Photogrammetric Record, xx(xxx): 000–000 (Month 20##)

POINT CLOUD REFINEMENT WITH SELF-CALIBRATION OF A MOBILE MULTI-BEAM LIDAR SENSOR

NOUIRA HOUSSEM (houssem.nouira@mines-paristech.fr)
DESCHAUD JEAN-EMMANUEL (jean-emmanuel.deschaud@mines-paristech.fr)
GOULETTE FRANÇOIS (francois.goulette@mines-paristech.fr)
MINES ParisTech, PSL - Research University, CAOR - Center for Robotics, 60 Bd St-Michel 75006 Paris, France

Abstract

lidar sensors are widely used in mobile mapping systems. With recent developments, these sensors provide large volumes of data, which are necessary for some applications that require a high level of detail. Multi-beam lidar sensors can provide this level of detail, but need a specific calibration routine to provide the best precision possible. Because they have many beams, the calibration of such sensors is difficult and is not well represented in the literature. In this work, we present an automatic method for the optimization of the calibration parameters of a multi-beam lidar sensor on a mobile platform: the proposed approach does not require any calibration target, and only uses information from the acquired point clouds, which makes it simple to use. The goal of the optimization is to find calibration parameters that will improve the structure of the data. At the end of the automatic process, we are able to give a confidence value for the calibration parameters found.

KEYWORDS: Calibration, Mobile Mapping, Automatic, Computer Processing, Velodyne

INTRODUCTION

Light Detection and Ranging (lidar) sensors are useful for many tasks: mapping (Nuchter et al., 2004), (Ridene et al., 2009), (Tarel et al., 2012), (Serna et al., 2014), (Craciun et al., 2017), localization (Narayana and Goulette, 2009) and autonomous driving (Geiger et al., 2012) are some of the applications in which such sensors are used. Recently, multi-beam lidar sensors have appeared: they are more precise than mono-beam sensors and give point clouds with high densities of points. In order to give correctly reconstructed data with the sensors mounted on a mobile platform, additional information from exteroceptive and proprioceptive sensors is needed. Also, a good calibration of the whole system is required; this can take some time, depending on the acquisition system, and it is difficult to evaluate the precision of the method.

PROBLEM STATEMENT

In this study, we present an unsupervised optimization method that can correct the calibration parameters of a multi-beam lidar sensor used to generate the point clouds; the method does not use any calibration target, and the goal is to refine the acquired point cloud by optimizing the calibration parameters. We call calibration of the multi-beam lidar sensor its complete calibration. This calibration is separated into two calibrations:

(a) Extrinsic calibration, which consists of optimizing six parameters: three of translation and three of rotation, giving the transformation between the sensor and the Inertial Measurement Unit (IMU). We start with the initial parameters, which do not need to be close to the true ones. With an iterative process, we look for "better" calibration parameters, which improve the quality of the point clouds and minimize an energy function. This method applies for any multi-beam sensor mounted on a mobile platform (i.e. a vehicle or a robot), the only condition being to have some data overlapping between the points acquired by the beams.

(b) Intrinsic calibration, which consists of finding some additional parameters for each beam of the lidar sensor, in order to reference the data in the sensor reference frame. We set some beams as references, and we optimize the intrinsic calibration parameters of the other beams regarding these references. The same energy function as for the extrinsic calibration is used.

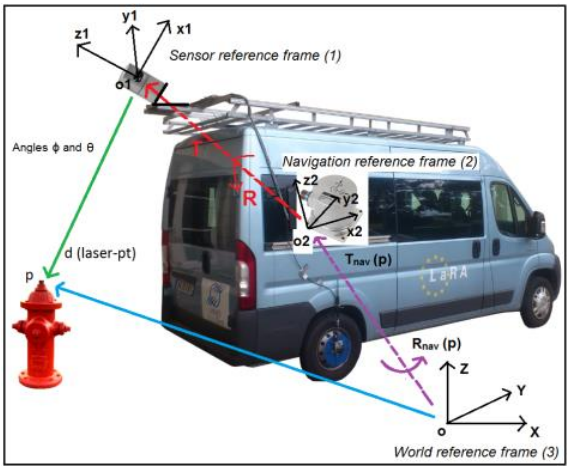


FIG. 1. Georeferencing of the data

Figure 1 shows our mobile mapping system, with a lidar sensor mounted on the roof (the Velodyne HDL-32E): we give its specificities in the section "Proposed optimization method". Fig. 1 also gives the different representations of an acquired point by the mapping system:

(a) Raw data are acquired by the multi-beam sensor mounted on the vehicle, which are the distance of the point acquired to the sensor and two angles.
(b) The raw data can be expressed in the Cartesian reference frame of the sensor: this is carried out using the intrinsic calibration parameters of the sensor.
(c) The extrinsic calibration gives the geometric transformation between the sensor and the IMU - which are mounted on the mobile platform - and is needed to record the coordinates in the navigation reference frame. There are six parameters to retrieve: three rotations and three translations.
(d) The data can also be geo-referenced by applying the transformation between the navigation reference frame and the global reference frame to these data. This transformation is given by the fusion of data from many sensors embedded on the vehicle, such as the IMU and a GPS.

This paper is organized as follows: in the section "Related work", we present the state-of-the-art concerning the algorithms for the calibration of multi-beam lidar systems. The section "Proposed optimization method" presents our optimization methods, for both the extrinsic and intrinsic calibration. The section "Optimization results" gives some experimental results obtained with our algorithm, and finally the Conclusions give a short discussion on the method we present.

RELATED WORK

The calibration of a lidar sensor is an important task, whether it has many beams or not. It allows the sensor to give correctly referenced data during the process of acquisition, which are necessary for many tasks, such as point cloud segmentation (Serna and Marcotegui, 2014) for example. In this section, we will talk about some of the calibration techniques for multi-beam lidar acquisition systems.

Calibration of multi-beam lidar sensors

Multi-beam lidar sensors can be separated into two categories:

(a) Sensors made of several mono-beam lidar sensors, for which the data are fusioned and which provide 3D information with a specific calibration routine, such as the RIEGL sensor.
(b) Multi-beam lidar sensors such as the Velodyne (Velodyne, 2017) or the Quanergy.

Because mono-beam lidar sensors may be cheaper than multi-beam sensors, some 3D mapping systems are constructed around several mono-beam sensors. This is for example the case with the mobile platform presented in Maddern et al. (2012), where the calibration of the 3D lidar is optimized by minimizing an entropy function: since the vehicle is moving during the acquisition, they acquire the same location at two different times; with a good calibration, this location should have the same global position.

We now only discuss the existing work on the calibration of Velodyne lidar sensors, because they have become popular since their advent in 2007; even so, the optimization method that we will present in the next section can be generalized to any multi-beam sensor, dependent on some small changes. The Velodyne multi-beam lidar sensor exists in three different versions, but on our mobile platform presented in Figure 1, we have the

32-beam version (Velodyne 32 beams datasheet, 2017). In the state of the art, several calibration techniques specific to multi-beam lidars exist. For the Velodyne sensor in particular, Glennie and Lichti (2010) and Muhammad and Lacroix (2010) propose an optimization of the intrinsic parameters for the 64-beam version, and Chan and Lichti (2013) propose an intrinsic calibration for the 32-beam model. In Glennie and Lichti (2010) and Muhammad and Lacroix (2010), a particular calibration environment is used to optimize the intrinsic parameters, which are respectively a specific area and a wide wall. In Chan and Lichti (2013), the optimization of the intrinsic parameters is performed statically, by using environmental information such as planar wall and vertical cylinders.

Zhu and Liu (2013) propose a method to optimize the three extrinsic parameters of rotation into two steps: first, the roll and pitch angles by estimating ground planes, and then the yaw angle by matching pole-like obstacles. This method is unsupervised and does not need a calibration target, as it uses only the information from the data gathered; however, the limitation is that it only estimates the rotation parameters, and does not take into account the translation ones. Huang et al. (2013) propose a full extrinsic calibration of a multi-beam lidar sensor, but they use calibration targets and infrared images to perform this task. The limitation with all the previous calibration methods is the use of a specific calibration environment, which takes time to set up.

Elseberg et al. (2013) want to optimize the calibration of the whole system used, which is composed of several lidar sensors. The energy function is a sum of Point Density Functions, which measures the compactness of their point clouds. They use an unsupervised and target-free method in post-processing, where the energy function they defined is minimized. The energy function was constructed in order to measure the compactness of the point clouds acquired.

Underwood et al. (2010) presents some error models in mapping applications, and propose some solutions to the different sources of error. Concerning the calibration of lidar sensors, an optimisation for a mono-beam lidar sensor is presented, with an extension to the case of multiple lidar sensors, which is similar to the case of a multi-beam sensor.

Finally, other approaches optimize both the intrinsic and the extrinsic calibration of a lidar sensor at the same time. This is the case in Levinson and Thrun (2010), where the authors present an intrinsic calibration and an extrinsic calibration method that use the same energy function for minimization. For both optimizations, the authors chose an energy function that penalizes points that are far away from planar surfaces extracted from the acquired data. The main difference with our optimization method is that they perform a grid search around their starting calibration parameters. Also, they optimize the intrinsic and extrinsic parameters separately.

