Analysis of Error Functions for the Iterative Closest Point Algorithm

Mémoire

Philippe Babin

Maîtrise en informatique - avec mémoire
Maître ès sciences (M. Sc.)

Québec, Canada

Analysis of Error Functions for the Iterative Closest Point Algorithm

Mémoire

Philippe Babin

Sous la direction de:

Philippe Giguère
François Pomerleau

Résumé

Dans les dernières années, beaucoup de progrès a été fait dans le domaine des voitures autonomes. Plusieurs grandes compagnies travaillent à créer un véhicule robuste et sûr. Pour réaliser cette tâche, ces voitures utilisent un lidar pour la localisation et pour la cartographie. *Iterative Closest Point (ICP)* est un algorithme de recalage de points utilisé pour la cartographie basé sur les lidars. Ce mémoire explore des approches pour améliorer le minimisateur d'erreur d'ICP. La première approche est une analyse en profondeur des filtres à données aberrantes. Quatorze des filtres les plus communs (incluant les M-estimateurs) ont été testés dans différents types d'environnement, pour un total de plus de 2 millions de recalages. Les résultats expérimentaux montrent que la plupart des filtres ont des performances similaires, s'ils sont correctement paramétrés. Néanmoins, les filtres comme *Var. Trim.*, *Cauchy* et *Cauchy MAD* sont plus stables à travers tous les types environnements testés. La deuxième approche explore les possibilités de la cartographie à grande échelle à l'aide de lidar dans la forêt boréale. La cartographie avec un lidar est souvent basée sur des techniques de Simultaneous Localization and Mapping (SLAM) utilisant un graphe de poses, celui-ci fusionne ensemble ICP, les positions Global Navigation Satellite System (GNSS) et les mesures de l'Inertial Measurement Unit (IMU). Nous proposons une approche alternative qui fusionne ses capteurs directement dans l'étape de minimisation d'ICP. Nous avons réussi à créer une carte ayant 4.1 km de tracés de motoneige et de chemins étroits. Cette carte est localement et globalement cohérente.

Abstract

In recent years a lot of progress has been made in the development of self-driving cars. Multiple big companies are working on creating a safe and robust autonomous vehicle . To make this task possible, theses vehicles rely on lidar sensors for localization and mapping. Iterative Closest Point (ICP) is a registration algorithm used in lidar-based mapping. This thesis explored approaches to improve the error minimization of ICP. The first approach is an in-depth analysis of outlier filters. Fourteen of the most common outlier filters (such as M-estimators) have been tested in different types of environments, for a total of more than two million registrations. The experimental results show that most outlier filters have a similar performance if they are correctly tuned. Nonetheless, filters such as *Var. Trim.*, *Cauchy*, and *Cauchy MAD* are more stable against different environment types. The second approach explores the possibilities of large-scale lidar mapping in a boreal forest. Lidar mapping is often based on the SLAM technique relying on pose graph optimization, which fuses the ICP algorithm, GNSS positioning, and IMU measurements. To handle those sensors directly within the ICP minimization process, we propose an alternative technique of embedding external constraints. We manage to create a crisp and globally consistent map of 4.1 km of snowmobile trails and narrow walkable trails. These two approaches show how ICP can be improved through the modification of a single step of the ICP's pipeline.

Contents

List of Figures

List of Tables

List of Acronyms

ICP Iterative Closest Point

ICRA IEEE International Conference on Robotics and Automation

FSR Field and Service Robotics

SLAM Simultaneous Localization and Mapping

SIFT Scale-Invariant Feature Transform

LOAM Lidar Odometry And Mapping

icp Iterative Closest Point

FGR Fast Global Registration

FRMSD Fractional Root Mean Squared Distance

RQE Rényi Quadratic Entropy

KC Kernel Correlation

RMT Relative Motion Threshold

NDT Normal Distribution Transformation

GM Geman-McClure

SC Switchable-Constraint

GICP Generalized-ICP

IRLS Iteratively Reweighted Least-Squares

MAD Median of Absolute Deviation

NDT Normal Distribution Transformation

ROS Robotic Operating System

IMU Inertial Measurement Unit

GNSS Global Navigation Satellite System

GTLS-ICP Generalized Total-Least-Squares ICP

BFGS Broyden-Fletcher-Goldfarb-Shanno

DoF Degrees of Freedom

ALS Airborne Lidar Scanning

RTK Real-Time Kinematic

ENU East-North-Up

"The method of Least Squares is seen to be our best course when we have thrown overboard a certain portion of our data – a sort of sacrifice which has often to be made by those who sail upon the stormy sea of Probability."

—F.Y. Edgeworth, 1888 *[1]*

Remerciements

Merci à Papa, Maman et Charles pour leur soutien. Merci à Philippe Giguère de m'avoir fait découvrir le monde de la recherche et de la robotique. Merci à François de m'avoir guidé tout au long de ma maîtrise. Merci à David pour m'avoir fait découvrir l'importance d'avoir un local avec une bonne covariance. Merci à Philippe pour ses suggestions de Scotches. Merci à Simon-Pierre pour ses backflips. Thanks to Vladimir for sharing his robotic wisdom. Merci à Jeff, Gari, Maxime, Dominique, Sébastien, Charles-Éric, Johann, Mathieu, Jonathan, Quan, Steffen, Sébastien et tous les membres du Norlab, Damas, Graal et Cobra. Merci à toute la gang pour les nombreuses games de Mölkky. Merci au Husky A200 pour avoir survécu à tous mes bidouillages électriques. Merci à Philippe(s), Simon(s), Gab et toute l'équipe de RoboCup ULaval pour m'avoir permis d'apprendre tant de choses (tout en brûlant beaucoup de microcontrôleurs).

Foreword

Chapter 2 of this work consists in an inserted article. It was submitted and accepted to the IEEE International Conference on Robotics and Automation (ICRA) on September 15$^{\text{th}}$, 2018 [2]. ICRA is a world-leading robotics conference, with an h5-index of 82 as of this year.[1] I am the main author of the publication, and as such I was responsible for its writing and the underlying experiments. My two co-authors, François Pomerleau and Philippe Giguère, are also the co-supervisors of my masters. They provided guidance for this endeavor and assisted me in the writing of the paper.

Likewise, Chapter 3 of this work consists in another inserted article. It was submitted and accepted to the 12$^{\text{th}}$ Conference of Field and Service Robotics (FSR) on April 10$^{\text{th}}$, 2019 [3]. FSR is one of the leading conference in field robotics. I am again the main author of the publication, and as such I was responsible for its writing and the underlying experiments. The second co-author, Philippe Dandurand, helped with the construction of the robotic platform and with the processing of GNSS readings. The third co-author, Vladimir Kubelka, a postdoctoral fellow, assisted in the writing of the paper.

Both articles are inserted almost exactly as submitted. To better integrate to the rest of this thesis, the figures and section numbers were modified. Also, the layout was changed to fit the standard masters thesis template of *Université Laval*.

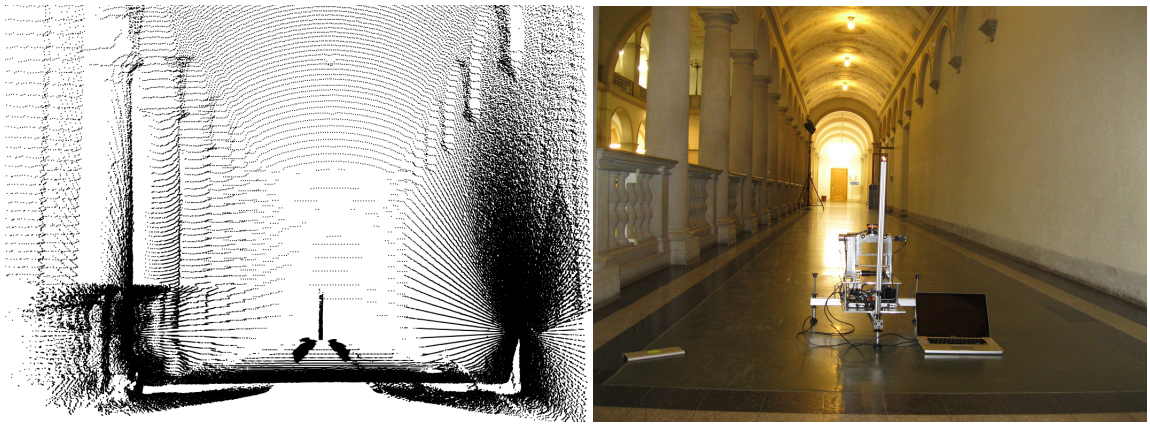
[1] `https://scholar.google.com/citations?view_op=top_venues`

Introduction

Over the past few years, self-driving cars and their future impact on society has become a common topic in news outlets. For instance, Singapore started a self-driving taxi pilot project two years ago, while Uber has kicked off a similar project in Pittsburgh. More recently, Waymo (formerly part of Google X) has started a pilot project in snowy Detroit to test more difficult weather conditions. As the first outdoor mass market autonomous vehicles, theses cars must be highly reliable and safe. Furthermore, the environment in which they navigate is highly dynamic. The weather, sporadic road work and even the sunlight itself can change drastically how the environment is perceived by the car's sensor [4], further compounding difficulties. Being able to navigate through these environmental changes is one of many requirements for a robust system. Navigation systems require detailed maps of the environment to plan decisions [5]. Most self-driving cars rely on lidar sensors to map the environment and to locate themselves within it. However, robust and large-scale mapping in a 3D environment with a lidar is still known to be a challenging task [4].

Lidars are quite similar to radar, as they both estimate the distance between the sensor and an object in the environment by measuring the time it takes for an electromagnetic pulse to come back. However in the case of lidar, the pulse is generated by a laser, instead of a radio transmitter. 2D lidars measure a planar cut of the environment, while 3D lidars measure a 3D field of view. The output of a scanning lidar is named a *point cloud*; see Figure 0.1 for an example of a point cloud compared to a picture of the same scene. Lidar is a key technology for robotic applications, since contrary to cameras, they are relatively immune to the ambient lighting conditions and, to a lesser extent, to weather conditions.

One of the uses of lidar technology in robotics is for localization. The geometric transformation (translation and rotation) between two positions can be estimated by aligning the lidar scans taken at these two positions. The problem of aligning two point clouds



(a) Point cloud from a lidar (b) Picture taken at a similar angle

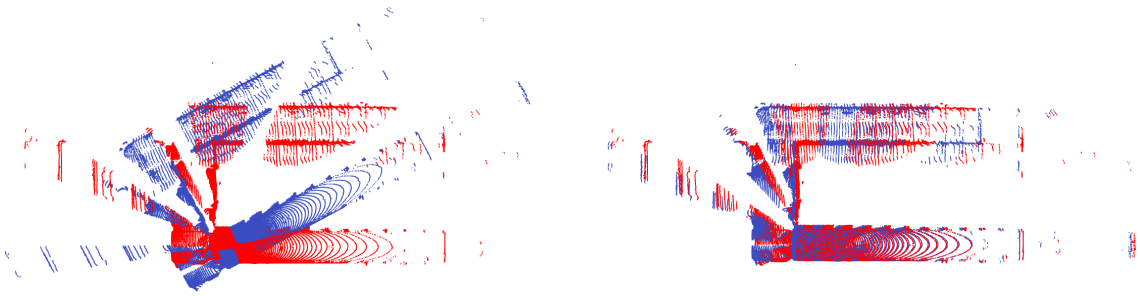
Figure 0.1 – Lidar scan compared to a picture of a camera. The point cloud and the picture are taken from the *hauptgebäude* dataset in the *Challenging data sets for point cloud registration algorithms* [6].

is called the *point set registration problem*. This general problem can be further subdivided into two issues: *i)* global registration and *ii)* local registration. The latter requires an initial estimate of the alignment, while the former does not. Figure 0.2 shows an example of local registration. Since for most robotic applications, an initial estimate can be acquired from other sensors (such as the wheel's odometry, a Global Navigation Satellite System (GNSS) or an Inertial Measurement Unit (IMU)), this thesis will focus *solely* on the problem of local registration.

In spite of having been introduced thirty years ago, the Iterative Closest Point (ICP) algorithm remains the most common solution to local registration [7], [8]. ICP attempts to find the best alignment by alternating between matching pairs of points close to each other and minimizing the distance between the pairs. Over the years, numerous variants of ICP algorithms have been created to increase the robustness, the speed or the precision of the original [9]. On the other hand, selecting an appropriate variant for a given application has become tedious, due in part to the lack of comparative benchmarks.

When registering point clouds, one of the main sources of errors is the presence of outliers. Outliers have various causes, such as lack of overlap, sensor noises, or dynamic elements. The problem of outlier rejection has a long history in the field of robust statistics [10]. Better outlier filtering naturally leads to better minimization results, which in turn generally leads to faster convergence.

This thesis will focus on the subject of improving the error minimization of ICP in the



(a) Initial alignment (b) Final alignment

Figure 0.2 – An example of pairwise local registration of two lidar point clouds, using ICP. This top view shows two lidar scans of the interior of a building. The static (i.e., reference) and dynamic (i.e., reading) point cloud are in red and blue, respectively.

context of mobile robotic applications. This goal is achieved through two approaches. The first approach explores the effect of outlier filtering on ICP . The second approach presents a novel sensor fusion technique, to be used directly within the error minimizer of ICP .

This thesis will consist primarily of two inserted articles, which correspond directly to the two previously-stated approaches. Chapter 1 provides the theoretical background required for the understanding of our two articles. In particular, it introduces the ICP algorithm, its individual steps and how it can be used for mapping. Then, Chapter 2, an inserted article published in IEEE International Conference on Robotics and Automation (ICRA), shows an analysis of the outlier filter step of ICP. Afterwards in Chapter 3, a novel approach to large-scale 3D mapping is presented, where sensor fusion between the IMU, the GNSS and the lidar are performed directly in the error minimizing step of ICP. Finally, the Appendix A shows a table of all outlier filters for ICP, with their various proprieties.

Chapter 1

Iterative Closest Point Algorithm

ICP was independently developed by Besl [7] and Chen *et al.* [8]. It is a solution to the point set rigid registration problem, which is as follows: given two overlapping point clouds (reading $\mathcal{P}$ and reference $\mathcal{Q}$), find the rigid transformation that maximizes their alignment. Rusinkiewicz *et al.* [11] introduced the idea of ICP as a pipeline of interchangeable blocks. Pomerleau *et al.* [12] built upon this idea by creating `libpointmatcher`, a library to test different configurations of the pipeline. Figure 1.1 shows an example of one such pipeline.

ICP is an iterative algorithm to solve this problem. In the first step, the data filtering step (1), the two point clouds are passed through filters which convert the raw data to a more useful form. In the matching step (2), each point of the reading $\mathcal{P}$ is matched to its closest neighbor in the reference $\mathcal{Q}$ and the errors between these matches are computed. Subsequently in the outlier filtering step (3), the outlier matches are rejected. In the minimization step (4), the geometric transformation $\widehat{\boldsymbol{T}}$ between the reference $\mathcal{Q}$ and the reading $\mathcal{P}$ that minimizes the error among the inlier matches is computed, and in the final step (5) the transformation $\widehat{\boldsymbol{T}}$ is applied to the reading $\mathcal{P}$ point cloud.

Finally, the steps (2) to (5) are repeated, until convergence or a pre-defined stopping criterion. Below, we present in greater details these five steps of ICP. We will restrict ourselves to the subject of descriptor-less point clouds taken from lidar. For more in-depth reviews of all types of registration, see Pomerleau *et al.* [9].

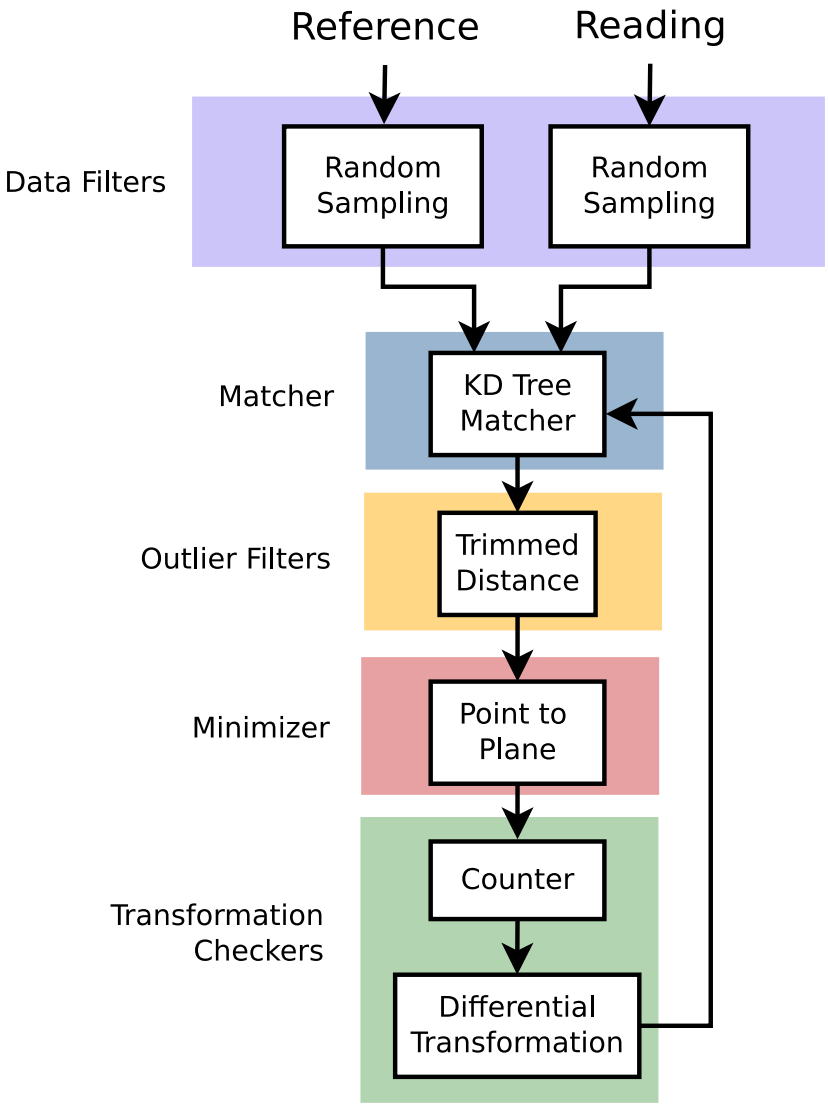


Figure 1.1 – An example of the default pipeline of ICP for the `libpointmatcher` library. At each iteration of the loop, the reading point cloud is moved by the estimated transformation. The loop is executed until the transformation checkers detect convergence.

1.1 ICP Pipeline

1.1.1 Data Filters

The first step of the pipeline is about converting the raw point cloud into a format required by the other steps. In this subsection, the two most common types of data filtering, feature reduction and feature augmentation [9], are presented. Before discussing data filtering, the point cloud representation used in this thesis is explained.

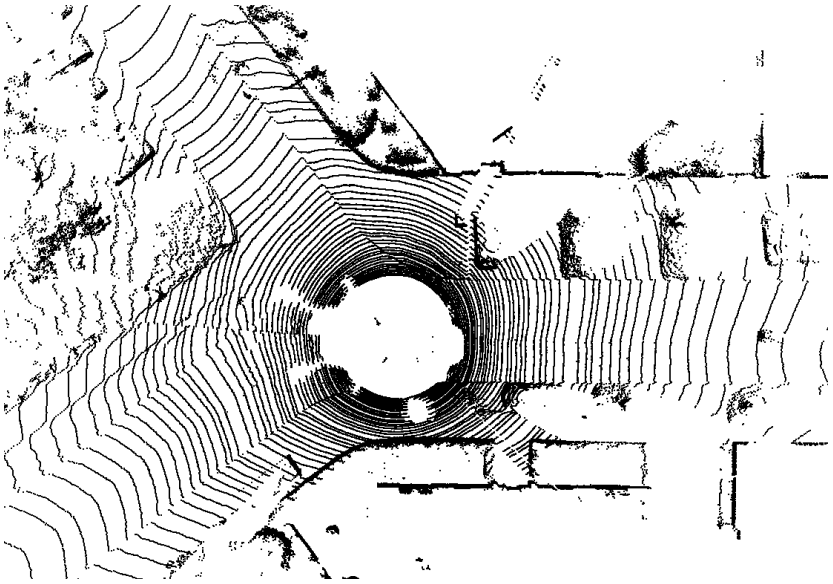
Point cloud representation

A 3D point cloud $\mathcal{P}$ is composed of 3D points $\boldsymbol{p} \in \mathbb{R}^3$. It is usually represented as a column-wise matrix $\begin{bmatrix}\boldsymbol{p}_0 & \boldsymbol{p}_1 & \boldsymbol{p}_2 & \dots & \boldsymbol{p}_n\end{bmatrix}$. A homogeneous representation of $\mathcal{P}$ is sometimes used in this thesis to simplify the application of a rigid transformation. The order of the points inside the matrix is not important: there is no relationship between how close two points are in the matrix and in Euclidean space. This representation has the advantages of being densely packed in memory and easily vectorizable. At the same time, the lack of spatial relationship between points makes lookup operations, such as the closest point search, computationally expensive.

