GLOBALLY OPTIMAL POINT CLOUD REGISTRATION FOR ROBUST MOBILE MAPPING

D. Skuddis[1], N. Haala[1]*

[1] Institute for Photogrammetry, University of Stuttgart, Germany - (david.skuddis, norbert.haala)@ifp.uni-stuttgart.de

Commission II, WG II/3

KEY WORDS: Point Cloud Registration, Scan Matching, Branch and Bound, Global Optimization, Mobile Mapping, LiDAR.

ABSTRACT:

Point cloud registration algorithms have been studied for several decades. In the SLAM domain, dense local convergence based methods are typically used to register consecutive scans. Since these procedures are not globally optimal, it happens that they converge to a wrong local minimum. This can lead to gross errors during mapping and can make entire datasets unusable. We introduce a new branch and bound based point cloud registration method that is globally optimal. The method is able to reliably determine the global optimum within a given parameter search space. We show how this method can be used in a mapping system as a fallback function to correct gross errors. Using various public datasets, we demonstrate the capabilities of the method.

1. INTRODUCTION

Point cloud registration is a problem that has been researched for a long time. Various algorithms have been developed for a wide range of applications.
In this paper we restrict to the field of mobile mapping and SLAM (Simultaneous Localization and Mapping). We address methods that use point clouds and scan matching to create maps of the environment as SLAM systems.

1.1 Motivation

Current SLAM systems such as (Zhang and Singh, 2014), (Behley and Stachniss, 2018) or (Koide et al., 2021) usually work very reliably. Impressive accuracies are achieved in public benchmark datasets like (Geiger et al., 2012).
While in most cases very good results can be achieved with such methods, gross errors occur from time to time. Corresponding errors can lead to strange behavior in the mobile robot domain or make whole datasets unusable in the mapping domain. In our opinion, there are two main reasons for gross errors:

1. The movement of the sensor is not observable from the point clouds. This means that an objective functional used in point cloud registration does not have a global extremum at the true correct pose.

2. The global extremum of an objective functional corresponds to the true correct pose but could not be discovered during the point cloud registration.

In this work, we present a solution to correct errors that occur as a result of the second reason.

1.2 Related Work

Here we first give a brief overview of algorithms for point cloud registration and then we go into more detail about globally optimal methods. Last we present previous works in the field of robust mapping.

* Corresponding author

In the SLAM area, local methods are usually used for the registration of consecutive scans. Prominent dense local methods are ICP (Besl and McKay, 1992), NDT (Biber and Straßer, 2003), CPD (Myronenko and Song, 2010) and GICP (Segal et al., 2009). We consider the latter to be a state of the art method and use it in the following for comparisons as a representative of convergence-based methods.
While dense methods use a continuous representation of the environment, feature-based methods only match features. An example of a feature based local method is the one presented in LOAM (Zhang and Singh, 2014).
All of the above methods have in common that they are local. That means they work convergence-based or make assumptions that nearby points or features in the separate point clouds correspond and therefore determine a solution very quickly. The disadvantage is that finding the global optimum is not guaranteed what can lead to gross errors from time to time.
In addition to the methods mentioned above, global methods for point cloud registration also exist. Here, there are also methods that rely on keypoints (see (Tombari et al., 2013) for a comparison of different keypoints types) and feature correspondences such as (Yang et al., 2020). Since we do not want to limit our applications to the presence of special keypoints or edges, we only consider dense methods. Known dense globally optimal methods are the one used in Google Cartographer for closing loops (Hess et al., 2016), the Go-ICP (Yang et al., 2015) and GOGMA (Campbell and Petersson, 2016). All of these methods use a branch and bound based approach. Since (Hess et al., 2016) was developed for two-dimensional grid maps, it is not applicable to 3D point clouds. Due to the point-to-point metric, the Go-ICP algorithm is not ideally suited for sparse point clouds with highly varying point density, such as those generated by mobile laser scanners. The GOGMA method is partially suitable, but has limited accuracy due to the approximation of the point cloud by Gaussian distributions. In addition, the quality of the approximation of the point cloud by gaussian distributions can be influenced by strongly varying point densities.
In this paper a globally optimal method for point cloud registration is presented. The method is inspired by the Go-ICP (Yang et al., 2015) algorithm. In contrast to Go-ICP, the proposed method runs in 6D while using unnested optimal search. By

taking into account normal vectors (point-to-plane distance), the new method is suitable for sparse point clouds, such as those generated by mobile laser scanners. To minimize the influence of outliers, a different metric is used than in the original Go-ICP. The metric chosen is related to that of NDT (Biber and Straßer, 2003). Our approach specifically addresses the application with point clouds from mobile laser scanners.

Robust LiDAR-based odometry or mapping systems are becoming increasingly important. In order to increase robustness, approaches exist that apply several point cloud registration methods in parallel and use the most plausible result based on a criterion (Reinke et al., 2021). Another approach combines data from multi sensors and monitors the health (health monitoring) of the odometry result (Palieri et al., 2020).

We present a robust mapping system that can detect gross errors in the results of local point cloud registration based on model assumptions without additional sensor information. In case of an error, the transform is corrected using the presented global registration method.

2. GLOBALLY OPTIMAL POINT CLOUD REGISTRATION

Globally optimal means that the global optimum of an objective functional is determined. In our case, we are looking for a set of parameters that describes a rigid transformation. The rigid transformation represents how a point cloud must be shifted and rotated in order to be correctly placed relative to a second point cloud. The objective functional evaluates how accurately the current set of parameters aligns the point clouds. To determine the global optimum, we use a branch and bound framework. In the following chapters, our approach for global point cloud registration is presented. We call it *GO-P2Plane* (Globally Optimal Point-2-Plane Registration).

First, the used data representation is described in detail. Subsequently, the objective functional is explained. The choice of the objective functional is elementary to enable a reliable point cloud registration.

2.1 Data Representation

In the registration we distinguish between the base point cloud and the point cloud that we want to register relative to the base point cloud. While the latter is still represented as a point cloud, the base point cloud is converted into a model consisting of planes at the beginning of the registration. First, for each point of the base point cloud, a local normal vector is estimated based on its k nearest neighbor points. The base point cloud is then spherically projected. Similar to the range image representation of point clouds, the space is then divided into discrete quadrangles. We refer to the quadrangles below as patches. The center of each patch is defined by an elevation angle and an azimuth. The width and height of a planar patch is given by the desired angular resolution δ. For each quadrant, the point whose elevation angle and azimuth are closest to the center is determined. The plane defined by the point and its normal vector is then considered as a representation of the space, which is valid within a planar patch. The entirety of planes then serves as a continuous environment representation. In Fig. 1 the planar patches representation is outlined. Each planar patch contains a center point $\overrightarrow{m}_{r,c}$ and a normal vector $\overrightarrow{N}_{r,c}$. r and c are the respective row and column indices that represent the azimuth and elevation angle. The planar patches are stored in a data structure similar to a range image representation to quickly determine projective correspondences. The

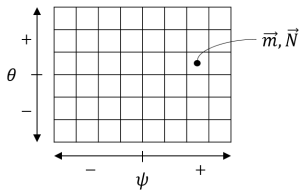

Figure 1. Planar patches representation.

division of the environment into discrete planar patches is elementary in the subsequent search for a global optimum. In Fig. 2 the planar patches representation is illustrated in 3D for an exemplary indoor scene.

