Foundations and Trends® in Robotics
Vol. X, No. X (2014) 1–XX

DOI: XXX

now
the essence of knowledge

A Review of Point Cloud Registration Algorithms for Mobile Robotics

François Pomerleau
Autonomous Space Robotics Laboratory – University of Toronto
Toronto, Canada
f.pomerleau@gmail.com

Francis Colas
Inria, Villers-lès-Nancy, F-54600, France
CNRS, Loria, UMR 7503, Vandœuvre-lès-Nancy, F-54500, France
Université de Lorraine, Vandœuvre-lès-Nancy, F-54500, France
francis.colas@inria.fr

Roland Siegwart
Autonomous Systems Lab – ETH Zurich
Zurich, Switzerland
rsiegwart@ethz.ch

Contents

Abstract

The topic of this review is geometric registration in robotics. Registration algorithms associate sets of data into a common coordinate system. They have been used extensively in object reconstruction, inspection, medical application, and localization of mobile robotics. We focus on mobile robotics applications in which point clouds are to be registered. While the underlying principle of those algorithms is simple, many variations have been proposed for many different applications. In this review, we give a historical perspective of the registration problem and show that the plethora of solutions can be organized and differentiated according to a few elements. Accordingly, we present a formalization of geometric registration and cast algorithms proposed in the literature into this framework. Finally, we review a few applications of this framework in mobile robotics that cover different kinds of platforms, environments, and tasks. These examples allow us to study the specific requirements of each use case and the necessary configuration choices leading to the registration implementation. Ultimately, the objective of this review is to provide guidelines for the choice of geometric registration configuration.

Keywords Survey; Review; Iterative Closest Point algorithm; Point set registration; Geometric registration; Mobile robotics; Laser odometry; Search and Rescue; Inspection; Environmental monitoring; Autonomous Driving.

F. Pomerleau, F. Colas and R. Siegwart. *A Review of Point Cloud Registration Algorithms for Mobile Robotics*. Foundations and Trends® in Robotics, vol. X, no. X, pp. 1–XX, 2014.

DOI: XXX.

1

Twenty years of Iterative Closest Point (ICP): The Legacy

The scope of this work is to present *registration* algorithms and their use in mobile robotics. Registration algorithms associate sets of data into a common coordinate system by minimizing the alignment error. This allows to integrate data from different sources into a bigger model.

Although they can be quite an abstract and technical concept, registration solutions already had an impact on the artistic field and popular culture. Photographers proficiently use image registration to build photograph composites achieving different looks-and-feels. The Brenizer method is an exemplary technique that is applied to achieve dramatic depth of field using panoramic image stitching (Figure 1.1 - Top). Another example is High Dynamic Range (HDR) photographs, where multiple images at different exposure levels need to be precisely overlaid to retrieve details in shaded and highlighted areas (Figure 1.1 - Bottom). Nowadays, even the latest cellphones have the capacity to build panoramic images from a series of pictures taken based on a visual guidelines that direct the user to move the camera viewfinder at the optimal position for the next picture. As for the specific case of 3D mapping application, cinematographers are depicting possible uses of registration algorithms in several recent science fiction movies. For

instance, in the remake of *Total Recall* (Colombia Pictures, 2012), an armed intervention team employed an array of hundreds of tiny cameras in a dangerous room leading to a 3D reconstruction of the area used to monitor potential threats within couple of seconds. Another closely related potential application was the used by a geologist of flying drones carrying laser rangefinders to explore an alien facility in *Prometheus* (Twentieth Century Fox, 2012).

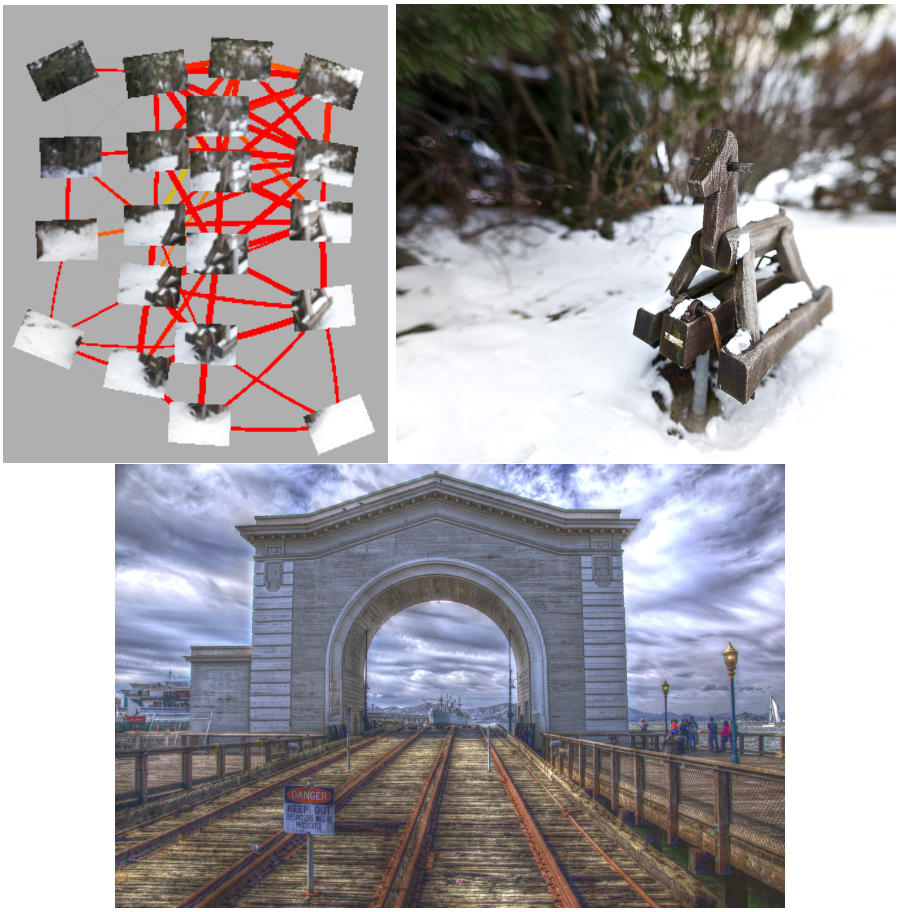

Figure 1.1: Example of image registrations used in photography. *Top*: Brenizer method using the open source software Hugin to stitch multiple images. *Bottom*: HDR composite of the San Francisco harbor using the open source software Luminance HDR to overlay three images.

More at the research level, current applications include: robotic exploration in harsh environments, organ reconstruction to improve medical diagnostics and object reconstruction for reverse engineering. Although registration using 2D images can be part of the same group of solutions, we focus on systems where depth information is already available (e.g., from laser rangefinders) and is mainly used for resolving misalignment error. We refer to the latter type as *geometric registration*[1]. However, some parallels with image registrations will be made throughout this work when relevant.

A simplified example of geometric registration is illustrated in Figure 1.2. A scene with a large tree, a lamppost and a bench was scanned using a laser rangefinder from two different poses. As laser points are indistinguishable, only their location information is available to resolve the alignment error. In that example, the point cloud in light green and with the horizontal ground is used as our fixed reference coordinates. Figure 1.2-*Left* shows the starting position of the two scans. The overlaid point cloud in dark blue has a misalignment error shifting it to the left with a tilt angle. This initial misalignment is represented with dark red lines in Figure 1.2-*Middle*. Although all individual points are similar, their proximity to other points gives enough information to automatically align the two point clouds (Figure 1.2-*Right*).

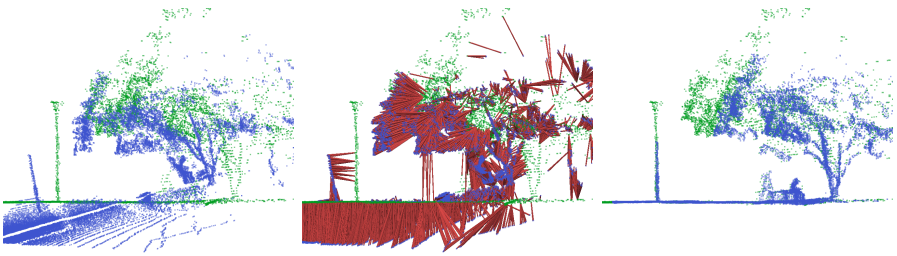

Figure 1.2: Examples of geometric registration between a reference point cloud (light green points) and a reading point cloud (dark blue points). *Left*: Initial position of the two point clouds. *Middle*: Alignment error (dark red lines). *Right*: Final alignment of the two point clouds.

[1]In general, image registration often has access to *labelled* points, which is less the case for geometric registration, either in 2D or 3D.

1.1 Early Solutions

As an interesting historical note, in an early publication by Hurley and Cattell [1962], registration is presented as an *Orthogonal Procrustes* problem. The name Procrustes refering to a bandit from the Greek mythology who made his victims fit on his bed by either stretching their limbs or cutting them off. Theseus eventually defeated Procrustes using the same violent procedure (Figure 1.3). Nowadays, the reference to the *Orthogonal Procrustes* problem is not often used in the scientific literature, but it illustrates well the idea.

Figure 1.3: Theseus *adjusting* Procrustes to the size of his bed. Photograph provided by Marie-Lan Nguyen / Wikimedia Commons.

While working more specifically on 3D-shape primitives, Faugeras and Hebert [1986] defined closed-form distances to minimize point-to-point and plane-to-plane alignment error. The proposed method solved translation and rotation as a two-step procedure. Later, a solution proposed by Walker et al. [1991] resolved together rotation and translation error using dual quaternions. The registration problem concretizes itself further in a survey of geometric matching algorithms and geometric representations for point sets, curves, surfaces, volumes, and their respective space-time trajectories [Besl, 1988]. At this time, the main ex-

perimental validation was using Computer-aided design (CAD) models with simple shapes. The first mention of the name ICP[2] was proposed by Besl and McKay [1992]. They expressed the problem as follows:

> "Given 3-D data in a sensor coordinate system, which describes a data shape that may correspond to a model shape, and given a model shape in a model coordinate system in a different geometric shape representation, estimate the optimal rotation and translation that aligns, or registers, the model shape and the data shape minimizing the distance between the shapes and thereby allowing determination of the equivalence of the shapes via a mean-square distance metric."

In their work, the proof of the solution convergence is demonstrated under the assumption that the number of associated points, or their weight, remains constant. Unless two identical shapes are registered together, outliers that are not present in both shapes need to be identified. This problems is observed by Champleboux et al. [1992] while developing early registration solutions for medical applications. They report failures when wrong initial transformations are used in combination with scans having low overlap ratio. During the same years, Chen and Medioni [1991] work with dense laser scans of statues and, shortly later, scans of tooth mockups [Chen and Medioni, 1992]. They propose a registration solution based on the minimization of point-to-plane alignment errors, which is still quite often used nowadays.

Even though a large volume of theoretical works was published on advanced geometric primitives (e.g., planes, curves, quadrics), Zhang [1994] states that primitives derived from points are too sensitive to noise and are not stable in moving systems with current (1994) sensing capabilities. Thus, he concludes that points were more reliable. Zhang [1994] pioneers the idea of using ICP-based solutions for outdoor robotic applications. He proposes a generic framework for symmetric match, but considers only one direction of registration as an

[2]In the remainder of this review, ICP and geometric registration have the same generic meaning.

approximation to save computation costs. He highly emphasizes the importance of removing spurious pairs and gives the first characterization of fast subsampling solutions. In addition, he highlighted the fact that outlier rejection is required for robotic applications, and that the proof of ICP convergence stated by Besl and McKay [1992] cannot hold for most of the robotics applications. In the outlook section of his work, he already mentions the use of uncertainty on the initial alignment, based on Kalman filters and Mahalanobis distance, and the need to handle dynamic elements.

1.2 Division and Explosion of the Field

Within only two years, four main application types already emerged from the possibilities to register 3D point clouds: object reconstructions [Chen and Medioni, 1991], non-contact inspections [Besl and McKay, 1992], medical and surgery support [Champleboux et al., 1992] and autonomous vehicle navigation [Zhang, 1993]. Publications in specialized journals for computer vision, robotics and medical imaging slowly divided the types of *interesting* problems to be solved. We can still read in current literature that the credits for being the first article to provide a solution differ from authors in different fields.