Overview of our optimization method

The optimization we want to propose use the assumption of a locally planar world: indeed, with the high resolution given by multi-beam lidar sensors, the acquired environment respects such an assumption, as stated in Levinson and Thrun (2010). Our approach is based on registering data coming from different beams of the multi-beam sensor, and this assumption allows us to use a point-to-plane distance for the registration. Also, we need a well computed trajectory for the vehicle: our optimization supposes that

the trajectory is correctly constructed, so that with correcting the calibration parameters, we can refine the point cloud.

To optimize the calibration parameters of the multi-beam sensor, we use an energy function that only requires information extracted from the acquired point clouds. No calibration target is used, and the process is unsupervised. The defined energy function is also minimized iteratively, as explained in the next section. However, there are many differences with respect to existing methods:

(a) First, the energy is defined as the sum of the squared distance of each point to the closest plane to which it should belong, and its expected optimal (minimum) value is related to the global covariance of the point cloud noise.
(b) We also introduce the energy weights, which exploit the local planarity of data.
(c) Our method leads to a more accurate point cloud by correcting the calibration parameters, and does not require a precise initialization.
(d) The numerical resolution is fast and is executed in an acceptable time of some minutes.
(e) We give an analysis of the precision obtained for the calibration parameters after optimization.

PROPOSED OPTIMIZATION METHOD

We use a mobile mapping system to perform our acquisitions, as presented in Figure 1: (Goulette et al., 2006) and (Yoo et al., 2010) present with some details the system and the sensors it is equipped with. With the actual version of the mobile system, many sensors are embedded, such as a BEI DHO5S odometer and an iXBlue LANDINS IMU to follow precisely the motion of the vehicle. We also have a Novatel FlexPak 6 GPS to retrieve the global position of the vehicle when possible, and finally a multi-beam lidar sensor, the 32-beam Velodyne, which is mounted on top of the vehicle. The Velodyne sensor provides up to 700 000 points/s, and covers a vertical field of view of 40° – from -8 to +32° – and an horizontal field of view of 360° Also, we know the georeferenced position of the vehicle at each control point, given by the frequency of the fusion IMU+GPS, which is 100 Hz. We assume that the localization of the vehicle is properly provided by the navigation sensors.

The points in a point cloud come from the combination of the acquisitions of each beam of the multi-beam lidar sensor: during the motion of the vehicle, adjacent beams on the sensor will acquire at different times points that belong to the same surface. Figure 2 shows the expected result: with incorrect calibration, points acquired by neighboring beams will not be aligned; with a good calibration, lines of points acquired by close beams will overlap.

Definition of the energy function

Locally, the points on the point cloud should be co-planar, and to optimize the calibration parameters, we minimize a point-to-plane distance, which should be small with the correct calibration parameters. We start the optimization process with an initial calibration, and only use information extracted from the point clouds. Eq. (1) gives the energy function we defined for the optimization of the calibration parameters:

$$J = \frac{\sum_{i=1}^{B}\sum_{j=i-N}^{i+N}\sum_{k} w_{i,j,k} * d_{i,j,k}^{2}}{\sum_{i=1}^{B}\sum_{j=i-N}^{i+N}\sum_{k} w_{i,j,k}} \quad (1)$$

where:

$$\begin{cases} d_{i,j,k} = n_{i,k}.\left(p_{i,k} - m_{j,k}\right) \\ p_{i,k} = R_{nav}(p'_{i,k}) * \left(R * p'_{i,k} + T\right) + T_{nav} \\ m_{j,k} = R_{nav}(m'_{j,k}) * \left(R * m'_{j,k} + T\right) + T_{nav} \end{cases}$$

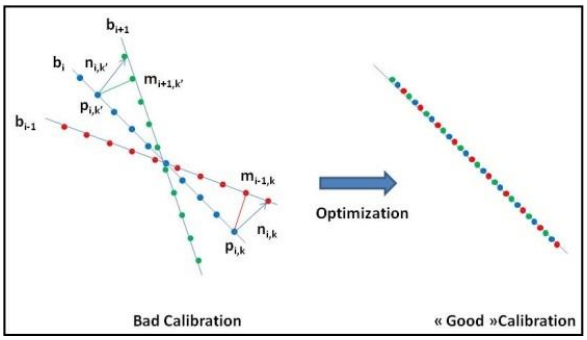


FIG. 2. Side-view of a planar surface

In Equation (1), the other terms are:

(a) B is a sample of the Velodyne sensor beams, with $B \subset [\![0\,;\,31]\!]$
(b) N is half the number of neighboring beams to beam i taken into account
(c) k iterates on a subset of the points of beam i
(d) $w_{i,j,k}$ is a weight whose value is between 0–1 depending on the local planarity – explained below, in the sub-section "Dimensionality of the points" – and on the distance between points $p_{i,k}$ and $m_{j,k}$.
(e) $n_{i,k}$ is the normal at point $p_{i,k}$ to the tangent plane to point $p_{i,k}$.
(f) $p_{i,k}$ and $m_{j,k}$. are respectively the k^{th} point of beam i, projected in the global reference frame and its nearest neighbor on beam j, also projected in the same reference frame.
(g) $p'_{i,k}$ and $m'_{j,k}$. are respectively the k^{th} point of beam i, projected in the sensor coordinate frame and its nearest neighbor on beam j, also projected in the same coordinate system.
(h) R_{nav} and T_{nav} are respectively the rotation matrix and translation vector from the navigation reference frame to the global reference frame. The matrix and vector depend on the time of the acquisition, and change from one point to another.
(i) The energy we defined has a relation to physics: indeed, the energy unit is length squared (m²), and the distance measured with the quantity $n_{i,k}.\left(p_{i,k} - m_{j,k}\right)$ gives an estimation of the noise between the two points, which is equal to 0 in the ideal case: Figure 2 illustrates this problem, where with a good calibration, and without noise, the energy should be close to 0. We suppose that this noise is independent for each point taken into account in the calculation of the energy J,

is centered, reduced and follows a normal distribution. With these hypotheses, the energy J follows a chi-squared distribution. Energy J gives an estimate of the variance σ^2 of the point cloud noise when the number of paired points taken into account is sufficiently large.

In this section, we first present the optimization of the extrinsic calibration parameters, followed by the intrinsic parameters. Finally, the optimization approach for all the calibration parameters is presented. For all the optimizations, the same energy function J is minimized.

Optimization of the calibration parameters

In Equation (1), R and T are respectively the rotation matrix and the translation vector from the sensor Cartesian reference frame to the navigation reference frame, which represents the extrinsic calibration of the sensor. R is represented with the three Euler angles, which are the calibration parameters of rotation roll, pitch and yaw, i.e. α, β and γ: this gives $R = R_z(\gamma) * R_y(\beta) * R_x(\alpha)$. For the translation, t_x, t_y and t_z represent the three translation parameters of the extrinsic calibration that we want to optimize. Thus, we have R(α, β, γ) and T(t_x, t_y, t_z).

For the intrinsic parameters' optimization, Figure 3 gives an illustration of the parameters that intervene in the acquisition of data by the lidar sensor. Indeed, the Velodyne HDL-32E is composed of 32 beams, which are placed on the same vertical plane. On Figure 3a, there is an example with two fibers. Fiber 15 is called the "reference": we choose it because its vertical angle is equal to zero. For each acquisition, the sensor gives the following information:

(a) The vertical angle ϕ_i of each beam, regarding the reference fiber.
(b) The distance $\rho_{i,k}$ between the origin of fiber i and the acquired point k.
(c) The horizontal angle θ_i, which is introduced by the motion of the sensor.

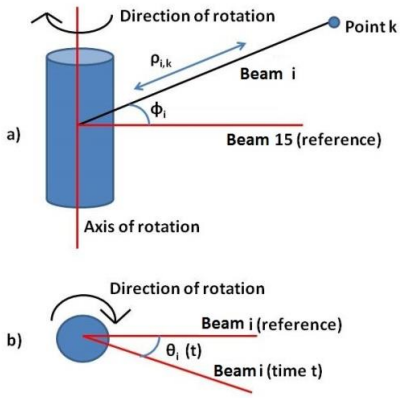


FIG. 3. Description of the intrinsic parameters for the Velodyne sensor

The intrinsic calibration of the Velodyne 32-beams can be represented with three equations that transform the spherical coordinates of each acquired point into Cartesian

coordinates. The three equations for a point p' acquired by a fiber i at time t are given by Equation (2):

$$p'_{i,k}(t) = \begin{pmatrix} x(t) \\ y(t) \\ z(t) \end{pmatrix} = \begin{pmatrix} \rho_i(k) * cos(\theta_i(t)) * cos(\phi_i) \\ -\rho_i(k) * sin(\theta_i(t)) * cos(\phi_i) \\ \rho_i(k) * sin(\phi_i) \end{pmatrix} \quad (2)$$

Linear approximation. The energy function we use penalizes points that are far from the local planar surfaces defined by points from the point cloud. If we have the optimum calibration parameters, the energy should be at a global minimum: this is not the case, because we start with initial parameters that are different from the optimal ones. We do not know how to solve energy J from Equation (1), because the energy is not linear due to the calibration parameters of rotation. We first change our problem by looking for small variations in the extrinsic parameters that reduce the energy J: starting from known parameters (t_x, t_y, t_z, α, β, γ), we look for small variations of the parameters (δt_x, δt_y, δt_z, $\delta\alpha$, $\delta\beta$, $\delta\gamma$), which give the following result: J(R($\alpha + \delta\alpha$, $\beta + \delta\beta$, $\gamma + \delta\gamma$), T($t_x + \delta t_x$, $t_y + \delta t_y$, $t_z + \delta t_z$)) < J(R(α, β, γ), T(t_x, t_y, t_z)).We replace the matrix R($\alpha + \delta\alpha$, $\beta + \delta\beta$, $\gamma + \delta\gamma$) by an approximation R(α, β, γ)+ $R_\alpha * \delta\alpha + R_\beta * \delta\beta + R_\gamma * \delta\gamma$, using the fact that the desired variations are supposed to be small.