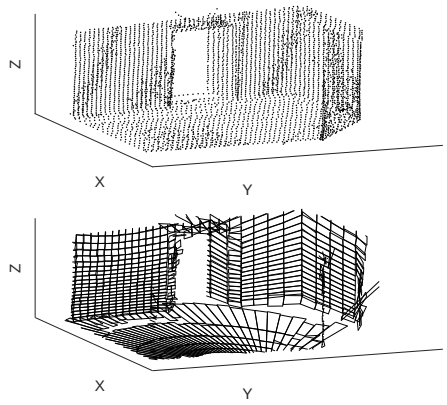

Figure 2. Transformation of a point cloud into planar patches. Top: Raw point cloud. Down: Planes of each patch.

2.2 Objective Functional

The objective functional is the function whose global maximum will be determined. The chosen objective functional consists of a weighted sum of the distances to the respective planar patches, normalized by the number of points (see equation (1)). The probability density function of the normal distribution is applied to weight the distances. The objective functional is related to the score presented in NDT (Biber and Straßer, 2003) but differs in that a constant value is assumed for the standard deviation and the point-to-plane error is used. The resulting score is a value in the range $[0, 1]$. Let P be a point cloud and n_p the number of points. Let $\overrightarrow{p}_i$ be a point from P and $\overrightarrow{m}_c$ the center point and $\overrightarrow{N}_c$ the normal vector of the corresponding planar patch for $\overrightarrow{p}_i$ of the base point cloud. For a rigid transformation defined by a translation $\overrightarrow{t}$ and a rotation R that is applied to P, the value of the objective functional $score(t, R)$, which is the mean of the probability density values of a normal distribution for all point-to-plane errors, is obtained as follows:

$$score(\overrightarrow{t}, R) = \frac{1}{n_p} \cdot \sum_{i=1}^{n_p} \exp(-\frac{e_i^2}{2\sigma^2}) \tag{1}$$

e_i represents the point-to-plane error:

$$e_i = |((R \cdot (\overrightarrow{p}_i + \overrightarrow{t})) - \overrightarrow{m}_c)^T \cdot \overrightarrow{N}_c| \tag{2}$$

σ can be considered as a design parameter. The weight function is displayed for an exemplary $\sigma = 0.17$ in Fig. 3. Points whose distance to the neighboring point is small get a high weight and

points far away have no influence on the objective functional. The functional thus behaves similar as measuring points within a consensus set, but is still derivable for local refinement. With a suitable choice of σ, rough outliers have no direct influence on the result. A great advantage of the selected objective function is that no inlier ratio has to be selected in advance. During the optimization, the transformation is to be determined for which the score is maximum.

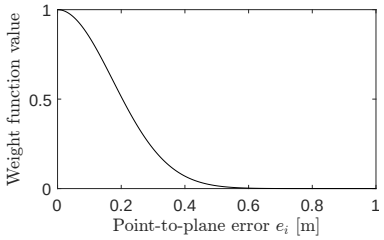

Figure 3. Scores weight function for the point-to-plane error.

2.2.1 Point Correspondence Within the approach points are projectively associated to a correspondent planar patch. This means that for each point for which a corresponding planar patch is required, the points angle of inclination and azimuth are first calculated. For a point $\overrightarrow{p} = [x, y, z]^T$ the angles can be calculated as follows:

$$\theta = \text{atan2}(z, \sqrt{x^2 + y^2}) \tag{3}$$

$$\psi = \text{atan2}(y, x) \tag{4}$$

The angles θ and ψ are then used to find the corresponding planar patch. This is done by converting the θ and ψ into rows and columns indices (r, c). With these indices the corresponding element can be selected in the planar patches representation. Figuratively, this type of correspondence means that it is determined which planar patch intersects the straight line defined by θ and ψ. The intersected planar patch is the correspondence searched for. This kind of correspondence was introduced by (Blais and Levine, 1995) and is also used in (Behley and Stachniss, 2018) to find corresponding surface elements during point cloud registration.

2.3 Branch and Bound

Branch and Bound (BnB) is a algorithm paradigm that is used for solving combinatorial optimization problems and mathematical optimization problems. In the BnB method, the solution space is searched successively by dividing it into subspaces. The evaluated subspaces result in a graph in the form of a root tree. For each subspace it is predicted which maximum value (in case of a maximization problem) the objective functional could have within the subspace. This predicted maximum value is called *upper bound*. The most promising subspace is then explored subsequently. In each exploration step the objective functional is evaluated in at least one location. In comparison to the predicted *upper bound*, this so called *lower bound* serves as a guess of the value range in the subspace. The *lower bounds* of the evaluated subspaces are used to determine the highest value $\overline{score}$ identified so far. The core idea of BnB is now to use the previous maximum of the *lower bounds* ($\overline{score}$) to remove explored subspaces whose *upper bound* is lower. This can be done since we know that the predicted maximum of the respective subspaces is smaller than our previous identified maximum value $\overline{score}$. More information and various examples of BnB can be found in (Clausen, 1999).
When designing a BnB based algorithm, a suitable approach that predicts the maximum value that the objective functional can assume within a subspace is required. This approach is called *bounding function*.