The field of registration crystalized with its first survey on medical image registration covering the years 1993 to 1998 [Maintz and Viergever, 1998]. It took 12 years for a specialized survey of 3D registration in computer vision to appear [Bowyer et al., 2006]. This work intends to be the first large scale review adapted for Robotics application.

ICP is a popular algorithm due to its simplicity: its general idea is easy to understand and to implement. However, the basic algorithm only works well in ideal cases. This led to hundreds[3] of variations around the original algorithm that were published on numerous different experimental scenarios (see Figure 1.4). This highlights both the usefulness of ICP and the difficulty to find a single versatile version.

[3]Close to 450 papers based on IEEE Xplore and around 1350 based on Scopus, between 1992 and 2013.

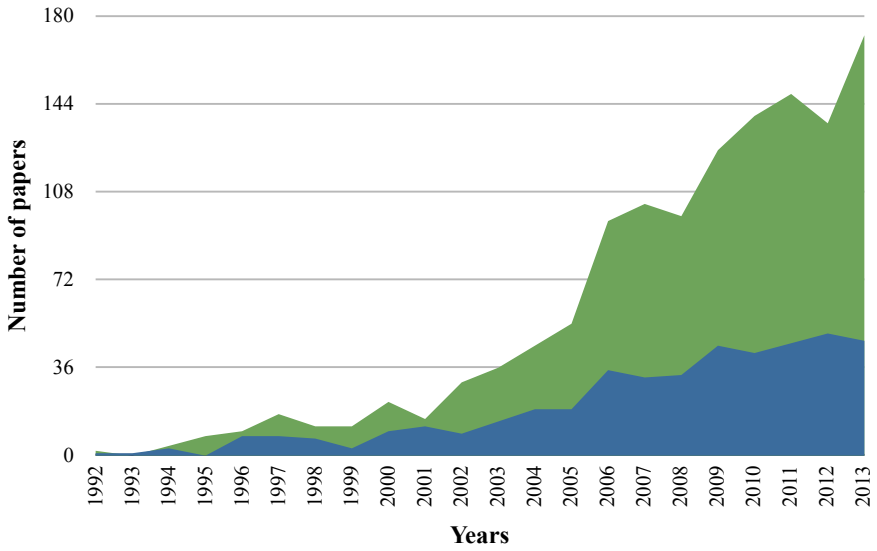

Figure 1.4: Evolution of the number of publications over the years. Results were obtained for the keywords *Iterative Closest Point* appearing in the abstract or the title of publications. The dark blue area is computed based on IEEE Xplore database and the light green area from the Scopus database.

In Figure 1.4, one can observe an increasing number of publications appearing around the year 2000. In robotics, this coincides with the advent of a 2D solution for pose estimations demonstrated with a SICK rangefinder [Lu and Milios, 1997] and of the basis of Simultaneous Localization and Mapping (SLAM) algorithms [Thrun et al., 1998]. Prior to the arrival of the SICK LMS-200 in robotics [Pfister et al., 2002], most of the sensors used were custom-made. This situation renders experiments difficult to replicate by other researchers. In those years, 2D lasers appeared as a viable solution for navigation over sonars, which were traditionally used [Thrun et al., 1998]. The 3D real-time applications were then not accessible due to high computation costs leading to an increased research focus toward 2D solutions for autonomous navigation, while other fields continued in 3D. At the same time in computer vision, the seminal work of Rusinkiewicz and Levoy [2001] on ICP algorithm comparisons led to significant progress in the scan

registration field. The experiments employed simulated 3D scans, highlighting different spatial constraints and sensor noises. Results mainly focused on the rapidity of convergence and the final precision of different solutions helping to select more appropriate solutions in further applications.

With the arrival of more standard sensors, researchers in robotics pushed the 2D registration algorithms so they could deal with larger environments with faster platforms [Bosse et al., 2004] and 3D slowly came back [Nüchter et al., 2004]. Since no comparison framework exists, the selection of an appropriate variant for particular experimental conditions is difficult. This is a major challenge because registration is at the front-end of the mapping pipeline, and the arbitrary nature of its selection affects the results of all subsequent steps of more advanced robotic tasks. Even the early work of Eggert et al. [1998] highlights the difficulty to compare with other solutions given the lack of metric over common data sets. In their survey, Maintz and Viergever [1998] point the fact that proper accuracy studies are just starting; the problem being that the results provided are too specific. In addition, they highlight the imprecise use of the terms accuracy, precision and robustness. They suggest to set up public databases and validation protocols, but foresee logistic, costs and efforts as incoming problem to those solutions.

Recently, the demand for a stronger experimental methodology in robotics was also stressed by Amigoni et al. [2009]. The authors survey different SLAM publications to highlight proper evaluation metrics that are applied to SLAM algorithms. Three principles of an experimental methodology in science (i.e, comparison, reproducibility/repeatability and justification/explanation) are translated in requirements for stronger SLAM results. As stated in their paper, a sound methodology should allow researchers to gain an insight about *intrinsic* (e.g., computational time, parameters used, parameter behaviors) and *extrinsic* (e.g., accuracy, precision) quantities. The authors report that, even though comparisons between algorithms are present in SLAM publications, very few researchers can reuse the same protocol and directly compare their results without having to re-implement other solutions.

With the introduction of the Microsoft Kinect in 2010, another wave of publications is expected, similar to what was observed following the widespread utilization of SICK rangefinders. The Kinect is a handheld camera sensor connected via USB to a computer that produces both depth and color readings. Such RGB-D sensors augment accessibility to object modeling and indoor mapping research [Henry et al., 2012]. This also opens the door to a mix of hybrid algorithms using features and descriptors without the need of expertise in sensor calibration. RGB-D cameras have different characteristics than laser-based sensors, such as a higher density of points at a higher frequency but covering a more restricted Field of View (FoV). A smaller FoV means less time to compute the registration before the sensor trajectory reduce the overlap to an unusable range. Having access to a higher frame rate with an optimized ICP solution shows that hand-waved sensor trajectory was trackable with real-time constraints [Pomerleau et al., 2011]. The Velodyne HDL-64E, first commercialized for the DARPA Urban Challenge in 2007, optimized its FoV to cover the expected trajectory of a ground vehicle. To cope with the high speed of a car, the sensor delivered a high data rate at 1.3 M points per second, bring the real-time constraint to another level. Those two sensors were the latest publication catalysts for the field of registration in mobile robotics, field often modulated by the development of new hardwares.

1.3 Algorithm Overview

The aim of geometric registration is to be able to represent a shape, called `reading`, in the same coordinate frame as another, called `reference`. This is equivalent to finding the transformation of `reading` that best aligns it to `reference`.

A shape $\mathcal{S}$ is a set of points including both geometric and non-geometric information. Geometric information is affected by a spatial transformation; this part of the dimension of a point will be called a *feature*. Features are typically coordinates of points, surface normals or tangent vectors. Non-geometric information is not affected by spatial

transformation; this part of the dimension of a point will be called a *descriptor*. Descriptors can be color, temperature, identifiers, etc.

Most algorithms actually apply some filters on the shapes in order to help the registration. There are mainly two uses of such filters. The first one is to remove some points that do not bring any valuable information for the registration. As the complexity of the algorithm is linear in the number of points, reducing this number can have a significant impact on the time of registration. The second use of filters can be to add information to the point. The typical example is the inference of local structural properties of the shape, such as normal information or curvature. This information, which is usually not present in the raw sensor data, can allow for better registration through a more precise association of the points, or the computation of the error to minimize.

More formally, let ${}^{\mathbb{A}}\mathcal{P}$ be the shape representing `reading` in a coordinate frame $\mathbb{A}$ and ${}^{\mathbb{B}}\mathcal{Q}$ the shape representing `reference` in its coordinate frame $\mathbb{B}$. The aim of registration is to estimate the transformation ${}^{\mathbb{B}}_{\mathbb{A}}\mathcal{T}$ by minimizing an error function $\text{error}(\mathcal{P}', \mathcal{Q})$:

$$ {}^{\mathbb{B}}_{\mathbb{A}}\hat{\mathcal{T}} = \arg\min_{\mathcal{T}} \left(\text{error} \left(\mathcal{T} \left({}^{\mathbb{A}}\mathcal{P} \right), {}^{\mathbb{B}}\mathcal{Q} \right) \right) \tag{1.1} $$

where $\mathcal{T}(\mathcal{S})$ is the application of the geometric transformation $\mathcal{T}$ to the shape $\mathcal{S}$.

One specificity of geometric registration is that the error function is computed on pairs of points that have been associated between the two shapes. The classical association is done by finding the closest point in `reference` of each point in `reading`. Ideally the association should be between points that, when the two shapes are aligned, are the closest in position. This problem is called *data association*, *point matching*, *correspondence finding* depending on the literature. Association solving can be done purely on the features but can also be improved by using the descriptors.

Formally, let $\mathcal{M} = \text{match}(\mathcal{P}, \mathcal{Q}) = \{(\boldsymbol{p}, \boldsymbol{q}) : \boldsymbol{p} \in \mathcal{P}, \boldsymbol{q} \in \mathcal{Q}\}$ be the set of matches between $\mathcal{P}$ and $\mathcal{Q}$. The error function is then of the form:

$$ \text{error}(\mathcal{P}, \mathcal{Q}) = \sum_{(\boldsymbol{p},\boldsymbol{q}) \in \mathcal{M}} \text{d}(\boldsymbol{p}, \boldsymbol{q}). $$

In order to make this error function more robust, *outliers* are sometimes identified and removed from the list of matches. In addition, weights $\mathcal{W} = \text{outlier}(\mathcal{M}) = \{w(\boldsymbol{p}, \boldsymbol{q}) : \forall (\boldsymbol{p}, \boldsymbol{q}) \in \mathcal{M}\}$ can be associated to the matches so as to increase or decrease their influence in the error function:

$$\text{error}(\mathcal{P}, \mathcal{Q}) = \sum_{(\boldsymbol{p},\boldsymbol{q})\in\mathcal{M}} w(\boldsymbol{p}, \boldsymbol{q})\, \text{d}(\boldsymbol{p}, \boldsymbol{q}).$$

It is clear that minimizing this error function with an ideal association yields the best estimate for ${}^{\mathbb{B}}_{\mathbb{A}}\mathcal{T}$ (Equation 1.1). However, unless the descriptors are discriminative enough (as with visual descriptors), the association can generally not be perfectly solved. The idea of ICP is that even with an imperfect association, minimizing the error yields a better estimates that, in turn, allows for better association. Concretely, the idea is to build a sequence of transformations ${}_{i-1}{}^{i}\mathcal{T}$ that are successively applied to $\mathcal{P}$. At a given iteration, a set of matches $\mathcal{M}_i$ is computed from the given relative position of the points. Then, based of those matches, a new transformation ${}^{i+1}_{i}\mathcal{T}$ is computed by minimizing the error:

$${}^{i+1}_{i}\mathcal{T} \leftarrow \arg\min_{\mathcal{T}} \left(\text{error}\left(\mathcal{T}\left({}^{i}\mathcal{P}'\right), \mathcal{Q}'\right)\right). \tag{1.2}$$

Finally, the estimate of the transformation between the two original shapes is the composition of all intermediary transformations:

$${}^{\mathbb{B}}_{\mathbb{A}}\hat{\mathcal{T}} = \left(\bigcirc_{i} {}_{i-1}{}^{i}\mathcal{T}\right) \circ \mathcal{T}_{init} \tag{1.3}$$

where $\bigcirc_{i} {}_{i-1}{}^{i}\mathcal{T} = \cdots \circ {}_{2}{}^{3}\mathcal{T} \circ {}_{1}{}^{2}\mathcal{T}$ is the iterative composition of the transformations, and $\mathcal{T}_{init}$ an initial transformation.

The generic procedure is summarized in Algorithm 1 and shown as a chart in Figure 1.5.

1.4 Overview of the Review

ICP is a framework where multiple variations and algorithms can be used to resolve geometric registration problems. In the light of this large corpus of work related to ICP and more generally to geometric

Algorithm 1 Summary of ICP algorithm.