We also want to correct the model for the intrinsic calibration presented in Equation (2); the model used by the constructor supposes that the sensor is perfect. We choose the following corrected model for each beam, which was presented in Chan and Lichti (2013), and we propose the following corrections for each beam:

(a) Between each beam, it is supposed that there is the same vertical angle separation. We add an offset $\delta\phi_i$ to each vertical angle ϕ_i to correct for the small errors that could exist.
(b) All the beams are supposedly placed on the same vertical plane. An error of alignment can exist, and we add an offset $\delta\theta_i$ to the horizontal angle θ_i.
(c) We add an offset $\delta\rho_{i,k}$ to the distance $\rho_{i,k}$ between the origin of beam i and the acquired point k.
(d) Finally, all the beams are supposed to have the same origin, which is not obvious. We add a small vertical offset to each beam, which takes into account small errors due to different origins.

All of the offsets that we add to Equation (2) give a new intrinsic transformation:

$$p'_{i,k}(t) = \begin{pmatrix} (\rho_i(k) + \delta\rho_i) * cos(\theta_i(t) + \delta\theta_i) * cos(\phi_i + \delta\phi_i) \\ -(\rho_i(k) + \delta\rho_i) * sin(\theta_i(t) + \delta\theta_i) * cos(\phi_i + \delta\phi_i) \\ (\rho_i(k) + \delta\rho_i) * sin(\phi_i + \delta\phi_i) + H_{z,i} \end{pmatrix}$$

We are looking for small offsets to correct the model given by the constructor: to simplify our problem, we perform a linearization at the first order of the intrinsic calibration equations, which gives the following transformation equations for point $p'_{i,k}$:

$$p'_{i,k}(t) = \begin{matrix} p'_{1,i,k}(t) + p'_{2,i,k}(t) * \delta\rho_i + p'_{3,i,k}(t) * \delta\theta_i \\ +p'_{4,i,k}(t) * \delta\phi_i + p'_{5,i}(t) \end{matrix} \quad (3)$$

where:

$$\begin{cases} p'_{1,i,k}(t) = \begin{pmatrix} \rho_i(k) * cos(\theta_i(t)) * cos(\phi_i) \\ -\rho_i(k) * sin(\theta_i(t)) * cos(\phi_i) \\ \rho_i(k) * sin(\phi_i) \end{pmatrix} \quad p'_{2,i,k}(t) = \begin{pmatrix} cos(\theta_i(t)) * cos(\phi_i) \\ sin(\theta_i(t)) * cos(\phi_i) \\ sin(\phi_i) \end{pmatrix} \\ p'_{3,i,k}(t) = \begin{pmatrix} \rho_i(k) * cos(\theta_i(t) + 90) * cos(\phi_i) \\ -\rho_i(k) * sin(\theta_i(t) + 90) * cos(\phi_i) \\ \rho_i(k) * sin(\phi_i) \end{pmatrix} \\ p'_{4,i,k}(t) = \begin{pmatrix} \rho_i(k) * cos(\theta_i(t)) * cos(\phi_i + 90) \\ -\rho_i(k) * sin(\theta_i(t)) * cos(\phi_i + 90) \\ \rho_i(k) * sin(\phi_i + 90) \end{pmatrix} \quad p'_{5,i}(t) = \begin{pmatrix} 0 \\ 0 \\ H_{z,i} \end{pmatrix} \end{cases}$$

As shown in Figure 3, we take one of the beams as a reference (i.e. beam 15), so we can reduce the degrees of freedom of the system and optimize the calibration parameters.

With the linear approximations we make on some elements of Equation (1), we change our minimization problem of a non-linear energy into that of a linear energy; we can re-write our energy into the following way:

$$J(\delta X) = \frac{\sum_{i=1}^{B} \sum_{j=i-N}^{i+N} \sum_k w_{i,j,k} * [D_{i,j,k} + C_{i,j,k}^T * \delta X + o(\delta X)]^2}{\sum_{i=1}^{B} \sum_{j=i-N}^{i+N} \sum_k w_{i,j,k}} \tag{4}$$

where:

$$\begin{cases} \delta X = \left(\begin{bmatrix} \delta\rho_i & \delta\theta_i & \delta\phi_i & H_{z,i} \end{bmatrix}_{i \in [\![0\,;\,31]\!] \setminus 15} \quad \delta t_x \quad \delta t_y \quad \delta t_z \quad \delta\alpha \quad \delta\beta \quad \delta\gamma \right)^T \\ D_{i,j,k} = n_{i,k}^T * \begin{pmatrix} T_{nav}\left(p'_{i,k}\right) - T_{nav}\left(m'_{j,k}\right) \\ + \left[R_{nav}\left(p'_{i,k}\right) * \left(R(\alpha,\beta,\gamma) * p'_{i,k} + T(t_x, t_y, t_z) \right) \right] \\ - \left[R_{nav}\left(m'_{j,k}\right) * \left(R(\alpha,\beta,\gamma) * m'_{j,k} + T(t_x, t_y, t_z) \right) \right] \end{pmatrix} \\ C_{i,j,k} = \begin{pmatrix} \dots \\ n_{i,k}^T * [R_{nav}(p'_{i,k}) * R(\alpha,\beta,\gamma) * p'_{2,i,k}] \\ n_{i,k}^T * [R_{nav}(p'_{i,k}) * R(\alpha,\beta,\gamma) * p'_{3,i,k}] \\ n_{i,k}^T * [R_{nav}(p'_{i,k}) * R(\alpha,\beta,\gamma) * p'_{4,i,k}] \\ n_{i,k}^T * (R_{nav}(p'_{i,k}) * R(\alpha,\beta,\gamma) * [0 \quad 0 \quad 1]^T) \\ \dots \\ n_{i,k}^T * [R_{nav}(m'_{j,k}) * R(\alpha,\beta,\gamma) * m'_{2,j,k}] \\ n_{i,k}^T * [R_{nav}(m'_{j,k}) * R(\alpha,\beta,\gamma) * m'_{3,j,k}] \\ n_{i,k}^T * [R_{nav}(m'_{j,k}) * R(\alpha,\beta,\gamma) * m'_{4,j,k}] \\ n_{i,k}^T * (R_{nav}(m'_{j,k}) * R(\alpha,\beta,\gamma) * [0 \quad 0 \quad 1]^T) \\ \dots \\ n_{i,k}^T * \left[R_{nav}(p'_{i,k}) - R_{nav}(m'_{j,k}) \right] * [1 \quad 0 \quad 0]^T \\ n_{i,k}^T * \left[R_{nav}(p'_{i,k}) - R_{nav}(m'_{j,k}) \right] * [0 \quad 1 \quad 0]^T \\ n_{i,k}^T * \left[R_{nav}(p'_{i,k}) - R_{nav}(m'_{j,k}) \right] * [0 \quad 0 \quad 1]^T \\ n_{i,k}^T * [R_{nav}(p'_{i,k}) * R_\alpha * p'_{i,k} - R_{nav}(m'_{j,k}) * R_\alpha * m'_{j,k}] \\ n_{i,k}^T * [R_{nav}(p'_{i,k}) * R_\beta * p'_{i,k} - R_{nav}(m'_{j,k}) * R_\beta * m'_{j,k}] \\ n_{i,k}^T * [R_{nav}(p'_{i,k}) * R_\gamma * p'_{i,k} - R_{nav}(m'_{j,k}) * R_\gamma * m'_{j,k}] \end{pmatrix} \end{cases}$$

The solution that minimizes the objective function (4) is the solution of the following linear system:

$$C * \delta X = -V \tag{5}$$

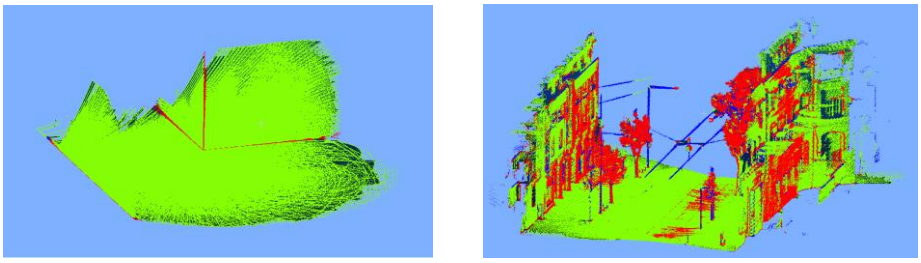
where:

$$\begin{cases} C = \sum_{i=1}^{B} \sum_{j=i-N}^{i+N} \sum_{k} w_{i,j,k} * C_{i,j,k} * C_{i,j,k}^{T} \\ V = \sum_{i=1}^{B} \sum_{j=i-N}^{i+N} \sum_{k} w_{i,j,k} * D_{i,j,k} * C_{i,j,k} \end{cases}$$