2.3.1 Domain Parametrization The search space consists of six dimensions. The first three dimensions correspond to the translation in x, y and z direction. The fourth to sixth dimension corresponds to the rotation. The rotation is parameterized by the axis-angle representation. The axis angle representation is a very compact way to describe a rotation. Rotations are therefore represented by a three elements vector $\overrightarrow{r} = [r_1, r_2, r_3]^T$. The angle of rotation is given by $||\overrightarrow{r}||$ and the normalized axis of a rotation by $\overrightarrow{r}/||\overrightarrow{r}||$. Any rotation can be described with the elements $r_1, r_2, r_3 \in [-\pi, \pi]$. Rotations defined in the axis-angle representation can be converted in a rotation matrix using Rodrigues' rotation formula. A rotation matrix, which is parameterized by an axis-angle representation vector $\overrightarrow{r}$, we write down with $R(\overrightarrow{r})$. In total, a vector with the following elements results: $\overrightarrow{V} = [x, y, z, r_1, r_2, r_3]^T$. A subspace is defined by its limits for each dimension. These are: $x \in [x_{min}, x_{max}]$, $y \in [y_{min}, y_{max}]$, $z \in [z_{min}, z_{max}]$, $r_1 \in [r_{1,min}, r_{1,max}]$, $r_2 \in [r_{2,min}, r_{2,max}]$, $r_3 \in [r_{3,min}, r_{3,max}]$.
In our 6 dimensional case, the exploration of a subspace results into $2^6 = 64$ subspaces. The extent of the resulting subspaces in each dimension is half of the original space. During recursive exploration, the search space is divided into subspaces in which the rotation components and the translation components respectively have identical edge lengths. Let a_t and a_r denote the translational and rotational edge lengths, we have the following definition:

$$a_t = x_{max} - x_{min} = y_{max} - y_{min} = z_{max} - z_{min} \tag{5}$$

$$a_r = r_{1,max} - r_{1,min} = r_{2,max} - r_{2,min} = r_{3,max} - r_{3,min} \tag{6}$$

If the rotation space and the translation space are considered separately, cube-shaped search areas result in each case.
A subspace can thus be completely described by a 6 dimensional position vector $\overrightarrow{V}$ pointing to the center of the subspace and by two edge lengths a_t and a_r.
To accelerate the search for a global optimum, the search space can be limited by a maximum rotation angle α_{max} and a maximum distance d_{max} between the origins of the point clouds. In this case, the search space is parametrized with $x, y, z \in [-d_{max}, d_{max}]$ and $r_1, r_2, r_3 \in [-\alpha_{max}, \alpha_{max}]$.

2.3.2 Bounding Function Derivation In *branch and bound*, an *upper bound* and a *lower bound* are used for optimization. The *lower bound* is the function value of an evaluation of the objective functional (usually in the middle of the subspace). When a new subspace is explored, a *lower bound* of it is calculated and the highest evaluated value $\overline{score}$ of all subspaces is updated.
The *upper bound function* predicts the maximum value that the objective functional can assume within a subspace. In the following we call the *upper bound* $\overline{B}$ and the *lower bound* $\underline{B}$. As described before, a subspace is defined by a position vector $\overrightarrow{V}$ to the center of the subspace and by the edge lengths a_t and a_r. Given a base point cloud in the planar patches representation as a set of planes defined by $\overrightarrow{N}_{r,c}, \overrightarrow{m}_{r,c}$ and a second point cloud P that is placed relative to it we calculate the upper bound as follows: Our approach to determine the

upper bound is done in two steps:
1. First, for each point in P all reachable corresponding planar patches are determined. Reachable means that for each point, the freedom of movement allowed by the extent of the subspace a_t and a_r is used to identify the set of planar patches that could potentially become a correspondence.
2. In a second step, for each possible planar patches correspondence, it is checked how small the point-to-plane error can become considering the degrees of freedom of the search space. For each point, the correspondence with the smallest error is selected and the score (see formula (1)) is calculated. This optimistic score represents the *upper bound*.

1. Identification of Correspondences
Let us first focus on the identification of possible planar patch correspondences. The identification of the reachable correspondences is performed sequentially for all points in P. $\overrightarrow{S}_r$ is the set of valid axis-angle representations and $\overrightarrow{S}_t$ the set of valid translation vectors within our subspace defined by $\overrightarrow{V}$, a_t and a_r. We define the different sets as follows:

$$\overrightarrow{S}_{ar} = \{ \begin{bmatrix} r_1 \\ r_2 \\ r_3 \end{bmatrix} \in \mathbb{R}^3 \mid r_1, r_2, r_3 \in [-a_r, a_r] \} \tag{7}$$

$$\overrightarrow{S}_r = \{ \overrightarrow{V}(4:6) + \overrightarrow{S}_{ar} \} \tag{8}$$

$$\overrightarrow{S}_{at} = \{ \begin{bmatrix} x \\ y \\ z \end{bmatrix} \in \mathbb{R}^3 \mid x, y, z \in [-a_t, a_t] \} \tag{9}$$

$$\overrightarrow{S}_t = \{ \overrightarrow{V}(1:3) + \overrightarrow{S}_{at} \} \tag{10}$$

Let $\overrightarrow{p}_i \in P$ be a point of the movable point cloud. Its possible movement resulting as a set of vectors $\overrightarrow{p}_m$ can be described by the following equation for a rigid transformation:

$$\overrightarrow{p}_m = R(\overrightarrow{S}_r) \cdot (\overrightarrow{p}_i + \overrightarrow{S}_t) \tag{11}$$

As (Yang et al., 2015) and (Campbell and Petersson, 2016) we resort to the theorem that for any vector $\overrightarrow{x}$ and two arbitrary axis angle respresentations $\overrightarrow{r}_1$ and $\overrightarrow{r}_2$ and their respective rotation matrix representations R_{r1} and R_{r2} holds:

$$\angle(R_{r1}x, R_{r2}x) \leq ||\overrightarrow{r}_1 - \overrightarrow{r}_2|| \tag{12}$$

Where $\angle(\overrightarrow{a}, \overrightarrow{b})$ represents the intersection angle of the two vectors $\overrightarrow{a}$ and $\overrightarrow{b}$. Next, we replace the rotation matrix with two separate rotation matrices using theorem (12):

$$R(\overrightarrow{S}_r) = R(\overrightarrow{S}_{ar}) \cdot R(\overrightarrow{V}(4:6)) \tag{13}$$

And we replace also the translation vector with two separate vectors:

$$\overrightarrow{S}_t = \overrightarrow{V}(1:3) + \overrightarrow{S}_{at} \tag{14}$$

If we substitute $R(\overrightarrow{S}_r)$ and $\overrightarrow{S}_t$ in equation (11) with (13) and (14) we obtain:

$$\overrightarrow{p}_m = R(\overrightarrow{S}_{ar}) \cdot R(\overrightarrow{V}(4:6)) \cdot (\overrightarrow{p}_i + \overrightarrow{V}(1:3) + \overrightarrow{S}_{at}) \tag{15}$$