Require: ${}^{\mathbb{A}}\mathcal{P}$ ▷ `reading`

Require: ${}^{\mathbb{B}}\mathcal{Q}$ ▷ `reference`

Require: $\mathcal{T}_{init}$ ▷ initial transformation

${}^{\mathbb{A}}\mathcal{P}' \leftarrow \text{datafilter}({}^{\mathbb{A}}\mathcal{P})$ ▷ data filters

${}^{\mathbb{B}}\mathcal{Q}' \leftarrow \text{datafilter}({}^{\mathbb{B}}\mathcal{Q})$ ▷ data filters

${}_{i-1}{}^{i}\mathcal{T} \leftarrow \mathcal{T}_{init}$

repeat

${}^{i}\mathcal{P}' \leftarrow {}_{i-1}{}^{i}\mathcal{T}({}^{i-1}\mathcal{P}')$ ▷ move `reading`

$\mathcal{M}_i \leftarrow \text{match}({}^{i}\mathcal{P}', \mathcal{Q}')$ ▷ associate points

$\mathcal{W}_i \leftarrow \text{outlier}(\mathcal{M}_i)$ ▷ filter outliers

${}^{i+1}_{i}\mathcal{T} \leftarrow \arg\min_{\mathcal{T}} \left(\text{error}\left(\mathcal{T}\left({}^{i}\mathcal{P}'\right), \mathcal{Q}'\right)\right)$

until convergence

Ensure: ${}^{\mathbb{B}}_{\mathbb{A}}\hat{\mathcal{T}} = \left(\bigcirc_{i} {}_{i-1}{}^{i}\mathcal{T}\right) \circ \mathcal{T}_{init}$

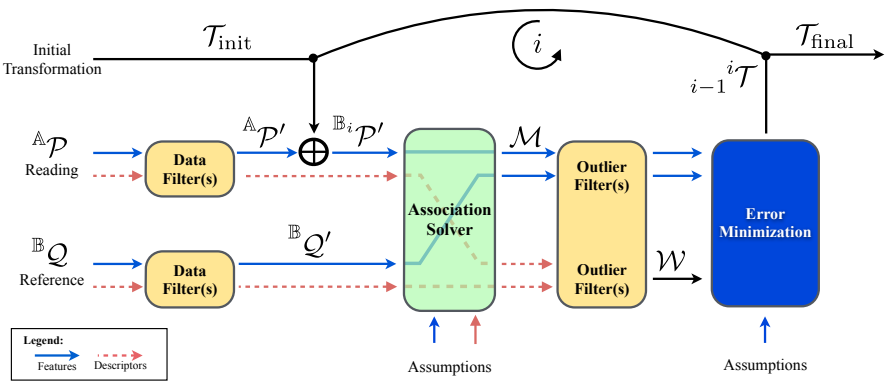

Figure 1.5: Generic scheme proposed for registration algorithms.

registration, we present a general framework to classify the existing solutions. We believe that after 20 years of *new* registration algorithms, it is time to evaluate what works best for which robotic systems. Therefore, our contributions aim at strengthening the current methodology and bring deeper analysis of current solutions. The timing is appropriate for such study given that computational power is now sufficient to support registration on embedded systems in real-time [Pomerleau et al., 2011]. Also, new advancements in electronics have improved the accuracy and speed of sensors. Improvements in battery technology have enabled longer autonomous operation time. Most importantly, researchers face a plethora of solutions from which a definition of usable solutions can be out of reach. This situation impedes the robotic field to progress on algorithms that rely on registration (e.g., path planning, autonomous exploration).

This review addresses this problem and is structured in two main sections:

Section 2 presents a literature review of different solutions with the aim to express ICP solutions in a common framework and validate our generic scheme proposed in Figure 1.5.

Section 3 describes case studies using five different robotic platforms. The requirements of each application are explained with some insight on how to tune parameters for specific applications. Those applications cover Search & Rescue activities, industrial inspection, shore monitoring and autonomous driving.

All sections close with a discussion in addition to a short summary. The main observations of those sections are recapitulated in Section 4 along with final remarks.

2

Formalization of the ICP Solution Family

The spread of registration solutions renders tedious the task of finding particular problems to address, knowing their limits and applying them properly. Although surveys exist to support registration in medical imagery [Maintz and Viergever, 1998, Markelj et al., 2012, Fluck et al., 2011, Pluim et al., 2003] and object reconstructions/recognition [Salvi et al., 2007, Bowyer et al., 2006], there is no scientific publication covering the evolution of registration applied to robotics.

Different application fields have different constraints, which in our case motivate a review specialized for robotics. For example, medical imaging has the advantages of controlled environments, standardized sensors, and precise sensor motions. On the other side, it has its own challenges: deformable objects, low descriptor discrepancy, multimodal sensing, and high impact of failure on human life. As for object reconstruction field, advantages can be: controlled to semi-controlled environments, human in the loop for sensor motions, low real-time requirements, and contained volume of objects; with challenges including loop closure and realistic representation of the model. On the other side, robotic applications often has an unbounded object of interest: the scene. This scene can be considered as a rigid body with spuri-

ous uncontrolled dynamic elements. Moreover, the focus is more on the stability of the 3D reconstruction linked to the localization of a mobile agent. In exploration, this mobility takes the form of a suite of sensors moved in an unknown environment by an autonomous system. This can imply a long sequence of measurements obtained through time in a range of different environments. Overlaps between measurements can largely vary and density of the point cloud should not be assumed to be uniform.

This chapter attempts to build a foundation for a survey specialized for the robotic field. Although this survey mainly focus on 3D registration applied to robots, having a broader overview of existing data-association solutions would be relevant at this early stage. We follow the generic scheme presented in Figure 1.5 to classify current publications. Different sources for `reading` and `reference` point clouds are surveyed in Section 2.1. Then, transformation functions mainly used in the Euclidean space are covered in Section 2.2. Next, Section 2.3 presents different preprocessing functions generally used to better approximate the environment or to enhance the discrepancy between salient points. Furthermore, Section 2.4 lists different way to associate a `reading` with a `reference` and some optimization techniques. Section 2.5 describes ways to handle outliers generated during the matching process. In Section 2.6, we finally expose diverse way to minimize alignment error.

Given the number of publications covering this field, we present examples of works for every modules instead of the complete list. Nevertheless, the classification used should be generic enough to allow further publications to be added in the future. In the summary presented in Section 2.7, we list the classification terminology used in our registration overview and highlight some relations between them.

2.1 Reading and Reference Sources

Sources where the `reading` and the `reference` come from are highly related to the application requiring data association. Historically, robotic

applications mostly utilize 2D laser rangefinders[1] for indoor [Yoshitaka et al., 2006] and outdoor [Bosse and Zlot, 2009b, 2008] localization. With the need for 3D localization, systems based on rotating 2D laser rangefinders recently received an increasing attention for both indoor [Armesto et al., 2010] and outdoor [Segal et al., 2009] environments. In computer vision, laser are used to reconstruct historical items in 3D [Pulli, 1999, Druon et al., 2006] or simply 3D models of small objects [Liu, 2010, Jost and Hugli, 2002]. Laser rangefinders are also used in face recognition [Pan et al., 2010]. Geomatic applications use lasers for 3D aerial mapping [Kumari et al., 2009] and, more exotically, depth data can also come from Atomic Force Microscopy (AFM) [Jost and Hügli, 2002].

The selection of a depth sensor depends on many different criteria. For example, when external power is not accessible, power consumption becomes critical. Also, a high maximum range of the sensor may become essential if an environment is large or difficult to access. Payload that can be carried by a mobile platform may be a limiting factor of small systems. FoV can also impact the design of mobile inspection tools since a too narrow FoV will require extra actuation to offer a better coverage of the environment. We propose, in Figure 2.1, a snapshot of the market in 2014 by representing 33 sensors with respect to their maximal range versus their weight. This figure is intended to help in defining a first group of sensors that a platform could carry, while quickly seeing the possible scanning range accessible to the platform. On the y-axis of the graph, we can divide the graph into airborne survey sensors with more than 1 km range, terrestrial survey sensors around 100 m to 1 km, and safety/robotics sensors under 100 m. Below a maximal range of 10 m, we can observe that a trend of having heavier sensors with reduced range. This type of sensors offers a better accuracy at the expense of range. Hand-held 3D scanners with sub-millimeter precision target mainly reverse engineering or culture heritage preservation.

There are also new technologies emerging in this growing field. It is worth noting the TigerCub 3D Flash LiDAR, proposed by Advanced

[1]Geomatic and aerospace field mostly refer to Light Detection And Ranging (LiDAR) instead of laser rangefinder.

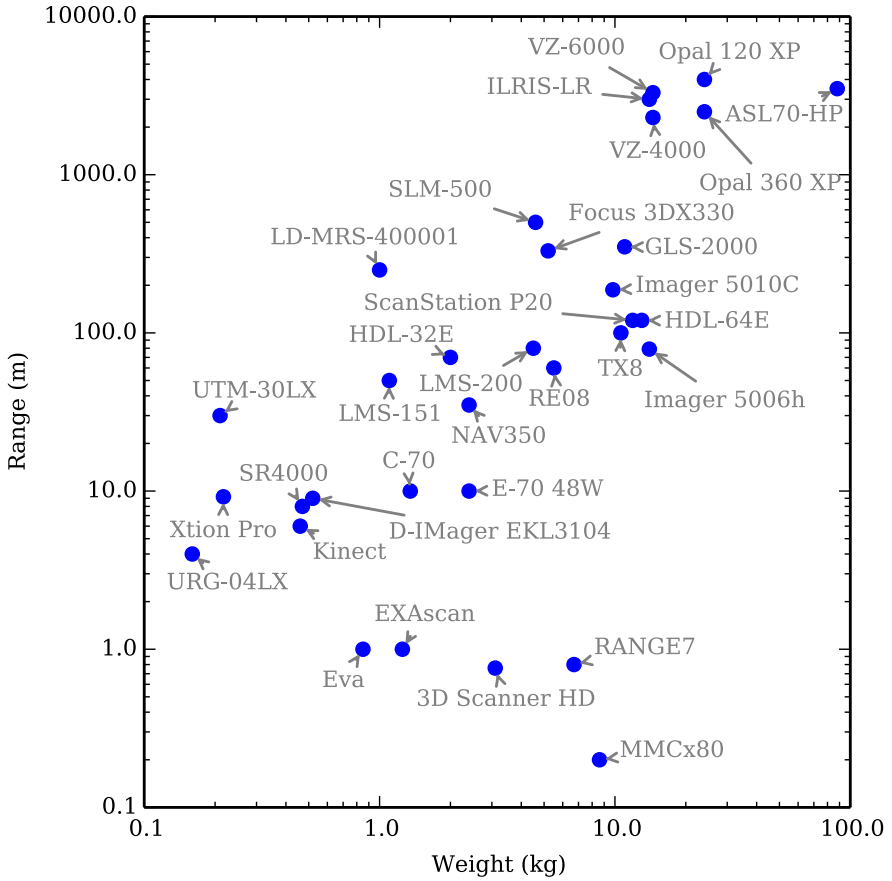

Figure 2.1: Classification of range sensors based on maximum range (m) and weight (kg). Manufacturers: Mesa Imaging (SR4000), Asus (Xtion Pro), Microsoft (Kinect), Ocular Robotics (RE08), Hokuyo (URG-04LX, UTM-30LX), Sick (LMS-151, LMS-200, NAV350, LD-MRS-400001), Velodyne (HDL-64E, HDL-32E), Faro (Focus 3DX 330), Leica (ScanStation P20, ASL70-HP), Riegl (VZ-4000, VZ-6000), Topcon (GLS-2000), Trimble (TX8), Optec (ILRIS-LR), Zoller+Froehlich (Imager 5006h, Imager 5010C), Konica Minolta (RANGE7), Panasonic (D-IMager EKL3104), Nikon (MMCx80), Artec (Eva), NextEngine (3D Scanner HD), Creaform (EXAscan), Renishaw (SLM-500), Fotonic (E-70 48W, C-70), Neptec (Opal 360 XP, Opal 120 XP). Note log scale.

Scientific Concepts, which can produce a depth image with a single laser beam. The system can read from a distance of up to 70 m with a 45° lens. Another company, Lytro, proposes a small and low cost camera producing depth images with a single lens. The system relied on a concept called *light field* to capture simultaneously multiple focus points and reconstruct images with different depth of fields out of the recorded data.