Dimensionality of the points

The dimensionality attributes are three values given by Equation (6). They are presented in Demantke et al. (2011) and can be written as:

$$\begin{cases} a_{1D} = \frac{\sigma_1 - \sigma_2}{\sigma_1} \\ a_{2D} = \frac{\sigma_2 - \sigma_3}{\sigma_1} \\ a_{3D} = \frac{\sigma_3}{\sigma_1} \end{cases} \tag{6}$$



(a) (b)

FIG. 4. Dimensionality computed for two point clouds: (a) simulated; (b) point cloud coming from a real acquisition. In blue, the dimensionality is 1; in green, the dimensionality is 2; in red, the dimensionality is 3

A Principal Component Analysis is carried out on the neighborhood to obtain the three best directions that describe the neighborhood. The corresponding eigenvalues are ordered in a decreasing order, from σ_1 to σ_3, where σ is the square root of the eigenvalue. The three attributes that are calculated for each point describe the nature of the surface to which the point belongs: if a_{1D} has the highest value, it means that the point is more likely to belong to a linear surface, such as a pole, and the dimensionality of the point would be 1; if it is a_{2D}, the point should belong to a planar surface, and the dimensionality of the point would be 2; finally, if it is a_{3D}, the point is more likely to belong to a volume surface, such as a tree, and the dimensionality of the point would be 3. Also, each parameter has a value between 0–1, and they are constructed so that their sum is equal to 1.

Figure 4 shows the repartition of the dimensionality attributes for a simulated and a real point cloud. For the real data, because the real point cloud possesses noise from different sources, there are points with a dimensionality of 3 on the ground or the façades.

What we call the "planarity value" at a point is the value of the dimensionality attribute a_{2D} calculated for this point, which gives information about the planarity of the surface to which the point belongs: the more planar the surface is, the closer to 1 is the value. When used, the planarity is introduced in the weights $w_{i,j,k}$ of each pair of points put in correspondance. Indeed, the weights are constructed as follows without the use of planarity:

$$w_{i,j,k} = \begin{cases} 1 & if \quad \|p_{i,k} - m_{j,k}\| < d_{max} \\ 0 & otherwise \end{cases}$$

With the use of the planarity, the weights become:

$$w_{i,j,k} = \begin{cases} max\left(a_{2D}(p),\, a_{2D}(m)\right) & if \quad \|p_{i,k} - m_{j,k}\| < d_{max} \\ 0 & otherwise \end{cases}$$

For the weights with the information of planarity, we tested the minimum value between the two attributes and also the mean value; there was no significant difference during our experiments, and the maximum gave slightly better optimization results.

Precision of the calibration parameters

In the state-of-the-art of the calibration of lidar sensors, we did not see values given to measure the precision of the optimized parameters: we generally compare our optimization result to a ground truth, which can be difficult and time-consuming, because the ground truth cannot be performed with an automatic process: human supervision is needed to validate the process. In our method, we give values to measure the precision of the parameters retrieved with our optimization process: we have an estimate of the precision for each parameter, given by the following terms:

$$\begin{cases} For\ the\ extrinsic\ parameters\ of\ translations: \sigma_x(m) = \sqrt{(C^{-1})_{124,124}} \\ For\ the\ extrinsic\ parameters\ of\ rotations: \sigma_\alpha(rad) = \sqrt{(C^{-1})_{127,127}} \end{cases} \quad (7)$$

In equation (7), we are presenting the precision formulas for the 6 extrinsic calibration parameters, and the subscripts 124 and 127 represent respectively the positions of the extrinsic parameters of translation along the X axis and the rotation parameter around the X axis in the covariance matrix C. The remaining precisions are defined in the same way as for the translation and rotation parameters. C is the matrix defined in Eq. (5), and its inverse $C^{-1} = Cov^{-1}(\delta X)$. With the square roots, we have the precisions of each parameter.

These precisions depend on the structure of the point cloud and the trajectory of the vehicle. For example:

(a) a trajectory with some changes, such as turns or a variation of altitude, will give a better precision for the calibration parameters.

(b) a trajectory that is straight, without changes, will not give a good precision for the parameters.

These parameters have the same order of magnitude whether we use the dimensionality attributes or not because we compute the precisions values after optimization, when the point cloud has already been refined.

Validity of the calibration results

We defined in the previous sections an energy function that should give better calibration parameters for our point clouds. We discuss in this section the conditions that validate the calibration obtained with our optimization process.

The value of the energy J should be small enough, under a threshold. As mentioned in the section "Definition of our energy function", our energy follows a chi-squared distribution. A validation threshold at 97% is $3\sigma^2$, with σ^2 the variance of the point cloud noise. For example, for real data, the noise comes from different sources; with our mobile mapping system, we have good precision, with a standard deviation for the noise of around 5 cm. This gives us a threshold of around 75 cm² for the value of energy J, in order to validate the calibration process.

We also define an error value for each category of intrinsic parameter, to characterize the difference between each offset of an intrinsic parameter and the associated ground truth when known. The error is the sum of the squares of the final offsets for the intrinsic parameters, which gives:

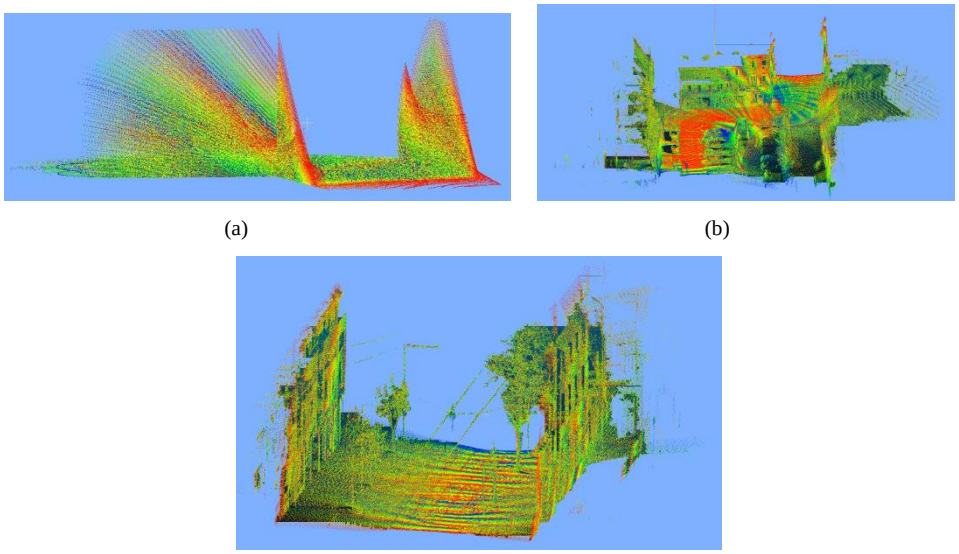
$$\begin{cases} \Delta\rho = \sqrt{\frac{1}{31} * \sum_{i=0/i\neq15}^{31} (\delta\rho_i)^2} \\ \Delta\theta = \sqrt{\frac{1}{31} * \sum_{i=0/i\neq15}^{31} (\delta\theta_i)^2} \\ \Delta\phi = \sqrt{\frac{1}{31} * \sum_{i=0/i\neq15}^{31} (\delta\phi_i)^2} \\ \Delta H_z = \sqrt{\frac{1}{31} * \sum_{i=0/i\neq15}^{31} (\delta H_z)^2} \end{cases} \quad (8)$$

In the ideal case, these errors should be close to 0 for each parameter. For a real point cloud, the errors that we are looking for should be small, which is also a validation criterion of the optimization result.

EXPERIMENTAL RESULTS

We now present the optimization results in detail for two point clouds: one is simulated, and the other comes from an acquisition in a real urban area. Another real point cloud is presented to confirm the optimization results. We tested the optimization on various simulated and real data, obtaining similar results.

Data used for the tests



(a) (b)

(c)

FIG. 5. Point clouds used for the tests: in the order, (a) simulated point cloud 1; (b) real point cloud 2; (c) real point cloud 3

The simulated data – point cloud 1 - is a point cloud that represents an acquisition in a urban area. The scene is made of planes: vertical – to represent walls and façades – and horizontal – the ground, representing the road. We use it to validate our algorithm. For the point cloud, the ground truth is known, and for the optimization, some error is added to each calibration parameter that we want to retrieve. The ground truth allows us to give the remaining error precisely for each parameter after the optimization. The simulated data is made of 5 million points, and the vehicle is making a turn; there is also a variation of altitude.

The real data are used to present our optimization results on data acquired in different environments:

(a) Point cloud 2 is part of an acquisition in the city of Montbeliard, in France. The point cloud contains some turns, several façades and a variation of altitude. The point cloud is made of 10 million points.

(b) Point cloud 3 is part of an acquisition in the city of Dijon, in France. There are façades and little variation of altitude, but no turns and some trees, along with electric cables for a tramway, which are non-planar elements. The point cloud is made of 5 million points.