Here $R(\overrightarrow{V}(4:6))$ is a rotation matrix defined by the angle axis representation in $\overrightarrow{V}$ at the center of the search space. $\overrightarrow{V}(1:3)$ is the translation defined at the center point $\overrightarrow{V}$ of a subspace. $R(\overrightarrow{S}_{ar})$ and $\overrightarrow{S}_{at}$ describe the remaining rotational and translational movement uncertainty quantified by the edge lengths a_r and a_t.
A correspondence of a point $\overrightarrow{p}_i$ to a planar patch exists if the elevation angle and the azimuth of the point are within the range of a planar patch. Thus, in order to determine the correspondences, it is necessary to determine what ranges of values the elevation angle and the azimuth of the point can assume considering the ability of movement enabled by the subspace.
For this, $\overrightarrow{p}_i$ is first transformed using the center parameters $\overrightarrow{V}$ of the search space:

$$\overrightarrow{p}_V = R(\overrightarrow{V}(4:6)) \cdot (\overrightarrow{p}_i + \overrightarrow{V}(1:3)) \tag{16}$$

If equation (15) is reshaped and (16) is inserted, the remaining range of movement around $\overrightarrow{p}_V$ can be described with:

$$\overrightarrow{p}_m = R(\overrightarrow{S}_{ar}) \cdot (\overrightarrow{p}_V + R(\overrightarrow{V}) \cdot \overrightarrow{S}_{at}) \tag{17}$$

Let θ_V and ψ_V be the elevation angle and the azimuth of $\overrightarrow{p}_V$. As can be seen in equation (17), there are two dependencies $R(\overrightarrow{S}_{ar})$ and $\overrightarrow{S}_{at}$ that can influence θ_V and ψ_V.
To determine the possible range of values for θ and ψ of $\overrightarrow{p}_m$, we first determine the maximum possible intersection angle between $\overrightarrow{p}_V$ and $\overrightarrow{p}_m$. The maximum distance by which $\overrightarrow{p}_V$ can be moved by $R(\overrightarrow{V}) \cdot \overrightarrow{S}_{at}$ is limited by the maximum length l_t of $\overrightarrow{S}_{at}$. The term $R(\overrightarrow{V})$ has no influence on the length and can therefore be neglected here.

$$\begin{aligned} l_t &= \text{argmax}(||\overrightarrow{S}_{at}|| = || \begin{bmatrix} x \\ y \\ z \end{bmatrix} || \text{ for } x, y, z \in [-\frac{a_t}{2}, \frac{a_t}{2}]) \\ &= \sqrt{3}\frac{a_t}{2} \end{aligned} \tag{18}$$

The maximum angular change between $\overrightarrow{p}_V$ and $\overrightarrow{p}_m$ using l_t can be approximated by considering l_t as part of the circumference of the circle with radius $||\overrightarrow{p}_v||$ around the point clouds origin. This gives a maximum angular change of:

$$\Delta\phi_t = 2\pi \cdot \frac{l_t}{2\pi \cdot ||\overrightarrow{p}_v||} = \frac{l_t}{||\overrightarrow{p}_v||} \tag{19}$$

The relationship is outlined in Fig. 4. If (18) is substituted

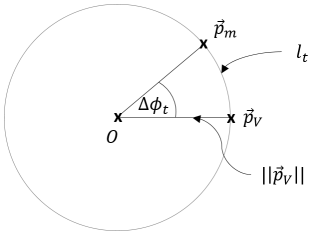

Figure 4. Derivation of $\Delta\phi_t$ from l_t.

into (19), the relation is obtained which describes the angle by which the direction of vector $\overrightarrow{p}_V$ can be influenced depending on the translation uncertainty a_t:

$$\Delta\phi_t = \frac{\sqrt{3}a_t}{2 \cdot ||\overrightarrow{p}_v||} \tag{20}$$

In addition to the translation term, the direction of $\overrightarrow{p}_V$ can also be changed by a rotation $R(\overrightarrow{S}_{ar})$.
As described in (12), the rotation angle of a point in space is $\leq$ as the vector norm of the axis angle representation. In our case, the maximum angle of rotation is:

$$\begin{aligned}\Delta\phi_r &= \text{argmax}(||\begin{bmatrix}r_1\\r_2\\r_3\end{bmatrix}|| \text{ for } r_1,r_2,r_3 \in [-\frac{a_r}{2},\frac{a_r}{2}]) \\ &= \sqrt{3}\frac{a_r}{2}\end{aligned} \tag{21}$$

The total possible rotation of a point $\overrightarrow{p}_m$ relative to $\overrightarrow{p}_V$ is thus:

$$\Delta\phi = \Delta\phi_t + \Delta\phi_r \tag{22}$$

To efficiently determine the possible range of values for θ and ψ of a point $\overrightarrow{p}_i$ within a subspace using the elevation angle θ_V and the azimuth ψ_V of the center point as well as $\Delta\phi$, we make the following approximation: Let $\overrightarrow{p}_1$ and $\overrightarrow{p}_2$ be two arbitrary vectors defined by their elevation angles θ_1, θ_2 and their azimuths ψ_1, ψ_2 with respective distances from the origin greater than zero. Then it is assumed that the following holds:

$$\angle(\overrightarrow{p}_1(\theta_1,\psi_1), \overrightarrow{p}_2(\theta_2,\psi_2)) \approx ||\begin{bmatrix}\theta_1\\\psi_1\end{bmatrix} - \begin{bmatrix}\theta_2\\\psi_2\end{bmatrix}|| \tag{23}$$

This simplification is approximately true as long as we are away from the poles $|\theta_1|, |\theta_2| << \pi/2$. Thus we get the set $C_{\theta,\psi}$ of all legal parameters for θ and ψ:

$$C_{\theta,\psi} = \{(\theta,\psi) \in \mathbb{R}^2 \mid \sqrt{(\theta_V-\theta)^2 + (\psi_V-\psi)^2} \leq \Delta\phi\} \tag{24}$$

This set $C_{\theta,\psi}$ will now be converted to the corresponding row and column indices (r, c) of our planar patches representation using the angular resolution δ of the planar patches:

$$C_{r,c} = \{(r,c) \in Z^2 \mid \sqrt{(\theta_V-r\delta)^2 + (\psi_V-c\delta)^2} \leq \Delta\phi\} \tag{25}$$

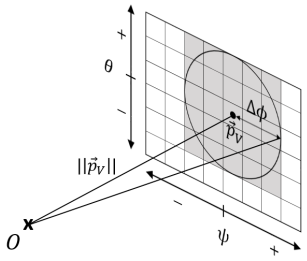

Figure 5. Identification of corresponding planar patches using $\Delta\phi$. Grey cells mark corresponding planar patches.