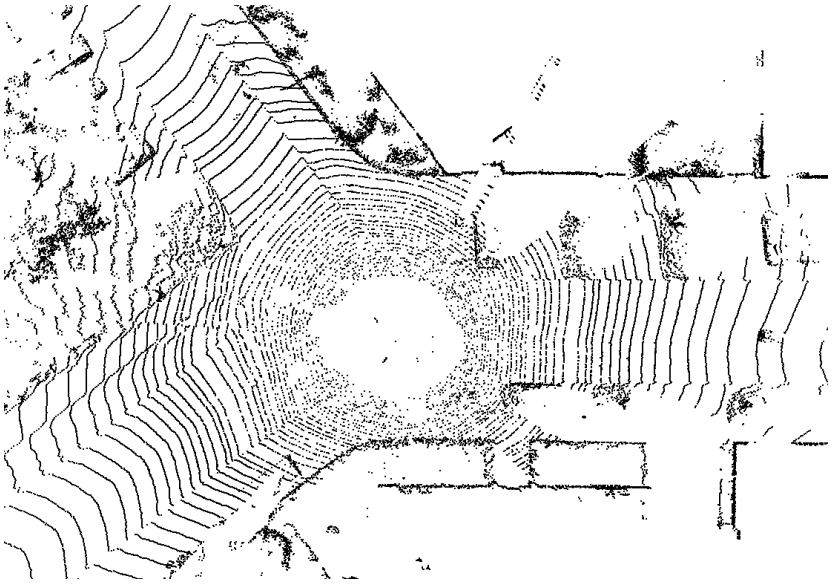
Filtering

Feature reduction is an operation that simplifies and compresses the point cloud from the lidar. Indeed, the point cloud outputted by a lidar scan is seldom directly used for registration. Since 3D lidars use a rotating array of laser beams, the raw lidar scans are composed of concentric circles. The circles close to the scanner will tend to have a higher density of points (see Figure 1.2). Thus during the registration, points closer to the scanner will be overrepresented, compared to those farther away. To circumvent this kind of problem, the raw scans must be downsampled in an appropriate manner. In particular, to uniformize the density of a point cloud, multiple data filters have been introduced over the years. They include random sampling, sampling over a uniform grid, or filtering out points with too many close neighbors. A cruder form of downsampling is to simply remove points which are outside a bounding box.

The data filtering step can also serve as a means of computing additional features. The most common example is computing the normal of a surface at each point, based on the distribution of its neighbors. These additional features can then be used by the other steps in the pipeline. For instance, the normals are used by a point-to-plane minimizer for outlier rejection.



(a) Raw lidar scan



(b) After maximum density filter

Figure 1.2 – A raw lidar scan is composed of concentric circles. The point cloud has a higher density of points when close to the scanner. A maximum density filter can uniformize the point cloud, thus reducing the amount of points closer to scan, while keeping the farthest points. The scan was taken with a Velodyne HDL-64e and is part of the Kitti dataset [13].

1.1.2 Matcher

The matching is the first step of the main loop of ICP. It is also the step with the highest computational cost. The matcher is a function which takes as input the reading $\mathcal{P}$ and reference $\mathcal{Q}$ point clouds, and returns for each point in $\mathcal{P}$ a corresponding point in $\mathcal{Q}$. The goal of the matcher is to solve the data association problem. With an RGB-D camera (e.g., a Kinect), the color and intensity of the neighboring points can be used to create image descriptors (e.g., Scale-Invariant Feature Transform (SIFT) [14]). The descriptor can then be used for matching purposes. With a lidar scan, the correspondence between the reference points in $\mathcal{Q}$ and the reading points in $\mathcal{P}$ is unknown. Thus, a heuristic is used, where each point in the reading $\mathcal{P}$ is matched with the closest point in the reference $\mathcal{Q}$. As a result, ICP can be quite sensitive to the initial estimate transformation $\check{\boldsymbol{T}}$, since being far from the correct alignment means matching with the wrong points. Figure 1.3 shows how the matching function slowly converges to the correct matches through the iterations.

Almost all variants of ICP use a kD-tree [15] as a matching algorithm. The kD-tree is a type of binary tree, which split space around axis-aligned planes. It is able to find the k nearest neighbors to a point in logarithmic time ($\mathcal{O}(k \log n)$), which is much faster than looking up each individual point. Usually a kD-tree is build out of the reference point cloud $\mathcal{Q}$ after the data filter step. Since the reference stays the same between iterations, this is done only once. Then at each iteration of the loop and for each of the points in the reading $\mathcal{P}$ the k closest points in the reference $\mathcal{Q}$ are found. The original ICP used the closest point ($k = 1$). As shown in Figure 1.3, using a higher value of k is more robust to mismatches, but at the cost of higher computation time. Modern implementations (such as `libnabo` [16]) of kD-tree relies on multithreading to alleviate this problem.

1.1.3 Outlier filters

As previously stated, outliers are an important source of errors for ICP. The third step of the ICP's pipeline is therefore devoted to reducing their effects on the registration. The outlier filter is a function which takes as input the matched point's distances, and outputs a weight for each of these matches. This weight is then used by the minimizer to determine which matches to filter out. ICP uses a least-square minimizer, which means that the influence of a given match increases by the square of the distance. Thus, one outlier with a sufficiently-large distance can dominate all of the inliers.

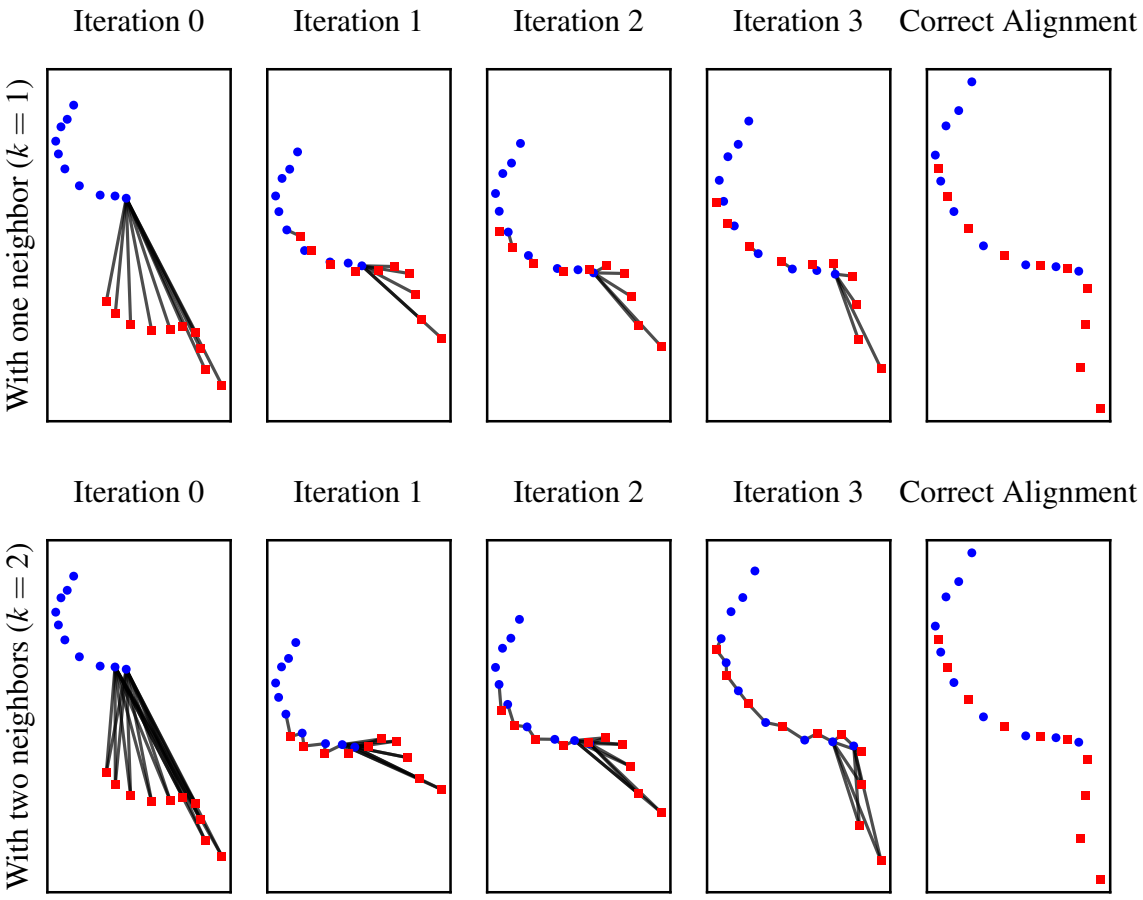


Figure 1.3 – Demonstrating the effect of the number of neighbors on the matching function for the registration of two overlapping curves. The blue circles and red squares are respectively the reference points $\mathcal{Q}$ and the reading points $\mathcal{P}$. The black lines represent the k associations between $\mathcal{Q}$ and $\mathcal{P}$. The reading points are initially far away from the correct alignment. At each iteration of ICP, they are moved closer to the correct alignment. With two neighbors, each point in the reading is matched to the two closest points in the reference. At the first iteration, all points are matched to the same point: it takes a few iterations before the points are matched to their correct neighbor. A higher neighbor count is more robust and precise, however each iteration takes longer to compute.

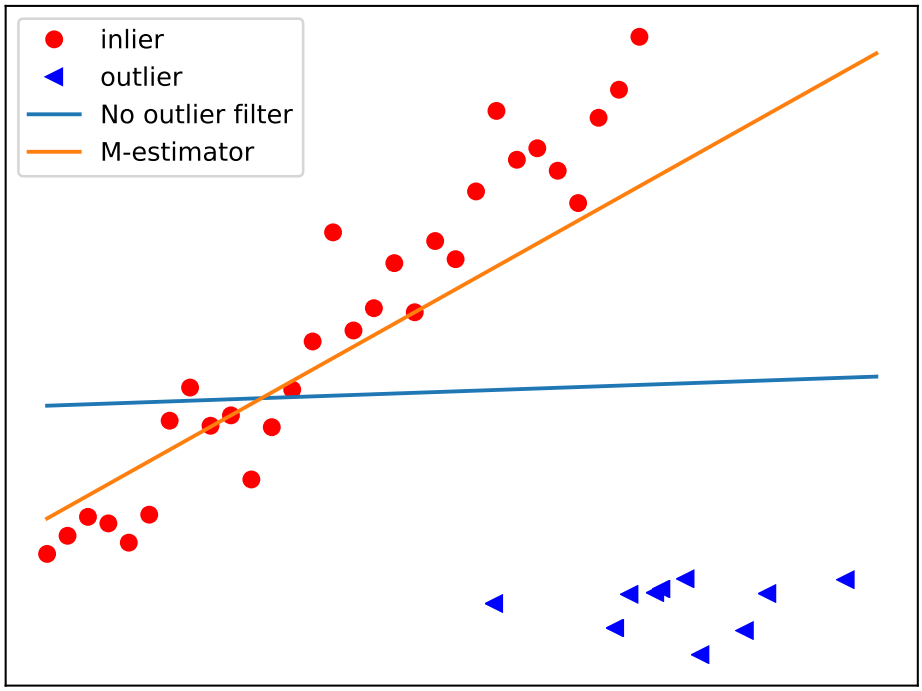


Figure 1.4 – An example of a least-squares fitting of a linear model in the presence of outliers. Without a filter, the minimization is influenced by the outliers. If an M-estimator is used, the influence of outliers is reduced, but not completely removed.

The field of robust statistics has created tools, such as M-estimator, to mitigate the effect of outliers. The M-estimator (also know as Maximum-likelihood estimator) is a common solution for outlier filtering of least-square problems [17]. Instead of completely removing outliers, they reduce their influence (see Figure 1.4 for a line fitting example). Chapter 2 explores in depth the effects and compromises of outlier filters for the ICP algorithm. Furthermore, Appendix A provides a detailed list of existing outlier filters and their respective properties.

1.1.4 Minimizer

The minimizer finds at each iteration the alignment which minimizes a given cost function $J(\cdot)$:

$$\widehat{\boldsymbol{T}} = \arg\min_{\boldsymbol{T}} \; J\left(\mathcal{Q}, \mathcal{P}, \check{\boldsymbol{T}}\right), \tag{1.1}$$

where $\check{\boldsymbol{T}}$ is the prior on the alignment (at the first iteration, this is the initial estimate) and $\widehat{\boldsymbol{T}}$ is the estimated transformation. The choice of the cost function $J(\cdot)$ depends on the properties of the environment. The original ICP paper [7] introduced the point-to-point cost function J_{p-p}, which minimizes the Euclidean distance between pairs of points:

$$J_{p-p}\left(\mathcal{Q}, \mathcal{P}, \check{\boldsymbol{T}}\right) = \sum_i \sum_j w_{ij}(\check{\boldsymbol{T}}) e_{ij}(\boldsymbol{T})^2, \tag{1.2}$$

$$e_{ij}(\boldsymbol{T}) = \left\| \boldsymbol{T}\boldsymbol{p}_i - \boldsymbol{q}_j \right\|, \tag{1.3}$$

where $w_{ij}(\cdot)$ is the weight function (the outlier filter output) for the i^{th} point in the reading ($\boldsymbol{p}_i \in \mathcal{P}$) and the j^{th} point in the reference ($\boldsymbol{q}_j \in \mathcal{Q}$), and $e_{ij}(\cdot)$ is the error function. As explained in Section 1.1.2, the double summation (corresponding to testing each reference $\boldsymbol{q}_j$ point against each reading point $\boldsymbol{p}_i$) is computationally expensive. Hence, only a subset of the points in $\mathcal{P}$ and $\mathcal{Q}$ are used. This subset is selected by the matcher.

The most common alternative to point-to-point is point-to-plane cost function J_{p-n} [8]:

$$J_{p-n}\left(\mathcal{Q}, \mathcal{P}, \check{\boldsymbol{T}}\right) = \sum_i \sum_j w_{ij}(\check{\boldsymbol{T}})((\boldsymbol{T}\boldsymbol{p}_i - \boldsymbol{q}_j) \cdot \boldsymbol{n}_j)^2,$$

where $\boldsymbol{n}_j$ is the normal of the j^{th} point in the reference. For robotic localization applications, point-to-plane outperforms point-to-point in most environments [6]. It uses the surface normal of the reference's points for a more robust registration. Section 3.3.1 goes into more depth about point-to-plane, and introduces a new cost function named point-to-Gaussian.

1.1.5 Transformation Checkers

The transformation checkers are responsible for verifying if ICP has converged and if so, will terminate the ICP iterations. Allowing a larger number of iterations increases the accuracy, but at the cost of computation time. The choice of the transformation checker depends mostly on the application. For instance, a real-time application can trade off accuracy for speed, by having fewer iterations. A common way of setting an upper bound on the computation time is to stop after a predetermined number of iterations. The number of iterations highly depends on factors such as the type of minimizer, or if the points are 2D or 3D.

Another way the transformation checker can detect convergence is by verifying if the previous iteration has moved the reading point cloud by more than a set distance. As shown in Figure 1.3, the reading point cloud moved faster in the first few iterations and slowly reached a local minimum towards the end. Thus a good sign of convergence is when the reading only moved by a small amount. Although, a sign of a diverging registration is when it moves far away from the initial estimate. Therefore, another type of transformation checker is used to detect when the registration has failed, by setting a bound on the distance from the initial estimate $\check{\boldsymbol{T}}$.

To conclude, the ICP algorithm can be divided into five steps. Each of these steps must be configured depending on the application, the geometric characteristics of the point clouds, while taking into account the expected registration speed. A few examples of each step have been shown in this section. From these examples, it is clear that selecting an appropriate solution is tedious, due to the large number of possible configurations for each step.

1.2 Mapping

Mapping is a fundamental research field in mobile robotics. Common tasks, such as path planning and obstacle avoidance, depend on it. Many solutions exist for the localization and mapping, with ICP used in some of these solutions. The most common approaches to ICP mapping that will be highlighted in our thesis are scan-to-scan, scan-to-map and graph-SLAM. While we do not use Simultaneous Localization and Mapping (SLAM) in this thesis (our work is much closer to odometry), we still provide a brief explanation to clearly differentiate our approach in Chapter 3 from SLAM.

Scan-to-scan is the simplest way of creating a map using ICP . As the output of ICP is a transformation between two scans, it can be used to find the robot's displacement between the same two scans. By running ICP between each pair of sequential scans, an odometry of the robot trajectory is estimated. However, like any other form of odometry, the localization error accumulates over time. Contrary to scan-to-map, a map is not required for the registration, and thus can be generated offline. The map is created by simply concatenating each scan together, based on the estimated trajectory.

A classic solution to the problem of trajectory estimation is SLAM, where the robot localizes itself inside a map and then uses this estimated position to update the same map. Above all, the main purpose of SLAM is to enable the fusion of multiple sen-

sor types and odometry sources within a probabilistic framework. SLAM is a large and complex field of research, with many frameworks and approaches. In our thesis, we only discuss graph-SLAM, since it is a general-purpose framework. Moreover, it can integrate ICP measurements with other sensors readily [18]. Graph-SLAM is commonly implemented using a factor graph [19] (i.e., a dependency graph), where poses at each point in time are connected to measurements of the environment, to odometry measurements and to other poses. Furthermore, the algorithm can use loop closures to correct previous poses. When a loop is detected, the current pose is connected to a previously visited pose in the map (i.e., the start of the loop). Thus, as long as an odometry measurement can be represented as a constraint between a pair of poses, it can be added to the graph. In the case of Iterative Closest Point (icp), the pose between each pair of scans becomes a constraint. However, each constraint also requires a covariance, which represents the degree of uncertainty of the measurement. For ICP , there is a closed-form solution to estimate this covariance [20], yet real life experiments show that it vastly underestimates the actual covariance [21].

Scan-to-map uses ICP to align the current scan with a map including previous ones. After alignment, the map is updated using the current scan. Since this mapping algorithm updates a map and then localizes itself within this map, it could qualify as a form of SLAM. However, it deviates from most SLAM algorithms in two crucial ways: *i)* scan-to-map is not a probabilistic state estimator and *ii)* it does not detect/take into account loop closures. But contrary to normal odometry, the error does not necessarily accumulate over time: when a new scan aligns to an already-visited part of the map, the localization error is reduced. This is similar to loop-closing, except that this error reduction is not propagated through the previous poses. Thus, scan-to-map could be qualified as a weak form of SLAM, while being much closer to odometry. The implementation of scan-to-map used in Chapter 2 and in Chapter 3 is a modified version of `ethzasl_icp_mapping`, which belongs to the category of scan-to-map. It is available as a Robotic Operating System (ROS) package, which uses `libpointmatcher` for registration.

In conclusion, we have the choice between two different approaches for ICP mapping: graph-SLAM and scan-to-map. The former is able to fuse multiple sensor types and odometry sources, however, estimating the correct covariance for ICP is problematic. On the one hand, the latter is able to create a locally-consistent map using ICP. On the other hand, it is unable to fuse multiple sensors. To remedy these problems, we propose a novel approach to ICP mapping, where the error minimizing step of ICP is

used for sensor fusion. This approach is presented in Chapter 3.

Chapter 2

Analysis of Robust Functions for Registration Algorithms

Résumé

L'exactitude du recalage de points est influencée par la présence de données aberrantes. Ainsi, un grand nombre de techniques ont été développés au fils des années afin de réduire leur effet. Par contre, sans une comparaison à grande échelle des filtres à données aberrantes, il est difficile de sélectionner un algorithme approprié pour une application donnée. Cet article présente une analyse approfondie des effets des filtres de données aberrantes pour l'algorithme ICP, dans le contexte d'applications en robotique mobile. Quatorze des filtres les plus communs (incluant les M-estimateurs) ont été testés dans différents types d'environnement, pour un total de plus de 2 millions de recalages. De plus, l'influence des paramètres a été explorée en profondeur. Les résultats expérimentaux montrent que la plupart des filtres ont des performances similaires, s'ils sont correctement paramétrés. Néanmoins, les filtres comme *Var. Trim.*, *Cauchy* et *Cauchy MAD* sont plus stable à travers tous les types environnements testés. Étonnamment, la norme L_1 produit une exactitude comparable aux autres filtres, tout en n'ayant aucun paramètre à configurer.