Datasets

In our experiments, we use the same information for both simulated and real data. We have raw information from the sensor, which is composed of:

(a) the position and orientation of the vehicle at a frequency of 100 Hz. This is the position of the IMU in the global reference frame, fused with other information from proprioceptive sensors, such as the GPS and the odometer.
(b) the coordinates of each acquired point in the spherical coordinate system of the sensor reference frame.
(c) the "beam" that acquired each point, since we work with a multi-beam sensor.

These pieces of information give us the position of the vehicle and its trajectory with good precision: indeed, we only work on the calibration parameters of the acquisition system; to have a well reconstructed point cloud at the end of the optimization, we need to know the trajectory of the vehicle precisely.

Implementation and algorithm parameters

Our algorithm was implemented in C++. The EIGEN library (Eigen, 2015) was used for all the operations on matrices or vectors, and the FLANN library (Flann, 2015) – Fast Library for Approximated Nearest Neighbor – was used for the nearest neighbor search. The different algorithms run on a computer with Windows 7: 64 bits OS, 32 GB of RAM and four Intel core-i7 processors, with clocks up to 2.80 GHz.

Our algorithm was tested with synthetic and real urban data: for the synthetic data, the parameters were known precisely, giving us a ground truth with which to compare the optimization results. For the real data, the calibration parameters were approximately known, and we did not have any ground truth.

In our algorithm, we have some parameters to set. We start with sub-sampling the data about one point out of three, because the point clouds have a high resolution and it reduces the computation times and the use of memory, without changing the results: some tests have been performed with a full resolution point cloud, and the final optimization results are the same, what changes is the number of iterations performed (less iterations with a higher resolution, but the optimization is longer). The number of neighboring beams for a beam b_i was fixed to four (N=2). Concerning the weights $w_{i,j,k}$, a threshold of 20 cm was chosen for the maximum distance d_{max} between a point $p_{i,k}$ and its nearest neighbor $m_{j,k}$ on the neighboring beam. We also tested our algorithm by taking into account the planar dimensionality of the points: if the distance condition was satisfied, the weight took as a value the highest one between the dimensionality of the two points. To calculate the dimensionality attributes, a neighborhood size of 200 was taken for the Principal Component Analysis, because we have a dense point cloud with the Velodyne sensor.

The parameters were fixed for all the tests performed: different values were tested, but those presented give both good optimization results and computation times. With the simulated data, we tested several values for the optimization parameters, and the observations were that we had the best compromise between the convergence speed and the final errors in comparison to the ground truth with the values presented.

Optimization of the extrinsic parameters

In this section, we present some results only for the optimization of the extrinsic parameters. A comparison is made between two optimizations: one with the use of dimensionality attributes and the other without. The optimization of the extrinsic calibration parameters without the use of the dimensionality attributes has already been discussed (Nouira et al., 2015). In this section, we want to prove that the use of the dimensionality attributes in the weights of our energy we minimize gives better optimization results. For the tests, the dimensionality attributes are updated every seven iterations to reduce the computation time. The optimization results are similar to those obtained with the update at each iteration; fewer iterations are performed, but the computational time is higher.

Results on simulated data

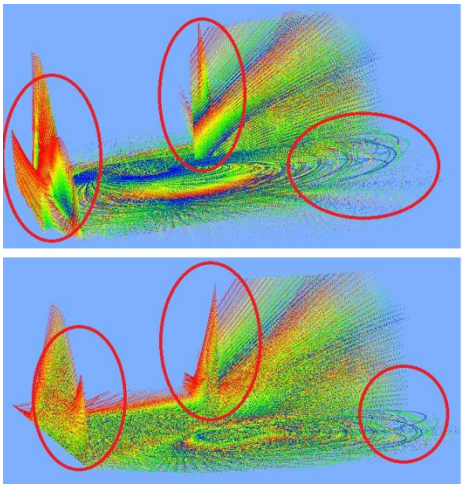


FIG. 6. Synthetic point cloud 1: on top, with the initial calibration; at the bottom, with our optimization. The two images have the same point of view

The first test was performed on point cloud 1. There is also a variation of altitude, and the vehicle is making a turn. Figure 6 presents the same point cloud, with the same point of view, but with two different calibrations. From Table I, we can see that we have parameters closer to the ground truth with the two optimizations, but we have the smallest global errors using the dimensionality attributes. Figure 7 gives the evolution of our energies during the optimization process: without the dimensionality attributes, the energy starts from a value of 243.07 cm², and reaches a value of 0.58 cm²; with the dimensionality attributes, the energy goes from a value of 86.63 cm² to 0.46 cm². The energy is always smaller using dimensionality attributes, and reaches its minimum faster. We also give on the figure the total weight with which the energy is normalized: with the dimensionality attributes, we see that the total weight considerably increases at each dimensionality attribute update, which shows that we are refining the point cloud.

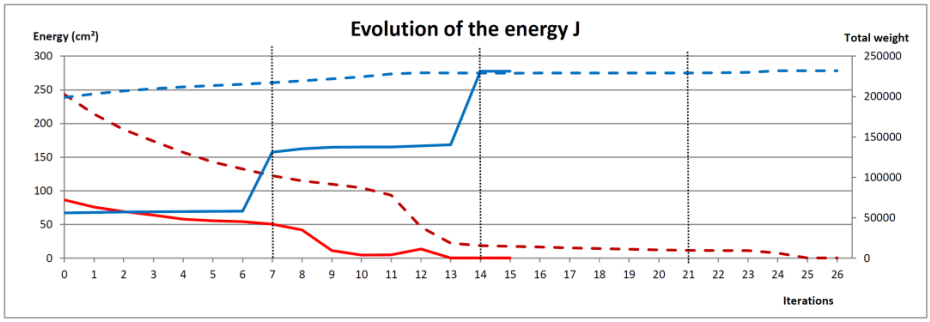


FIG. 7. Evolution of the energy for synthetic point cloud 1 with the optimization of the extrinsic parameters. The solid red line represents the evolution of the energy using the dimensionality attributes, the dashed red line without; the straight blue line represents the total weight used to normalize our energy using the dimensionality attributes, the dashed blue line without; finally, the vertical dotted lines show the iterations where the dimensionality attributes are updated

TABLE I. Final errors of the optimized extrinsic parameters of synthetic point cloud 1, after the optimization of the extrinsic calibration parameters

	t_x (cm)	t_y (cm)	t_z (cm)	α (°)	β (°)	γ (°)
Initial Extrinsic Calibration (ground truth)	*-200.00*	*240.00*	*-150.00*	*5.00*	*-37.00*	*-5.50*
Difference between the ground truth and our optimization (without dimensionality)	*-0.003*	*-0.016*	*0.033*	*-0.000*	*0.000*	*0.001*
Difference between the ground truth and our optimization (with dimensionality)	*-0.001*	*-0.023*	*0.008*	*-0.000*	*0.000*	*0.000*

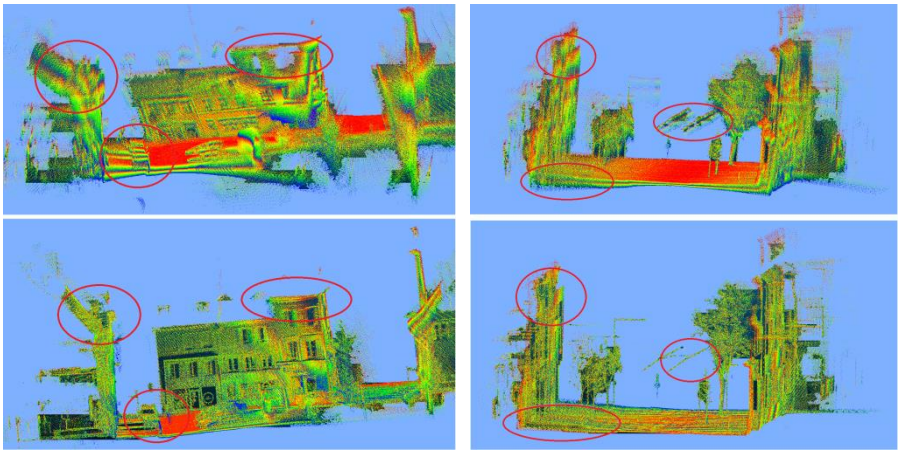
TABLE II. Precision of the parameters for synthetic point cloud 1

	σ_{t_x}(cm)	σ_{t_y} (cm)	σ_{t_z} (cm)	σ_α (°)	σ_β (°)	σ_γ (°)
Synthetic point cloud 1	*1.00*	*2.18*	*3.43*	*0.06*	*0.04*	*0.07*

We can see that our optimization gives good results, since the final energy is small. Table II gives the precision of our calibration parameters retrieved with our optimization for point cloud 1. As we could expect, we obtain good precision for all the parameters, which complements the observations made in Table I: the difference between the ground truth and the parameters retrieved with our optimizations is small.

We also present the computational time of our method for the total number of iterations: our optimization took about 75 s to give the result without the use of dimensionality attributes, and about 5 minutes with: the computational time is longer because of the calculation of the dimensionality attributes, but we have improved the optimization results, and it is still reasonable. Also, this computational time can be reduced by reducing the number of neighbors taken into account in the computation of the dimensionality, depending on the desired precision for the attributes.

