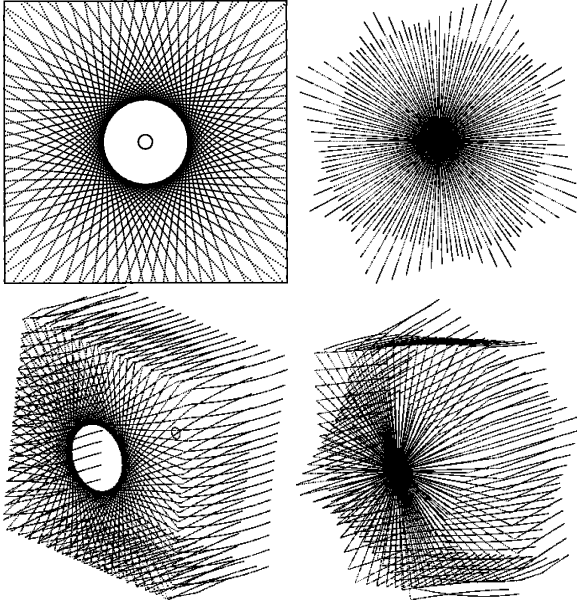
Bosse, M., Zlot, R., and Flick, P. (2012). Zebedee: Design of a spring-mounted 3-d range sensor with application to mobile mapping. *IEEE Transactions on Robotics*, 28(5):1104–1119.

Browning, B., Deschaud, J.-E., Prasser, D., and Rander, P. (2012). 3d mapping for high-fidelity unmanned ground vehicle lidar simulation. *The International Journal of Robotics Research*, 31(12):1349–1376.

Censi, A. (2007). An accurate closed-form estimate of ICP's covariance. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 3167–3172, Rome, Italy.

Chen, Y. and Medioni, G. (1992). Object modelling by registration of multiple range images. *Image Vision Computing*, 10(3):145–155.





**Figure 22:** Illustration of large calibration offsets. The ground truth is shown on the left column. The right column shows a large calibration offset with  $\frac{1}{2}$  meter translation and  $10^\circ$  rotation about the Y-axis, with the points triangulated assuming an identity calibration. The top row shows a front view of the simulated scene, while the bottom row shows a 3D view. While the magnitude of the offset could be managed with standard registration problem, the problem is exacerbated in the case of calibration. This is because the calibration values *produce* the point cloud. Notice the distortions in the point cloud shown at the bottom right corner where scene planes are severely warped. In this case, our calibration algorithm fails to find the correct solution. The algorithm correctly estimates  $10^\circ$  rotation about the Y-axis, but adds an extra  $15^\circ$  rotation about the X-axis at the expense of translation estimates.

Dong, H., Anderson, S., and Barfoot, T. (2013). Two-axis scanning lidar geometric calibration using intensity imagery and distortion mapping. In *the IEEE International Conference on Robotics and Automation (ICRA)*, pages 3672–3678.

Fairfield, N. (2009). *Localization, Mapping, and Planning in 3D Environments*. PhD thesis, Robotics Institute, Carnegie Mellon University, Pittsburgh, PA.

Fitzgibbon, A. W. (2003). Robust registration of 2d and 3d point sets. *Image Vision Computing*, 21(13-14):1145–1153.

Habib, A. and Kersting, A. (2010). Impact of lidar system calibration on the relative and absolute accuracy of the adjusted point cloud. In *European Calibration and Orientation Workshop on Integrated Systems for Sensor Georeferencing and Navigation*.

Hansen, P., Alismail, H., Rander, P., and Browning, B. (2012). Online continuous stereo extrinsic parameter

estimation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1059–1066.

Haralick, R. M. (2000). Propagating covariance in computer vision. In *Performance Characterization in Computer Vision*, pages 95–114. Springer.

Hartley, R., Trunpf, J., Dai, Y., and Li, H. (2013). Rotation averaging. *International Journal of Computer Vision (IJCV)*, 103(3):267–305.

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

Jian, B. and Vemuri, B. C. (2011). Robust point set registration using gaussian mixture models. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 33(8):1633–1645.