Abstract

Registration accuracy is influenced by the presence of outliers and numerous robust solutions have been developed over the years to mitigate their effect. However, without a large scale comparison of solutions to filter outliers, it is becoming tedious to select an appropriate algorithm for a given application. This article presents a comprehensive analysis of the effects of outlier filters on the ICP algorithm aimed at a mobile robotic application. Fourteen of the most common outlier filters (such as M-estimators) have been tested in different types of environments, for a total of more than two million registrations. Furthermore, the influence of tuning parameters has been thoroughly explored. The experimental results show that most outlier filters have a similar performance if they are correctly tuned. Nonetheless, filters such as *Var. Trim.*, *Cauchy*, and *Cauchy MAD* are more stable against different environment types. Interestingly, the simple norm L_1 produces comparable accuracy, while being parameterless.

2.1 Introduction

A fundamental task in robotics is finding the rigid transformation between two overlapping point clouds. The most common solution to the point cloud registration problem is the ICP algorithm, which alternates between finding the best correspondence for the two point clouds and minimizing the distance between those correspondences [7], [8]. Based on the taxonomy of Rusinkiewicz *et al.* [11], Pomerleau *et al.* [12] proposed a protocol and a framework to test and compare the common configurations of ICP. They simplified the process to four stages: 1) data point filtering, 2) data association, 3) outlier filtering, and 4) error minimization. The stage on outlier filtering is necessary as the presence of a single outlier with a large enough error could have more influence on the minimization outcome than all the inliers combined. To solve this generic problem, Huber [10] extended classical statistics with robust cost functions. A robust cost function reduces the influence of outliers in the minimization process. The most common class of these robust functions is the maximum likelihood estimator, or M-estimator. Other solutions exist, which rely either on thresholds (i.e., hard rejection) or on continuous functions (i.e., soft rejection).

To the best of our knowledge, the current research literature is missing a comprehensive comparison of outlier filters for ICP (i.e., Stage 3 of the ICP pipeline). In the case of ICP used in mobile robotics, outliers are mainly caused by non-overlapping regions, sensor noises and shadow points produced by the sensor. Most papers about outlier filters compare their own algorithm with only two other algorithms [22]–[24] or only on a single dataset [23]–[26]. Few papers evaluate the influence of the overlap between the two point clouds and the initial perturbation, which are leading error causes for ICP [9]. Furthermore, the dataset selected for evaluation varies depending on research fields. For instance, papers targeting object reconstruction will use the Stanford dataset [24], [27], [28]. Results obtained with a dataset containing exclusively objects might be too specific and consequently not translate well to the field of mobile robotics, because of the difference in structure, density, and scale. Also, experiments on registration performances tend to modify multiple stages of ICP at once, making it difficult to estimate the impact of outlier rejection algorithms on the overall performance. Additionally, few papers on outlier filters evaluate the impact of the tuning parameters within those outlier rejection algorithms. The influence of an outlier for a given error can change drastically, as a function of the tuning parameter value. Finally, with the rise of the number of ICP variants [9], it is becoming tedious to select the appropriate robust solution for an application in mobile robotics.

To mitigate these problems, we propose a comprehensive analysis of outlier filter algorithms. To this effect, the main contributions of this paper are: 1) a large scale analysis investigating 14 outlier filters subject to more than two million registrations in different types of environment; 2) a consolidation of the notion of point cloud *crispness* [29] to a special case of a common M-estimator, leading to a better understanding of its use in the outlier filtering stage; 3) a support to better replication of our results with the open-source implementations of tested outlier filters in `libpointmatcher`[1].

2.2 Related Works

Prior to the existence of ICP, a seminal work on M-estimators was conducted by Welsch [17], where he surveyed eight functions now known as the classic M-estimators. In the context of camera-based localization, MacTavish *et al.* [30] made a thorough effort by comparing seven robust cost functions, but their conclusions do not translate directly to point cloud registrations. For ICP, Pomerleau *et al.* [9] proposed an in-depth review of the literature explaining how outlier filters must be configured depending on the robotic application at hand. Unfortunately, their investigation is limited to listing current solutions, without comparison. These surveys guided us in the selection of our list of solutions analyzed in this paper.

When it comes to comparing outlier filters, the most common baseline is vanilla ICP (i.e., labeled L_2 hereafter), which does not have any outlier filter and directly minimizes a least-squared function.

In terms of hard rejections, Phillips *et al.* [27] compared a solution using an adaptive trimmed solution against a manually-adjusted trimmed threshold [11] and L_2. However, they limited their analysis to a pair of point clouds with overlap larger than 75% and using simulated outliers. Another adaptive threshold solution, this time related to simulated annealing algorithms [25], was compared to five types of hard rejection algorithms. Unfortunately, they used a limited 2D dataset relying on ten pairs of scans with similar overlap. As for soft rejections, Bergström *et al.* [31] compared the effect of three M-estimators on ICP and proposed an algorithm to auto-tune them, but they only provided results based on simulated data of simple geometric shapes. Agamennoni *et al.* [32] proposed a soft rejection function based on the Student's T-distribution for registration between a sparse and a dense point cloud. But, they compare their algorithm to another complete ICP solution, where multiple stages changed. Bouaziz

[1] `https://github.com/ethz-asl/libpointmatcher`

et al. [28] introduced a soft rejection function based on L_p norms, however, they did not provide qualitative evaluation. Moreover, their solution does not use a standard least-squared minimizer. Closer to the topic of our analysis, Bosse *et al.* [26] compared five M-estimators and another robust function against L_2, for a variety of minimization problems, one of which was ICP. However, the ICP analysis was limited to examples relying on a single 2D pair of point clouds. In our analysis, a common 3D benchmark is used for all solutions allowing a fair comparison of robust cost functions.

When an outlier filter relies on fixed parameters, it is important to know their effects on registration performance. Segal *et al.* [33] removed matching points with residual errors higher than a given value (i.e., named *max. distance* hereafter) and evaluated the influence of this parameter on ICP. They concluded that the value used for this parameter is a trade-off between accuracy and robustness. Bosse *et al.* [34] compared the same outlier filter against one of the classic M-estimator *Cauchy* in an 2D outdoor environment. They concluded that *max. distance* is less robust to the parameter value than *Cauchy*, but it provided better accuracy when correctly tuned.

For the most part, a robust cost function relies on fix parameters, configured by trial and error. To sidestep this issue, some outlier filters are designed so as to be auto-tunable. For instance, Haralick *et al.* [35] and Bergström *et al.* [31] have both proposed an algorithm to auto-scale M-estimators. However, Bergström *et al.* [31] requires two additional hyper-parameters to tune the auto-scaler: one representing the sensor's standard deviation and the other specifying the decreasing rate to reach the standard deviation. In the case of outlier filters based on a threshold following a given quantile of the residual error, Chetverikov *et al.* [36] proposed an estimator to tune the overlap parameter, which uses an iterative minimizer. Unfortunately, having an iterative minimizer inside ICP's iterative loop becomes computationally intensive [37], thus most of the implementation resorts to manually fixing the quartile. Phillips *et al.* [27] improved on [36] by minimizing the Fractional Root Mean Squared Distance (FRMSD) to tune the parameter representing the overlap ratio. It removes the need for an inner loop to estimate the parameter, thus speeding up the execution by a factor of at least five. But, this algorithm performance also depends on a hyper-parameter that needs to be tuned [37]. In this paper, we also explore the influence of tuning parameters, testing even the stability of hyper-parameters, and provide a methodology to tune them.

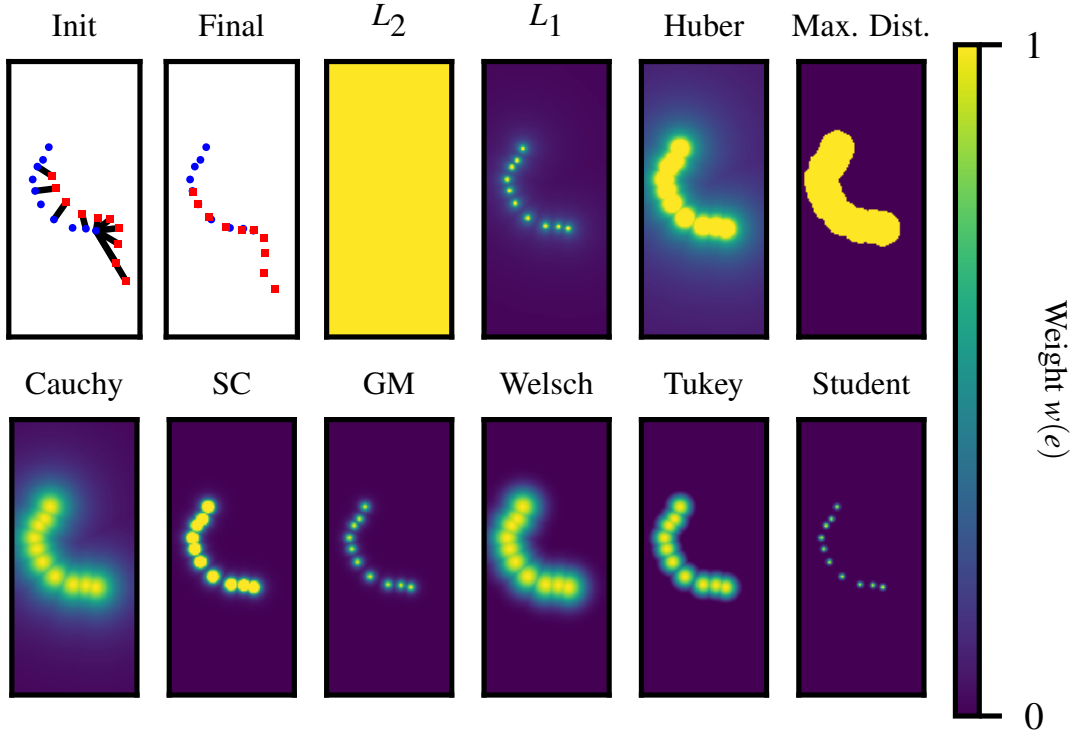


Figure 2.1 – The effect of different outlier filters on the registration of two overlapping curves. The blue circles and red squares are respectively the reference points and the reading points. The initial graph depicts the data association from the nearest neighbor at the first iteration. The final graph shows the registration at the last iteration. The other graphs depict scalar fields of the weight function $w(e)$ of a reading point. For L_2, all points have the same weight. L_1 gives an infinite weight to a point directly on top of a reference point.

2.3 Theory

The ICP algorithm aims at estimating a rigid transformation $\hat{\boldsymbol{T}}$ that best aligns a *reference* point cloud $\mathcal{Q}$ with a *reading* point cloud $\mathcal{P}$, given a prior transformation $\check{\boldsymbol{T}}$. The outlier filtering stage of ICP has strong ties to error minimization. The former reduces the influence of wrongful data association, while the latter finds a solution that respects the constraints of the previous stage. In the context of ICP, these two stages can be summarized as estimating the rigid transformation $\hat{\boldsymbol{T}}$ by minimizing

$$\hat{\boldsymbol{T}} = \arg\min_{\boldsymbol{T}} \sum_{i=1} \sum_{j=1} \rho\Big(e(\boldsymbol{T}, \boldsymbol{p}_i, \boldsymbol{q}_j)\Big), \tag{2.1}$$

where $\rho(\cdot)$ is the cost function. The error function $e(\cdot)$ is the scaled distance between matched points, defined as

$$e(\boldsymbol{T}, \boldsymbol{p}_i, \boldsymbol{q}_j) = \frac{\left\| \boldsymbol{T}\boldsymbol{p}_i - \boldsymbol{q}_j \right\|}{s}, \tag{2.2}$$

where $\boldsymbol{p}_i \in \mathcal{P}$ and $\boldsymbol{q}_j \in \mathcal{Q}$. This is equivalent to a simplified version of the Mahalanobis distance, where the scale s is uniform on all dimensions. For the original version of ICP, s is set to one. The double summation of Equation 2.1 is expensive to compute and is typically approximated using a subset of pairs, using nearest neighbor points of each $\boldsymbol{p}_i$. To simplify the notation, we will use $e_m(\cdot)$ for each error to be minimized, with m being the index of this subset. Since $\rho(\cdot)$ is non-convex, the minimization must be solved iteratively, in a manner similar to the Iteratively Reweighted Least-Squares (IRLS) [31] by decomposing $\rho(\cdot)$ as a weight w and a squared error term, such that

$$\hat{\boldsymbol{T}} \approx \arg\min_{\boldsymbol{T}} \sum_{m=1} w\big(e_m(\check{\boldsymbol{T}})\big)\ e_m(\boldsymbol{T})^2. \tag{2.3}$$

At each iteration, the prior transformation $\hat{\boldsymbol{T}}$ is assigned to the last estimated transformation until convergence. Figure 2.1 shows a toy example of registration. The subsequent scalar fields show the impact of different weight functions $w(\cdot)$ in the neighborhood of $\mathcal{Q}$. A notable example is L_2, where the lack of weights is equivalent to using a constant weight for all pairs. Beyond L_2, outlier filtering is all about the choice of this weight function $w(\cdot)$ and its configuration. Table 2.1 shows a list of robust cost functions used in this paper, with a dedicated column highlighting their implementation of $w(\cdot)$.

2.3.1 Hard rejection

Outlier filters categorized as hard rejection define the result of $w(\cdot)$ to be binary (i.e., either zero or one). The two most commons solutions are *Max. distance* and *Trimmed*. The solution *Max. distance* rejects any match with a distance larger than a threshold. *Trimmed* only keeps the error below the f^{th} percentile P_f of the matches, where f is the *overlap ratio parameter*. This makes the registration accuracy directly related to how close this parameter f is to the actual overlap between reference and reading. In that sense, if f deviates from the true overlap, the accuracy will degrade. In applications where the overlap is unknown or changes often, selecting a fix overlap ratio f becomes challenging. A variant of *trimmed*, *Var. Trimmed* [27], calculates the FRMSD for all possible overlap ratios, and selects the ratio with the minimum FRMSD value as f. The *Median filter* is also used, but is a special case of *Trimmed*, where $f = 50\,\%$.

Table 2.1 – Descriptive table of robust cost functions used in this analysis expressed with respect to their tuning parameter k and the scaled error e.

Functions	Conditions	Cost $\rho(e)$	Weight $w(e)$	M
L_2		$\frac{e^2}{2}$	1	✓
L_1		$\lvert e\rvert$	$\frac{1}{\lvert e\rvert}$	✗
Huber	$\begin{cases} \lvert e\rvert \leq k \\ \text{otherwise} \end{cases}$	$\begin{cases} \frac{e^2}{2} \\ k(\lvert e\rvert - k/2) \end{cases}$	$\begin{cases} 1 \\ \frac{k}{\lvert e\rvert} \end{cases}$	✓
Cauchy		$\frac{k^2}{2}\log(1+(e/k)^2)$	$\frac{1}{1+(e/k)^2}$	✓
GM		$\frac{e^2/2}{k+e^2}$	$\frac{k^2}{(k+e^2)^2}$	✓
SC	$\begin{cases} e^2 \leq k \\ \text{otherwise} \end{cases}$	$\begin{cases} \frac{e^2}{2} \\ \frac{2ke^2}{k+e^2} - k/2 \end{cases}$	$\begin{cases} 1 \\ \frac{4k^2}{(k+e^2)^2} \end{cases}$	✓
Welsch		$\frac{k^2}{2}(1-\exp(-(\frac{e}{k})^2))$	$\exp(-(e/k)^2)$	✓
Tukey	$\begin{cases} \lvert e\rvert \leq k \\ \text{otherwise} \end{cases}$	$\begin{cases} \frac{k^2(1-(1-(\frac{e}{k})^2)^3)}{2} \\ \frac{k^2}{2} \end{cases}$	$\begin{cases} (1-(e/k)^2)^2 \\ 0 \end{cases}$	✓
Student			$\frac{(k+3)(1+\frac{e^2}{k})^{-\frac{k+3}{2}}}{k+e^2}$	✗
Max. Dist.	$\begin{cases} \lvert e\rvert \leq k \\ \text{otherwise} \end{cases}$	$\begin{cases} \frac{e^2}{2} \\ \frac{k^2}{2} \end{cases}$	$\begin{cases} 1 \\ 0 \end{cases}$	✓
Trimmed	$\begin{cases} e \leq P_f \\ \text{otherwise} \end{cases}$		$\begin{cases} 1 \\ 0 \end{cases}$	✗

Legend: M = Is the function a M-Estimator?

2.3.2 Soft Rejection and M-estimators

Outlier filters using soft rejection output a weight where $w(\cdot) \in \mathbb{R}^+$. The most common soft rejection algorithms are M-estimators. To be an M-estimator, a cost function $\rho(\cdot)$ must fulfill three conditions, which are to be *1)* symmetric, *2)* non-negative, and *3)* monotonically-increasing [35]. Those conditions do not limit M-estimators to robust cost functions as, for example, L_2 satisfies all of them. Moreover, cost functions associated with M-estimators are analyzed using their influence function $\psi(\cdot)$ and weight function $w(\cdot)$, such that

$$\psi(e) = \frac{\partial \rho(e)}{\partial e} \qquad \text{and} \qquad w(e) = \frac{\psi(e)}{e}. \tag{2.4}$$

The influence function $\psi(\cdot)$ is used to evaluate whether an M-estimator is robust or not. If $\psi(\cdot)$ is non-monotonic (i.e., redescending) and is null for an error that tends to infinity, the M-estimator is considered robust. As for the weight function $w(\cdot)$, it is given for convenience since it is the only part required to implement an M-estimator for an IRLS solution, as in Equation 2.3. In the soft rejection algorithms considered in

this paper (shown in Table 2.1), two are not M-estimators: L_1 has a singularity for an error that is equal to zero, and *Student* has an undefined $\rho(\cdot)$.

It is worth noting that some soft rejection functions are related. For instance, *Huber* uses a parameter k to combine L_1 and L_2, in order to avoid the singularity at $e = 0$ of L_1. Also, *Switchable-Constraint* (labeled *SC* hereafter), typically used in pose graph SLAM, was expressed as a combination of L_2 and *Geman-McClure* (labeled *GM* hereafter) using a parameter k. It shares the same cost function as Dynamic Covariance Scaling [30]. *SC* is expected to have similar results to *GM* for extreme values of k.

2.3.3 Estimating the scale

As expressed in Equation 2.2, we used a scaled error in our ICP implementation. Contrary to the filter parameter k, which should be globally constant, the scale s is related to the point clouds and can be either fixed or estimated at every iteration. The scale s relates to the uncertainty for which paired points with a certain error should be considered as outliers. There are multiple estimators for the scale, two of the most interesting are: 1) Haralick *et al.* [35] used the Median of Absolute Deviation (MAD) as a scale estimator and calculated it at each iteration; 2) Bergström *et al.* [31] starts with $s = 1.9 \cdot \text{median}(e)$ and then gradually decrease s at each iteration, to asymptotically reach a standard deviation σ_*. The parameter ξ controls the convergence rate of the scale.

2.3.4 Relating *crispness* to the M-estimator Welsch

The notion of *crispness* as a measure of how well two point clouds are aligned was introduced by Sheehan *et al.* [38] in the context of sensor calibration. It originates from the use of a Gaussian kernel in a measurement of Rényi Quadratic Entropy (RQE) for a kernel correlation approach to registration [39]. RQE has the following cost function:

$$\rho_{\text{rqe}}(e_*) = \exp\left(-\frac{e_*^2}{4\sigma^2}\right), \tag{2.5}$$

where e_* is the unscaled error and σ is a tuning parameter. RQE has been described as an M-estimator by [39], however, it has not been related to an existing M-estimator. This cost function is a special case of *Welsch* with $k = 2$ and $s = \sigma$. This means that minimizing for RQE is the same thing as using a *Welsch* M-estimator with this configuration. Thus, this configuration is expected to have good accuracy.

2.4 Experiments

Our study focuses on registration-based localization in the context of mobile robotics. Our analysis follows a similar methodology as in Pomerleau *et al.* [12], with a strict focus on changing the outlier filtering stage of ICP. The other stages were kept the same. Table 2.2 describes the ICP pipeline used.