Results on real urban data



(a) (b)

FIG. 8. Real point clouds 2(a) and 3(b): on top, before optimization of the extrinsic parameters; at the bottom, after optimization. The two images have the same point of view

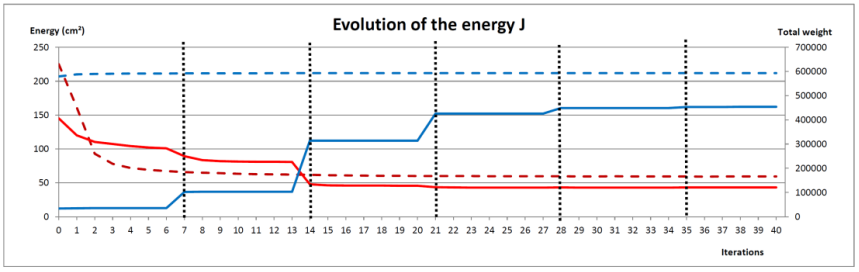


FIG. 9. Evolution of the energy for real point cloud 2 with the optimization of the extrinsic parameters.

TABLE III. Final results for the extrinsic calibration parameters of real point cloud 2, with the optimization of the extrinsic parameters

	t_x (cm)	t_y (cm)	t_z (cm)	α (°)	β (°)	γ (°)
Initial Calibration	*0.00*	*0.00*	*0.00*	*0.00*	*-45.00*	*90.00*
Hand-Measured Calibration	*-0.21*	*-1.22*	*0.95*	*0.00*	*-60.00*	*90.00*
Calibration after optimization (without dimensionality)	*-0.46*	*-1.56*	*-7.21*	*4.49*	*-60.06*	*89.59*
Calibration after optimization (with dimensionality)	*-0.48*	*-1.37*	*-7.60*	*-0.27*	*-59.84*	*93.74*

TABLE IV. Precision of the parameters for real point cloud 2

	σ_{t_x}(cm)	σ_{t_y} (cm)	σ_{t_z} (cm)	σ_α (°)	σ_β (°)	σ_γ (°)
Real point cloud 2	*7.77*	*3.10*	*157.66*	*0.50*	*0.33*	*0.47*

The first real point cloud is point cloud 2. In the upper panel of Figure 8(a), we have the point cloud with a bad calibration, and without assumptions on the calibration parameters, we find optimized parameters that improve the structure of the point cloud, as shown at the bottom of the figure.

As for the simulated data, we present a comparison between two optimizations: without the use of dimensionality attributes on the one hand, and with the attributes on the other hand. Figure 9 gives the evolution of the energies, with and without the dimensionality attributes: without, the energy goes from a value of 224.81 cm² to of 59.26 cm². With the dimensionality attributes, the energy starts from a value of 145.11 cm² and reaches 43.26 cm², which is smaller. If we consider a standard deviation for the point cloud noise of 5 cm, we have a threshold for the energy of 75 cm², and both final energies are under this threshold. We see that the total weight with the use of dimensionality does not end close to that without dimensionality: it is smaller, because we are in an urban area and the point cloud contains some noise. Also, the point cloud has some non-planar objects, for which the planarity values are small. This time, there are many iterations for which the energy does not change much. The parameters are still modified, but because of the precisions of the parameters, those changes do not modify the energy: this is why we stopped the optimizations at 40 iterations. Table III gives the parameters after optimization for point cloud 2. We have:

(a) the starting parameters, which are chosen arbitrarily;
(b) the hand-measured parameters. These parameters are not precisely measured with a calibration target, and are used to get an idea of the values to which the optimized calibration parameters should be close;
(c) the two optimized parameters, using dimensionality attributes.

The optimized parameters between the two optimizations are close to each other, but are also close to the hand-measured ones: this is the expected result because the precisions are globally good. There is still an important difference for the z parameter of translation, since there is a bad observability for this parameter: for the z parameter of translation, we have a final error of some meters, which could be predictable because the precision is on the order of meters. The precisions of the extrinsic parameters for point cloud 2 are given in Table IV. The precisions are good for most of the parameters, since we have turns in the point cloud. For the z parameter of translation, the precision is bad; this is unsurprising because there is no significant variation of altitude in an urban environment. The energy is still small and the point cloud well structured at the end of the optimization, because the precision in the Z direction is bad and any variation in this direction does not change the energy.

Finally, the computational times are respectively 2 minutes without the use of dimensionality attributes and 7 minutes with, which are acceptable times for a point cloud with 10 million points.

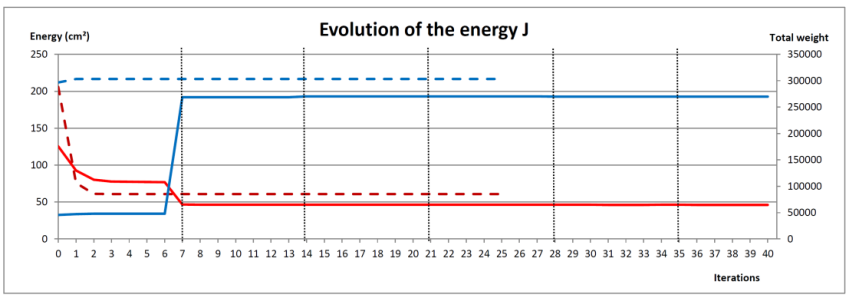


FIG. 10. Evolution of the energy for real point cloud 3 with the optimization of the extrinsic parameters.

The second real point cloud is point cloud 3. We have the same observations as for point cloud 2, i.e. the optimization with dimensionality attributes gives better results, since the final energy on Figure 10 is smaller. Also, the computational times are similar to those presented for point cloud 2.

Optimization of the extrinsic and intrinsic calibration parameters

In the previous section, we showed that optimization with dimensionality attributes gives better optimization results. In this section, we will present the results of the optimization of all the calibration parameters, extrinsic and intrinsic, as presented in the section "Proposed optimization method". We make a comparison with the optimization of the extrinsic parameters only.

Results of simulated data. For the simulated data, and because the calibration parameters are "perfect" – this is the way the simulated data are constructed – we added errors to the calibration parameters, and compared the results between the optimization of the extrinsic calibration parameters only, and the optimization of all the calibration parameters. The errors added to the intrinsic parameters were between -0.3–0.3° for the angle parameters (ϕ and θ), and between -3–3 cm for the distance (ρ and H_z). As for the errors added to the extrinsic parameters, they were around 1 m for the translation parameters, and 2–3° for the rotation parameters.

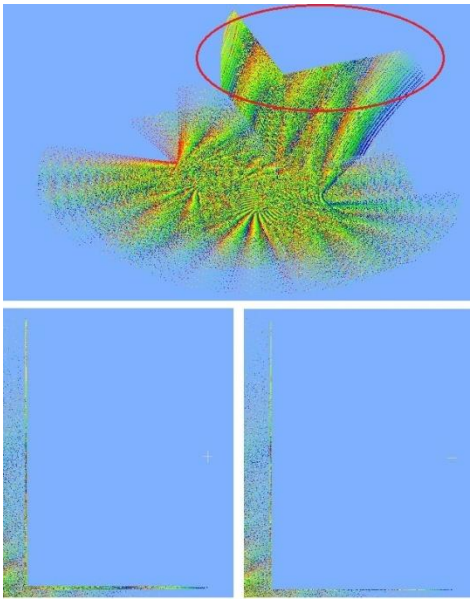


FIG. 11. Synthetic point cloud 1: on top, the point cloud with good extrinsic and intrinsic calibration parameters; at the bottom left, the point cloud with a corrected extrinsic calibration and bad intrinsic parameters; at the bottom right, the same point cloud but with corrected extrinsic and intrinsic parameters. The two images at the bottom have the same point of view.

For simulated point cloud 1, Figure 12 gives the evolution of the energies when we optimize the extrinsic calibration parameters only, and when we optimize all the calibration parameters, both with bad intrinsic and extrinsic parameters to start with. With the optimization of the extrinsic parameters only, the energy goes from a value of 261.69 cm² to 178.48 cm², and with the optimization of the complete calibration parameters, the energy goes from the same value 261.69 cm² to 0.59 cm². The energy is significantly smaller at the end of the optimization when we optimize all the calibration parameters. From Figure 11, at the bottom left, we can see some problem with the structure of the point cloud when the intrinsic parameters are not optimized, with a plane that is not completely planar. With the optimization of all the calibration parameters, the structure of the point cloud is improved, as shown at the bottom right. As expected, the optimization of all the calibration parameters gives better results, with a better-structured point cloud, which complements the energy difference.