2. Determination of the Minimum Error

The smallest possible error of a point taking into account the degrees of freedom of the current subspace is determined by checking all correspondences of a point step by step. When checking correspondences, we now assume that the point was rotated in the direction of the planar patch. We can do this because the correspondence is enabled through the rotation of a point. To avoid the parametrization of a rotation matrix for the point rotation, we use the distance of the point $\overrightarrow{p}_V$ from the origin and compare it with the distance of the planar patches center point $\overrightarrow{m}_{r,c}$ from the origin. Using a unit vector $\frac{\overrightarrow{m}_{r,c}}{||\overrightarrow{m}_{r,c}||}$ in the direction of the planar patch we then calculate the point-to-plane error of the point:

$$\epsilon_{r,c} = |\ (\ (\ ||\overrightarrow{m}_{r,c}|| - ||\overrightarrow{p}_V||)\ \frac{\overrightarrow{m}_{r,c}}{||\overrightarrow{m}_{r,c}||}\ )^T \cdot \overrightarrow{N}_{r,c}\ | \tag{26}$$

The error calculated in (26) is a prediction of the error defined in equation (2) for one of the possible correspondence. A geometrical derivation of $\epsilon_{r,c}$ is shown in Fig. 6. The error $\epsilon_{r,c}$

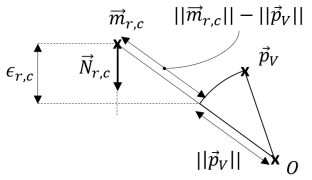

Figure 6. Derivation of the predicted error $\epsilon_{r,c}$ for a corresponding planar patch. The planar patch is defined by its center point $\overrightarrow{m}_{r,c}$ and its normal vector $\overrightarrow{N}_{r,c}$.

can be further reduced by a rotational component l_δ compensating the coarse angular resolution of the planar patches and by the translational component l_t (see equation (18)). The rotation compensates for the circumstance that the rotation parameters only allow discrete steps, namely the selection of the planar patches. Angle changes within a planar patch would otherwise have no effect. The maximum possible reduction of the error $\epsilon_{r,c}$ within a planar patch is calculated by using the same relation as in (19):

$$l_\delta = \frac{\delta}{2}\,||\overrightarrow{p}_V|| \tag{27}$$

The smallest possible error between a point $\overrightarrow{p}_i$ and its current correspondence is therefore:

$$\epsilon_{r,c,min} = \max(\epsilon_{r,c} - l_t - l_\delta, 0) \tag{28}$$

The minimum error $\epsilon_{i,min}$ of a point $\overrightarrow{p}_i$ results from the minimum of the values for epsilon $\epsilon_{r,c,min}$ of all its correspondences $C_{r,c}$:

$$\epsilon_{min}(\overrightarrow{p}_i) = \text{argmin}(\epsilon_{r,c,min} \text{ for } (r,c) \in C_{r,c}) \tag{29}$$

Calculation of the Upper Bound

The *upper bound* is a prediction of the maximum possible value of the score as it is defined in equation (1) within a subspace. To calculate this prediction, we now determine the minimum error based on the subspace defined by $\overrightarrow{V}, a_t, a_r$ for each point $\overrightarrow{p}_i$ in the point cloud P. The minimum error is then used to calculate the *upper bound* score. The procedure how to calculate the *upper bound* is defined in detail in algorithm 1.

Calculation of the Lower Bound

The *lower bound* is calculated by evaluating the value of the objective functional at the center point $\overrightarrow{V}$ of a subspace:

$$\underline{B} = score(R(\overrightarrow{V}(4:6)), \overrightarrow{V}(1:3)) \tag{30}$$

As mentioned before, this value serves as a measure of the range of values within the subspace.

Algorithm 1 Calculate Upper Bound

Input: Set of planar patches defined by $\overrightarrow{m}_{r,c}$ and $\overrightarrow{N}_{r,c}$, point cloud P, subspace defined by $\overrightarrow{V}$, a_r and a_t
Output: Upper bound $\overline{B}$

$\overline{B} = 0$
for all $\overrightarrow{p}_i \in P$ **do**
  $\overrightarrow{p}_V = R(\overrightarrow{V}(4:6)) \cdot (\overrightarrow{p}_i + \overrightarrow{V}(1:3))$
  Identify correspondences $C_{r,c}$ based on $\overrightarrow{p}_V$, a_r and a_t
  $\epsilon_{min} = \infty$
  for all $(r,c) \in C_{r,c}$ **do**
    Calculate $\epsilon_{r,c,min}$ as defined in equation (28)
    $\epsilon_{min} = \min(\epsilon_{min}, \epsilon_{r,c,min})$
  end for
  $\overline{B} = \overline{B} + \frac{1}{n_p} \cdot \exp(-\frac{\epsilon^2_{min}}{2\sigma^2})$
end for
return $\overline{B}$

2.4 Local Alignment

When searching for a global maximum, a local alignment is performed to speed up the procedure. Higher *lower bound* values usually allow more areas to be removed during global optimization by Branch and Bound. A local alignment within a subspace is performed when the *lower bound* $\underline{B}$, i.e. the evaluation of the score in the center of a subspace, exceeds a threshold $\lambda_{local} = 0.5 \cdot \overline{score}$. Where $\overline{score}$ is the highest identified score so far. The local alignment follows a ICP (Besl and McKay, 1992) related scheme. Instead of minimizing the point-to-point distance, the score is maximized based on the corresponding planar patches in each step. The Newton method is used for this purpose. To use the same metric as in our branch-and-bound optimization, planar patches are projectively associated to each point. This also speeds up the determination of the correspondences.

2.5 Algorithm Overview

Next, we present a pseudo code describing the sequence of the *GO-P2Plane* algorithm. As already mentioned, a subspace S is defined by a vector to the center point $\overrightarrow{V}$ and the edge lengths a_r and a_t. $\overline{B}$ is the *upper bound* of a subspace and $\underline{B}$ the *lower bound*. The list L is a data structure containing all subsequent subspaces S_i of our search space S_0 with their respective upper bounds $\overline{B_i}$. R is a rotation matrix and $\overrightarrow{t}$ a translation vector with three elements. The sequence of the algorithm is defined in algorithm 2.

3. INTEGRATION INTO MAPPING

We used the presented *GO-P2Plane* algorithm to develop a robust mapping system. Current mapping methods usually use convergence-based methods to register scans. If the convergence-based registration converges to a wrong local minimum, gross errors occur. Our system is characterized by the fact that errors can be detected and corrected. A globally optimal registration of all scans would be too time-consuming. That's why we combine local convergence-based registration with global registration. Based on various criteria, we decide scan by scan whether the result of the local registration is plausible.