Kelly, A. (1994). Essential kinematics for autonomous vehicles. Technical Report CMU-RI-TR-94-14, Robotics Institute, Pittsburgh, PA.

Kelly, A., Chan, N., Herman, H., Huber, D., Meyers, R., Rander, P., Warner, R., Ziglar, J., and Capstick, E. (2011). Real-time photorealistic virtualized reality interface for remote mobile robot control. *The International Journal of Robotics Research*, 30(3):384–404.

Kneip, L., Tâche, F., Caprari, G., and Siegwart, R. (2009). Characterization of the compact hokuyo URG-04LX 2d laser range scanner. In *the IEEE International Conference on Robotics and Automation (ICRA)*, ICRA, pages 2522–2529, Piscataway, NJ, USA. IEEE Press.

Kuffner, J. (2004). Effective sampling and distance metrics for 3d rigid body path planning. In *the IEEE International Conference on Robotics and Automation (ICRA)*. IEEE.

Levenberg, K. (1944). A method for the solution of certain non-linear problems in least squares. *Quarterly Journal of Applied Mathematics*, II(2):164–168.

Levinson, J. and Thrun, S. (2014). Unsupervised calibration for multi-beam lasers. In Khatib, O., Kumar, V., and Sukhatme, G., editors, *Experimental Robotics*, volume 79 of *Springer Tracts in Advanced Robotics*, pages 179–193. Springer Berlin Heidelberg.

Marquardt, D. W. (1963). An algorithm for least-squares estimation of nonlinear parameters. *Journal of the Society for Industrial and Applied Mathematics*, 11(2):pp. 431–441.

Meng, X., Currit, N., and Zhao, K. (2010). Ground filtering algorithms for airborne lidar data: A review of critical issues. *Remote Sensing*, 2(3):833–860.

Mertz, C., Navarro-Serment, L. E., Duggins, D., Gowdy, J., MacLachlan, R., Rybski, P., Steinfeld, A., Suppe, A., Urmson, C., Vandapel, N., Hebert, M., and Thorpe, C. (2013). Moving object detection with laser scanners. *Journal of Field Robotics*, 30(1):17 – 43.