A number of key factors were selected to be tested jointly, in order to determine their influence. These factors were: **1)** the **outlier filter**, selected based on their popularity and their interesting properties (total of 12); **2)** the **configuration parameter** of the filters; **3)** the **environment** type (indoor, outdoor, etc.); and **4)** the **overlap** between the reference and reading scans. For each factor selection above, 128 registrations were computed with a **random perturbation** from the ground truth. How some of these factors were sampled during experiments is detailed below.

Configuration parameter sampling Filter parameter values were sampled in a way to ensure efficient exploration of their configuration space. For instance, the tuning parameters k of all M-estimators (i.e., *Huber*, *Cauchy*, *SC*, *GM*, *Welsch* and *Tukey*) and *Student* were sampled evenly on a log scale. Two sampling value regions were defined. The first region was $Z_1 \in \left[1 \times 10^{-6}, 0.1\right[$ and the second one was $Z_2 \in \left[0.1,\ 100\right]$. Z_1 explores the asymptotic behavior of the algorithm for near-zero k values, while Z_2 is where the parameter with the best accuracy is located for most estimators. They are sampled 20 times for Z_1 and 30 times for Z_2. For the error scale s for M-estimator (see Eq. 2.2), two auto-scalers (*Berg.* and MAD) and one fixed scale have been tested. All three permutations were tested on *Cauchy*, while all other M-estimators were tested only with MAD. In the case of the *Berg.* auto-scaler, we used a convergence rate of $\xi = 0.85$, as it was found to be the best one in our analysis. Its parameter σ_* was sampled in the same two zones as the M-estimator (Z_1 and Z_2). *Cauchy Berg.* used the tuning parameter $k_{cauchy} = 4.304$, and it was selected based on [31]. If the estimator used was MAD, then $s = MAD(\boldsymbol{e})$, otherwise $s = 1$. In the case of the *Trimmed* filter, its overlap parameter f has been sampled linearly 20 times in the range $\left[1 \times 10^{-4}, 100\right]\%$. For the *Var. Trim.* filter, the minimum and maximum overlap parameters have been set to 40% and 100% respectively. Its λ parameters have been sampled linearly 20 times between 0.8 and 5. The filter *Max. Dist.* has been sampled 20 times linearly in the range $\left[0.1, 2\right]$. Finally, the parameter-less L_1 and L_2 filters were used as-is.

Environments Experiments were performed on the *Challenging Datasets* [6]. These provide a ground truth with *mm*-level of precision. Our analysis used 3 sets of point

Table 2.2 – Listing of the configuration of `libpointmatcher` used.

Stage	Configuration	Description
Data association	`KDTree`	Three matches per point
Data filtering	`SurfaceNormal`	Density with 20 neighbors
	`MaxDensity`	Limit density to 10k pts/m^3
	`RandomSampling`	Keep 75% of points
Error Min.	`PointToPlane`	Point-to-plane error
Trans. Checking	`Differential`	Stop below 1 mm and 1 mrad
	`Counter`	Max. iteration count is 40

clouds from this dataset, one per type of environment: structured (*Hauptgebäude*), semi-structured (*Gazebo Summer*) and unstructured (*Wood Summer*). As each of them contains around 35 point clouds, it allows a fine control of the ratio of overlap between point clouds. For each set, 12 pairs of point clouds where selected, to uniformly sample the overlap between 40% and 100%.

Initial perturbation on $\boldsymbol{T}_0$ For fine registration, ICP requires a prior on the transformation between the *reading* and *reference* point clouds. The performance accuracy of ICP is directly impacted by the distance between the initial transformation $\boldsymbol{T}_0$ and the ground truth. For each of the previous factors, 128 initial transformations $\boldsymbol{T}_0$ were generated by adding a random perturbation sampled from a uniform distribution, and centered at the ground truth. To stress-test ICP, we chose a perturbation as challenging as the hard perturbation of [12]. The perturbation in translation was generated by sampling a point in a sphere with a 1 m radius, while the one in rotation was generated by first sampling from an uniform angle distribution between 0 and 25 deg and then applying it around a random 3D vector.

Our evaluation of accuracy used the transformation error $\boldsymbol{\Delta}$ defined as $\boldsymbol{\Delta} = \boldsymbol{T}_{gt}^{-1}\boldsymbol{T}_{final}$ where $\boldsymbol{T}_{gt}$ is the ground truth and $\boldsymbol{T}_{final}$ is the transformation at the last iteration. $\boldsymbol{\Delta}$ is further separated into two components, for easier interpretation: $\boldsymbol{\Delta}_R$ for the 3x3 rotation matrix and $\boldsymbol{\Delta}_T$ for the 3x1 translation vector. Finally, the translation error was evaluated with the Euclidean distance of $\boldsymbol{\Delta}_T$, and the rotation error metric is $\theta = \arccos(\frac{trace(\boldsymbol{\Delta}_R)-1}{2})$.

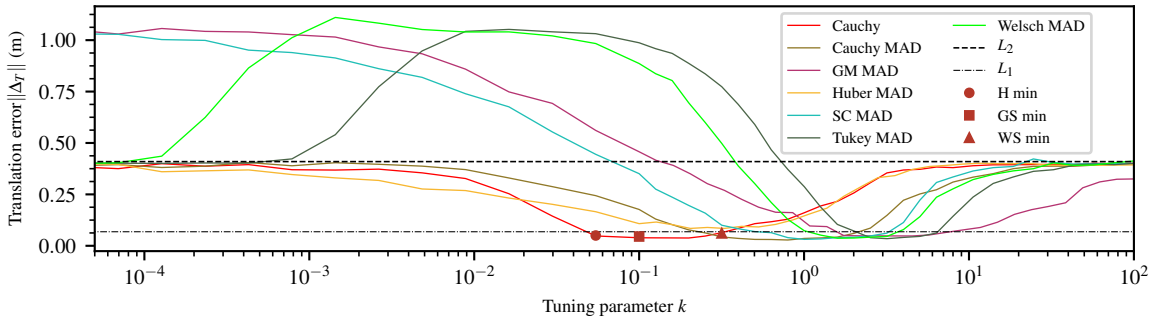


Figure 2.2 – Influence of the parameter's value on the translation registration accuracy of M-estimators. Each parameter sample point is the median of the error for all datasets for all overlap. The H min, GS min and WS min correspond to the parameter of *Cauchy* with the minimum median error for the environment *Hauptgebäude*, *Gazebo Summer*, and *Wood Summer*.

2.5 Results

In the first set of tests (Section 2.5.1), we performed over 2.3 million pairwise registrations to exhaustively evaluate the outlier filter solutions. These registrations were computed offline using Compute Canada's Supercomputers. In the second set of experiments (Section 2.5.2), we collected data with a mobile robot on an indoor-outdoor trajectory. We then tested on this trajectory the two best performing filters (Section 2.5.1) for a real-time mapping task (Figure 2.4).

2.5.1 Pairwise Outlier Filter Tests

Pairwise registration results are broken down into three parts. We first discuss the performance (median error) of each filter as a function of its parameter, for all environments compounded (Figure 2.2). We then closely analyze the distribution of errors for the best parameter values found from this search, for the three environments (Figure 2.3). In the third part, we evaluate how robust the best parameters are to a change in the environment (Table 2.3).

2.5.1.1 Parameter Search over All Environments Compounded

We computed the error metrics $\|\boldsymbol{\Delta}_T\|$ for all registrations, irrespective of the environment. Figure 2.2 shows the median translation error for M-estimators, as a function of the filter's tuning parameter value. We can see that all filters with parameters have a single global minimum, located in a relatively flat valley. Most solutions have a similar best performance, while *Huber MAD* slightly underperforms.

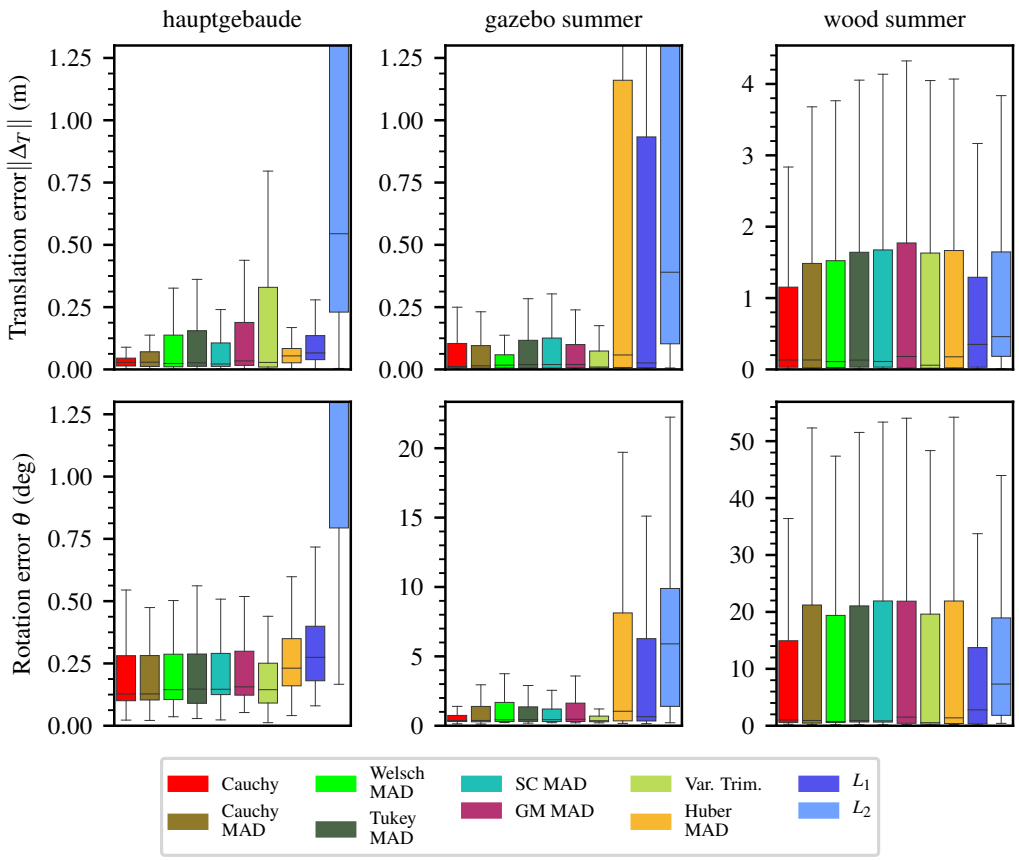


Figure 2.3 – Box plot of the performance of each outlier filter for the parameter value with lowest median error in that particular environment.

Comparison with L_2 (Vanilla ICP): For large values of configuration parameter k, all filters (excepted *GM*) performed similarly to L_2 (top dashed line), i.e. outlier filtering is effectively disabled. This is not surprising, as these M-estimators have the property that $\lim_{k\to\infty} w(e,k) \approx 1$. For small values of k, the performance depends on the estimator used. It can nevertheless be categorized into two trends. For *Huber MAD*, *Cauchy* and *Cauchy MAD*, the translation error degrades smoothly towards L_2 as $k \to 0$, without surpassing it. For the four other filters, performance can become *much worse* than L_2 with $k \to 0$, although for *Welsch* and *Tukey* the performance eventually goes back down to L_2. From all this, we can conclude that: *1)* it is preferable to overestimate the parameter for all M-estimators than to underestimate it, to avoid too much rejection of inliers; *2)* peak performance varied little from one filter to another (less than 2 cm), except for *Huber MAD*; and *3) Huber MAD*, *Cauchy* and *Cauchy MAD* never performed worse than L_2.

Comparison with L_1: Despite being unsophisticated, L_1 almost always outperforms

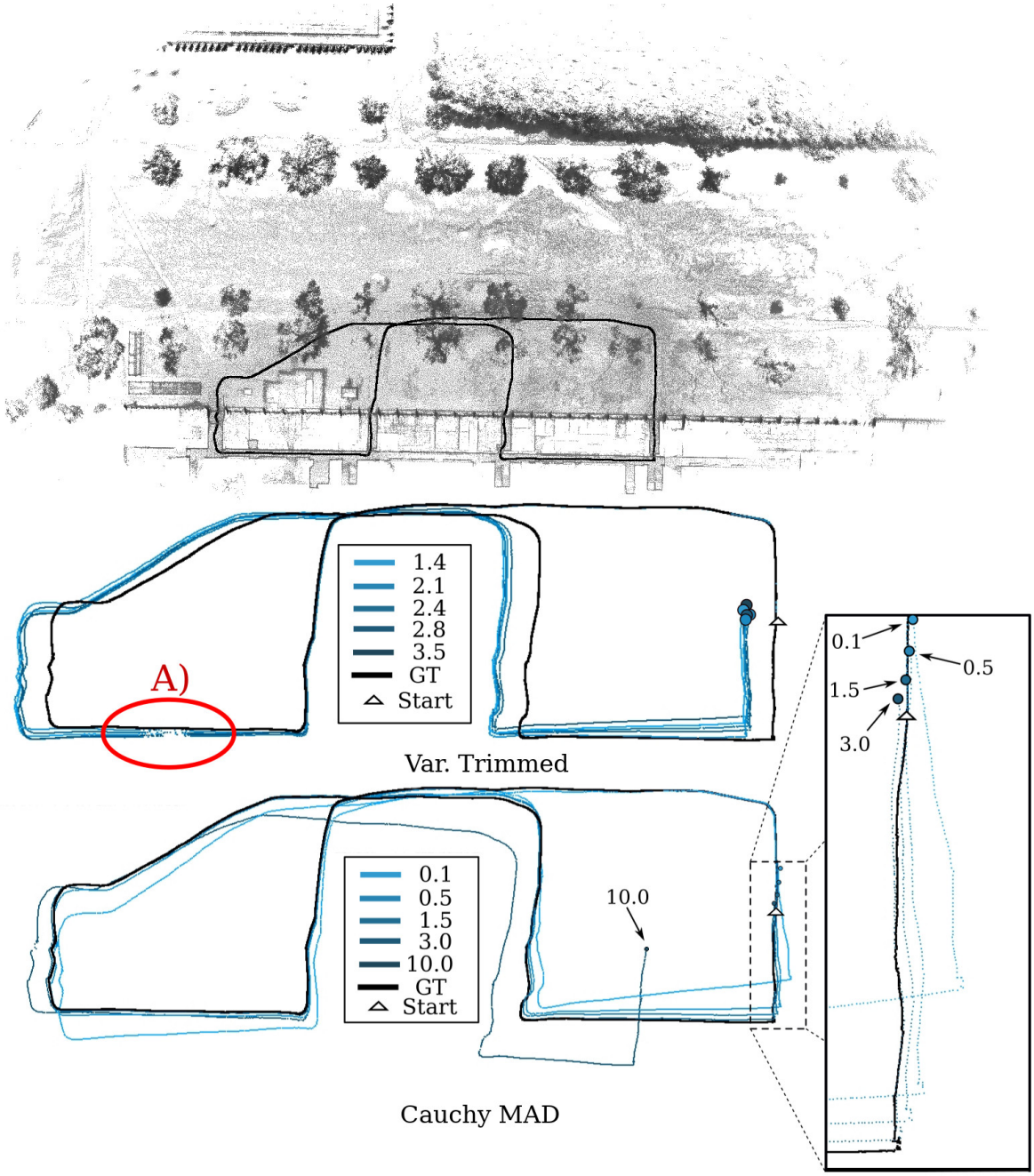


Figure 2.4 – Testing the ICP accuracy in a real time 3D SLAM application with a Husky A200 following a challenging indoor-outdoor route on Université Laval's campus. The start and end of the route are at the same location. The black trajectory (ground truth) was calculated offline. The end position of the trajectory for each parameter value is represented by a circle. In A), all configurations of *Var. Trimmed* could not converge correctly in the corridor.

all outlier filters approaches, except for a narrow band of parameter values. *Huber* is the only M-estimator that does not outperform L_1. However, as we will see later in Figure 2.3, L_1 exhibits a much greater variance in certain environments, which could be problematic if one is interested in minimizing the risk of being lost.

Other observations: *Cauchy* and *Cauchy MAD* have the same curve, but with an offset, with *Cauchy MAD*'s minimum centered at $k = 1$ by the auto-scaler. Since *GM* and *SC* shares the same function, they have a similar curve, differing only by a slight offset. *Welsch* and *Tukey* also have similar curves with an offset, despite having very different weight functions $w(e)$.

2.5.1.2 Distribution of Errors for the Best Fixed Filter Parameter, for Three Environments

In Figure 2.2, we showed only the median error, for all environments combined together. As such, this median does not tell the whole story. We thus computed in Figure 2.3 the distribution of errors of all filtering approaches, for three different environments. It is important to note that the filter parameters were fixed in these experiments, and were established from the best results found in Figure 2.2. From these experiments, the environment influence on the registration is clear: the structured one (*Hauptgebäude*) is noticeably easier in rotation, while the unstructured one (*Wood Summer*) is by far the hardest in rotation and in translation. Furthermore, all error distributions are asymmetric (heavy-tailed). Apart from the unstructured environment of *Wood Summer*, all outlier filters have error spreads and median errors significantly smaller than L_2. This further confirms the lack of robustness of L_2. Although L_1 has a favorable median error in the semi-structured environment *Gazebo Summer*, its error spread is much greater that most other M-estimators. The performance of *Huber* is even worse than that of L_1 in that environment despite being a "robust" version of the former. Finally, we observed that *Cauchy* has an error spread noticeably smaller than its auto-scaled counterpart (*Cauchy MAD*).

2.5.1.3 Robustness of Best Fixed Parameter Across Environments

To determine if a filter is robust across environment changes, we used the following metric: if the best parameters of a filter for each of the three environments are all within the global flat valley, this filter is considered robust. We define this global flat valley as the range of parameters that perform better than L_1, when comparing the median translation error. For instance, Figure 2.2 shows the best parameters of *Cauchy*

Table 2.3 – All 14 outlier filters were tested on the three environments and for a variety of parameter values. The parameter corresponding to the smallest median error for each dataset is shown. In the median error columns, the outlier filter with the best performance is in bold. In the parameter columns, if the parameter value is located inside the global flat valley, then it is in bold. If all three environments are within the valley, then 'All' is in bold.

Outlier Filters	Median Error (mm)				Parameters			
	H	GS	WS	All	H	GS	WS	All
L_2	544	390	459	409	n/a	n/a	n/a	n/a
L_1	66	25	350	68	n/a	n/a	n/a	n/a
Huber MAD	54	58	176	84	0.05	0.33	0.67	0.33
Cauchy	28	11	131	37	**0.05**	**0.10**	**0.32**	**0.20**
Cauchy MAD	28	14	132	**28**	**0.40**	**1.00**	**1.59**	**0.80**
Cauchy Berg	39	18	99	39	**0.01**	**0.00**	0.05	0.01
SC MAD	**21**	19	111	31	0.50	**2.53**	**3.18**	1.00
GM MAD	34	19	180	47	1.08	**4.52**	11.72	4.52
Welsch MAD	23	17	109	36	**2.00**	**2.00**	**3.18**	**1.59**
Tukey MAD	26	19	130	34	**2.53**	**5.04**	**6.35**	**3.18**
Student	40	32	178	60	0.10	0.13	1.37	0.16
Max. Distance	45	38	112	58	0.30	**0.40**	0.60	0.40
Trim	39	24	284	63	0.63	**0.68**	0.89	0.68
Var. Trimmed	28	**9**	**58**	**27**	**1.91**	**2.35**	**2.35**	**1.91**

Legend: H = Hauptgebäude, GS = Gazebo Summer, WS = Wood Summer

for three environments. Since all three parameter values have an error below L_1, *Cauchy* is robust as per our metric.

In Table 2.3, the best performing parameters for all filters are presented, for three environments. The best overall filter for these experiments is *Var. Trim*. It has half of the error of the second best filter in *Wood Summer*. It is also the only hard rejection filter that met our robustness criteria describes above. Filters such as *Cauchy*, *Cauchy MAD*, *Welsch MAD*, and *Tukey MAD* are also robust across our environment changes, as they all have values below L_1 (indicated by bold notation). On the subject of auto-scaling, *Cauchy Berg* is out performed by *Cauchy* and *Cauchy MAD* for all environments except *Wood Summer*. The optimal parameter of *Welsch* for two environments is $k = 2$, the exact value for RQE kernel function (as discussed in Section 2.3.4), which means that our experiments agree with the theory around kernel function. The parameters of *Max distance* and *Trim* depend on the environment, possibly indicating a lack of robustness. We can attribute this to the fact that they are hard rejection, and that they lack the adaptability of *Var. Trimmed*.