3.1 Overview

In Fig. 7, our embedding of the globally optimal algorithm in a mapping system is sketched. For each new point cloud R_t, the

Algorithm 2 GO-P2Plane

Input: planar patches $\overrightarrow{m}_{r,c}$, $\overrightarrow{N}_{r,c}$, moving point cloud P and search space defined by $\overrightarrow{V}$, a_r and a_t
Output: Maximum $\overline{score}$, rotation matrix $\overline{R}$ and translation $\overline{t}$

Init open list L for subspaces and respective upper bounds
Add the search space defined by $S_0 = (\overrightarrow{V}, a_r, a_t)$ to L
while $\overline{score} < \overline{B_i}$ **do**
  Choose element i_{max} with maximum upper bound in L
  Divide subspace of list element i_{max} into 64 subspaces $S_1, ..., S_{64}$
  for all $S_j,\ j \in \{1, ..., 64\}$ **do**
    Calculate $\overline{B_j}$ and $\underline{B_j}$ for S_j using algorithm 1 and equation (30)
    if $\overline{B_j} > \overline{score}$ **then**
      Add $(S_j, \overline{B_j})$ to L
      if $\underline{B_j} > \lambda_{local}$ **then**
        $[score, R, \overrightarrow{t}] = \text{localAlignment}()$
        if $\frac{\overline{score}}{score} < 1$ **then** $[score, R, \overrightarrow{t}] \mapsto [\overline{score}, \overline{R}, \overline{t}]$
      end if
    end if
  end for
end while
return $\overline{score}$, $\overline{R}$ and $\overline{t}$

processing steps are executed consecutively. $_t$ is a time index that is incremented by one for each new point cloud. A new point cloud R_t is preprocessed first using a voxel grid filter for downsampling. Then, the point cloud P_i is registered locally relative to last keyframe K_i using the Generalized-ICP (Segal et al., 2009). The result of the registration donated by transformation T_t is then checked for plausibility by comparing it with the results of the last time step. The plausibility check used is described in section 3.2. If the results are plausible, a keyframe update is performed with it. If they are not, a globally optimal registration is performed using *GO-P2Plane*. The selector is a placeholder that symbolizes the passing of the respective transformation parameters. If the distance to the last keyframe exceeds a threshold the keyframe is updated. This means the current point cloud P_t is saved and provided as K_i for further processing. The index $_i$ indicates the keyframe number.

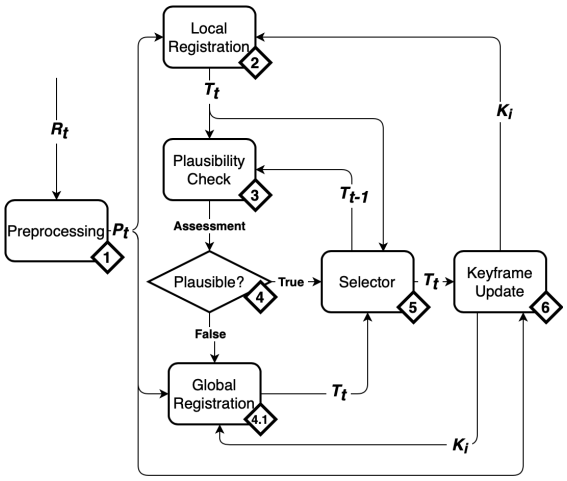

Figure 7. Overview of the mapping process. The numbers in the diamond-shaped boxes represent the order.

3.2 Plausibility Check

We use 3 criteria to check the plausibility of local registration results. Let T_t be the transformation of the current scanner pose relative to the map origin that we want to evaluate. It consists of the translation vector $\overrightarrow{t}_t$ and a rotation matrix R_t. T_{t-t} is the corresponding transformation of the previous time step. $\Delta\tau$ is the time difference between the two points in time τ_{t-1} and τ_t of the transformations.
Our plausibility check is mainly based on model assumptions for the movement of the LiDAR sensor during data recording. For this we define a maximum velocity v_{max} of the sensor during recording, a maximum acceleration a_{max} and a maximum rotation rate ω_{max}. Our model assumption is that the sensor moves continuously during recording and that these values are not exceeded. The sensor velocity is considered constant between two scans and we calculate it as follows:

$$\overrightarrow{v}_t = \frac{(\overrightarrow{t}_t - \overrightarrow{t}_{t-1})}{\Delta\tau} \tag{31}$$

The absolute rotation rate between two scans can be calculated with:

$$|\omega_t| = \arccos \frac{\mathrm{tr}(R_{t-1}^T R_t) - 1}{2\Delta\tau} \tag{32}$$

The result of the registration is considered plausible if the following conditions are met:

$$||\overrightarrow{v}_t|| \leq v_{max} \ \wedge \ ||\frac{\overrightarrow{v}_t - \overrightarrow{v}_{t-1}}{\Delta\tau}|| \leq a_{max} \ \wedge \ |\omega_t| \leq \omega_{max} \ \rightarrow plausible \tag{33}$$

3.3 Global Registration

For global registration we use our proposed *GO-P2Plane* approach. The procedure serves as a fallback function if the result determined with the use of local registration does not seem plausible. In order to use the globally optimal method as efficiently as possible, we limit the search range using the model parameters. This can significantly reduce the computation time of the method. For the translation range, the search distance is limited to $d_{max} = \Delta\tau\ v_{max}$, where $\Delta\tau$ is the time difference between two consecutive LiDAR scans and v_{max} the maximum assumed velocity. For the rotational search range a maximum rotation of $\alpha_{max} = \Delta\tau\ \omega_{max}$ is assumed between the point clouds. Here, ω_{max} is the maximum assumed angular velocity of the scanner.

3.4 Keyframe Update

To determine the global pose relative to the very first point cloud, the point clouds are always registered relative to the last so-called keyframe. The keyframes are a subset of the total set of point clouds. If the distance to the last keyframe exceeds a limit, a new keyframe is saved.

4. RESULTS

Since we are aiming at mapping in our application, we have limited our choice of experiments to scans with partial overlap. We implemented the *GO-P2Plane* algorithm in C++ and during the tests the software runs on an Intel i7 4. generation.

Our results consists of two parts. First, we applied the presented algorithm in isolation to point clouds of the public *Stairs* dataset (Pomerleau et al., 2012) containing laser scans obtained with custom rotating setup using a Hokuyo UTM-30LX laser range finder. The contained scans are related to terrestrial laser scans in terms of point density and field-of-view. Furthermore, we tested the mapping system presented in chapter 3 using the first sequence of the KITTI dataset (Geiger et al., 2012).