Data association is also used in a variety of applications based on images acquired by black and white or color cameras [Zitová and Flusser, 2003]. In medical applications, other specialized images, like red-free or fluorescein angiogram, are used to help diagnostics over several years of observations [Stewart et al., 2003, Tsai et al., 2010]. Two cameras can be registered together to produce a depth image, which can then be used to track human body motions [Kim, 2010] or create map of the environment [Diebel et al., 2004].

Zitová and Flusser [2003] propose a general classification of data acquisition as follow: different viewpoints (multi-view analysis), different times (multi-temporal analysis), different sensors (multimodal analysis), and also scene-to-model registration. Robotic and object reconstruction applications are more related to the registration of different viewpoints, while medical imagery tend to do more time analysis to evaluate, for example, the status of a growing tumor. Calibration problem falls into the category of different sensor registration and search for a known object in the environment relate to scene-to-model registration.

Example In the simplest case with a laser range sensor, shapes $\mathcal{P}$ are point clouds and can be written in a matrix form with each column a point vector:

$$\mathcal{P} = \boldsymbol{P} = \begin{bmatrix} \boldsymbol{p}_1 & \boldsymbol{p}_2 & \cdots & \boldsymbol{p}_N \end{bmatrix}$$

where $\boldsymbol{p}_i$ is a point and N the number of points in the point cloud.

2.2 Transformation Functions

Transformation functions allow for the expression of an entity defined in a reference frame $\mathbb{A}$ in another reference frame $\mathbb{B}$. A first type of transformation functions without parameter can directly map spaces to another by following some common convention (e.g., mapping function from Cartesian to spherical, Cartesian to cylindrical, homogeneous to Cartesian, etc.). A second type of transformation functions uses a set of parameters to morph an entity to express it in another frame.

2.2.1 Parametrized Transformation

Basic parametrized transformation functions are: translation, uniform scaling, rotation, nonuniform scaling, shear and projection. Typically, a combination of those basic functions is used. A short nomenclature can be introduced as follow:

Rigid transformation is a combination of translation and rotation. It is also known as a Euclidean transformation.

Similarity transform is a combination of *rigid transformation* and uniform scaling.

Affine transform is a combination of *rigid transformation*, nonuniform scaling and shear.

Orthogonal projection is a group of transformation based on vector and planar projection.

The projective transformation is not listed above because it is linked to a larger field, the projective geometric, which has ramifications beyond basic parametrized transformation functions described above.

Most of the registration algorithms used in robotics are based on *rigid transformation* parametrization. Nonlinear transformations are often expressed as a set of *rigid transformation* with some limited spatial influence. This type of registration can be referred as *flexible* registration and were considered first in medical imaging for organ reconstructions [Maurer et al., 1996, Feldmar and Ayache, 1996]. It is

interesting to note that those flexible registrations resemble the graph relaxation used for error back propagation in SLAM algorithm [Grisetti et al., 2007].

Example In case of a rigid transformation, if points are represented using homogeneous coordinates, a transformation $\mathcal{T}$ can be represented as a matrix $\boldsymbol{T}$ such that:

$$\mathcal{T}(\mathcal{P}) = \boldsymbol{T}\boldsymbol{P} = \begin{bmatrix} \boldsymbol{T}\boldsymbol{p}_1 & \boldsymbol{T}\boldsymbol{p}_2 & \cdots & \boldsymbol{T}\boldsymbol{p}_N \end{bmatrix}.$$

$\boldsymbol{T}$ is then of the form:

$$\boldsymbol{T} = \begin{bmatrix} \boldsymbol{R} & \boldsymbol{t} \\ \boldsymbol{0}^\top & 1 \end{bmatrix}$$

where $\boldsymbol{R}$ is a rotation matrix and $\boldsymbol{t}$ a translation vector.

The generic formula computing the final transform Equation 1.3 becomes a simple matrix product:

$$\begin{aligned} {}^{\mathbb{B}}_{\mathbb{A}}\hat{\mathcal{T}} &= \left(\bigcirc_i {}^{i-1}{}^{i}\mathcal{T} \right) \circ \mathcal{T}_{init} \\ \Leftrightarrow {}^{\mathbb{B}}_{\mathbb{A}}\hat{\boldsymbol{T}} &= \left(\prod_i {}^{i-1}{}^{i}\boldsymbol{T} \right) \boldsymbol{T}_{init}. \end{aligned}$$

2.2.2 Initial Transformation Sources

The `initial transformation` is a sensitive part of registration algorithms when the data association is realized mainly based on geometric features. Large error in the initial guess may trigger multiple local minima generating the wrong final transformation. Many strategies are used to overcome this limitation. Here, we defined three types of `initial transformation` sources:

External – The transformation parameters come from an external source. While some papers present general algorithms assuming that a reasonable transformation will be given [Armesto et al., 2010, Druon et al., 2006, Schutz et al., 1998, Jost and Hügli, 2002], others include humans in the loop [Godin et al., 1994, Pulli, 1999]. More integrated solutions rely on external sensor (wheel

encoders, Inertial Measurement Unit (IMU), Global Positioning System (GPS), etc.) or a fusion of them as first guess [Diebel et al., 2004, Yoshitaka et al., 2006]. Multiple initial guesses from the same external motion model and based on particle filters are also used in robotics [Grisetti et al., 2005] and in human body motion tracking [Kim, 2010].

Parameter cascade – This type of system rely on the same registrations algorithm, while varying parameters through the process to achieve, for example, faster or more accurate results. After some exit criteria, the registration parameters are changed and the output transformation is fed to the subsequent system. One example of cascade system is the coarse-to-fine strategy, where the `reading` or the `reference` are downsampled using different levels of compression. The same registration technique is used to minimize the error sequentially from the most compressed layer to the least compressed [Zhang, 1994, Jost and Hugli, 2002, Magnusson et al., 2009, Bosse and Zlot, 2009b]. An alternative approach is the fine-to-coarse strategy, where the registration starts from a small bounding box. The expansion of the bounding box follows the uncertainty of the minimization until the bounding box covers the overlapping section of the `reading` with the `reference` [Stewart et al., 2003, Tsai et al., 2010].

Registration cascade – The initial transformation comes from a different registration algorithm. The first registration is usually heavily based on descriptor matching because it is independent from the `initial transformation` (always identity). Then, other types of registration, either using descriptors or features, are used to minimize the alignment error. The motivation for the selection of those registration techniques is to reduce local minima possibilities with coarse alignment methods and continue with more precise but computationally more expensive methods. From the papers reviewed, the system can have two [Bosse and Zlot, 2008, Censi, 2008, Godin et al., 1994, Stewart et al., 2003, Tsai et al., 2010] or three layers [Pan et al., 2010, Bosse and Zlot,

2009b], but nothing seems to limit the number of layers implemented on a given system.

Example In a robotic system, it is typical to have an estimation of the state of the robot given odometry of inertial sensors, for example using variants of Kalman filtering [Hitz et al., 2014]. The result of this estimation can then be taken as the initial transformation. But the result of the registration can also be integrated in a filtering approach, for instance as an observation of an extended Kalman filter [Kubelka et al., 2014].

2.3 Data Filters

The different types of `data filters` try to augment the distinctiveness of the inputs usually by reducing the number of features and by augmenting the dimension of either the features or the descriptors. For example, a black and white image has a uniform distribution of features (a grid) and one dimension descriptor (the intensity) associated to each feature. After some `data filters` are applied, only few points in the image will be kept as features[2] and the descriptors will be increased with information from neighboring pixels, for example to 64 dimensions when using Scale Invariant Feature Transform (SIFT) descriptors [Lowe, 2004]. In the case of a point cloud, it might be necessary to extract surface normal vectors (*feature enhancement*), while uniformizing the density of points (*features reduction*). This can also be viewed as *lossy data compression*.

2.3.1 Feature Enhancement

When only geometric information is available, there are still ways to extract some level of distinctness by using differential geometry. We shortly introduce key concepts related to the use of geometric information. In this work, the notion of a shape $\mathcal{S}$ is used as a representation of a generic object in the Euclidean space with a certain set of properties. For example, those properties can be photometric, thermic, and

[2] In computer vision, remaining features are often called *key points*.

semantic. Simple shapes, such as points, lines, quadrics, can be easily parametrized but most of the shapes encountered in the a real environment are too complex to be completely synthesized with parameters. To allow a certain representation of the world, a complex shape $\mathcal{S}$ can be approximated by a set of other shapes only if they can be expressed in the same frame of reference $\mathbb{F}$.

Sensors measuring depth produce such approximations by discretizing the environment in a set of points. From smooth areas defined by those points, we present five types of primitives that can be extracted based on Differential Geometry: point, line, plane, curve and quadric. The first derivative group rely on normal vectors n (i.e. orthogonal to a line or plane) and tangent vectors t (i.e. parallel to a line or plane) to express the area. Since they are defined with respect to a point p, normal and tangent vectors can be represented with a minimal set of 2 angles in polar coordinates. Normal and tangent vectors can be seen as a dual representation. The choice of using either one or the other is defined by the minimum information required to express a primitive. In the case of a line in 3D, the normal vectors needs to define a plane perpendicular to that line. Therefore, only using the tangent vector is more convenient. The same reasoning holds for using a normal vector for the surface. The notion of direction also needs to be defined depending of the primitive. It is reasonable to use unsigned direction in the case of tangents and signed direction in the case of normals where positive sign defined the outer surface of the shape. The motivation behind this choice is that we want to keep track of which side of a surface we are measuring while moving. In the 2D case, it is equivalent to represent a line by a tangent or a normal in term of the number of parameters. However, it is usually assumed that the perceived 2D plane cuts perpendicular surfaces, so it makes more sense to use normal vector to also track the outer side of the line. Figure 2.2 illustrates this choice of representation for the 2D case. Viewpoints $\boldsymbol{v_n}$ (i.e., where the sensor was when a point $\boldsymbol{p}$ was measured on a surface $\mathcal{S}$) are used to determine the direction of the normal vectors $\boldsymbol{n}$, whereas no extra information is needed for a tangent vector $\boldsymbol{t}$ laying on a line that can be observed from both sides.

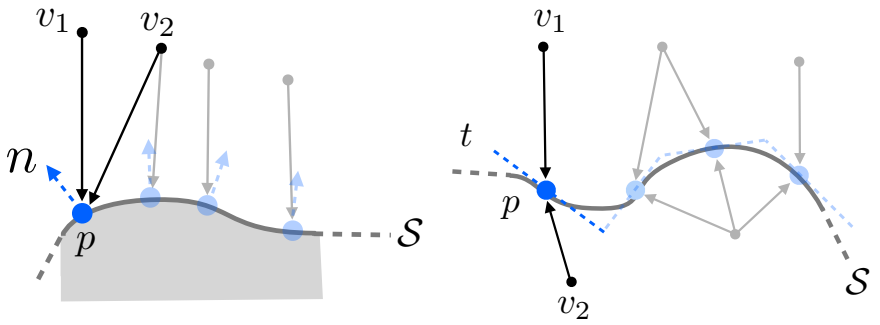

Figure 2.2: Difference between using normal vectors and tangent vectors to represent a 2D shape $\mathcal{S}$. *Left*: The 2D line is known to be measured from a volume, which can only view from one side as illustrated with the black arrows $\boldsymbol{v_1}$ and $\boldsymbol{v_2}$. The direction of the surface make sense to be encoded in its representation leading to the choice of normal vectors $\boldsymbol{n}$ (dashed blue arrow). *Right*: The same line can be observe from both direction leading to the selection of an undirected tangent vector $\boldsymbol{t}$ (dashed blue line).

As for the second derivative group, a curve is parametrized by a curvature κ (a scalar) representing an osculating circle parallel to the tangent of a line. In the case of a quadric (i.e. a curved surface), it is parametrized by principal direction vectors $\boldsymbol{t_{min}}$, $\boldsymbol{t_{max}}$ and the principal curvature scalars $\kappa_{min}, \kappa_{max}$. Those principal directions rely on a point $\boldsymbol{p}$ and on the normal vector $\boldsymbol{n}$, so they could be expressed only with one angle. In other words, the principal directions are tangent vectors to the surface complementing the normal vector. In the case of curves, the curvature is always positive as opposed to quadrics for which a positive κ means that the surface bend in the same direction of the normal vector and vice-versa. For simplicity in further computation, most of the publications use normalized 3D vectors to express normals and tangents leading to a larger set of parameters, as shown in Table 2.1.