2.5.2 Test on a full Indoor-Outdoor Trajectory

In Figure 2.4, two filters (*Cauchy MAD* and *Var. Trim.*) were tested in a challenging route containing both indoor and outdoor portions. These filters were chosen for their good performance in our offline tests. The robot used a Velodyne HDL-32e for scanning and used a mix of wheel encoder and an Xsens MTi-30 IMU for odometry. To estimate the trajectory, an ICP-based SLAM was used with a moving window map. The moving window map was created by randomly decimating the map's point. For ground truth, the ICP-based SLAM was done offline without moving windows. The outlier filter performance was evaluated by comparing the cumulative registration error to the ground truth. Multiple parameter values were tested, all near the best value from Table 2.3. For *Var. Trim.*, all parameter values had difficulties in the highlighted zone A in Figure 2.4, which occurs in a small corridor. This confirmed that *Var. Trim.* has problems in structured environments as in Figure 2.3-*Hauptgebäude*. For *Cauchy MAD*, all tested parameters finished close to the ground truth. The parameter with the best performance was $k = 3.0$, which demonstrates that the valley for this experiment is shifted to higher k values than our offline tests. We think that this change is caused by the different sensors used, and because this test is a registration between scan to (small) map, while our offline tests were between two scans.

2.6 Conclusion

In this paper, we performed exhaustive robustness experiments on a wide range of outlier filters, in the context of ICP. After analysis, we concluded that all robust solutions have similar performances, with a number of particularities worth noting. For instance, L_1 exhibits good overall performance, despite having no parameter. Also, using MAD as auto-scale improves accuracy, but does not relieve from parameter tuning. When appropriately tuned, *Var. Trim.* has the best accuracy, with a translation error under 27 mm, while the best M-estimator is *Cauchy MAD* with roughly the same error. Moreover, we demonstrated the necessity of a well-tuned outlier filter for robust registration. In particular, *Welsch*, *Tukey*, *GM*, and *SC* should be employed carefully, as they have the potential to produce worse estimate than L_2 when mis-tuned. This fact should be kept in mind when assessing the risks associated with parameter selection.

Encouraged by the result of L_1, further investigation will be made to adapt registration solutions related to L_p norms [28] into the standard ICP pipeline. Furthermore, the link between RQE, a Gaussian kernel and *Welsch* opens the door to a family of kernels to

be studied for outlier filters along with an integration of the kernel correlation solution [39] and EM-ICP [23] into a generic version of ICP.

Acknowledgment

This work was partially financed by the Fonds de Recherche du Québec - Nature et technologies (FRQNT).

References (Chapter 2)

[6] F. Pomerleau, M. Liu, F. Colas, and R. Siegwart, "Challenging data sets for point cloud registration algorithms", *The International Journal of Robotics Research*, vol. 31, no. 14, pp. 1705–1711, 2012.

[7] P. J. Besl, "Geometric Modeling and Computer Vision", *Proceedings of the IEEE*, vol. 76, no. 8, pp. 936–958, 1988.

[8] Y. Chen and G. Medioni, "Object modeling by registration of multiple range images", *Proceedings. 1991 IEEE International Conference on Robotics and Automation*, no. 3, pp. 2724–2729, 1992.

[9] F. Pomerleau, F. Colas, and R. Siegwart, "A Review of Point Cloud Registration Algorithms for Mobile Robotics", *Foundations and Trends in Robotics*, vol. 4, no. 1, pp. 1–104, 2015.

[10] P. J. Huber, "Robust Estimation of a Location Parameter", *The Annals of Mathematical Statistics*, vol. 35, no. 1, pp. 73–101, 1964.

[11] S. Rusinkiewicz and M. Levoy, "Efficient variants of the ICP algorithm", *Proceedings of International Conference on 3-D Digital Imaging and Modeling, 3DIM*, pp. 145–152, 2001.

[12] F. Pomerleau, F. Colas, R. Siegwart, and S. Magnenat, "Comparing ICP variants on real-world data sets: Open-source library and experimental protocol", *Autonomous Robots*, vol. 34, no. 3, pp. 133–148, 2013.

[17] R. E. Welsch, "Robust regression using iteratively reweighted least-squares", *Communications in Statistics - Theory and Methods*, vol. 6, no. 9, pp. 813–827, 1977.

[22] S. Nobili, R. Scona, M. Caravagna, and M. Fallon, "Overlap-based ICP tuning for robust localization of a humanoid robot", in *Proceedings - IEEE International Conference on Robotics and Automation*, 2017, pp. 4721–4728.

[23] S. Granger and P. Xavier, "Multi-scale EM-ICP : A Fast and Robust Approach for Surface Registration", *Proceedings of the European Conference on Computer Vision*, pp. 418–432, 2002.

[24] A. W. Fitzgibbon, "Robust registration of 2D and 3D point sets", *Image and Vision Computing*, vol. 21, no. 13-14, pp. 1145–1153, 2003.

[25] F. Pomerleau, F. Colas, F. Ferland, and F. Michaud, "Relative Motion Threshold for Rejection in ICP Registration", *Field and Service Robotics*, vol. 62, pp. 229–238, Jul. 2015.

[26] M. Bosse, G. Agamennoni, and I. Gilitschenski, "Robust Estimation and Applications in Robotics", *Foundations and Trends in Robotics*, vol. 4, no. 4, pp. 225–269, 2016.

[27] J. M. Phillips, R. Liu, and C. Tomasi, "Outlier robust ICP for minimizing fractional RMSD", in *3DIM 2007 - Proceedings 6th International Conference on 3-D Digital Imaging and Modeling*, 2007, pp. 427–434.

[28] S. Bouaziz, A. Tagliasacchi, and M. Pauly, "Sparse iterative closest point", *Eurographics Symposium on Geometry Processing*, vol. 32, no. 5, pp. 113–123, 2013.

[29] M. Sheehan, A. Harrison, and P. Newman, "Continuous vehicle localisation using sparse 3D sensing, kernelised rényi distance and fast Gauss transforms", in *IEEE International Conference on Intelligent Robots and Systems*, 2013, pp. 398–405.

[30] K. MacTavish and T. D. Barfoot, "At all Costs: A Comparison of Robust Cost Functions for Camera Correspondence Outliers", in *Proceedings of the 12th Conference on Computer and Robot Vision*, 2015, pp. 62–69.

[31] P. Bergström and O. Edlund, "Robust registration of point sets using iteratively reweighted least squares", *Computational Optimization and Applications*, vol. 58, no. 3, pp. 543–561, 2014.

[32] G. Agamennoni, S. Fontana, R. Y. Siegwart, and D. G. Sorrenti, "Point Clouds Registration with Probabilistic Data Association", *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 4092–4098, 2016.

[33] A Segal, D Haehnel, and S Thrun, "Generalized-ICP", in *Robotics: Science and Systems*, vol. 5, 2009, pp. 168–176.

[34] M. Bosse and R. Zlot, "Map matching and data association for large-scale two-dimensional laser scan-based SLAM", *International Journal of Robotics Research*, vol. 27, no. 6, pp. 667–691, 2008.

[35] R. Haralick, H. Joo, C.-N. Lee, X. Zhuang, V. G. Vaidya, and M. B. Kim, "Pose Estimation from Corresponding Point Data", *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 19, no. 6, pp. 1426–1446, 1989.

[36] D. Chetverikov, D. Stepanov, and P. Krsek, "Robust Euclidean alignment of 3D point sets: The trimmed iterative closest point algorithm", *Image and Vision Computing*, vol. 23, no. 3, pp. 299–309, 2005.

[37] S. Du, J. Zhu, N. Zheng, Y. Liu, and C. Li, "Robust iterative closest point algorithm for registration of point sets with outliers", *Optical Engineering*, vol. 50, no. 8, p. 087 001, 2011.

[38] M. Sheehan, A. Harrison, and P. Newman, "Self-calibration for a 3D laser", *International Journal of Robotics Research*, vol. 31, no. 5, pp. 675–687, 2012.

[39] Y. Tsin and T. Kanade, "A Correlation-Based Approach to Robust Point Set Registration", in *8th European Conference on Computer Vision*, vol. 0, 2004, pp. 558–569.

Chapter 3

Large-scale 3D Mapping of Subarctic Forests

Résumé

La capacité de cartographier un environnement subarctique difficile ouvre la voie à des déploiements de robots dans des industries tels que la foresterie, la surveillance et l'exploitation minière à ciel ouvert. Dans cet article, nous explorons la possibilité de cartographie à grande échelle en forêt boréale, à l'aide de lidar. La cartographie avec un lidar est souvent basée sur des techniques de SLAM utilisant un graphe de poses. Celui-ci fusionne ensemble l'ICP, les positions GNSS et les mesures de l'IMU. Nous proposons une approche alternative qui fusionne les informations issues des capteurs directement dans l'étape de minimisation de l'ICP. De plus, une nouvelle formulation de la fonction de coût de l'ICP est présentée et est utilisée afin de compenser les incertitudes provenant du GNSS et des nuages de points du lidar. Pour tester cette approche, nous avons enregistré un ensemble de données à grande échelle à la *Forêt Montmorency*. Nous présentons les problèmes techniques rencontrés lors du déploiement en hiver. Les cartes générées grâce à notre nouvelle technique sont globalement et localement cohérentes, même sur un parcours de 4.1 km.

Abstract

The ability to map challenging subarctic environments opens new horizons for robotic deployments in industries such as forestry, surveillance, and open-pit mining. In this paper, we explore the possibilities of large-scale lidar mapping in a boreal forest. Computational and sensory requirements with regards to contemporary hardware are considered as well. The lidar mapping is often based on the SLAM technique relying on pose graph optimization, which fuses the ICP algorithm, GNSS positioning, and IMU measurements. To handle these sensors directly within the ICP minimization process, we propose an alternative approach of embedding external constraints. Furthermore, a novel formulation of a cost function is presented and cast into the problem of handling uncertainties from GNSS and lidar points. To test our approach, we acquired a large-scale dataset in the *Forêt Montmorency* research forest. We report on the technical problems faced during our winter deployments aiming at building 3D maps using our new cost function. These maps demonstrate both global and local consistency over 4.1 km.

Figure 3.1 – The boreal forest *Forêt Montmorency* presents an environment ideal to test robotic applications in a subarctic region. For lidar mapping, it offers challenging dense forest conditions. Credit: *Forêt Montmorency*.

3.1 Introduction

Autonomous mobile robots require a representation (i.e., a map) of the environment in order to perform specific tasks. For instance, maps are needed internally to plan motions and avoid obstacles. The map itself can also be the objective, captured by robots and used for later analysis [40], including forestry inventories [41]. Although many solutions exist for localization and mapping, the environment itself influences the complexity of the task, and thus dictates which algorithm to use. In this study, we targeted snowy, subarctic forests to explore new challenges to large-scale mapping. Indeed, this type of environment is minimally-structured, making registration more difficult. Moreover, the ruggedness of terrains dictates the need for a full six Degrees of Freedom (DoF) solution, with little assumption on trajectory smoothness.

Another difficulty brought by subarctic environments is the lack of distinctive visual features during snowy periods [42]. With images of snowy surfaces, it is challenging to extract enough features in order to support visual odometry or vision-based Simultaneous Localization and Mapping (SLAM). This precludes the use of passive camera-based localization systems, making lidar the sensing modality of choice for these conditions. A natural approach to mapping in this case is to incrementally build a 3D point cloud map from scans taken at different locations, using the Iterative Closest Point (ICP)

algorithm [43]. The ICP algorithm iteratively finds corresponding points between two point clouds and looks for a rigid transformation minimizing an alignment error. This approach ensures local consistency, yet inevitably suffers from global drift [12]. This drift problem can be mitigated by SLAM techniques based on pose graph optimization [44], [45]. The key idea behind the latter is to identify loop closures and optimize all estimated transformations between the individual lidar scans to ensure global consistency. Unfortunately, in environments which do not allow loop closure (such as a long straight trajectory), drift cannot be avoided without additional external localization sources, such as Global Navigation Satellite System (GNSS). Moreover, the pose graph optimizer requires uncertainty estimations in the form of covariance matrices for all transformations, including those from ICP. As shown by Landry *et al.* [21], this uncertainty can be modeled, learned or sampled, but the Gaussian distribution assumption does not hold up well in complex 3D environments.

Furthermore, autonomous robots operating on polar ice sheets [46] often rely on GNSS as their main source of positioning. In unstructured environments (e.g., boreal forests, taiga), GNSS cannot be used this way due to high uncertainty of position estimates. This uncertainty is caused by interference of the canopy with the signals from satellites [47]. Still, the main advantage of GNSS is that it provides a global source of positioning, which shows minimal and bounded bias compared to the ICP. Meanwhile, ICP creates maps that are crisp (i.e. locally consistent).

The goal of our paper is to demonstrate large-scale mapping of difficult environments, while generating maps that are *i)* crisp, *ii)* without long-term drifts, and *iii)* that can be updated swiftly. To satisfy the first two criteria, we experimented with embedding external constraints directly within the ICP minimization process using a novel formulation to handle uncertainty on positions. More precisely, we propose to augment the ICP algorithm by adding penalty terms based on the global GNSS positioning and the Inertial Measurement Unit (IMU) attitude estimates, weighed by their uncertainty. This formulation has the advantage that the uncertainty associated with these external constraints, contrary to the ICP's, can usually be readily estimated. Second, it avoids undue oscillations induced by alternating between graph minimization and point cloud registration, as both algorithms have no guarantees of sharing the same minimum. The third criterion pertains to the need for fast point-cloud map update in autonomous systems. The main bottleneck of the ICP algorithm is the update of the KD-tree, a data structure used for a fast nearest-neighbor search. Since our objective is to build large maps, this slowdown becomes unacceptable. We investigate a simple optimiza-

tion technique to reduce the execution time of registrations by reducing the size of the map portion used for the KD-tree update. We tested our approach on large-scale maps recorded in a boreal forest during the winter (shown in Figure 3.1). A particular emphasis has been placed on discussing problems related to the different aspects of lidar mapping for this type of environment.

3.2 Related Work

The context our work involves robotic deployments in harsh, snowy environments. This problem of mapping and localization in these conditions has only been investigated by a few publications. For example, visual SLAM for a robotic snowmobile platform was deployed by [48] on a glacier in Alaska. The authors report difficulties relating to the relatively low number of visual features in close vicinity of the mapping platform, compared to visual features located on the horizon. Since the nearby features are vital for translation estimation, image processing techniques are proposed to improve their extraction. Effects of changing shape and appearance of snowy areas on a path-following algorithm are also discussed in [49]. Their findings further motivate the use of lidars for mapping in these conditions. In our approach, the deployment of the lidar sensor translates the problem of extracting image features for localization into the problem of locating against 3D geometry. On the one hand, areas covered by the boreal forests comply well with this requirement. On the other hand, on open plateaus where 3D features are sparse, the GNSS constraints assure consistent localization and mapping. Moreover, similarly to [48] and [49], we do not require wheel or track odometry measurements from the mobile platform. This feature simplifies integration of the mapping system into different mobile vehicles which do not offer any odometry (in our case, the sleigh). We, however, benefit from an IMU which provides attitude prior with unbiased roll and pitch angles.

Laser scans can be captured from a ground-based static sensor as in the work of [50] who employed a stationary high density lidar sensor for terrain classification and identification of tree stems. Alternatively, the Airborne Lidar Scanning (ALS) approach allows the mapping of vast forested areas from the air. Besides the ALS, *Structure from Motion* technique [51] and stereo imagery [52] are further alternatives to creating 3D maps, suitable mainly for light aerial drones. Our goal is creating and maintaining globally consistent 3D maps for autonomous ground robots. In the case of ALS, the global consistency is easier to achieve because of the high-altitude point of view and

unobstructed GNSS reception. Contrarily, ground robots only observe a limited portion of the area at a time and their GNSS reception is partially occluded by the canopy. On the other hand, ground-based 3D scans offer high details, also useful for in-depth vegetation analysis. The problem of storing and managing large amounts of data is common to all of the mentioned works. In our approach, we propose a technique to limit the computation demands during the mapping process.

Fueled by the increasing interest in self-driving cars, multiple large-scale urban datasets, most notably [13], (containing lidar and GNSS information beside other sensors) have become available. These datasets have accelerated development, refining a variety of visual- and lidar-based SLAM algorithms. Contrary to structured urban environments, we investigate the characteristics of mapping in unstructured ones (forests) in harsh winter conditions. Additionally, any improvement on the accuracy of registration algorithms reduces pressure on loop-closure and graph minimization algorithms, leading to more robust lidar-based SLAM algorithms overall.

From the extensive family of ICP variants [9], our contribution relates mainly to incorporating generalized noise models into the ICP algorithm. Since the GNSS positioning provides a confidence estimate in the form of a covariance matrix, simplification to an isotropic noise model discards potentially important information. Ohta *et al.* [53] were first to consider anisotropic and inhomogeneous noise models when estimating optimal rotation of features extracted from stereo-pair depth images for 3D reconstruction. Later, the Generalized Total-Least-Squares ICP (GTLS-ICP) algorithm was introduced by Estépar *et al.* [54] for registering medical fiduciary markers. The work considers an anisotropic noise model in the registration phase of ICP and accounts for optimizing translation component as well. Further improvements were introduced by Maier-hein *et al.* [55], where the matching phase is modified to benefit both from KD-tree search speed and Mahalanobis-distance metric. This technique has been eventually enhanced by the introduction of a new kind of KD-tree which directly supports Mahalanobis-distance and a new minimizer [56]. In our approach, the anisotropic noise is strictly limited to the GNSS position measurements, making the problem slightly different. We look for a way to integrate this positioning information together with its anisotropic noise into the ICP. More closely related to robotic applications, the Generalized-ICP (GICP) algorithm [33] preserves the possibility to model measurement noise, while focusing on minimizing the plane-to-plane metric. However, using an iterative minimizer such as Broyden-Fletcher-Goldfarb-Shanno (BFGS) inside the matching loop of ICP is prohibitively slow. In this paper, we investigate how to link point set registration to

include penalty terms brought by the GNSS and IMU measurements within the same mathematical framework generic to anisotropic noise.

3.3 Theory

3.3.1 New Formulation for Point-to-Gaussian Cost Function

The ICP algorithm aims at estimating a rigid transformation $\widehat{\boldsymbol{T}}$ that best aligns a set of 3D points $\mathcal{Q}$ (i.e., a *map* point cloud) with a second set of 3D points $\mathcal{P}$ (i.e., *scan* point cloud), given a prior transformation $\breve{\boldsymbol{T}}$. For a better representation of surfaces, the points of the map point cloud can be represented locally by planes. The problem of rigid registration using points from the scan and planes from the map [8] can be summarized as minimizing the point-to-plane cost function $\mathrm{J}_{\text{p-n}}(\cdot)$ following

$$\widehat{\boldsymbol{T}} = \arg\min_{\boldsymbol{T}} \; J_{\text{p-n}}\left(\mathcal{Q}, \mathcal{P}, \breve{\boldsymbol{T}}\right), \quad \text{with} \tag{3.1}$$

$$J_{\text{p-n}} = \sum_{i=1} \sum_{j=1} w_{ij} (\boldsymbol{e}_{ij}^T \boldsymbol{n}_i)^2, \quad \text{and} \quad \boldsymbol{e}_{ij} = \boldsymbol{q}_i - \breve{\boldsymbol{R}} \boldsymbol{p}_j - \breve{\boldsymbol{t}}, \tag{3.2}$$

where $\boldsymbol{e}_{ij}$ is the error vector between the i^{th} point $\boldsymbol{q}$ of $\mathcal{Q}$ and the j^{th} point $\boldsymbol{p}$ of $\mathcal{P}$, $\boldsymbol{n}_i$ is the normal of the plane, $\breve{\boldsymbol{R}}$ and $\breve{\boldsymbol{t}}$ are respectively the rotation and translation part of $\breve{\boldsymbol{T}}$, and w_{ij} is a weight limiting the impact of outliers as surveyed by Babin *et al.* [2]. The double summation in (3.1) is expensive to compute and is typically approximated using a subset of pairs using nearest neighbor points of each scan point. To simplify the notation, we will use $\boldsymbol{e}_m$ for each error to be minimized, with m being the index of this subset. Point-to-plane error outperforms point-to-point error in most cases [12]. However, it does not represent non-planar surface well. Point-to-Gaussian provides a more versatile representation [57]. Instead of being represented by a plane, each point in $\mathcal{Q}$ is the mean of a Gaussian and its incertitude is represented by a covariance. The point-to-Gaussian cost function $\mathrm{J}_{\text{p-g}}$ thus becomes the following:

$$\mathrm{J}_{\text{p-g}} = \sum_{m=1} \left(w \boldsymbol{e}^T \boldsymbol{W}^{-1} \boldsymbol{e} \right)_m, \tag{3.3}$$

where $\boldsymbol{W}^{-1}$ is the inverse of the covariance. In point-to-Gaussian, the Mahalanobis distance is minimized instead of the Euclidean distance (point-to-point) or the projected distance to a plan (point-to-plane). Instead of using a second iterative solver within the matching loop of ICP [33], [54], [56], we propose a novel decomposition to minimize the point-to-Gaussian error (3.3) directly using the equations for point-to-plane error (3.1).