4.1 Stairs Dataset

The *Stairs* dataset (Pomerleau et al., 2012) consists of 31 laser scans with a mean of 191,000 points per scan. The scans were recorded indoors in a staircase and the recording points of the sequence are approx. 0.4 m away from each other. The challenges of the dataset are large volume changes and large variations in the overlaps. Global positions with mm-range precision are available as ground truth for all scans. In this experiment we registered all the individual scans one after the other relative to each other using our method. The methods are evaluated by comparing the errors of the relative transformations between each two consecutive point clouds with the ground truth. The average error of the translation and the average rotation error are used as error metrics. The mean overlap of the consecutive point clouds is 91 % and the minimum overlap 62 %. The Go-ICP (Yang et al., 2015) and GOGMA (Campbell and Petersson, 2016) registration results for this dataset were taken from (Campbell and Petersson, 2016). For the Go-ICP we display the results for N=500 points. For the GOGMA algorithm there are also presented results with a subsequent refinement in (Campbell and Petersson, 2016), but we have not included these in the table, since methods for refinement are not the focus of this work. Our *GO-P2Plane* algorithm used also N=500 points for its model point cloud. As search space of the algorithm we allowed any kind of rotation in these experiments $\alpha_{max} = 180\,^\circ$ and set a maximum distance of $d_{max} = 1\,m$. As a state-of-the-art representative of convergence-based registration algorithms, the Generalized-ICP (GICP) (Segal et al., 2009) was tested. For this purpose, the publicly available implementation in the Point Cloud Library (Rusu and Cousins, 2011) was used with the default settings. Using this example, it can

Table 1. Translation [m] and rotation error [°] results and mean runtime [s] per scan for *Stairs* dataset

Method	*GO-P2Plane*	GOGMA	Go-ICP	GICP
Mean Tra.	0.06	0.26	1.17	0.03
Mean Rot.	2.02	1.25	19.4	0.67
Max. Tra.	0.15	-	-	**0.79**
Max. Rot.	4.12	-	-	**15.97**
Runtime	127.3	49.6	103.0	1.27

be shown that a conventional convergence-based registration algorithm can fail from time to time. The low average angle and translation error indicates that in most cases the correct transformation could be determined. The maximum translation and rotation error (marked in bold in the table) shows that at least in one case the GICP has converged into a false local minimum. This happened when processing the scans with IDs 24 and 25 in the dataset. The scene with the respective solutions of GICP and our method are shown in Fig. 8.
The low maximum translation and rotation errors of our presented method show that the approach could register all point clouds correctly. The results obtained are comparable to those of the GOGMA algorithm.

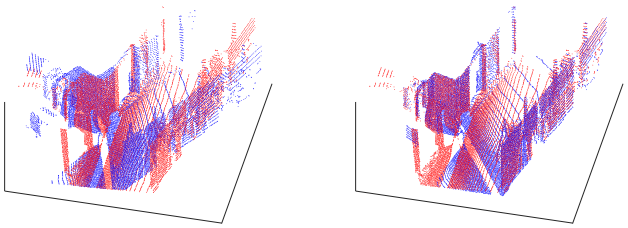

Figure 8. Scan 24 and scan 25 of the *Stairs* dataset. Left: GICP registration result. Right: Result of our *GO-P2Plane* algorithm.

4.2 KITTI dataset

Here we want to present the results of how we tested our mapping system using the KITTI dataset (Geiger et al., 2012). First of all, it is important to clarify that our goal with this system is not to achieve a particularly high accuracy. We are also aware that the procedures of the authors leading the ranking with their odometry results have run reliably within the KITTI dataset and that there were most likely no gross errors in the registration. Situations in which gross errors occur during scan matching differ depending on the environment setup, the methods and may also behave differently depending on the parameter settings. Our goal with this experiment is to demonstrate the functionality of a robust mapping system
In our experiments, we applied our system to the first sequence of the KITTI dataset. To enforce difficult conditions, we reduced the data set to one-third of the scans by using only every 3rd scan. This reduces the overlap and increases the distance between scans.

Validation of Plausibility Check
First, we tested our plausibility check conditions which are described in chapter 3.2. For this purpose, we processed the modified dataset as described without global registration with our pipeline. Each time the result of the local registration violated the plausibility conditions, we saved this information. In Fig. 9 translation and rotation errors for each pose are shown. We marked identified implausible poses red, while plausible poses are green. With this experiment, we can show that we can successfully identify poses with large translation error using our plausibility check. On further analysis, we found that in this dataset it was mainly the maximum assumed acceleration (we use $a_{max} = 10 \frac{m}{s^2}$) that allowed the identification of the poses with faulty translations. In Fig. 11 we share the resulting trajectory in den x-y plane with outlined locations of implausible states.

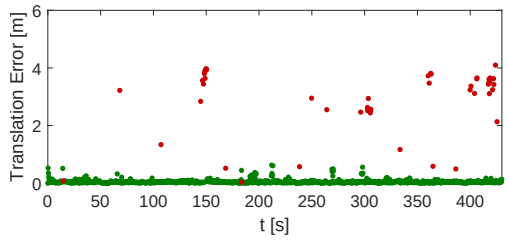

Figure 9. Translation error for each transform of the first sequence of the KITTI dataset using only every 3rd scan without global registration. Red points represent identified implausible poses and green points plausible poses.

Application of our System
Here we present our mapping results applying our complete mapping pipeline as introduced in chapter 3 to the modified first sequence of the KITTI dataset (Geiger et al., 2012). For the presented *GO-P2Plane* algorithm we used an angular resolution of $\delta = 3\,^\circ$ for the planar patches. We reduced the second point cloud to 2,000 points by choosing random points. To further reduce the computation time of our global optimal registration algorithm, we added an additional stop condition. After $N = 300,000$ iterations, the search for the global optimum is terminated. In our configuration, this corresponds to 20 s of computation time. It should be noted that the algorithm is no longer globally optimal as a result. Nevertheless, the number of iterations was in our experiments sufficient to find suitable transformations.

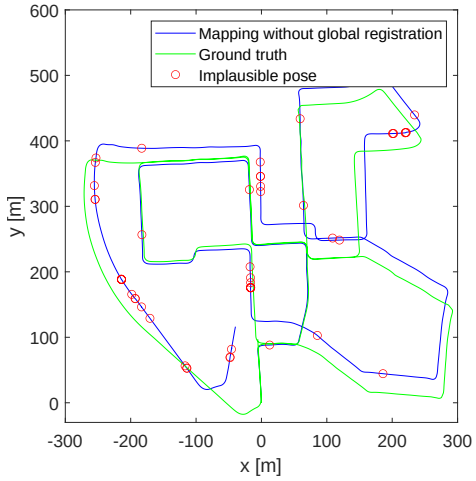

Figure 10. GICP mapping results plotted in the horizontal plane of the modified first sequence of the KITTI dataset without global registration. Red circles mark locations, where the registration result was classified as implausible.