Those parameters (point, tangent, normal, principal direction, curvature, and principal curvatures) bring us to a group of parametrized primitives (point, line, plane, curve and quadric), which can be helpful to approximate other complex 3D shapes. Those geometric primitives

Table 2.1: Comparison of different parametrizations used to represent geometric primitives.

Minimum parametrization

Name	*Parameters*	*Constraints*	*Reference*
Point	$\boldsymbol{p} = \{x, y, z\}$	$\boldsymbol{p} \in \Re^3$	$\mathbb{F}$
Tangent	$\boldsymbol{t} = \{\theta, \phi\}$	$\theta \in [-\pi, \pi)$ $\phi \in [-\frac{\pi}{2}, \frac{\pi}{2}]$	$\mathbb{F}$
Normal	$\boldsymbol{n} = \{\theta, \phi\}$	$\theta \in [-\pi, \pi)$ $\phi \in [-\frac{\pi}{2}, \frac{\pi}{2}]$	$\mathbb{F}$
Principal Directions	ψ	$\psi \in [-\pi, \pi)$	$\{\boldsymbol{n}, \mathbb{F}\}$
Curvature	κ	$\kappa \in \Re_+$	$\mathbb{F}$
Principal Curvatures	$\boldsymbol{\kappa} = \{\kappa_{\min}, \kappa_{\max}\}$	$\boldsymbol{\kappa} \in \Re^2$	$\{\boldsymbol{n}, \mathbb{F}\}$

Typical parametrization

Point	$\boldsymbol{p} = \{x, y, z\}$	$\boldsymbol{p} \in \Re^3$	$\mathbb{F}$
Tangent	$\boldsymbol{t} = \{t_x, t_y, t_z\}$	$\lvert\boldsymbol{t}\rvert = 1$	$\mathbb{F}$
Normal	$\boldsymbol{n} = \{n_x, n_y, n_z\}$	$\lvert\boldsymbol{n}\rvert = 1$	$\mathbb{F}$
Principal Directions	$\boldsymbol{\gamma} = \{t_{\min}, t_{\max}\}$	$n \perp t_{min} \perp t_{max}$	$\mathbb{F}$
Curvature	κ	$\kappa \in \Re_+$	$\mathbb{F}$
Principal Curvatures	$\boldsymbol{\kappa} = \{\kappa_{\min}, \kappa_{\max}\}$	$\boldsymbol{\kappa} \in \Re^2$	$\{\boldsymbol{n}, \mathbb{F}\}$

are listed in Table 2.2 with their characteristics. A graphical representation of those primitive is also showed in Figure 2.3 in the case of a shape considered as a 1D manifold and in Figure 2.4 in the case of a 2D manifold. In their current configuration, lines, planes, curves and quadrics are unbounded, which means that they can reach infinity on their unconstrained direction. One can constrain a primitive by using a primitive with a lower dimensionality [Besl, 1988]. Thus, a plane can be bounded by a set of lines, a line can be bounded by a set of points, etc.

At a higher level of organization, a group of geometric primitives can be processed without any proximity assumption (*unstructured*) or with some smoothness constrains (*structured*). Examples of representation for a 1D manifold is a spline, while for a 2D manifold a mesh or Non-uniform rational B-spline (NURBS) can be used. When it comes to a noisy group of points, tensor voting [Medioni et al., 2000] can be used

Table 2.2: Characteristics of primitives used for shape approximations. The number in parenthesis of the column *Nb Param.* corresponds to the minimum number of parameters that can be used to express the same primitive.

Primitives	*Parameters*	*Derivative*	*Manifold*	*Bound*	*Nb Param.*
Point	$\boldsymbol{p}$	0	0	-	3(3)
Line	$\mathcal{L} = \{\boldsymbol{p}, \boldsymbol{t}\}$	1	1	point	6(5)
Plane	$\Phi = \{\boldsymbol{p}, \boldsymbol{n}\}$	1	2	line, curve	6(5)
Curve	$\mathcal{C} = \{\boldsymbol{p}, \boldsymbol{t}, \boldsymbol{n}, \kappa\}$	2	1	point	10(7)
Quadric	$\mathcal{Q} = \{\boldsymbol{p}, \boldsymbol{n}, \boldsymbol{\gamma}, \boldsymbol{\kappa}\}$	2	2	line, curve	14(8)

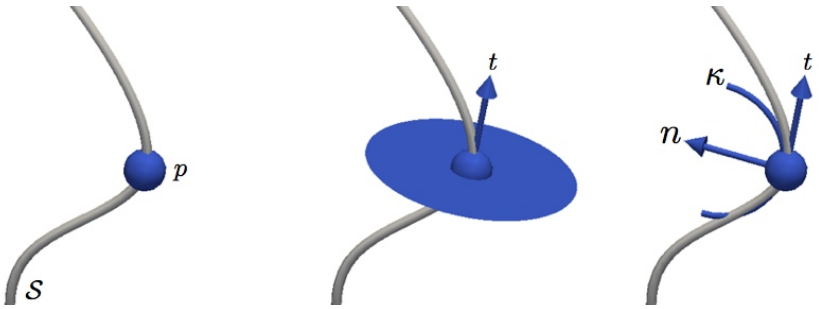

Figure 2.3: Representations (in dark blue) of a complex shape $\mathcal{S}$ (in light gray) approximated as a 1D manifold. *Left*: no derivative (point). *Middle*: first derivative (line). *Right*: second derivative (curve).

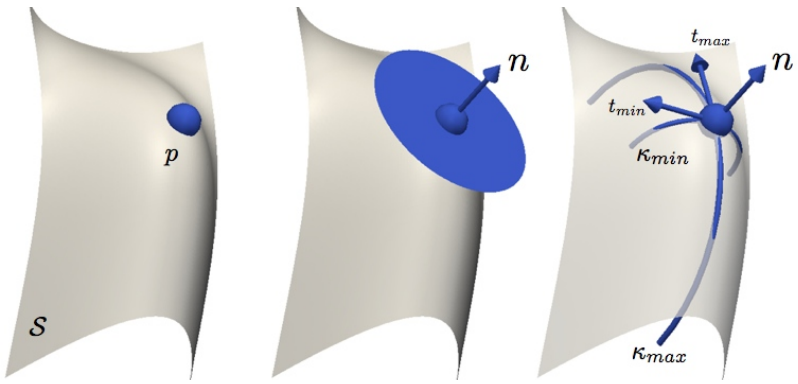

Figure 2.4: Representations (in dark blue) of a complex shape $\mathcal{S}$ (in light gray) approximated as a 2D manifold. *Left*: no derivative (point). *Middle*: first derivative (plane). *Right*: second derivative (quadric).

to interpolate shape on a dense 3D grid. The voting results can then be later process to extract 1D and 2D manifolds out of the dense volume.

Higher derivatives could also approximate a shape at one point with more precision. Unfortunately, high derivatives are very sensitive to noise when applied outside of a theoretical context. In this review, we limit ourselves to the second derivative given that a non-negligible noise level is expected from the sensor measurements and from the motion of the sensor. As shown in [Pomerleau et al., 2012a], even the first derivative needs a large supporting surface to overcome the sensor noise of typical laser rangefinders. For example, extracting the surface normal from a flat area of 10-cm radius already leads to an expected error of 1.6° (0.03 rad) for the rangefinder Sick LMS-151. This is one of the characteristics of robotic applications when it comes to selecting a surface parametrization. The ratio of noise to signal is often higher in robotics than in object modeling, thus rendering many complex modeling algorithms ineffective. This could explain why most registration algorithms applied to robotics tend to select a shape representation very close to the raw measurements (i.e., points) instead of relying on faulty surface reconstruction.

Sensitivity to transformation functions

The shape representations are affected differently by transformation functions. At a more generic level, transformation functions affect geometric quantities. Examples of quantities are: coordinate, orientation, length, angle, and length ratio. Those geometric quantities with examples of associated primitives are listed in Table 2.3. As examples of lengths, we used κ, which is the inverse of a radius, and the eigenvalues λ, which define a scale over a vector. Having geometric parameters invariant to as many transformations as possible helps the matching function during registration because the association will be less sensitive to large initial alignment error. Table 2.4 relates the different geometric quantities to the basic transformation functions affecting them.

Most of the time, point cloud features come without external descriptors (i.e. as opposed to an image containing photometric information), so the proximity of other features is used to extend the shape

Table 2.3: Examples of geometric quantities susceptible to be affected by a transformation function.

Quantity	*Single Entity*	*Relationship in a Set*
Coordinate	$\boldsymbol{p}$	-
Orientation	$\boldsymbol{n}, \boldsymbol{t_{\min}}, \boldsymbol{t_{\max}}$	-
Length	κ, λ	$\|\|p_a - p_b\|\|$
Angle	-	$\arccos \boldsymbol{n_a} \cdot \boldsymbol{n_b}$
Length Ratio	-	κ_a / κ_b

Table 2.4: Influence of a transformation function on quantities defining a geometric primitive. Cells marked with a "X" means that the transformation affect the values of the entity.

Function	*Coordinate*	*Length*	*Orientation*	*Angle*	*Length Ratio*
Translation	X	-	-	-	-
Uniforme Scaling	X	X	-	-	-
Rotation	X	-	X	-	-
Nonuniform Scaling	X	X	X	X	-
Shear	X	X	X	X	X
Orthogonal Projection	X	X	X	X	X
Perspective Projection	X	X	X	X	X

approximation to support further derivatives. Surface orientations (or line orientations in 2D) are mainly used in literature [Pulli, 1999, Censi, 2008, Bosse and Zlot, 2009b, Jost and Hugli, 2002, Schutz et al., 1998, Jost and Hügli, 2002, Segal et al., 2009]. Line orientations are also used in image registration where the environment presents very few salient points when considering only intensity variation [Stewart et al., 2003]. Work based on surface normal vector distributions of surrounding points are also used by Magnusson et al. [2009] and Fairfield and Wettergreen [2009].

2.3.2 Descriptor Enhancement

A comparison of descriptors extracted from 2D point clouds can be found in [Bosse and Zlot, 2009a]. It is proposed that moment grid is better than 2D shape context, Gestalt, Hough transform peaks, ori-

entation and projection histograms, and normal orientation histogram grid. Extension to the 2D shape context can be found in [Tsai et al., 2010]. Another study for 3D point clouds also concludes that moment grid is better than 3D shape context, spin image, shell image and local covariance [Bosse and Zlot, 2009b].

Usually, ICP is done using only geometric features, but some works also present results using the intensity remission from an Hokuyo [Yoshitaka et al., 2006] and from specialized system using three different wavelengths [Godin et al., 1994]. Laser range finders are also combined with cameras to add color information to measured points [Schutz et al., 1998, Druon et al., 2006]. When other sensors are used to provide descriptors, calibration is required. Interestingly, calibration also relies on registration solutions. Terrestrial survey scanners often have a calibrated camera associating color to 3D points, similar to RGB-D cameras. With the larger availability of photometric information, descriptors developed by the computer vision community can be used. In [Tuytelaars and Mikolajczyk, 2008], characteristics of descriptive features are listed as *rotation*, *scale*, and *affine invariance*. Evaluation criteria are listed as *repeatability*, *distinctiveness*, *locality*, *quantity*, *accuracy*, and *efficiency*. In image registration, the list of most common tools for extracting descriptors are: Harris, Hessian, SUSAN, Harris-Laplace, Hessian-Laplace, DoG (SIFT), SURF, Harris-Affine, Hessian-Affine, Salient Regions, Edge-based, MSER, Intensity-based and Superpixel [Tuytelaars and Mikolajczyk, 2008]. It is interesting to note that descriptors based on photometric information rely on passive illumination to ensure invariance. This assumption of illumination sources remaining static, which is mostly true for indoor lights, needs to be treated carefully for outdoor illumination, where the sun moves and clouds can shade light. Laser's intensity measurements are even more sensitive to transformation functions because the illumination source follows the sensor position.