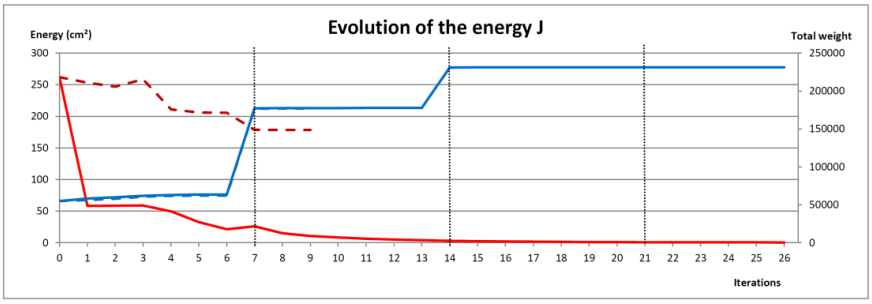


FIG. 12. Evolution of the energy of synthetic point cloud 1. The solid red line represents the evolution of the energy when both the extrinsic and intrinsic calibration parameters are optimized, and the dashed red line when only the extrinsic parameters are optimized; the solid blue line represents the total weight used to normalize our energy, with the optimization of both calibration parameters, and the dashed blue line with the optimization of the extrinsic parameters only; finally, the vertical dotted lines show the iterations where the dimensionality attributes are updated

TABLE V. Final errors of the optimized extrinsic parameters of synthetic point cloud 1, with the optimization of both intrinsic and extrinsic calibration parameters

	t_x (cm)	t_y (cm)	t_z (cm)	α (°)	β (°)	γ (°)
Initial Calibration	*-100.00*	*50.00*	*-30.00*	*2.50*	*-27.00*	*-2.00*
Difference between the ground truth and our optimization (extrinsic parameters optimization only)	*-0.181*	*-0.742*	*-0.463*	*-0.201*	*0.014*	*0.220*
Difference between the ground truth and our optimization (extrinsic and intrinsic parameters optimization)	*-0.682*	*-0.046*	*1.116*	*-0.008*	*0.006*	*0.039*

TABLE VI. Final errors of the optimized intrinsic parameters of synthetic point cloud 1, with the optimization of both intrinsic and extrinsic calibration parameters

	$\Delta\rho$(cm)	$\Delta\theta$ (°)	$\Delta\phi$ (°)	ΔH_z(cm)
Initial errors	*2*	*0.2*	*0.3*	*3*
Difference between the ground truth and our optimization (extrinsic and intrinsic parameters optimization)	*0.11*	*0.036*	*0.018*	*0.70*

We also present similar results for point cloud 1. Table V gives the remaining errors for the extrinsic parameters for both optimizations: for the rotation parameters, we have smaller remaining errors when all the calibration parameters are optimized. Regarding the intrinsic calibration parameters, for the optimization of the complete calibration, the remaining errors are between -0.07–0.01° for the rotations (ϕ and θ), and between -2–0 cm for the translation parameters (ρ and H_z). The errors have been reduced with the complete optimization, which is what we aimed for. Table VI gives the global error for the intrinsic parameters we defined in the section "Validity of the calibration". For the optimization of the extrinsic parameters only, the errors for each set of parameters do not change, since we do not correct the intrinsic parameters. On the other hand, when we optimize all the calibration parameters, the errors are significantly reduced: this explains the results of Table V, where there are no real improvements for the optimized extrinsic parameters with the complete optimization. Globally, with the optimization of all the calibration parameters, we significantly reduce the errors of the parameters, and the energy is also the smallest.

The computational time for the optimization of all the calibration parameters is about 20 minutes, instead of 5 minutes for the extrinsic parameters only: this is longer than the optimization of the extrinsic parameters, but still acceptable, regarding the important number of parameters optimized and the improvements made to the point cloud.

Results on real urban data

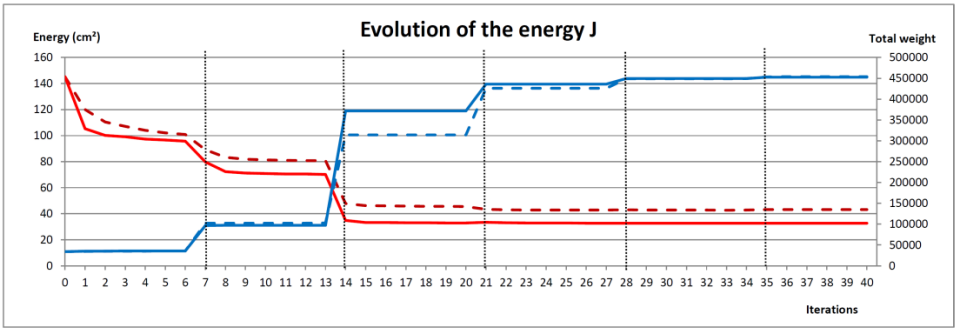


FIG. 13. Evolution of the energy of real point cloud 2

TABLE VII. Final results for the extrinsic calibration parameters of real point cloud 2, with the optimization of both intrinsic and extrinsic calibration parameters

	t_x (m)	t_y (m)	t_z (m)	α (°)	β (°)	γ (°)
Initial Calibration	*0.00*	*0.00*	*0.00*	*0.00*	*-45.00*	*90.00*
Hand-Measured Calibration	*-0.21*	*-1.22*	*0.95*	*0.00*	*-60.00*	*90.00*
Calibration after optimization (extrinsic parameters optimization only)	*-0.48*	*-1.37*	*-7.60*	*-0.27*	*-59.84*	*93.74*
Calibration after optimization (extrinsic and intrinsic optimization)	*-0.52*	*-1.39*	*-6.26*	*2.74*	*-57.31*	*90.99*

We first carried out the complete calibration parameters optimization on point cloud 2; we have visually the same results as those presented in Figure 8(a). If we take a look at the energies presented in Figure 13, they start at 145.11 cm² and reaches a value of 43.26

cm² when only the extrinsic calibration parameters are optimized, and 32.70 cm² when all the calibration parameters are optimized. With the energy variation, we can see that with the optimization of all the calibration parameters, we have a smaller energy that shows some improvements in the consistency of the point cloud. The final total weight with the complete optimization is nearly the same as that with the optimization of the extrinsic parameters: this shows that the consistency of the point cloud is close between the two optimizations, and that there is visually no difference on the structure of the point cloud. Still, there are some differences between the extrinsic parameters optimized with the two approaches: Table VII gives these differences between the optimized extrinsic parameters and those measured for point cloud 3, when the optimization was performed on all the calibration parameters of our acquisition system. The extrinsic parameters obtained with the two optimizations are close to each other, except for the extrinsic parameter of translation z because of the problem of observability in this direction. Regarding the intrinsic parameters, the starting error value was 0 for each parameter: we did not make any assumptions about its starting value and took the constructor default intrinsic calibration, which we want to improve slightly. At the end of the optimization, we obtained offsets for the intrinsic parameters between -22–20 cm for the translation offsets H_z; between -1–2 cm for the offsets ρ, and between -0.30–0.80° for the rotation offsets (ϕ and θ). These offsets are small and coherent, except for the offsets H_z, which are too high: we can explain this result with the observability in the z direction. As for the extrinsic translation parameter z, there is a bad observability of the offset H_z, and therefore important errors in the parameters; still, the structure of the point cloud is unchanged, and the energy value is even reduced with the complete calibration compared to the extrinsic parameters optimization.

Finally, the computational times are longer when we optimize all the calibration parameters: it took about 35 minutes, instead of 7 minutes for the optimization of the extrinsic parameters. Still, the computational time is acceptable regarding the important number of parameters optimized.

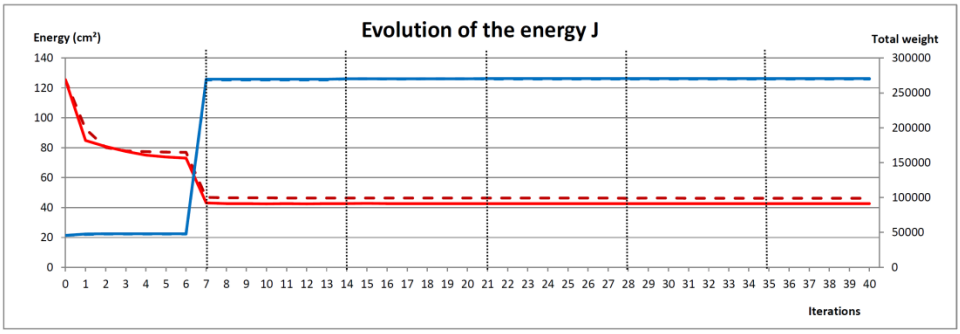


FIG. 14. Evolution of the energy of real point cloud 3

For point cloud 3, we visually have the same results as those presented in Figure 8(b). Figure 14 gives the evolution of the energy for the optimization with the dimensionality attributes, starting at 125.50 cm² and ending at 46.22 cm². With the complete calibration, the energy reaches a value of 42.61 cm². As for point cloud 2, the final energy is smaller

than that obtained with the optimization of the extrinsic calibration parameters only. However, with this point cloud, the difference between the two optimizations is small, because of the bad observability of the translation parameters. Still, we note that it does not affect the structure of the point cloud: the energy is the smallest when we optimize all the calibration parameters and when we take into account the dimensionality attributes of the points. Also, we can notice some high differences between the corrected intrinsic parameters for the two datasets: the same sensor was used, and the big differences come from both the difference of trajectory and the differences in the structure of the environment:

- ➔ For the first real dataset, the vehicle is doing some turns, and the environment is made of orthogonal facades.
- ➔ For the second dataset, the vehicle does not turn, and the facades are parallel to each other.

The differences in the estimated intrinsic parameters come from these differences, where for the second point cloud, the calibration parameters are less observable.

The computational time is about 25 minutes for the optimization of all the calibration parameters, instead of around 6 minutes for the optimization of the extrinsic parameters only.