The inverse of the covariance $\boldsymbol{W}^{-1}$ can be expressed as a matrix $\boldsymbol{N}$ of eigenvectors and a diagonal matrix $\boldsymbol{\Lambda}$ holding the sorted eigenvalues, with $\lambda_1 < \lambda_2 < \lambda_3$, using

$$\boldsymbol{W} = \boldsymbol{N}\boldsymbol{\Lambda}\boldsymbol{N}^T \Rightarrow \boldsymbol{W}^{-1} = \boldsymbol{N}\boldsymbol{\Lambda}^{-1}\boldsymbol{N}^T.$$

The decomposition can be inserted inside the cost function and reformulated as three point-to-plane errors using a projection for each of the eigenvector. Dropping the summation and the indices for clarity, we obtain for a single pair of points

$$\begin{aligned} \mathrm{J}_{\text{p-g}} &= \sum w\boldsymbol{e}^T\boldsymbol{N}\boldsymbol{\Lambda}^{-1}\boldsymbol{N}^T\boldsymbol{e} && (3.4)\\ &= \sum w\boldsymbol{e}^T \begin{bmatrix}\boldsymbol{n}_1 & \boldsymbol{n}_2 & \boldsymbol{n}_3\end{bmatrix} \operatorname{diag}\left(\tfrac{1}{\lambda_1}, \tfrac{1}{\lambda_2}, \tfrac{1}{\lambda_3}\right) \begin{bmatrix}\boldsymbol{n}_1 & \boldsymbol{n}_2 & \boldsymbol{n}_3\end{bmatrix}^T \boldsymbol{e} \\ &= \sum w\Bigg(\underbrace{\frac{1}{\lambda_1}\Big(\boldsymbol{e}^T\boldsymbol{n}_1\Big)^2}_{\mathrm{J}_{\text{p-n}}} + \frac{1}{\lambda_2}\Big(\boldsymbol{e}^T\boldsymbol{n}_2\Big)^2 + \frac{1}{\lambda_3}\Big(\boldsymbol{e}^T\boldsymbol{n}_3\Big)^2\Bigg), && (3.5) \end{aligned}$$

where λ_i is an eigenvalue and $\boldsymbol{n}_i$ is its associated eigenvector. Thus, point-to-Gaussian can be used with any point-to-plane minimizer. In fact, point-to-plane is a special case of point-to-Gaussian, where the first eigenvalue λ_1 is small enough compared to λ_2 and λ_3. This formulation can be used to also minimize Gaussian-to-Gaussian by setting $\boldsymbol{W}$ to the sum of the uncertainty of point $\boldsymbol{q}$ with the rotated uncertainty of its associated $\boldsymbol{p}$.

3.3.2 Adding Penalty Terms to ICP

ICP mapping creates crisp maps by taking into account local geometric characteristics contained in each new point cloud. Therefore, global consistency is not enforced. On the other hand, GNSS provides globally consistent positioning, but yields low local precision, especially when compared to ICP in forested areas. Furthermore, there is a disproportion between the altitude, latitude and longitude positioning components, the altitude being the least precise. By fusing ICP, GNSS and IMU information, we propose to compensate for the ICP drift.

Penalties are a natural way to add a constraint to the minimization step of ICP. They can be seen as imaginary points added to the point cloud during minimization for which the association is known. The minimization problem thus becomes:

$$\widehat{\boldsymbol{T}} = \arg\min_{\boldsymbol{T}} \underbrace{\frac{1}{M}\sum_{m=1}\left(w\boldsymbol{e}^T\boldsymbol{W}^{-1}\boldsymbol{e}\right)_m}_{\text{point clouds}} + \underbrace{\frac{1}{K}\sum_{k=1}\left(\boldsymbol{e}^T\boldsymbol{W}^{-1}\boldsymbol{e}\right)_k}_{\text{penalties}}, \qquad (3.6)$$

where $\boldsymbol{e}_k$ and $\boldsymbol{W}_k$ are respectively the error and the covariance of the k^{th} penalty, M is the number of matched points and K the number of penalty added. The penalty error $\boldsymbol{e}_k$ consists of two points added to their respecting frames of reference

$$\boldsymbol{e}_k = {}^{\text{M}}\boldsymbol{q}_k - {}^{\text{M}}_{S}\breve{\boldsymbol{T}}\,{}^{\text{S}}\boldsymbol{p}_k, \tag{3.7}$$

where ${}^{\text{M}}\boldsymbol{q}_k$ is the position of the k^{th} penalty in the map frame, ${}^{\text{M}}_{S}\boldsymbol{T}$ is the transformation from the scan frame (S) to the map frame (M) at the current iteration and ${}^{\text{S}}\boldsymbol{p}_k$ is the position of the k^{th} penalty in the scan frame. For instance for the GNSS penalty, ${}^{\text{M}}\boldsymbol{q}_k$ is the GNSS's global position and ${}^{\text{S}}\boldsymbol{p}_k$ is the origin of the scan frame. Effects of adding penalty points are presented in Section 3.5.1.

Since the GNSS penalty points come in the form of a Gaussian distribution and point clouds contain plane normal information, equation (3.6) is approximated using a constant scale s in our implementation. This constant ensures the conversion from projected distances to Mahalanobis distances by assuming that $\frac{1}{\lambda_1}$ is constant for all points and neglecting $\frac{1}{\lambda_2}$ and$\frac{1}{\lambda_3}$ in (3.5). In this setting, the final cost function to be optimized becomes:

$$\text{J}_{\text{p-g}} \approx \frac{s}{M}\sum_{m=1}\left(w(\boldsymbol{e}^T\boldsymbol{n}_1)^2\right)_m + \frac{1}{K}\sum_{k=1}\left(\boldsymbol{e}^T\boldsymbol{W}^{-1}\boldsymbol{e}\right)_k. \tag{3.8}$$

3.3.3 Iterative Closest Point mapping

The mapping was achieved using a modified version of `ethz-icp-mapping` [58]. The mapper performs the following steps: 1) Move the scan to the initial estimate, 2) register the scan with the map using ICP, and 3) insert the scan inside the map. The initial estimate $\breve{\boldsymbol{T}}$ is composed of a translation increment based on the GNSS positioning and change in orientation based on the IMU. The IMU heading is corrected by the GNSS positioning as long as the platform moves forward. Justification of this correction and a possible alternative are discussed in Section 3.5.1. Since this initial estimate $\breve{\boldsymbol{T}}$ is utilized in an incremental manner, the mapping can diverge over time. As for the construction of the global map $\mathcal{Q}$, the whole scan $\mathcal{P}$ is not directly concatenated. Rather, only points that are farther than ϵ from any points in $\mathcal{Q}$ are inserted. This helps in keeping the global map uniform, without sacrificing registration precision. As the robot explores the environment, the complexity of registration grows linearly with the number of points in the map due to the KD-tree structure updates. To stabilize the mapping complexity, a scan $\mathcal{P}$ is not registered against the whole map $\mathcal{Q}$, but only against a subsection of the map within a radius $r_{\max}$ equal to the maximum range of the lidar. The effects of this optimization is shown in Section 3.5.3.

Figure 3.2 – The data acquisition platform mounted on a sleigh behind the snowmobile (*left*). This configuration limited transfer of engine vibrations to the sensors. *Right*: close-up of the GNSS receiver and the RS-16 lidar.

3.4 Experimental Setup

3.4.1 Data Acquisition Platform

For our experiments, we developed a rugged data acquisition platform which can withstand snow and sub-zero temperatures. It comprises an Xsens MTI-30 IMU, a Robosense RS-16 lidar, and a REACH RS+ GNSS antenna powered by two 20Ah 12V AGM batteries (10 h battery life). A small, low-power computer (AIV-APL1V1FL-PT1) records the sensor data using the Robotic Operating System (ROS) framework. This platform can be attached to most of mobile vehicles (see Figure 3.2). The rotation axis of the lidar sensor is at an angle of 27° from the vertical. This orientation has been chosen for two reasons: 1) the lidar does not see the mobile vehicle nor its operators as long as they are in front of the platform, and 2) as mentioned in [59] a lower incidence angle with the ground reduces the odometry drift. The GNSS antenna is coupled with a fixed station (also a REACH RS+ antenna) mounted on a tripod to provide a Real-Time Kinematic (RTK) solution.

The Université Laval owns the *Forêt Montmorency*, the largest research forest in the world with over 412 km^2 of boreal forest (Figure 3.1). For our experiments, we collected data along three large loops (see Figure 3.3). Two of them (i.e., `lake` and `forest`) consist of a mix of narrow walkable trails and wider snowmobile trails, while the last one (i.e., `skidoo`) followed exclusively a wide snowmobile trail.

Figure 3.3 – The three large loops completed in the Montmorency forest in order to collect the `lake` (0.8 km), `forest` (1.3 km) and `skidoo` (2 km) datasets. The map was adapted from ©Mapbox and ©OpenStreetMap contributors.

More specifically, the dataset `lake` was recorded by mounting our platform to a snowmobile (see Figure 3.2) and then having two operators pull the sleigh through a pedestrian trail. The snowmobile drove through a cross-country skiing trail, which is an open area with good GNSS coverage (`A` to `B` in Figure 3.3). The pedestrian trail was a dense forest path (`B` to `C`), where the platform suffered from poor GNSS reception due to the tree canopy. The overlap between scans diminished abruptly each time branches came near the sensor. Similarly, the dataset `forest` consists of a pedestrian trail and a cross-country skiing trail. In the first part of the trajectory, a pedestrian trail in the dense forest was traversed with a sleigh (`D` to `E`), followed by an untapped path through an even denser forest (`E` to `F`) and finally the sleigh was attached to a snowmobile and driven back to the starting point (`F` to `D`). Lastly, the dataset `skidoo` follows a 2 km long snowmobile trail. The data were gathered during a light snow fall. This loop was the easiest to map because it provided clear GNSS signal and was relatively flat from beginning to end (`G` to `H` to `G`).

3.5 Field Results

3.5.1 Effects of Adding GNSS Penalty to ICP

In order to inject the GNSS positioning information as a constraint into the ICP algorithm, it is necessary to find the correct transformation between the ICP map coordinate frame and the local tangent East-North-Up (ENU) frame. The translation, roll

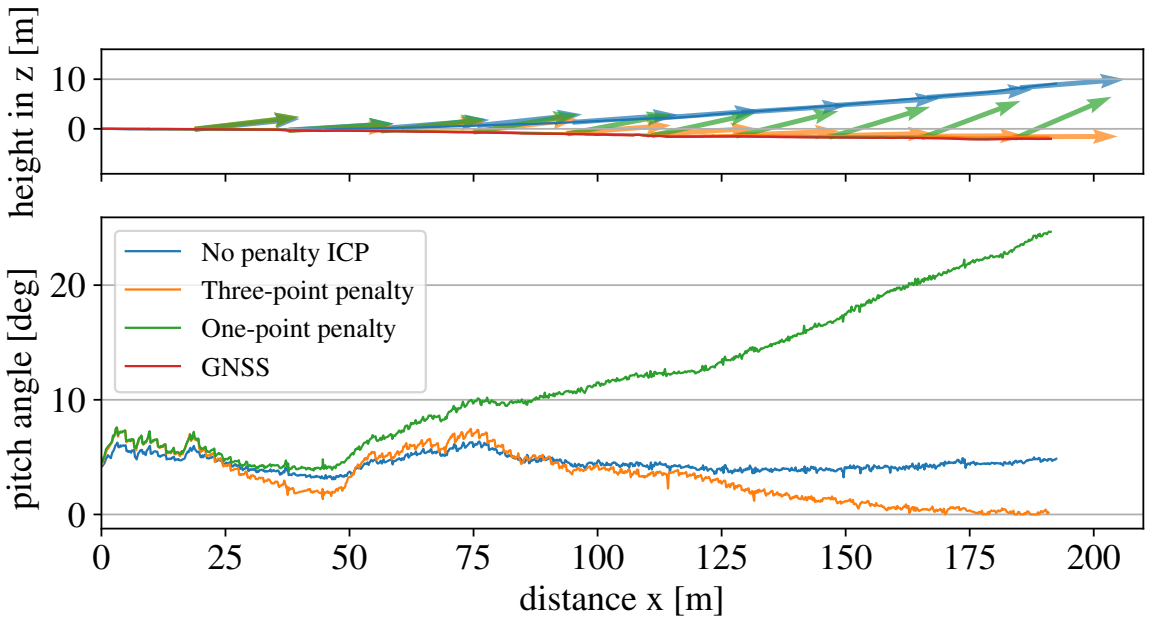


Figure 3.4 – Effect of adding penalty points to the ICP. *Top*: side view of a straight trajectory segment along with orientations represented by arrows. *Bottom*: corresponding pitch angles, unavailable for GNSS.

and pitch angles are directly observable from the GNSS receiver and the IMU. The yaw angle (heading) measurement provided by the magnetometer part of the IMU is, however, affected by soft- and hard-iron errors induced by the mobile platform itself and by local deviations of the Earth's magnetic field. A practical solution to this problem was to observe a short initial portion of the GNSS trajectory to estimate the magnetometer heading offset. This way, the typical error of between 15° and 20° could be reduced to 3° or less, which led to a satisfactory initial alignment. Another approach would be attaching a second GNSS receiver antenna to the mobile platform and estimating the heading angle from the relative positions of the two antennas. Both approaches, however, require a precise RTK GNSS solution. The standard uncorrected GNSS operating under the tree canopy yields excessive error and cannot be used for this purpose.

We first only applied a single penalty point to the ICP, based on the GNSS positioning. As the green trajectory in Figure 3.4 demonstrates, this approach does not provide satisfactory results. On a short straight trajectory, we see that the single-point penalty forces the ICP to follow the GNSS reference, however, the orientation estimate drifts leading to a malformed map. In the Figure 3.4, this effect manifests itself as a slow rise in the pitch angle.

To fully constrain the ICP and avoid both orientation and position drift, we increased

the number of penalty points to three, still following (3.8). The additional two points lie on the gravity and on the heading vectors as follows: In the map frame, one point lies below the GNSS position in the direction of the z axis. The second point lies in the x-y plane, in the direction of the current heading as indicated by the IMU and GNSS. In the scan frame, these two new points are accordingly projected using the IMU orientation information. In the ideal case, all three penalty points in the scan frame coincide with their map counterparts. Otherwise, the penalty is forcing the ICP solution towards the ideal state. The effect is demonstrated in the Figure 3.4 by the orange trajectory; the ICP output follows the GNSS positioning while keeping the correct orientation as well.

3.5.2 The Forêt Montmorency Dataset Results and Discussion

For each dataset, three mapping configurations were evaluated: GNSS+IMU (i.e., prior), ICP with penalties (i.e., penalty) and ICP without penalty (i.e., baseline). When processing, the `ethz-icp-mapping` was used with $r_{\max} = 100\,\text{m}$ and with a minimum point distance $\epsilon = 5\,\text{cm}$ for `lake` and `forest`. The `skidoo` dataset uses $\epsilon = 10\,\text{cm}$ due to its size and memory requirements. The resulting maps are shown in Figure 3.5.

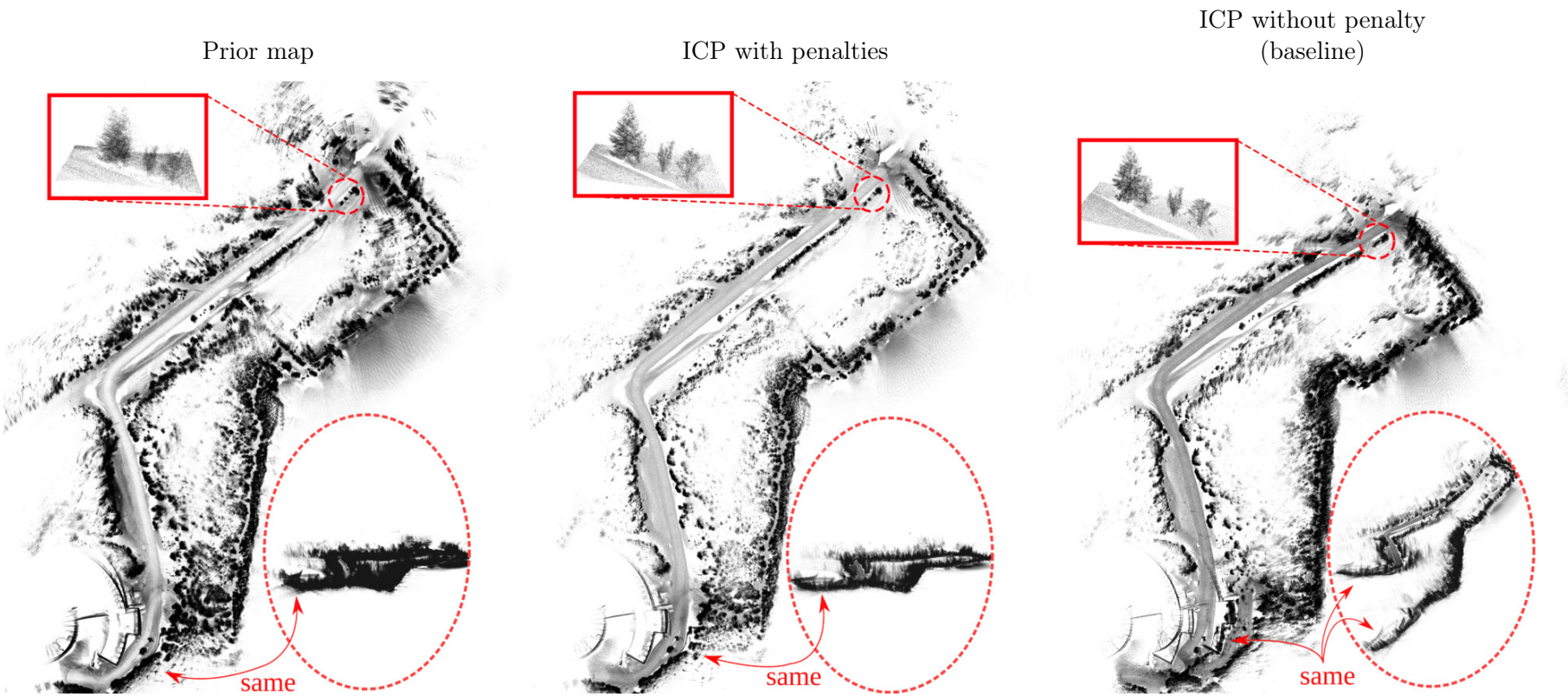


Figure 3.5 – Top view of the point cloud maps of the three datasets; prior, penalty and baseline configurations. The figure on this page show the `lake` dataset. A side view of the map (red dashed ellipse) shows the misalignment between the start and end. Red insets show the local (in)consistencies otherwise not visible at the full scale. Some trees in the insets are up to 15 m in height.