2.3.3 Feature Reduction

In applications using point clouds, features are sparse but not uniformly distributed. Nevertheless, the fact that sensors can provide a huge num-

ber of readings on a short period of time creates a bottleneck in term of computation power for the association as explained later. Several techniques are used to reduce the number of features: random sampling [Jost and Hugli, 2002, Pan et al., 2010], uniform grid [Magnusson et al., 2009, Bosse and Zlot, 2009b], grid projection [Pan et al., 2010], octree [Fairfield and Wettergreen, 2009, Wurm et al., 2010], and bounding box [Stewart et al., 2003, Tsai et al., 2010]. All these techniques reduce the number of features without considering their distinctiveness.

Having that criteria in mind, Bosse and Zlot [2009a] present results showing that keeping a representative point per curvature cluster is better than segment centroids and mean-shift for 2D point clouds. Also, Gelfand et al. [2003] propose a sampling method selecting points leading to a better geometric stability of the minimization. Relying on non-geometric information, Druon et al. [2006] use seven clusters based on hue values and select only one cluster carrying the most information for registration. The same type of solution was previously presented before by Godin et al. [1994], who clustered point based on laser intensity values.

It is also possible to find more application-specific methods in the literature. For example, the tip of the nose, inner corners of the eyes, and nose corners are directly extracted for face detection [Pan et al., 2010]. In medical imagery, blood vessel crossings are also used to reduce features. Moreover, the main orientation of the blood vessel crossings and its number of branching is used to construct descriptors [Stewart et al., 2003]. The complete point cloud can also be reduced to its first and second statistical moments [Liu, 2010] or with orientation and projection histograms [Bosse and Zlot, 2008].

2.3.4 Sensor Noise

Sensor noise is also taken into account at this stage of the process. Models representing noises are intended to evaluate the uncertainty of a measured point based on the limitations of the sensor used. They may try to identify if a point is a measurement artifact or how accurately the position is measured. To cope with stereo reconstruction noise, Diebel et al. [2004] remove points with distance and surface angle to neighbors

larger than two times the median of all distances and surface angles within the point cloud. When using laser's remission intensity, which is not invariant to distance and angle, Yoshitaka et al. [2006] propose to keep points only close to the laser to reduce the impact of distance on the intensity value. For color images, points with low saturation value tend to be gray and are removed before applying clustering technique based on hue [Druon et al., 2006]. Points on boundaries of the sensor reading can also be removed to avoid misleading interpretation of neighbor points [Armesto et al., 2010]. When an error model is available, it is also possible to add noise information based on measurement distance, incidence angle, reflectivity, etc. Examples of noise models on distance reading are investigated for Sick LMS-151, Hokuyo URG and UTM in [Pomerleau et al., 2012a].

Example 1 The simplest of the filters is a random subsampling in order to decimate the point cloud:

$$\mathcal{P}' = \text{datafilter}(\mathcal{P}) = \{\boldsymbol{p} \in \mathcal{P} : \eta(\boldsymbol{p}) < \theta\}$$

where $\eta \in [0, 1)$ is a uniform-distributed random value and $\theta \in [0, 1]$ a fixed threshold, corresponding to the fraction of points to keep.

Example 2 A second example is the computation of normal vectors in a point cloud:

$$\mathcal{P}' = \text{datafilter}(\mathcal{P}) = \left\{ \begin{bmatrix} \boldsymbol{p} \\ \boldsymbol{n} \end{bmatrix} : \forall \boldsymbol{p} \in \mathcal{P}, \boldsymbol{n} = \text{normal}(\boldsymbol{p}) \right\}$$

where $\text{normal}(\boldsymbol{p})$ is the normal vector inferred around point $\boldsymbol{p}$.

2.4 Association Solver

As explained in Section 1.3, points from `reading` and `reference` have to be associated into pairs that will be the basis of the computation of the error to be minimized. The association solver (called also *match function*) then yields a set of pairs of points:

$$\mathcal{M} = \text{match}(\mathcal{P}, \mathcal{Q}) = \{(\boldsymbol{p}, \boldsymbol{q}) : \boldsymbol{p} \in \mathcal{P}, \boldsymbol{q} \in \mathcal{Q}\}$$

2.4.1 Types of association

The association of the `reading` with the `reference` can be divided into three types: 1) feature matching, 2) descriptor matching and 3) mixed. Feature matching is mainly done using Euclidean distance between a point in the `reference` and a point in the `reading` [Pulli, 1999, Druon et al., 2006, Censi, 2008, Segal et al., 2009, Pan et al., 2010, Kim, 2010], a point and a plane [Champleboux et al., 1992] and quadrics [Feldmar and Ayache, 1994]. Custom distances based on point positions and angles is also used [Armesto et al., 2010]. Descriptors are matched based on their Euclidean distances [Lowe, 2004, Bosse and Zlot, 2009b]. The concept of measuring distance between two entities can take multiple forms (ex., correlation matching, earth mover distance, L^1, L^{inf}, etc.) In the current literature reviewed, other distances used in matching functions were Mahalanobis [Stewart et al., 2003] and χ^2-test statistic [Tsai et al., 2010]. Using only features or descriptors in the association have their advantages and inconveniences. In laser rangefinder based matching, feature positions are quite accurate compared to descriptor uniqueness but the `initial transformation` needs to be within a maximum range to avoid local minima. When using descriptors, the matching becomes independent of the initial position, but may fail for repetitive elements (e.g., checkerboard, building facades with repetitive windows).

A logical extension is to mix both types of matching. One way is to express descriptors in the feature space using a conversion factor. This was used with surface orientations [Jost and Hügli, 2002, Bosse and Zlot, 2009b], surface orientations and color [Schutz et al., 1998], color [Johnson and Kang, 1997] and laser intensity [Godin et al., 1994, Yoshitaka et al., 2006]. The other way around is also possible (i.e., expressing feature positions in descriptor space). In [Mortensen et al., 2005], a descriptor, called Global Context, is created using surrounding feature positions. The distance is computed as the sum of the Euclidean distance of a SIFT descriptor with the χ^2 distance of the Global Context descriptor. A ratio between feature and descriptor distances is required to join the different spaces together, but it is often implicitly defined as unity [Tsai et al., 2010].

Other parameters to consider during the matching stage are the match direction and the number of matches used. The match direction refers to either match from the `reference` to the `reading` or from the `reading` to the `reference`. Most of the time, one of those two possibilities is used without further consideration but some techniques use both directions [Pulli, 1999, Godin et al., 1994]. As for the number of matches, most of the applications consider only the closest point but some others process a certain percentage of the lowest distance [Stewart et al., 2003] or the complete matching matrix. The complete matching matrix is often used in loop closing detection [Bosse and Zlot, 2009a] but it can be used for local matching, as with the SoftAssign method [Gold et al., 1998, Liu, 2010].

2.4.2 Implementation optimization

The association solver deals with the Nearest Neighbor (NN) problem, which typically has a complexity of $O(nm)$ where n and m are respectively the number of elements in the `reading` and in the `reference`. This stage is generally the most time-consuming and a lot of papers present variations of NN search to reduce its complexity. A dynamic space partitioning can by applied using kD-trees to reduce the search complexity to $O(n \log m)$ after a $O(m \log m)$ building phase. Approximate kD-trees decreases the computational time by employing a distance threshold to limit the search at the risk of returning sub-optimal neighbors [Arya and Mount, 1993]. This increases the overall speed of the search, while the redundancy between points prevents large accuracy degradation [Nüchter et al., 2005]. In an iterative context, Nüchter et al. [2007] propose to use cached kD-tree for faster search. NN from the previous iteration are feed to the current search as starting points to accelerate the computation. Additionally, Zlot and Bosse [2009] compare kD-tree, locality-sensitive hashing and spill-trees and concluded that the kD-tree is better in terms of accuracy, query time, build time, and memory usage. They also observed that approximations can reduce the query time by two orders of magnitude while maintaining sufficient accuracy.

KD-trees provide very little acceleration for high dimension vectors like the ones used for image based descriptors. Static space partitioning, usually based on grid or hashing, offer less adaptation but can compensate with their computation speed. Approximate search based on Best-Bin-First can be used instead for optimization [Lowe, 2004]. Other techniques use a dual proximity hypothesis (e.g., laser points ordered by a temporal sequence [Censi, 2008], n-search [Jost and Hugli, 2002]), projection on one grid [Pan et al., 2010] or on multiple grid (called multi Z-buffer) [Benjemaa and Schmitt, 1997] to also reduce the search time. Although very useful, kD-tree also limits the distance metric used to be Euclidean. This forces some approximations when using Euclidean distances between components of unit surface normal vectors instead of the angle between them [Feldmar and Ayache, 1996, Eggert et al., 1998, Gelfand et al., 2003]. Figure 2.5 illustrates the error between the angle distance of surface normal vectors against the approximation using Euclidean distance. One can notice that, as long as the distance is low, the approximation can hold but, with large error the approximation is much less accurate. Moreover, the construction of the tree requires some time, and often only the `reference` is used as seeding points. Limiting the number of kD-tree constructions by the use of keyframes or metascans can help to decrease the registration time for a sequence of scans, while limiting the drift of the final path [Wulf et al., 2008].

Cascade systems, presented in Section 2.2.2, is also another research direction to accelerate the search. Jost and Hugli [2003] compute ICP several times while varying the resolution from coarse to fine. At a coarse resolution (i.e., with a limited number of points) ICP converges faster but with less accuracy than at a fine resolution. However, by initializing a finer-resolution ICP with the result of the coarser one, the convergence of the fine-resolution ICP is much faster than with a single-shot ICP, as the initial alignment is mostly correct. These authors also used a pre-computed list of NN to approximate the matching step. With both of these techniques, they showed a significant increase of the speed of ICP while maintaining adequate robustness. For the same absolute performance as standard ICP, Li et al. [2010] use fewer

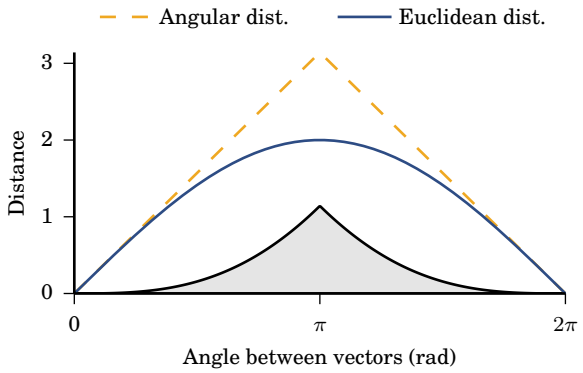

Figure 2.5: Impact of using Euclidean distance (solid blue line) instead of angle (dashed yellow line) between surface normal vectors. The shaded gray area represents the error.

iterations at high resolution, which decreases the total time by a factor of 1.5 in 2D and 2.5 in 3D. The multi-resolution approach can also increase the search speed for the closest point by using a hierarchical-model point selection with a stereo camera [Kim, 2010]. By subsampling the space and with the help of the sensor structure, this solution can achieve a speed gain of factor 3 with respect to standard ICP when using a kD-tree search. In this case, the use of the structure of the depth image increases the matching speed. In the same direction, the specificity of a 2D laser scanner acquisition structure can help optimize the search [Censi, 2008]. However, these optimizations are oriented toward specific sensors, which makes them hard to generalize, and are not suitable for a multi-sensor setup.

Example When only one match is kept from the `reading` $\mathcal{P}$ to the `reference` $\mathcal{Q}$ based on the Euclidean distance, the association is defined as:

$$\mathcal{M} = \text{match}(\mathcal{P}, \mathcal{Q}) = \left\{ (\boldsymbol{p}_n, \boldsymbol{q}_m) : \forall \boldsymbol{p}_n \in \boldsymbol{P}, \boldsymbol{q}_m = \arg\min_j \left(\text{d}(\boldsymbol{p}_n, \boldsymbol{q}_j) \right) \right\}$$

with $\text{d}(\boldsymbol{p}, \boldsymbol{q}) = \|\boldsymbol{p} - \boldsymbol{q}\|_2$.

2.5 Outlier Filters

While `data filters` are used to reduce the impact of sensor noise, `outlier filters` are used to reduce the impact of erroneous matches mainly caused by partial observations, dynamic elements and poor environment information extraction. Match quality can be evaluated based on feature pairs, descriptor pairs or even both together, independent of the match distance used (i.e., based on features or on descriptors).