CONCLUSIONS

We have presented a novel method for performing the automatic optimization of the calibration parameters of a terrestrial lidar system, in a post-processing application. We optimize two types of calibration parameters: the intrinsic parameters of a multi-beam sensor, and the extrinsic parameters which describe the transformation between the laser sensor reference and the vehicle reference frame. The intrinsic calibration parameters are specific to a sensor, as we show in this article by constructing a specific corrected model for the Velodyne HDL-32E: still, by adapting the model to the used lidar sensor, one can apply our method to optimize the intrinsic parameters of the sensor. Concerning the extrinsic parameters, they are independent from the mobile vehicle and the lidar sensor: one can directly apply our optimisation method to optimize the extrinsic calibration parameters.

Our optimization can be applied to any multi-beam lidar sensor configuration, as long as there are overlapping data between the beams: as we show in this paper, our optimization method is based on the registration of data coming from different beams of the sensor. The optimization process gives good results: for the datasets we presented, we get calibration parameters close to the ground truth when available, and a well-structured point cloud in general, where the global noise is reduced.

The results show that all the calibration parameters cannot be always well retrieved, as we show with the real datasets where the optimized parameters are far from the ones hand-measured: this is not a ground truth, but we know that we should be close to these parameters. This problem comes from the observabilities of the parameters we optimize: indeed, depending on the trajectory of the vehicle, some parameters will be well observable or not. The observability of a parameter is the capacity of a numeric system to detect the changes of this parameter. If we consider a mobile acquisition, we have 6 degrees of freedom: 3 translations, and 3 rotations. In general, the rotations are well

observable, except the rotation around the axis which is orthogonal to the ground. The translations are less observable, and the elevation is the parameters which is the less observable. To be able to retrieve well optimized parameters, the trajectory of the vehicle should be composed of some altitude variations and some turns. The precision we presented for the experimental results are linked to the observabilities of the parameters: the less precise a parameter is, the less observable it is, and the results we presented confirm the observations we made.

Because of the observabilities of the parameters, it is hard to correctly retrieve the "true" calibration parameters through optimization: the value of a parameter with a bad observability will not significantly change the structure of a point cloud, and any value would provide the same structure. With our optimization method, we are able to correctly refine a point cloud, whatever the observabilities of the parameters are.

REFERENCES

BESL, P. J., MCKAY, N. D., 1992. A method for registration of 3d shapes. *Transactions on Pattern Analysis and Machine Intelligence, IEEE, Vol. 14, pp. 239–256.*

CHAN, T. O., LICHTI, D. D., 2013. Feature-based self-calibration of velodyne hdl- 32e lidar for terrestrial mobile mapping applications. *The 8th International Symposium on Mobile Mapping Technology, Tainan, Taiwan.*

CRACIUN, D., DESCHAUD, J.-E., GOULETTE, F., 2017. Automatic Ground Surface Reconstruction from mobile laser systems for driving simulation engines. *Published in Simulation, vol.93 (3), pp 201—211.*

DEMANTKE, J., MALLET, C., DAVID, N., VALLET, B., 2011. Dimensionality based scale selection in 3d lidar point clouds, *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*

EIGEN LIBRARY, http://eigen.tuxfamily.org/index.php?title= Main_Page, (Accessed 15th November 2015)

ELSEBERG, J., BORRMANN, D., NUCHTER, A., 2013. Algorithmic solutions for computing precise maximum likelihood 3d point clouds from mobile laser scanning platforms, *Remote sensing, Vol. 5(11), pp. 5871–5906.*

FLANN LIBRARY, 2015. http://www.cs.ubc.ca/research/flann, (Accessed 12th February 2017)

GEIGER, A., LENZ, P., URTASUN, R., 2012. Are we ready for autonomous driving? The KITTI vision benchmark suite.,*published in IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*

GLENNIE, C., LICHTI, D. D., 2010. Static calibration and analysis of the velodyne hdl-64e s2 for high accuracy mobile scanning. *Remote Sensing, Vol. 2(6), pp. 1610–1624.*

GOULETTE, F, NASHASHIBI, F, ABUHADROUS, I, AMMOUN, S, LAURGEAU, C., 2006. An integrated on-board laser range sensing system for on-the-way city and road modelling, *in the International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences, vol. 34.*

GRAND DARPA CHALLENGE, 2007. http://en.wikipedia.org/wiki/DARPA_ Grand_Challenge, (Accessed 12th February 2017)

HUANG, L., BARTH, M., 2009. A novel multi-planar lidar and computer vision calibration procedure using 2d patterns for automated navigation. *Intelligent Vehicles Symposium, Xi'an, China, IEEE.*

HUANG, P.-S., HONG, W.-B., CHIEN, H.-J., CHEN, C.-Y., 2013. Extrinsic Calibration of a multi-beam lidar System with improved Intrinsic Laser Parameters using V-Shaped Planes and Infrared Images, *IVMSP, Seoul, 2013.*

LEVINSON, J., THRUN, S., 2010. Unsupervised calibration for multi-beam lasers, *International Symposium on Experimental Robotics.*

MADDERN, A., HARRISON, A., NEWMAN, P., 2012. Lost in translation (and rotation): Rapid extrinsic calibration for 2d and 3d lidars. *International Conference on Robotics and Automation (ICRA).*

MUHAMMAD, N., LACROIX, S., 2010. Calibration of a rotating multi-beam lidar. *IEEE/RSJ International Conference on Intelligent Robots and Systems, Taipei, Taiwan.*

NARAYANA K. S, K., CHOI, S., GOULETTE, F., 2009. Localization for mobile mapping systems, experimental results, analysis and post-processing improvements. *6th International Symposium on Mobile Mapping Technology, Sao Paulo, Brazil.*

NOUIRA, H., DESCHAUD, J.E., GOULETTE, F., 2015, Target-free extrinsic calibration of a mobile multi-beam lidar

system, *Proceedings of the ISPRS Geo-spatial week, la Grande Motte, France*

NUCHTER, A., SURMANN, H., LINGEMANN, K., HERTZBERG, J., THRUN, S., 2004. 6D SLAM with an Application in Autonomous Mine Mapping. *Proceedings of the 2004 IEEE International Conference on Robotics and Automation, New Orleans, L.A. (April)*, pp. 1998–2003.

RIDENE, T., GOULETTE, F.， 2009． Registration of fixed-and-mobile-based terrestrial laser data sets with DSM， *IEEE International Symposium on Computational Intelligence in Robotics and Automation (CIRA), pp. 375--380*．

SERNA, A., MARCOTEGUI, B., 2014. Detection, segmentation and classification of 3d urban objects using mathematical morphology and supervised learning, *ISPRS Journal of Photogrammetry and Remote Sensing* vol. 93, pp. 243–255.

SERNA, A., MARCOTEGUI, B., GOULETTE, F., DESCHAUD, J.-E., 2014. Paris-rue-madame database: A 3D mobile laser scanner dataset for benchmarking urban detection, segmentation and classification methods, *ICPRAM 2014 - Proceedings of the 3rd International Conference on Pattern Recognition Applications and Methods, pp. 819-824*

TAREL, J.-P., CHARBONNIER, P., GOULETTE, F., DESCHAUD, J.-E., 2012. 3D road environment modeling applied to visibility mapping: An experimental comparison, *proceedings of the IEEE International Symposium on Distributed Simulation and Real-Time Applications*, *pp. 19-26.*

UNDERWOOD, James P., HILL, A., PEYNOT, T., SCHEDING, Steven J., 2010, Error modelling and calibration of exteroceptive sensors for accurate mapping applications, *published in the journal of Fied Robotics, pp 2-20, volume 27, issue 1.*

VELODYNE 32 BEAMS DATASHEET, 2017. http://velodynelidar.com/lidar/hdlproducts/97-0038d%20HDL-32E_datasheet.pdf (Accessed 12th February 2017)

VELODYNE WEBSITE, 2017. http://velodynelidar.com/products.html, (Accessed 12th February 2017)

YOO, H.-J., GOULETTE, F., SENPAUROCA, J., LEPERE, G., 2009. Simulation based comparative analysis for the design of laser terrestrial mobile mapping systems | [Simulação baseada numa análise comparativa para o projeto de um sistema movel de mapeamento terrestre a laser]. *Published in the Boletim de Ciencias Geodesicas, vol. 15 (5), pp. 839-854.*

ZHU, Z., LIU, J., 2013. Unsupervised extrinsic parameters calibration for multi- beam lidars. *Proceedings of the 2nd International Conference on Computer Science and Electronics Engineering, Paris, France*.

Résumé

Nous présentons dans cet article une nouvelle méthode pour affiner des données issues d'acquisitions par un système mobile équipé d'un lidar multi-couches en corrigeant les paramètres de calibrage du système. L'optimisation des paramètres de calibrage permet de corriger la structure des nuages de points, et nous présentons une méthode robuste à l'initialisation. Nous montrons aussi des résultats sur des données simulées et réelles, avec des temps de calculs raisonnables de quelques minutes, ainsi que des indices de confiance sur les paramètres optimisés.

Zusammenfassung

Die digitale Bildzuordnung hat seit den ersten analogen Ansätzen für die automatisierte ...

Resumen

La correspondencia de imágenes tiene una historia de más de 50 años, desde los primeros …

摘要

影像匹配技术在模拟摄影测量中首次应用开始，已经有50年的发展 …