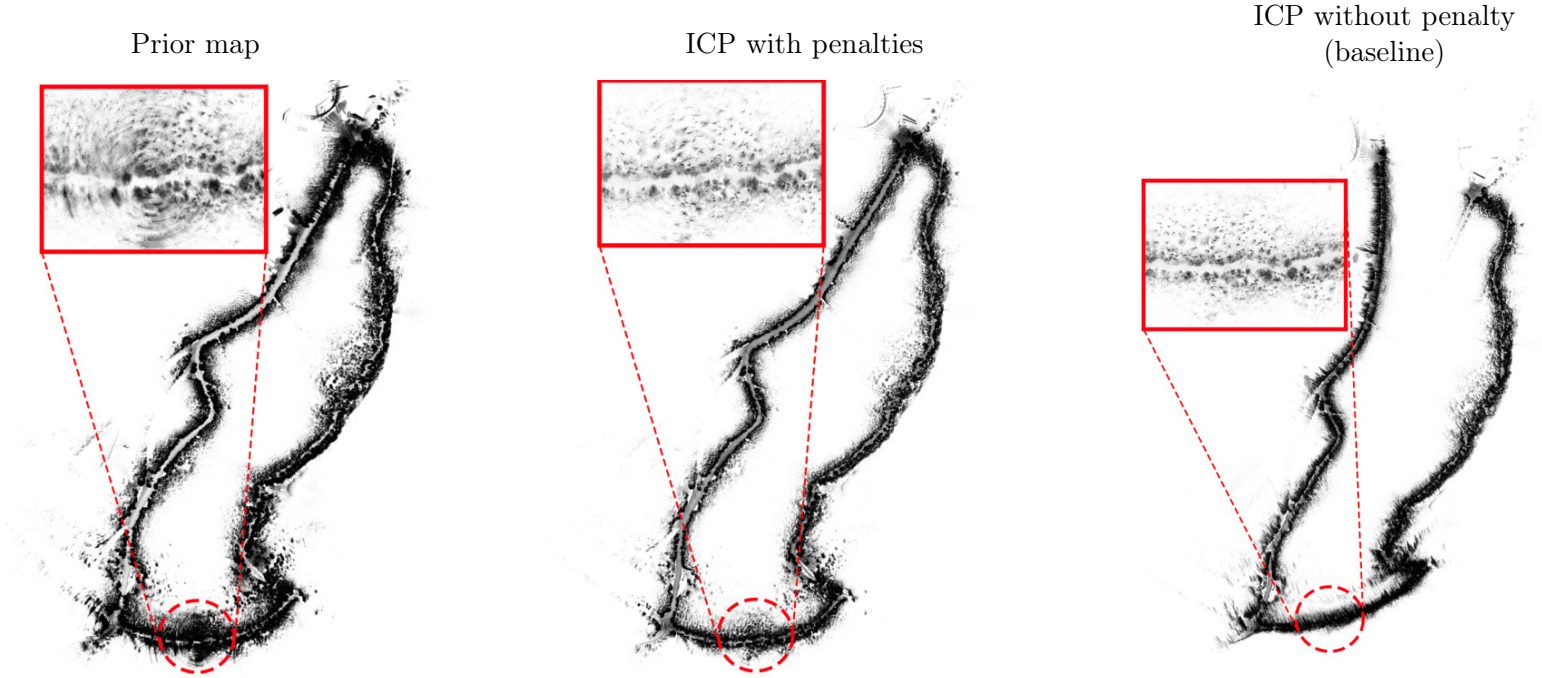


Figure 3.5 – `forest` dataset

Prior map

ICP with penalties

ICP without penalty
(baseline)

Figure 3.5 – `skidoo` dataset

(I) Local Wildlife (II) Snow fall

(III) Path Obstacles (IV) Uneven Path

Figure 3.6 – Aspects of mapping a subarctic boreal forest. (II) shows snow fall caused by the snow in the tree branches. (I) was taken in skidoo, (II) in lake, and (III)-(IV) in forest.

Lake – In this dataset, the baseline map looks similar to the penalty map when observed from top. However, a side view clearly shows that the buildings at the start and the end of the trajectory do not match. This effect is avoided by applying the penalty. For each configuration, we highlighted a pine tree in the top part of the map. While the tree is crisper in the baseline, the penalty map's tree is clearly sharper than the one from the prior map. **Forest** – The particularity of this map is the rough trail at the bottom of the map (see subfig. III and IV of Figure 3.6). It suffers from large circular artifacts caused by the platform being immobile and by major changes of orientation along that trail. Again, the penalty is less clear than the baseline, but quite an improvement compared to the prior map. The circular-shaped building at the top part of the map is

quite blurry in both the prior and penalty map. This lack of crispiness is caused by the start and end of the trajectory not correctly matching. **`Skidoo`** – The baseline map is the most bent of all three datasets. This bending has two probable causes: 1) the map was created with lower density $\epsilon = 10\,\text{cm}$ contrary to the other maps 2) the trajectory is twice as long as the `lake` dataset. Otherwise, the behavior is similar to that of the other datasets with the similar circular-shaped artifacts in the prior map.

None of the baseline maps manages to close the loop, they all drift and bend over time. Because of the magnetometer-based heading, the prior maps show large circular artifacts at several locations. These artifacts are especially noticeable in zones where the platform stops moving—the heading cannot be corrected by GNSS in this case. All penalty and prior maps closed the loop in a similar fashion. Moreover, the penalties manage to achieve a trade-off between the global and local consistency.

The field trials at the Forêt Montmorency presented a number of challenges. As `(I)` of Figure 3.6 shows, local wildlife might hinder your experiments. We had a pair of moose blocking one of our trajectories and the experiment had to be rescheduled for another day. Also, wild birds used our static GNSS antenna as a perch. Another challenge is snow fall, as even a light snow fall will be visible in the map. Furthermore, even when our experiments were done on a clear day, they were still affected by snow falling from the trees (see subfig. `II` of Figure 3.6). In the `forest` trajectory, we had to pass through an untapped trail with trees blocking our way `(III)`. Because of the roughness of that trail, the platform almost tumbled over multiple time `(IV)`. The snowmobile trails, on the other hand, were easy to pass through.

Cold can cause hindering issues with lidars that are not properly rated for low temperatures. We tested a lidar rated for a minimum of $-10\,^{\circ}\text{C}$, with exterior temperature during our field experiments varying between $-17\,^{\circ}\text{C}$ and $-7\,^{\circ}\text{C}$. The lidar started to malfunction when the temperature dropped below $0\,^{\circ}\text{C}$, producing spurious measurements to a level where the environment could not be seen anymore. Providing an exterior source of heat could mitigate the problem temporary. A second lidar rated for $-20\,^{\circ}\text{C}$ was used for our final experiments. Through the development process, we observed that sun glare was more apparent at low temperatures. More tests are required to fully understand the impact of cold on lidar, but one should be careful regarding the temperature ranting of sensors deployed as it can cause serious safety issues.

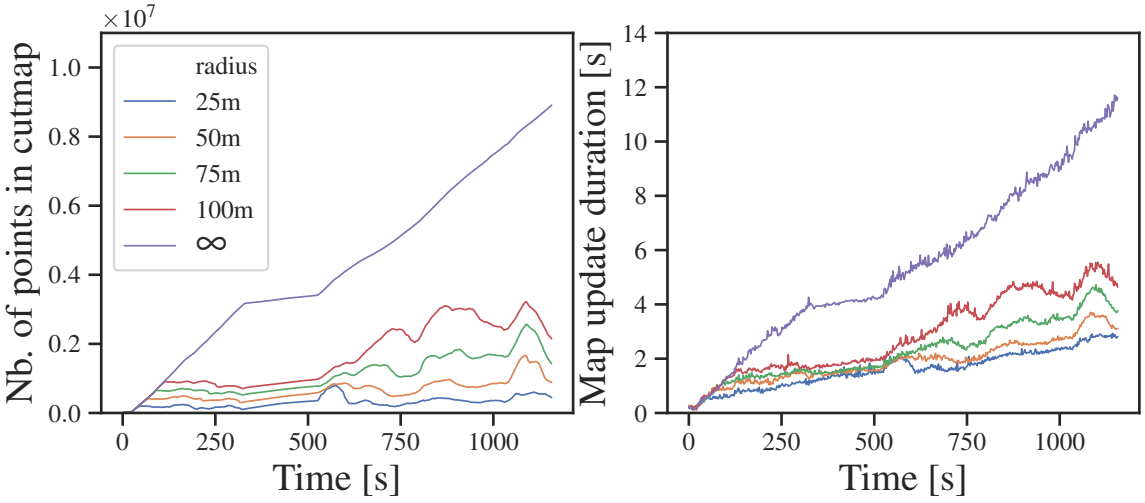


Figure 3.7 – Effect of the cut-map radius $r_{\max}$ on the complexity over time for the `lake` trajectory. *Left*: number of points used as a reference point cloud for the ICP. *Right*: computation time used by the mapper for each new lidar scan.

3.5.3 Steps towards Real-Time Large-Scale Lidar Mapping

The large-scale point cloud maps bear several problems that complicate real-time deployment on autonomous vehicles. The processing time required to register the scans and update the map may limit the agility of the vehicle. Moreover, memory management needs to be taken into consideration. For example, the ∼23 700 000 data points of the `forest` final prior map consumed 1.8 GB when stored in RAM. To improve the mapping speed, we have implemented the $r_{\max}$ cut-map radius as defined in Section 3.3.3. As shown in Figure 3.7, the mapper execution time is reduced. One can observe a flattening in the number of points in the cut-map around 250 s to 500 s. This situation occurs when the mobile platform is immobile.

Finally, in order to achieve globally consistent maps, we used the RTK GNSS solution consisting of two receivers, one static and the other attached to the mobile platform. The precise positioning information is obtained by combining the information from both receivers. In our case, it was done during post-processing of the dataset. For real-time deployment, it is necessary to reliably transmit the static receiver information to the mobile vehicle, which may be difficult in dense vegetation.

3.6 Conclusion

In this paper, we explored the process of creating large globally consistent maps of subarctic boreal forests by adding external constraints to the ICP algorithm. The maps remained crisp even through 4.1 km of narrow walkable and snowmobile trails. We also discussed problems encountered with the environment and the lidar sensor during the field trials. Moreover, we introduced a computation optimization for very large maps, allowing real-time deployments. Encouraged by the results, this opens the door to further comparison with Normal Distribution Transformation (NDT) and GICP using better experimental validation and external tracking systems. Furthermore, studying the impact of penalties within ICP against graph minimization would lead to a better understanding of their pros and cons.

Acknowledgment

This works was financed by the Fonds de Recherche du Québec – Nature et technologies (FRQNT) and the Natural Sciences and Engineering Research Council of Canada (NSERC) through the grant CRDPJ 527642-18 SNOW.

References (Chapter 3)

[2] P. Babin, P. Giguère, and F. Pomerleau, "Analysis of Robust Functions for Registration Algorithms", in *IEEE International Conference on Robotics and Automation (ICRA)*, 2019, pp. 1451–1457.

[8] Y. Chen and G. Medioni, "Object modeling by registration of multiple range images", *Proceedings. 1991 IEEE International Conference on Robotics and Automation*, no. 3, pp. 2724–2729, 1992.

[9] F. Pomerleau, F. Colas, and R. Siegwart, "A Review of Point Cloud Registration Algorithms for Mobile Robotics", *Foundations and Trends in Robotics*, vol. 4, no. 1, pp. 1–104, 2015.

[12] F. Pomerleau, F. Colas, R. Siegwart, and S. Magnenat, "Comparing ICP variants on real-world data sets: Open-source library and experimental protocol", *Autonomous Robots*, vol. 34, no. 3, pp. 133–148, 2013.

[13] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "Vision meets robotics: the KITTI dataset", *The International Journal of Robotics Research*, vol. 32, no. 11, pp. 1231–1237, 2013.

[21] D. Landry, F. Pomerleau, and P. Giguère, "CELLO-3D: Estimating the Covariance of ICP in the Real World", in *IEEE International Conference on Robotics and Automation (ICRA)*, 2019, pp. 8190–8196.

[33] A Segal, D Haehnel, and S Thrun, "Generalized-ICP", in *Robotics: Science and Systems*, vol. 5, 2009, pp. 168–176.

[40] R. Zlot and M. Bosse, "Efficient large-scale three-dimensional mobile mapping for underground mines", *Journal of Field Robotics*, vol. 31, no. 5, pp. 758–779, 2014.

[41] M. Pierzchała, P. Giguère, and R. Astrup, "Mapping forests using an unmanned ground vehicle with 3d lidar and graph-slam", *Computers and Electronics in Agriculture*, vol. 145, pp. 217–225, 2018.

[42] S. Williams, L. T. Parker, and A. M. Howard, "Terrain Reconstruction of Glacial Surfaces : Robotic Surveying Techniques", *IEEE Robotics and Automation Magazine*, vol. 19, no. 4, pp. 59–71, 2012.

[43] P. Besl and N. McKay, "A Method for Registration of 3-D Shapes", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 14, no. 2, pp. 239–256, 1992.

[44] S. Thrun and M. Montemerlo, "The GraphSLAM Algorithm with Applications to Large-Scale Mapping of Urban Structures", *The International Journal of Robotics Research*, vol. 25, no. 5-6, pp. 403–429, 2006.

[45] C. Cadena, L. Carlone, H. Carrillo, Y. Latif, D. Scaramuzza, *et al.*, "Past, Present, and Future of Simultaneous Localization and Mapping: Toward the Robust-Perception Age", *IEEE Transactions on Robotics*, vol. 32, no. 6, pp. 1309–1332, 2016.

[46] J. H. Lever, A. J. Delaney, L. E Ray, E. Trautmann, L. A. Barna, and A. M. Burzynski, "Autonomous GPR Surveys using the Polar Rover Yeti", *Journal of Field Robotics*, vol. 30, no. 2, pp. 194–215, 2013.

[47] G. Jagbrant, J. P. Underwood, J. Nieto, and S. Sukkarieh, "LiDAR Based Tree and Platform Localisation in Almond Orchards", in *Proceedings of the Conference on Field and Service Robotics*, 2015, pp. 469–483.

[48] S. Williams and A. M. Howard, "Developing Monocular Visual Pose Estimation for Arctic Environments", *Journal of Field Robotics*, vol. 71, no. 5, pp. 486–494, 2009.

[49] M. Paton, F. Pomerleau, and T. Barfoot, "In the dead of winter: challenging vision-based path following in extreme conditions", *Springer Tracts in Advanced Robotics*, vol. 113, pp. 563–576, 2016.

[50] M. W. McDaniel, N. Takayuki, C. A. Brooks, P. Salesses, and K. Lagnemma, "Terrain Classification and Identification of Tree Stems Using Ground-based LiDAR", *Journal of Field Robotics*, vol. 29, pp. 891–910, 2012.

[51] L. Wallace, A. Lucieer, Z. Malenovský, D. Turner, *et al.*, "Assessment of Forest Structure Using Two UAV Techniques: A Comparison of Airborne Laser Scanning and Structure from Motion (SfM) Point Clouds", *Forests*, vol. 7, no. 3, 2016.

[52] J. Tian, R. P., P. d'Angelo, and M. Ehlers, "Region-based automatic building and forest change detection on cartosat-1 stereo imagery", *International Society for Photogrammetry and Remote Sensing*, vol. 79, pp. 226 –239, 2013.

[53] N. Ohta and K. Kanatani, "Optimal Estimation of Three- Dimensional Rotation and Reliability Evaluation", *European Conference on Computer Vision*, pp. 175–187, 1998.

[54] R. S. J. Estépar, A. Brun, and C.-F. Westin, "Robust Generalized Total Least Squares Iterative Closest Point Registration", in *Medical Image Computing and Computer Assisted Intervention*, 2004, pp. 234–241.

[55] L. Maier-hein, A. M. Franz, T. R. Santos, M. Schmidt, M. Fangerau, *et al.*, "Convergent ICP Algorithm to Accomodate Anisotropic and Inhomogenous Localization Error", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 34, no. 8, pp. 1520–1532, 2012.

[56] S. D. Billings, E. M. Boctor, and R. H. Taylor, "Iterative Most-Likely Point Registration (IMLP): A Robust Algorithm for Computing Optimal Shape Alignment", *PLoS ONE*, pp. 1–45, 2015.

[57] R. Balachandran and J. M. Fitzpatrick, "Iterative solution for rigid-body point-based registration with anisotropic weighting", vol. 7261, M. I. Miga and K. H. Wong, Eds., pp. 1051 –1060, 2009.

[58] F. Pomerleau, P. Krüsi, F. Colas, P. Furgale, and R. Siegwart, "Long-term 3D map maintenance in dynamic environments", in *IEEE International Conference on Robotics and Automation (ICRA)*, 2014, pp. 3712–3719.

[59] J. Laconte, S.-P. Deschênes, M. Labussière, and F. Pomerleau, "Lidar measurement bias estimation via return waveform modelling in a context of 3d mapping", in *IEEE International Conference on Robotics and Automation (ICRA)*, 2019, pp. 8100–8106.

Conclusion

In this thesis, we aimed at improving the error minimization of the ICP algorithm. We presented our improvements through two inserted articles. The first article is an analysis of 14 of the most common outlier filters for ICP. Each outlier filter has been tested in different types of environments and with multiple types of overlap. Furthermore, the influence of the tuning parameters present in many of these filters has been thoroughly explored. In the end, our analysis concluded that `Variable Trimmed`, `Cauchy` and `Cauchy MAD` are more robust to different environments and to change in overlap than the other outlier filters. Surprisingly, L_1 demonstrated a comparable performance despite being parameter-less. Meanwhile, the second inserted article explored the creation of large, globally-consistent maps by modifying the error minimizer of ICP. Our improvements enabled the sensor fusion of an IMU and a GNSS, directly within the ICP's minimizer. We managed to create a crisp and globally-consistent map of 4.1 km of snowmobile and narrow walkable trails.

While the contributions presented in this thesis improve existing methods, there are still challenges and avenues for research left to explore. For the outlier filters analysis, a remaining challenge is to demonstrate that our conclusions also apply to scan-to-map registration. Since all of the quantitative evaluations were done using a scan-to-scan dataset, it is difficult to truly know if our tests generalize to scan-to-map. The main hurdle to such an analysis is the lack of datasets with a millimeter precision. *Challenging data sets for point cloud registration algorithms* [6] mostly consist of small loops (less than 50 m long), which is insufficient for large-scale mapping experiments. As for the second article, it lacks quantitative comparisons to existing mapping frameworks. Future works should conduct further comparisons with SLAM frameworks such as g2o [60] and Lidar Odometry And Mapping (LOAM) [61]. Again, such comparisons require large-scale datasets of forest environments with a precise ground truth. Our improved version of ICP described in Chapter 3 can be used in conjunction with graph-SLAM. It would be interesting to see if our more globally-consistent ICP could reduce

the pressure on the loop-closure and on the graph minimization of SLAM.

To conclude, we presented improvements to the ICP algorithm in this thesis. We have proposed possible research avenues left to explore. This shows that the ICP algorithm will still provide research opportunities in the future, despite its age.

Bibliography

[1] F. Edgeworth, *Metretike: Or, The Method of Measuring Probability and Utility*. Temple Company, 1888.

[2] P. Babin, P. Giguère, and F. Pomerleau, "Analysis of Robust Functions for Registration Algorithms", in *IEEE International Conference on Robotics and Automation (ICRA)*, 2019, pp. 1451–1457.

[3] P. Babin, P. Dandurand, V. Kubelka, P. Giguère, and F. Pomerleau, "Large-scale 3D mapping of sub-arctic forests", in *Proceedings of the Conference on Field and Service Robotics (FSR). Springer Tracts in Advanced Robotics*, 2019.

[4] W. Maddern, G. Pascoe, and P. Newman, "Leveraging experience for large-scale LIDAR localisation in changing cities", *Proceedings - IEEE International Conference on Robotics and Automation*, vol. 2015-June, no. June, pp. 1684–1691, 2015.

[5] M.-F. Chang, J. Lambert, P. Sangkloy, J. Singh, S. Bak, A. Hartnett, D. Wang, P. Carr, S. Lucey, D. Ramanan, and J. Hays, "Argoverse: 3d tracking and forecasting with rich maps", in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[6] F. Pomerleau, M. Liu, F. Colas, and R. Siegwart, "Challenging data sets for point cloud registration algorithms", *The International Journal of Robotics Research*, vol. 31, no. 14, pp. 1705–1711, 2012.

[7] P. J. Besl, "Geometric Modeling and Computer Vision", *Proceedings of the IEEE*, vol. 76, no. 8, pp. 936–958, 1988.

[8] Y. Chen and G. Medioni, "Object modeling by registration of multiple range images", *Proceedings. 1991 IEEE International Conference on Robotics and Automation*, no. 3, pp. 2724–2729, 1992.

[9] F. Pomerleau, F. Colas, and R. Siegwart, "A Review of Point Cloud Registration Algorithms for Mobile Robotics", *Foundations and Trends in Robotics*, vol. 4, no. 1, pp. 1–104, 2015.

[10] P. J. Huber, "Robust Estimation of a Location Parameter", *The Annals of Mathematical Statistics*, vol. 35, no. 1, pp. 73–101, 1964.

[11] S. Rusinkiewicz and M. Levoy, "Efficient variants of the ICP algorithm", *Proceedings of International Conference on 3-D Digital Imaging and Modeling, 3DIM*, pp. 145–152, 2001.