2.5.1 Rejection

When using descriptors, it is possible to apply an `outlier filter`. Regarding SIFT descriptors, the original method proposed to reject all matches where the distance ratio with the second best match is higher than 80 % [Lowe, 2004]. In [Stewart et al., 2003], all matches under 95 % confidence based on χ^2 uncertainty bound are rejected.

In the case where outliers are filtered based on features, rejection techniques mostly use a threshold based on Euclidean distance, the main difference is how thresholds are chosen. A naïve approach is to use a maximal distance between points [Segal et al., 2009]. This technique is sometimes hidden by using a fixed radius directly in the matching function. Likewise, surface orientation differences between paired points can be limited to a fixed value [Pulli, 1999, Zhang, 1994] or an adaptive one based on the median [Diebel et al., 2004]. Adaptive methods can be based on the mean distance between points and their Standard Deviation (STD) [Druon et al., 2006, Zhang, 1994], STD only [Masuda et al., 1996], the quartile position (a.k.a trimmed) [Chetverikov et al., 2002, Censi, 2008, Armesto et al., 2010] and the median [Diebel et al., 2004] of the distances of all paired points. In an iterative system for data association, Pulli [1999] proposed to manually reduce the threshold at each iteration based on the convergence of the system. An automatic extension to this approach is presented in [Pomerleau et al., 2010]. A different type of `outlier filter` evaluates whether there are multiple matches from the `reading` to the `reference` and keeps only the match with the smallest distance [Zinsser et al., 2003].

2.5.2 Weighting

All the methods above utilize *hard* assignment to identify outliers. This means that, beyond a certain threshold, the feature pair (i.e., tuple) is discarded prior to being considered by the minimization procedure. Assignments can also be made *soft* by using a weighting function promoting inliers during the minimization. Those weighting functions can consist of the ratio of mean distance over each paired distances [Pan et al., 2010] or function such as Gaussian [Godin et al., 1994] or Cauchy (a.k.a. Lorentzian) [Bosse and Zlot, 2009b]. Mix between *soft* and *hard* assignment are also used like one minus the ratio of the tuple distance divided by a maximum distance (with a saturation to zero when the tuple distance is over the maximum distance) [Diebel et al., 2004] or directly using the bisquare (a.k.a Tukey or Beaton-Tukey) function [Masuda, 2001, Stewart et al., 2003]. All those techniques use only the feature information to weight outliers. One example of mixing features and descriptors proximity can be founded in the work of Godin et al. [1994], where the total weight is computed by multiplying the feature distance by a reflectance similarity function.

2.5.3 Robust statistics

Dealing with outliers during a minimization process falls into the field of robust statistics. A suite of tools is proposed to robustly estimate the position (i.e., robust variant of the mean) and the scale (i.e., robust variant of the STD). For the scale estimation, some possibilities were found: Huber estimate, Median Absolute Deviation (MAD), interquartile range, Tukey estimator, trimmed estimator and Winsorised estimator.

The utilization of weights for a minimization process is based on a class of functions called M-estimators. Here is a list of M-estimators found in the robust regression literature: Least-squares (a.k.a L_2) (not robust but is commonly used), Least-absolute (a.k.a L_1), $L_1 - L_2$, Least-power (a.k.a L_p), Fair, Huber, Geman-McClure, Logistic, Median, Talworth, Welsch, Cauchy (a.k.a Lorentzian). A category of M-estimator, called redescending M-estimators, have a saturation point

to reject gross outliers. This category is equivalent to a mix of soft and hard weights. Those functions are called: Hampel, bisquare and Andrews. Note that only the Cauchy [Bosse and Zlot, 2009b] and the bisquare [Masuda, 2001, Stewart et al., 2003] functions were found in data-association papers.

Example Outlier filters can be defined as assigning a weight to each pair: $\mathcal{W} = \text{outlier}(\mathcal{M}) = \{w(\boldsymbol{p}, \boldsymbol{q}) : \forall (\boldsymbol{p}, \boldsymbol{q}) \in \mathcal{M}\}$. Outlier rejection based on a fixed threshold τ on the distance between the points can then be written:

$$w(\boldsymbol{p}, \boldsymbol{q}) = \begin{cases} 0 & \text{if } \mathrm{d}(\boldsymbol{p} - \boldsymbol{q}) > \tau, \\ 1 & \text{otherwise.} \end{cases}$$

In the case of a hard outlier filter such as this one, one can equivalently define a new set of matches $\mathcal{M}'$ with only the pairs with a weight of 1.

2.6 Error Minimization

The aim of `error minimization` is to solve Equation 1.2:

$${}^{i+1}_{i}\mathcal{T} \leftarrow \arg\min_{\mathcal{T}} \left(\text{error} \left(\mathcal{T} \left({}^{i}\mathcal{P}' \right), \mathcal{Q}' \right) \right).$$

This step relies on the definition of an error metric calculated from the association of features and needs to be resolved using an error model. The error model can be sometimes the same as the distance metric used at the matching stage but the main difference is that error is only defined in the feature space and not in the descriptor space. This is because only features are influenced by transformation parameters, as listed in Table 2.4. So, if the association is based on descriptor distances, another error must be defined to correct the misalignment. Parameters selected for the minimization should follow an expected deformation model. Zitová and Flusser [2003] present two generic types to classify error metrics: global (rigid, affine transform, perspective projection model) and local (radial basis functions, elastic registration, fluid registration, diffusion-based, level sets, optical-flow-based registration).

2.6.1 Shape Morphing

Most of the data association algorithms based on point clouds use global-rigid error. This error metric is parametrized by 3 translations and 3 rotations parameters for a total of 6 Degrees of Freedom (DoF), when dealing with 3D point clouds (3 DoF in 2 dimensions). Point-to-point error uses the most basic primitive and was first introduced in a registration context by Besl and McKay [1992] and used subsequently in multiple solutions [Godin et al., 1994, Pulli, 1999, Druon et al., 2006, Pan et al., 2010, Kim, 2010]. During the matching step, it might happen that different kind of geometric primitives (e.g., point, line, curve, plane, quadric) are matched together. Multiple error metrics were developed for those situations and we want to bring them under the same concept that we introduce as *Shape Morphing*. Essentially, when a primitive with higher dimensionality is matched with a lower one, it is morphed via projective geometry to adapt to its counterpart. Figure 2.6 presents the list of possible combination for a 2D space and illustrates the concept for different errors. Using the subfigure labeled point-to-line as an example, a point in solid red matches a line in dashed blue. To generate an alignment error, a virtual point (i.e., the empty blue circle) is generated by projection. The same principle applies to point-to-curve and line-to-curve. Although not depicted in Figure 2.6, their 3D counterparts (i.e., points, planes, quadrics) follow the same projection principle.

The most represented example is the point-to-plane introduced by Chen and Medioni [1992] and then reused in multiple works [Champleboux et al., 1992, Gagnon et al., 1994, Bergevin et al., 1996, Gelfand et al., 2003]. Its 2D version, point-to-line, is also used in robotics [Bosse and Zlot, 2009a] and a closed-form solution was presented by Censi [2008]. Using higher complexity to represent 3D primitives, Segal et al. [2009] propose the use of plane-to-plane, while early work of Feldmar and Ayache [1996] uses quadric-to-quadric.

It is also possible to find extensions to those error metrics: point-to-point with extrapolation and damping [Zinsser et al., 2003], a mix of point-to-line with odometry error [Diebel et al., 2004], a mix of point-to-point, point-to-line or point-to-plane with angle [Armesto et al., 2010]

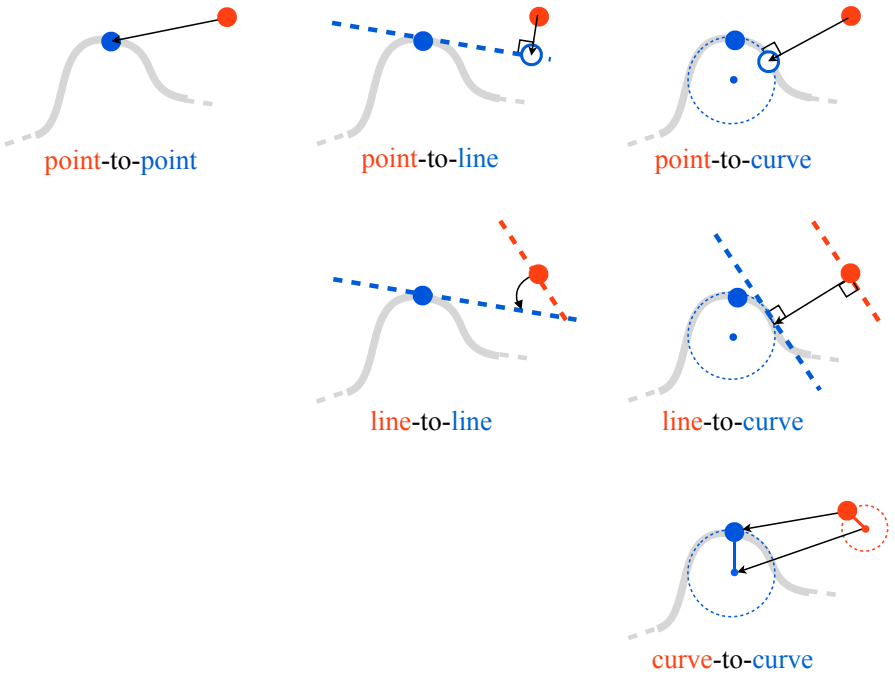

Figure 2.6: Possible morphing in 2D. The real underlaying shape is represented in light gray with its approximation in dark blue. The misaligned surface is represented by a point in light red. The resulting errors are represented with black arrows.

and mix of point-to-point with Boltzmann-Gibbs-Shannon entropy and Burg entropies [Liu, 2010]. Entropy based methods used in medical registration were reviewed by Pluim et al. [2003] as being: Shannon, Rodriguez and Loew, Jumarie, Rényi entropies. All those techniques rely on mean squared error.

Recently, Silva et al. [2005] introduce a novel error called Surface Interpenetration Measure (SIM), which presents more robustness against different noise types. This measure was then applied later by Pan et al. [2010] for face recognition. Image registrations mainly use affine transformations including skew and scale deformations like in [Lowe, 2004]. A more complex hierarchy of error models, presented by Stewart et al. [2003], increases the transformation parameter complexity from similarity to affine, reduced quadratic and finally quadratic. Those error

models allow them to achieve higher precision on the final alignment, while avoiding heavy computation at the beginning of the minimization.

2.6.2 Optimization

Once the error model is defined, the problem is to select a strategy or scheme to find the transformation with the minimum error. Different optimization strategy are reviewed and discussed by Rusinkiewicz and Levoy [2001]. The authors mention the possible use of Singular Value Decomposition (SVD) [Arun et al., 1987], quaternions [Horn, 1987], orthonormal matrices [Horn et al., 1988], and dual quaternions [Walker et al., 1991] for the point-to-point objective function. It is noted that the results provided by those solutions are quite similar when the association between points is unknown [Eggert et al., 1997]. This is why those optimization solutions are only briefly listed in this review. In the case of the point-to-plane error, linearization based on small angle approximation is mainly used following its original implementation [Chen and Medioni, 1991]. Other objective functions for point cloud alignment rely on histogram correlation [Bosse and Zlot, 2008], tensor voting [Reyes et al., 2007], or Hough transform [Lowe, 2004, Censi, 2006].

Example 1 In the case of the point-to-point error, the error is the Euclidean distance:

$$\begin{aligned} \text{error}(\mathcal{P}, \mathcal{Q}) &= \sum_{(\boldsymbol{p},\boldsymbol{q})\in\mathcal{M}'} \|\boldsymbol{p} - \boldsymbol{q}\|_2 \\ &= \sum_{k=1}^{K} \|\boldsymbol{p}_k - \boldsymbol{q}_k\|_2 \end{aligned}$$

where K is the number of points in $\mathcal{M}'$.