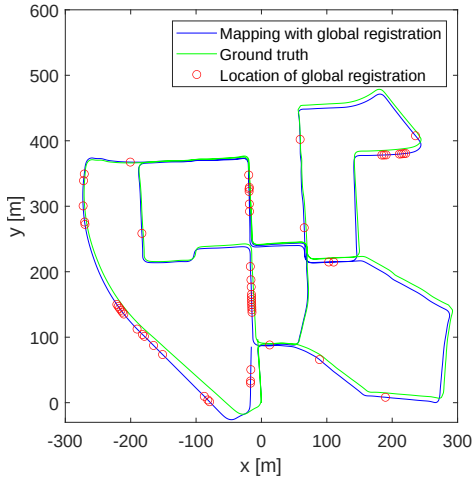

Figure 11. Mapping results plotted in the horizontal plane of the modified first sequence of the KITTI dataset with our complete mapping pipeline. Red circles mark locations, where the *GO-P2Plane* was used for global registration.

5. CONCLUSION

We have presented a new globally optimal point cloud registration algorithm (chapter 2). Like some other registration algorithms, it works according to the branch and bound paradigm. However, it is the first branch and bound that optimizes according to the point-to-plane metric. Like the point-to-plane metric, the newly developed upper bound function is particularly well suited for sparse point clouds from mobile laser scanners. Using the Stairs dataset, we were able to show that our method achieves similar results to related methods.
Furthermore, we have developed a mapping system for point clouds that can detect gross errors and also correct them with the help of the presented global registration algorithm (chapter 3). We were able to test the system successfully using a public data set.

Limitations
As long as the correct pose has a global maximum in our objective functional at registration, it can be identified with our algorithm. However, if the movement of the sensor in the point clouds is not observable, because the sensor is moving on a free surface or in a tunnel with the same shape, the presented algorithm cannot find a reliable solution. Another problem that can occur with a small overlap of two point clouds is that points that do not actually belong together can be staged by a wrong pose in such a way that a maximum in the objective functional is generated. Such a maximum can be obtained, for example, by superimposing two observations of the ground that do not actually represent the same location.
Another challenge is the computing time of the presented method. By reducing the parameter search space on the basis of assumptions about motion limitations, it can be reduced, but the method is still far away from real-time. The results presented were not calculated in real time. One possibility would be to integrate the procedure into a real-time mapping system and, if necessary, to calculate error corrections parallely in a separate thread.

Future Work
In order to enable a broader applicability of the method, a drastic reduction of the computational costs is necessary. This can be achieved, for example, by more compact representations of the environment than point clouds or planar patches. Another way to reduce the computational costs is a more exact *upper bound function* for maximum value predictions. Integration into real-time mapping systems is also being considered.

REFERENCES

Behley, J., Stachniss, C., 2018. Efficient surfel-based slam using 3d laser range data in urban environments. *Robotics: Science and Systems*, 2018, 59.

Besl, P. J., McKay, N. D., 1992. Method for registration of 3-d shapes. *Sensor fusion IV: control paradigms and data structures*, 1611, International Society for Optics and Photonics, 586–606.

Biber, P., Straßer, W., 2003. The normal distributions transform: A new approach to laser scan matching. *Proceedings 2003 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2003)(Cat. No. 03CH37453)*, 3, IEEE, 2743–2748.

Blais, G., Levine, M. D., 1995. Registering multiview range data to create 3D computer objects. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17(8), 820–824.

Campbell, D., Petersson, L., 2016. Gogma: Globally-optimal gaussian mixture alignment. *Proceedings of the IEEE conference on computer vision and pattern recognition*, 5685–5694.

Clausen, J., 1999. Branch and bound algorithms-principles and examples. *Department of Computer Science, University of Copenhagen*, 1–30.

Geiger, A., Lenz, P., Urtasun, R., 2012. Are we ready for autonomous driving? the kitti vision benchmark suite. *Conference on Computer Vision and Pattern Recognition (CVPR)*.

Hess, W., Kohler, D., Rapp, H., Andor, D., 2016. Real-time loop closure in 2d lidar slam. *2016 IEEE international conference on robotics and automation (ICRA)*, IEEE, 1271–1278.

Koide, K., Yokozuka, M., Oishi, S., Banno, A., 2021. Globally Consistent 3D LiDAR Mapping With GPU-Accelerated GICP Matching Cost Factors. *IEEE Robotics and Automation Letters*, 6(4), 8591–8598.

Myronenko, A., Song, X., 2010. Point set registration: Coherent point drift. *IEEE transactions on pattern analysis and machine intelligence*, 32(12), 2262–2275.

Palieri, M., Morrell, B., Thakur, A., Ebadi, K., Nash, J., Chatterjee, A., Kanellakis, C., Carlone, L., Guaragnella, C., Agha-Mohammadi, A.-a., 2020. Locus: A multi-sensor lidar-centric solution for high-precision odometry and 3d mapping in real-time. *IEEE Robotics and Automation Letters*, 6(2), 421–428.

Pomerleau, F., Liu, M., Colas, F., Siegwart, R., 2012. Challenging data sets for point cloud registration algorithms. *The International Journal of Robotics Research*, 31(14), 1705–1711.

Reinke, A., Chen, X., Stachniss, C., 2021. Simple but effective redundant odometry for autonomous vehicles. *2021 IEEE International Conference on Robotics and Automation (ICRA)*, IEEE, 9631–9637.

Rusu, R. B., Cousins, S., 2011. 3D is here: Point Cloud Library (PCL). *IEEE International Conference on Robotics and Automation (ICRA)*, Shanghai, China.

Segal, A., Haehnel, D., Thrun, S., 2009. Generalized-icp. *Robotics: science and systems*, 2number 4, Seattle, WA, 435.

Tombari, F., Salti, S., Di Stefano, L., 2013. Performance evaluation of 3D keypoint detectors. *International Journal of Computer Vision*, 102(1), 198–220.

Yang, H., Shi, J., Carlone, L., 2020. Teaser: Fast and certifiable point cloud registration. *IEEE Transactions on Robotics*, 37(2), 314–333.

Yang, J., Li, H., Campbell, D., Jia, Y., 2015. Go-ICP: A globally optimal solution to 3D ICP point-set registration. *IEEE transactions on pattern analysis and machine intelligence*, 38(11), 2241–2254.

Zhang, J., Singh, S., 2014. Loam: Lidar odometry and mapping in real-time. *Robotics: Science and Systems*, 2number 9, Berkeley, CA, 1–9.