- Mirzaei, F. M., Kottas, D. G., and Roumeliotis, S. I. (2012). 3D LIDAR-camera intrinsic and extrinsic calibration: Identifiability and analytical least-squares-based initialization. *The International Journal of Robotics Research*, 31(4):452–467.
- Muhammad, N. and Lacroix, S. (2010). Calibration of a rotating multi-beam lidar. In *The IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 5648–5653.
- Muja, M. and Lowe, D. (2014). Scalable nearest neighbour algorithms for high dimensional data. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, PP(99):1–1.
- Okubo, Y., Ye, C., and Borenstein, J. (2009). Characterization of the hokuyo URG-04LX laser rangefinder for mobile robot obstacle negotiation. In *SPIE Defense, Security, and Sensing*, pages 733212–733212.
- Pradeep, V., Konolige, K., and Berger, E. (2010). Calibrating a multi-arm multi-sensor robot: A bundle adjustment approach. In *International Symposium on Experimental Robotics (ISER)*, New Delhi, India.
- Renslow, M., Greenfield, P., and Guay, T. (2000). Evaluation of multi-return lidar for forestry applications. *US Department of Agriculture Forest Service-Engineering, Remote Sensing Applications*.
- Roy, N. and Thrun, S. (1999). Online self-calibration for mobile robots. In *the IEEE International Conference on Robotics and Automation (ICRA)*, volume 3, pages 2292–2297. IEEE.
- Scherer, S., Rehder, J., Achar, S., Cover, H., Chambers, A. D., Nuske, S. T., and Singh, S. (2012). River mapping from a flying robot: state estimation, river detection, and obstacle mapping. *Autonomous Robots*, 32(5).
- Sheehan, M., Harrison, A., and Newman, P. (2012). Self-calibration for a 3d laser. *The International Journal of Robotics Research*, 31(5):675–687.
- Singh, S. and West, J. (1991). Cyclone: A laser scanner for mobile robot navigation. Technical Report CMU-RI-TR-91-18, Robotics Institute, Pittsburgh, PA.
- Stentz, A., Bares, J., Pilarski, T., and Stager, D. (2007). The crusher system for autonomous navigation. In *AUVSI’s Unmanned Systems North America*, Washington D.C.
- Thrun, S., Haehnel, D., Ferguson, D., Montemerlo, M., Triebel, R., Burgard, W., Baker, C. R., Omohundro, Z., Thayer, S., and Whittaker, W. R. L. (2003). A system for volumetric robotic mapping of abandoned mines. In *the IEEE International Conference on Robotics and Automation (ICRA)*, volume 3, pages 4270 – 4275.
- Tong, C. H., Furgale, P., and Barfoot, T. D. (2013). Gaussian process gauss-newton for non-parametric simultaneous localization and mapping. *The International Journal of Robotics Research*, 32(5):507–525.
- Triggs, B., Mclauchlan, P. F., Hartley, R. I., and Fitzgibbon, A. W. (2000). Bundle adjustment – a modern synthesis. *Lecture Notes in Computer Science*, 1883:298 – 372.
- Tsin, Y. (2003). *Kernel Correlation as an Affinity Measure in Point-Sampled Vision Problems*. PhD thesis, Robotics Institute, Carnegie Mellon University, Pittsburgh, PA.
- Underwood, J. P., Hill, A., Peynot, T., and Scheduling, S. J. (2010). Error modeling and calibration of exteroceptive sensors for accurate mapping applications. *Journal of Field Robotics*, 27(1):2–20.
- Unnikrishnan, R. and Hebert, M. (2005). Fast extrinsic calibration of a laser rangefinder to a camera. Technical Report CMU-RI-TR-05-09, Robotics Institute.
- Unnikrishnan, R., Lalonde, J.-F., Vandapel, N., and Hebert, M. (2010). Scale selection for geometric fitting in noisy point clouds. *The International Journal on Computational Geometry and Applications*, 20(5).
- Urmson, C., Anhalt, J., Bae, H., Bagnell, J. A. D., Baker, C. R., Bittner, R. E., Brown, T., Clark, M. N., Darms, M., Demitrish, D., Dolan, J. M., Duggins, D., Ferguson, D., Galatali, T., Geyer, C. M., Gittleman, M., Harbaugh, S., Hebert, M., Howard, T., Kolski, S., Likhachev, M., Litkouhi, B., Kelly, A., McNaughton, M., Miller, N., Nickolaou, J., Peterson, K., Pilnick, B., Rajkumar, R., Rybski, P., Sadekar, V., Salesky, B., Seo, Y.-W., Singh, S., Snider, J. M., Struble, J. C., Stentz, A. T., Taylor, M., Whittaker, W. R. L., Wolkowicki, Z., Zhang, W., and Zigar, J. (2008). Autonomous driving in urban environments: Boss and the urban challenge. *Journal of Field Robotics: Special Issue on the 2007 DARPA Urban Challenge, Part I*, 25(8):425–466.
- van de Geer, S. (2005). Least squares estimators. In Everitt, B. S. and Howell, D. C., editors, *Encyclopedia of Statistics in Behavioral Science*, volume 2, pages 1041–1045. Wiley and Sons.
- Weingarten, J. (2006). *Feature-based 3D SLAM*. PhD thesis, École polytechnique fédérale de Lausanne (EPFL). Thesis No. 3601.
- Wong, U., Morris, A., Lea, C., Lee, J., Whittaker, C., Gurney, B., and Whittaker, R. (2011). Comparative evaluation of range sensing technologies for underground void modeling. In *the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3816–3823.
- Wulf, I. and Wagner, B. (2003). Fast 3D Scanning Methods for Laser Measurement Systems. In *International Conference on Control Systems and Computer Science*, volume 1.
- Zlot, R. and Bosse, M. (2014). Efficient large-scale three-dimensional mobile mapping for underground mines. *Journal of Field Robotics*.