[12] F. Pomerleau, F. Colas, R. Siegwart, and S. Magnenat, "Comparing ICP variants on real-world data sets: Open-source library and experimental protocol", *Autonomous Robots*, vol. 34, no. 3, pp. 133–148, 2013.

[13] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "Vision meets robotics: the KITTI dataset", *The International Journal of Robotics Research*, vol. 32, no. 11, pp. 1231–1237, 2013.

[14] D. G. Lowe, "Object recognition from local scale-invariant features", in *Proceedings of the Seventh IEEE International Conference on Computer Vision*, vol. 2, 1999, 1150–1157 vol.2.

[15] J. L. Bentley, "Multidimensional binary search trees used for associative searching", *Communications of the Association for Computing Machinery (ACM)*, vol. 18, no. 9, pp. 509–517, Sep. 1975.

[16] J. Elseberg, S. Magnenat, R. Siegwart, and A. Nüchter, "Comparison of nearest-neighbor-search strategies and implementations for efficient shape registration", *Journal of Software Engineering for Robotics (JOSER)*, vol. 3, no. 1, pp. 2–12, 2012.

[17] R. E. Welsch, "Robust regression using iteratively reweighted least-squares", *Communications in Statistics - Theory and Methods*, vol. 6, no. 9, pp. 813–827, 1977.

[18] E. Mendes, P. Koch, and S. Lacroix, "ICP-based pose-graph SLAM", in *SSRR 2016 - International Symposium on Safety, Security and Rescue Robotics*, IEEE, 2016, pp. 195–200.

[19] F. R. Kschischang, B. J. Frey, and H. Loeliger, "Factor graphs and the sum-product algorithm", *IEEE Transactions on Information Theory*, vol. 47, no. 2, pp. 498–519, 2001.

[20] A. Censi, "An accurate closed-form estimate of ICP's covariance", in *IEEE International Conference on Robotics and Automation*, 2007, pp. 3167–3172.

[21] D. Landry, F. Pomerleau, and P. Giguère, "CELLO-3D: Estimating the Covariance of ICP in the Real World", in *IEEE International Conference on Robotics and Automation (ICRA)*, 2019, pp. 8190–8196.

[22] S. Nobili, R. Scona, M. Caravagna, and M. Fallon, "Overlap-based ICP tuning for robust localization of a humanoid robot", in *Proceedings - IEEE International Conference on Robotics and Automation*, 2017, pp. 4721–4728.

[23] S. Granger and P. Xavier, "Multi-scale EM-ICP : A Fast and Robust Approach for Surface Registration", *Proceedings of the European Conference on Computer Vision*, pp. 418–432, 2002.

[24] A. W. Fitzgibbon, "Robust registration of 2D and 3D point sets", *Image and Vision Computing*, vol. 21, no. 13-14, pp. 1145–1153, 2003.

[25] F. Pomerleau, F. Colas, F. Ferland, and F. Michaud, "Relative Motion Threshold for Rejection in ICP Registration", *Field and Service Robotics*, vol. 62, pp. 229–238, Jul. 2015.

[26] M. Bosse, G. Agamennoni, and I. Gilitschenski, "Robust Estimation and Applications in Robotics", *Foundations and Trends in Robotics*, vol. 4, no. 4, pp. 225–269, 2016.

[27] J. M. Phillips, R. Liu, and C. Tomasi, "Outlier robust ICP for minimizing fractional RMSD", in *3DIM 2007 - Proceedings 6th International Conference on 3-D Digital Imaging and Modeling*, 2007, pp. 427–434.

[28] S. Bouaziz, A. Tagliasacchi, and M. Pauly, "Sparse iterative closest point", *Eurographics Symposium on Geometry Processing*, vol. 32, no. 5, pp. 113–123, 2013.

[29] M. Sheehan, A. Harrison, and P. Newman, "Continuous vehicle localisation using sparse 3D sensing, kernelised rényi distance and fast Gauss transforms", in *IEEE International Conference on Intelligent Robots and Systems*, 2013, pp. 398–405.

[30] K. MacTavish and T. D. Barfoot, "At all Costs: A Comparison of Robust Cost Functions for Camera Correspondence Outliers", in *Proceedings of the 12th Conference on Computer and Robot Vision*, 2015, pp. 62–69.

[31] P. Bergström and O. Edlund, "Robust registration of point sets using iteratively reweighted least squares", *Computational Optimization and Applications*, vol. 58, no. 3, pp. 543–561, 2014.

[32] G. Agamennoni, S. Fontana, R. Y. Siegwart, and D. G. Sorrenti, "Point Clouds Registration with Probabilistic Data Association", *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 4092–4098, 2016.

[33] A Segal, D Haehnel, and S Thrun, "Generalized-ICP", in *Robotics: Science and Systems*, vol. 5, 2009, pp. 168–176.

[34] M. Bosse and R. Zlot, "Map matching and data association for large-scale two-dimensional laser scan-based SLAM", *International Journal of Robotics Research*, vol. 27, no. 6, pp. 667–691, 2008.

[35] R. Haralick, H. Joo, C.-N. Lee, X. Zhuang, V. G. Vaidya, and M. B. Kim, "Pose Estimation from Corresponding Point Data", *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 19, no. 6, pp. 1426–1446, 1989.

[36] D. Chetverikov, D. Stepanov, and P. Krsek, "Robust Euclidean alignment of 3D point sets: The trimmed iterative closest point algorithm", *Image and Vision Computing*, vol. 23, no. 3, pp. 299–309, 2005.

[37] S. Du, J. Zhu, N. Zheng, Y. Liu, and C. Li, "Robust iterative closest point algorithm for registration of point sets with outliers", *Optical Engineering*, vol. 50, no. 8, p. 087 001, 2011.

[38] M. Sheehan, A. Harrison, and P. Newman, "Self-calibration for a 3D laser", *International Journal of Robotics Research*, vol. 31, no. 5, pp. 675–687, 2012.

[39] Y. Tsin and T. Kanade, "A Correlation-Based Approach to Robust Point Set Registration", in *8th European Conference on Computer Vision*, vol. 0, 2004, pp. 558–569.

[40] R. Zlot and M. Bosse, "Efficient large-scale three-dimensional mobile mapping for underground mines", *Journal of Field Robotics*, vol. 31, no. 5, pp. 758–779, 2014.

[41] M. Pierzchała, P. Giguère, and R. Astrup, "Mapping forests using an unmanned ground vehicle with 3d lidar and graph-slam", *Computers and Electronics in Agriculture*, vol. 145, pp. 217–225, 2018.

[42] S. Williams, L. T. Parker, and A. M. Howard, "Terrain Reconstruction of Glacial Surfaces : Robotic Surveying Techniques", *IEEE Robotics and Automation Magazine*, vol. 19, no. 4, pp. 59–71, 2012.

[43] P. Besl and N. McKay, "A Method for Registration of 3-D Shapes", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 14, no. 2, pp. 239–256, 1992.

[44] S. Thrun and M. Montemerlo, "The GraphSLAM Algorithm with Applications to Large-Scale Mapping of Urban Structures", *The International Journal of Robotics Research*, vol. 25, no. 5-6, pp. 403–429, 2006.

[45] C. Cadena, L. Carlone, H. Carrillo, Y. Latif, D. Scaramuzza, *et al.*, "Past, Present, and Future of Simultaneous Localization and Mapping: Toward the Robust-Perception Age", *IEEE Transactions on Robotics*, vol. 32, no. 6, pp. 1309–1332, 2016.

[46] J. H. Lever, A. J. Delaney, L. E Ray, E. Trautmann, L. A. Barna, and A. M. Burzynski, "Autonomous GPR Surveys using the Polar Rover Yeti", *Journal of Field Robotics*, vol. 30, no. 2, pp. 194–215, 2013.

[47] G. Jagbrant, J. P. Underwood, J. Nieto, and S. Sukkarieh, "LiDAR Based Tree and Platform Localisation in Almond Orchards", in *Proceedings of the Conference on Field and Service Robotics*, 2015, pp. 469–483.

[48] S. Williams and A. M. Howard, "Developing Monocular Visual Pose Estimation for Arctic Environments", *Journal of Field Robotics*, vol. 71, no. 5, pp. 486–494, 2009.

[49] M. Paton, F. Pomerleau, and T. Barfoot, "In the dead of winter: challenging vision-based path following in extreme conditions", *Springer Tracts in Advanced Robotics*, vol. 113, pp. 563–576, 2016.

[50] M. W. McDaniel, N. Takayuki, C. A. Brooks, P. Salesses, and K. Lagnemma, "Terrain Classification and Identification of Tree Stems Using Ground-based LiDAR", *Journal of Field Robotics*, vol. 29, pp. 891–910, 2012.

[51] L. Wallace, A. Lucieer, Z. Malenovský, D. Turner, *et al.*, "Assessment of Forest Structure Using Two UAV Techniques: A Comparison of Airborne Laser Scanning and Structure from Motion (SfM) Point Clouds", *Forests*, vol. 7, no. 3, 2016.

[52] J. Tian, R. P., P. d'Angelo, and M. Ehlers, "Region-based automatic building and forest change detection on cartosat-1 stereo imagery", *International Society for Photogrammetry and Remote Sensing*, vol. 79, pp. 226 –239, 2013.

[53] N. Ohta and K. Kanatani, "Optimal Estimation of Three- Dimensional Rotation and Reliability Evaluation", *European Conference on Computer Vision*, pp. 175–187, 1998.

[54] R. S. J. Estépar, A. Brun, and C.-F. Westin, "Robust Generalized Total Least Squares Iterative Closest Point Registration", in *Medical Image Computing and Computer Assisted Intervention*, 2004, pp. 234–241.

[55] L. Maier-hein, A. M. Franz, T. R. Santos, M. Schmidt, M. Fangerau, *et al.*, "Convergent ICP Algorithm to Accomodate Anisotropic and Inhomogenous Localization Error", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 34, no. 8, pp. 1520–1532, 2012.

[56] S. D. Billings, E. M. Boctor, and R. H. Taylor, "Iterative Most-Likely Point Registration (IMLP): A Robust Algorithm for Computing Optimal Shape Alignment", *PLoS ONE*, pp. 1–45, 2015.

[57] R. Balachandran and J. M. Fitzpatrick, "Iterative solution for rigid-body point-based registration with anisotropic weighting", vol. 7261, M. I. Miga and K. H. Wong, Eds., pp. 1051 –1060, 2009.

[58] F. Pomerleau, P. Krüsi, F. Colas, P. Furgale, and R. Siegwart, "Long-term 3D map maintenance in dynamic environments", in *IEEE International Conference on Robotics and Automation (ICRA)*, 2014, pp. 3712–3719.

[59] J. Laconte, S.-P. Deschênes, M. Labussière, and F. Pomerleau, "Lidar measurement bias estimation via return waveform modelling in a context of 3d mapping", in *IEEE International Conference on Robotics and Automation (ICRA)*, 2019, pp. 8100–8106.

[60] R. Kümmerle, G. Grisetti, H. Strasdat, K. Konolige, and W. Burgard, "G2o: a general framework for graph optimization", in *2011 IEEE International Conference on Robotics and Automation*, 2011, pp. 3607–3613.

[61] J. Zhang and S. Singh, "LOAM: Lidar Odometry and Mapping in Real- time", in *IEEE Transactions on Robotics*, vol. 32, 2015, pp. 141–148.

[62] J. Zhu, D. Wang, X. Bai, H. Lu, C. Jin, and Z. Li, "Registration of point clouds based on the ratio of bidirectional distances", *Proceedings of the 4th International Conference on 3D Vision*, pp. 102–107, 2016.

[63] M. Bosse and R. Zlot, "Continuous 3D scan-matching with a spinning 2D laser", *2009 IEEE International Conference on Robotics and Automation*, pp. 4312–4319, 2009.

[64] S. Geman, D. E. McClure, and D. Geman, "A nonlinear filter for film restoration and other problems in image processing", *CVGIP: Graphical Models and Image Processing*, vol. 54, no. 4, pp. 281 –289, 1992.

[65] N. Sünderhauf and P. Protzel, "Switchable constraints for robust pose graph slam", in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2012, pp. 1879–1884.

[66] P. Agarwal, G. D. Tipaldi, L. Spinello, C. Stachniss, and W. Burgard, "Robust map optimization using dynamic covariance scaling", *IEEE International Conference on Robotics and Automation*, pp. 62–69, 2013.

[67] D. Chetverikov, D. Svirko, D. Stepanov, and P. Krsek, "The Trimmed Iterative Closest Point algorithm", *Object recognition supported by user interaction for service robots*, vol. 3, no. c, pp. 545–548, 2002.

[68] A. Censi, "An ICP variant using a point-to-line metric", *IEEE International Conference on Robotics and Automation (ICRA)*, pp. 19–25, 2008.

[69] M. A. Fischler and R. C. Bolles, "Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography", *Communications of the ACM*, vol. 24, no. 6, pp. 381–395, 1981.

[70] Q. Y. Zhou, J. Park, and V. Koltun, "Fast global registration", *Lecture Notes in Computer Science*, vol. 9906 LNCS, pp. 766–782, 2016.

Appendix A

List of Outlier Filters for Iterative Closest Point

Table A.1 – Extended list of 26 outlier filters for ICP. In some case the specific outlier filter does not have a name, hence the name of its variant of ICP is used. The breakdown point is the maximum proportion of outliers beyond which the estimator may develop an arbitrarily large bias [26]. The influence function $\psi(\cdot)$ is not included, but can be determined by simply multiplying the weight function $w(\cdot)$ by the scaled error e (see Equation 2.4). The other variables are the tuning parameter k, the unscaled error (aka the Euclidean distance) e', the f^{th} percentile of the matches P_f, where f is the overlap ratio parameter, the ratio between the closest match distance and the closest match's closest match distance β [62], and the distance threshold at the t iteration d_t (it updated at each iteration based on the relative translation motion of the previous iteration [25]).

Name	M-Estimator	Binary Weights	Adaptative	Breakdown point	Condition	Cost function(ρ)	Weight function(w)
L_2	✓	✓	✗	0 %[17]		$\frac{e^2}{2}$	1
L_1	✗	✗	✗	0 %		$\lvert e\rvert$	$\frac{1}{\lvert e\rvert}$
Huber [10]	✓	✗	✗	50 %[26]	$\begin{cases} \lvert e\rvert \leq k \\ \text{otherwise} \end{cases}$	$\begin{cases} \frac{e^2}{2} \\ k(\lvert e\rvert - k/2) \end{cases}$	$\begin{cases} 1 \\ \frac{k}{\lvert e\rvert} \end{cases}$
Cauchy [63]	✓	✗	✗	25 %[26]		$\frac{k^2}{2}\log(1+(e/k)^2)$	$\frac{1}{1+(e/k)^2}$
Geman-McClure (GM) [64]	✓	✗	✗	10 %		$\frac{e^2/2}{k+e^2}$	$\frac{k^2}{(k+e^2)^2}$
Switchable-Constraint (SC) [65][66]	✓	✗	✗	n/a	$\begin{cases} e^2 \leq k \\ \text{otherwise} \end{cases}$	$\begin{cases} \frac{e^2}{2} \\ \frac{2ke^2}{k+e^2} - k/2 \end{cases}$	$\begin{cases} 1 \\ \frac{4k^2}{(k+e^2)^2} \end{cases}$
Welsch [17]	✓	✗	✗	10 % [26]		$\frac{k^2}{2}(1-\exp(-(\frac{e}{k})^2))$	$\exp(-(e/k)^2)$

§ The breakdown point is calculate as $\min(100\,\% - f, 50\,\%)$, where f is the overlap ratio parameter. For RANSAC and Trimmed this parameter is manually tuned. For Var. Trim. and AICP, f is estimated using an heuristic.

Name	M-Estimator	Binary Weights	Adaptative	Breakdown point	Condition	Cost function(ρ)	Weight function(w)
Tukey [17]	✓	✗	✗	n/a	$\begin{cases} \lvert e\rvert \leq k \\ \text{otherwise} \end{cases}$	$\begin{cases} \frac{k^2(1-(1-(\frac{e}{k})^2)^3)}{2} \\ \frac{k^2}{2} \end{cases}$	$\begin{cases} (1-(e/k)^2)^2 \\ 0 \end{cases}$
Andrew [17]	✓	✗	✗	?	$\begin{cases} \lvert e\rvert \leq \pi k \\ \text{otherwise} \end{cases}$	$\begin{cases} k^2(1-\cos(e/k)) \\ 2k^2 \end{cases}$	$\begin{cases} \frac{k}{e}\sin(e/k) \\ 0 \end{cases}$
Fair [17]	✓	✗	✗	?		$k^2(e/k - \log(1+e/k))$	$\frac{1}{1+e/k}$
Logistic [17]	✓	✗	✗	?		$k^2 \log(\cosh(e/k))$	$\frac{\tanh e/k}{e/k}$
Student [32]	✗	✗	✗	n/a			$\frac{(k+3)(1+\frac{e^2}{k})^{-\frac{k+3}{2}}}{k+e^2}$
Maximum Distance	✓	✓	✗	n/a	$\begin{cases} \lvert e'\rvert \leq k \\ \text{otherwise} \end{cases}$	$\begin{cases} \frac{e'^2}{2} \\ \frac{k^2}{2} \end{cases}$	$\begin{cases} 1 \\ 0 \end{cases}$
Trimmed [67]	✗	✓	✗	§	$\begin{cases} e \leq P_f \\ \text{otherwise} \end{cases}$		$\begin{cases} 1 \\ 0 \end{cases}$
Median [68]	✗	✓	✗	§	$\begin{cases} e \leq P_{50\,\%} \\ \text{otherwise} \end{cases}$		$\begin{cases} 1 \\ 0 \end{cases}$
FICP (i.e. Var. Trim.) [27]	✗	✓	✓	§	$\begin{cases} e \leq P_f \\ \text{otherwise} \end{cases}$		$\begin{cases} 1 \\ 0 \end{cases}$
RANSAC [69]	✗	✓	✗	§			

§ The breakdown point is calculate as $\min(100\,\% - f, 50\,\%)$, where f is the overlap ratio parameter. For RANSAC and Trimmed this parameter is manually tuned. For Var. Trim. and AICP, f is estimated using an heuristic.

Name	M-Estimator	Binary Weights	Adaptative	Breakdown point	Condition	Cost function(ρ)	Weight function(w)
GICP [33]	✓	✓	✗	n/a	$\begin{cases} \lvert e\rvert \leq k \\ \text{otherwise} \end{cases}$	$\begin{cases} \frac{e^2}{2} \\ \frac{k^2}{2} \end{cases}$	$\begin{cases} 1 \\ 0 \end{cases}$
BiDistance [62]	✗	✗	✗	n/a			$\exp(k(\beta-1))$
AICP [22]	✗	✓	✓	§	$\begin{cases} e \leq P_f \\ \text{otherwise} \end{cases}$		$\begin{cases} 1 \\ 0 \end{cases}$
LM-ICP [24]	✓	✗	✗	50 %[26]	$\begin{cases} \lvert e\rvert \leq k \\ \text{otherwise} \end{cases}$	$\begin{cases} \frac{e^2}{2} \\ k(\lvert e\rvert - k/2) \end{cases}$	$\begin{cases} 1 \\ \frac{k}{\lvert e\rvert} \end{cases}$
EM-ICP [23]	✓	✗	✓	n/a	$\begin{cases} \lvert e\rvert \leq 3 \\ \text{otherwise} \end{cases}$	$\begin{cases} \lvert e\rvert \\ 0 \end{cases}$	$\begin{cases} \frac{1}{e} \\ 0 \end{cases}$
Kernel Correlation (KC) [39]	✓	✗	✗	10 %[26]		$\frac{k^2}{2}(1-\exp(-(\frac{e}{k})^2))$	$\exp(-(e/k)^2)$
Fast Global Registration (FGR) [70]	✓	✗	✓	10 %		$\frac{e^2/2}{k+e^2}$	$\frac{k^2}{(k+e^2)^2}$
Relative Motion Threshold (RMT) [25]	✗	✓	✓	n/a	$\begin{cases} e \leq d_t \\ \text{otherwise} \end{cases}$		$\begin{cases} 1 \\ 0 \end{cases}$
Sparse ICP [28]	✗	✗	✗	n/a	$0 \leq p \leq 1$	e^p	pe^{p-2}

§ The breakdown point is calculate as $\min(100\,\% - f, 50\,\%)$, where f is the overlap ratio parameter. For RANSAC and Trimmed this parameter is manually tuned. For Var. Trim. and AICP, f is estimated using an heuristic.