The error minimization is then:

$$\begin{aligned} {}^{i+1}_{i}\boldsymbol{T} &= \arg\min_{\boldsymbol{T}} \left(\sum_{k=1}^{K} \|\boldsymbol{T}\boldsymbol{p}_k - \boldsymbol{q}_k\|_2 \right) \\ &= \arg\min_{\boldsymbol{T}} \left(\sum_{k=1}^{K} \|\boldsymbol{R}\boldsymbol{p}_k + \boldsymbol{t} - \boldsymbol{q}_k\|_2 \right). \end{aligned}$$

In that case, this minimization problem can be solved analytically by computing the centroids (average) of the point clouds, and the singular value decomposition of the covariance [Arun et al., 1987]. More precisely, let $\boldsymbol{\mu}_p = \frac{1}{K}\sum_{k=1}^{K} \boldsymbol{p}_k$ and $\boldsymbol{\mu}_q = \frac{1}{K}\sum_{k=1}^{K} \boldsymbol{q}_k$ be the centroids of both point clouds. The covariance is then:

$$\boldsymbol{H} = \sum_{k=1}^{K} (\boldsymbol{p}_k - \boldsymbol{\mu}_p)(\boldsymbol{q}_k - \boldsymbol{\mu}_q)^\top.$$

Let $\boldsymbol{U}\boldsymbol{\Lambda}\boldsymbol{V}^\top$ be the singular value decomposition of $\boldsymbol{H}$. It can be shown that the optimal transformation can be computed with:

$$\begin{cases} \hat{\boldsymbol{R}} &= \boldsymbol{V}\boldsymbol{U}^\top \\ \hat{\boldsymbol{t}} &= \boldsymbol{\mu}_q - \hat{\boldsymbol{R}}\boldsymbol{\mu}_p. \end{cases}$$

Example 2 Another error often used is point-to-plane error, which is only the distance between a point and the plane defined by another point and the normal associated to it:

$$\text{error}(\mathcal{P}, \mathcal{Q}) = \sum_{k=1}^{K} \|(\boldsymbol{p}_k - \boldsymbol{q}_k) \cdot \boldsymbol{n}_k\|_2$$

where $\boldsymbol{n}_k$ is the normal vector around the 3D point $\boldsymbol{q}_k$ in `reference`.

The usual method relies on the linearization of the rotation matrix:

$$\boldsymbol{R} = R(\alpha, \beta, \gamma) \approx \begin{bmatrix} 1 & -\gamma & \beta \\ \gamma & 1 & -\alpha \\ -\beta & \alpha & 1 \end{bmatrix} = [\boldsymbol{r}]_\times + \boldsymbol{I}.$$

The full transformation is parametrized by 6 degrees of freedom:

$$\mathcal{T} = \boldsymbol{\tau} = \begin{bmatrix} \boldsymbol{r} \\ \boldsymbol{t} \end{bmatrix} = \begin{bmatrix} \alpha \\ \beta \\ \gamma \\ t_x \\ t_y \\ t_z \end{bmatrix}.$$

Under these assumptions, the optimal can be obtained by solving the following linear system (see Appendix A for more details):

$$\boldsymbol{G}\boldsymbol{G}^\top \boldsymbol{\tau} = \boldsymbol{G}\boldsymbol{h} \tag{2.1}$$

where

$$\boldsymbol{G} = \begin{bmatrix} \dots & \boldsymbol{p}_k \times \boldsymbol{n}_k & \dots \\ & \boldsymbol{n}_k & \end{bmatrix}$$

is a $6 \times K$ matrix and

$$\boldsymbol{h} = \begin{bmatrix} \vdots \\ (\boldsymbol{q}_k - \boldsymbol{p}_k) \cdot \boldsymbol{n}_k \\ \vdots \end{bmatrix}$$

is a column vector of K elements. The linear system of Equation 2.1 can be resolved for $\boldsymbol{\tau}$ using the Cholesky decomposition.

2.7 Summary

In this section, we presented a review of current registration solutions expressed in a common framework. Publications on this topic have a large variety of contributions, from the adaptation of a generic solution to a specific application, up to a detailed theoretical solution of a single registration module. The concepts related to registration problem touch also a variety of mathematical tools (e.g., differential geometry, statistics, probabilities, (robust) regression, etc.). We also noted that same concepts have a different nomenclature depending on the field: computer vision, robotics or medical imagery. Although the concepts

are very similar, there is also a gap to be bridged between the terminology used for geometric registration and image registration. This situation seems to have been better handled in medical imagery than in robotics and computer vision. It could be explained by the fact that, from the beginning, registrations were applied for 2D/2D, 3D/3D and 3D/2D to cover the different standardized sensors that were being used to provide diagnostics. We provide a summary of key concepts in Table 2.5 to facility an overview of the classifications used.

The actual design of a geometric registration solution can be represented as an assembly of modules. However, judging from the number of publications, it can be difficult to choose them for a given problem. It is important to first clarify the characteristics of the application and its implication regarding the registration framework (i.e., recall Figure 1.5). A given application defines `reading` and `reference` point cloud sources as well as how the `initial transformation` can be provided. `Data filters` are typically defined to enhance the point clouds with additional information used for matching or outlier rejection, such as surface normals. They are also used to reduce the number of points to decrease the computational load and to bound the local density of the points leading to algorithms more robust to different sensor configurations. The `matching function` might need to be tuned if the initial transformation cannot be guaranteed to be bounded. Finally, the choice of `outlier filters` has an impact on the robustness of the error minimization and might accommodate changing overlap between reading and reference.

Simple solutions can often work well in practice provided the underlying assumptions are well understood. In the next section, using concrete examples, we will demonstrate what those assumptions can be and how to motivate the selection of a particular solution.

Table 2.5: Possible classifications for registration algorithms.

`reading` and `reference`	
Sensor types	Photometric, time-of-flight, phase-shift, triangulation
Applications	Scene/object reconstruction, identification, tracking
Data acquisition	Different viewpoints, different times, different sensors, scene to model
`Initial Transformations`:	
Sources	External sensors, users, other registration algorithms
Types	Single hypothesis, multiple hypotheses
Systems	Iterative, cascade
`Data Filters`:	
Goals	Enhance discrepancy, reduce time, reduce noise
Descriptor invariance	Rotation, translation, scale, affine
Feature Relationship	Unstructured, structured
Support	Laser intensity, color, geometry
`Association Solvers`:	
Types	Feature, descriptor, mixed
Direction	Reading to reference, reference to reading, both
Distance metric	Euclidean, Mahalanobis, χ^2 test statistic, custom
Optimization	Hashing/indexing, static space partitioning, dynamic space partitioning, feature reduction
`Outlier Filters`:	
Outlier sources	Partial observations, dynamic elements, sensor noises
Support	Features, descriptors, mixed
Assignment	Hard, soft, mixed
`Error Minimizations`	
Error	Geometric, morphing, entropy
Deformation	Global, local
Minimization schemes	Closed form, small angle approximation, voting

3

Registration Use Cases

Mobile robotic platforms with 3D sensing capabilities can be used to build a 3D map of their environment. The 3D representation of an environment can be used to support other algorithms or complement other sensors. In this section, we present applications of geometric registration on various prototype platforms. The intention is to illustrate the choice of specific modules to compose an ICP solution tuned to the task.

Following the registration analysis detailed in Section 2, we describe mobile robotic applications by focussing on the environment, platform velocity, available localization information and type of sensors. For all applications, the same computation scheme was applied. A sequence of 3D scans was streamed to a registration module, which firstly appliced a set of filters directly on the input 3D scan in local coordinate frame (i.e., the origin being the center of the sensor). Secondly, a standard ICP solution is used to registered the latest input scan with a global map . Finally, the newly registered scan is concatenated with the global map and filters are applied to the resulting map before making it available to the next scan registration. This type of processing pipeline was first proposed by Wulf et al. [2008] under the name of

metascans, which brings the random walk error as a function of distance newly explored instead of time. All of those processing steps were made using the library `libpointmatcher` presented in [Pomerleau et al., 2013]. Only text-based files with parameters were changed to achieve the presented reconstruction results. No loop-closing detection nor error back-propagation algorithms were used to post-process the resulting maps. This three-step procedure could be considered as *laser odometry* (as compared to visual odometry) instead of a globally consistent solution. The goal is to present to the reader qualitative mapping results, evaluate for which applications a local registration is sufficient and to abstract a general list of registration challenges that results from the experiences gather in those applications. We had the opportunity to cover multiple types of platforms moving on the ground, air and water, with different sizes and velocities.

The first application presented in Section 3.1 addresses a critical task for real-time registration: Search & Rescue. Field deployments on firemen training site were published at the *2012 International Conference on Field and Service Robotics* [Kruijff et al., 2012], while the results specific to registration were published in [Pomerleau et al., 2013]. A second application relevant to registration is the automation of inspection as described in Section 3.2. The mapping capability of the platform used to demonstrate this task was first published at *1st International Conference on Applied Robotics for the Power Industry* [Tâche et al., 2010], and the extended results were issued in the *Journal of Field Robotics* [Tâche et al., 2011]. The third application employed a novel autonomous vessel, which was described in *IEEE Robotics and Automation Magazine* [Hitz et al., 2012], to demonstrate shoreline monitoring (Section 3.3). Finally, autonomous-driving car is shortly addressed in Section 3.4. We end in Section 3.5 with some lessons learned from the experimentation in those different conditions.

3.1 Search and Rescue

Within the framework of the European project NIFTi (FP7-ICT-247870), novel solutions were assembled together and tested in order

to support firemen and first responders in Search & Rescue missions. A first use of 3D maps is to help plan the of strategic deployment of responders in environment were humans can only intervene for a limited time. Those situations include nuclear incident, chemical spill, unstable supporting structures, and excessive heat conditions. When tele-operated, 3D maps can be used to provide the user with situation awareness and support critical decisions about risky platform motions. This type of application often has a limited communication range, leading to an increased need for autonomous behavior such as navigation [Colas et al., 2013]. More autonomy also means more onboard computation in case of communication breakdown. In such situations, onboard localization is essential to return the platform to a location were wireless communication can be reestablished.

Apart from increasing pressure on real-time solutions, Search & Rescue environments cover a large spectrum of possibilities. For example, deployments can happen in a well-structured nuclear plant, in a partially-collapsed building or outdoors in a case of a train chemical spill. In an advanced robot-human collaborative intervention, dynamic elements created by the other agents (e.g., firemen, other robots, etc.) acting on the field need to be considered in the registration solution. Moreover, dynamic elements at a global level (e.g., building collapsing during the exploration) can happen. In the case of relatively contained situations, where humans have difficulty accessing a restricted intervention area, very little dynamic elements are expected (e.g., the Fukishima Daiichi nuclear disaster in 2011). In the cases presented in this section, applications were demonstrated with few dynamic elements and without the need to identify them.

The platform deployed in the field is called NiftiBot (Figure 3.1). Its mobility is provided by two tracked bogies linked by a differential to facilitate the crossing of uneven terrain. Moreover, two flippers per track allow an active control of the platform orientation and offer an extended range for gap traversals. The mechanical configuration of the robot enables it to climb slopes up to 45°, including stairs. The robot fits in a bounding box of $0.17\,\mathrm{m}^3$ volume and weights approximately 20 kg. The primary sensor used for registration was a 2D rotating SICK

LMS-151 laser, with its rotation axis pointing toward the front of the robot. The aggregation of 2D scans used the motion information of the platform to reconstruct 3D scan at 0.35 Hz. A typical 3D scan contained 55,000 points. Two other sensors can be found on the platform, namely a GPS-aided IMU (X-sens MTI-G) and an omnidirectional camera (PointGrey Ladybug2). The velocity of the platform can be considered slow (0.3 m/s), especially during teleoperation where delicate motion are required. The pre-alignment information used as input for the registration was based on a Kalman filter fusing the IMU and the odometry information. The large error on motion estimates came from the vibration of the tracks, the large contact surface of the tracks on the ground, and the fact that the platform often collides with obstacles. Therefore, smooth and continuous motion models can easily break, thus simple prediction models (e.g., constant velocity) are not applicable.

Figure 3.1: Photograph of NiftiBot, the main platform used for Search & Rescue demonstrations in an unstructured environment. The 3D sensor, Sick LMS-151, is positioned in front of the robot, in-between the two white tracks. The extra motorized rotation axis is pointing in the forward direction of the robot.

The configuration of the rotating laser produces a high density of points in front of the robot, which is desirable to predict collision, but not beneficial to the registration minimization. Thus, we forced the maximal density to 100 points per m^3 after having randomly subsampled the point cloud in order to finish the registration and the map maintenance within 2 seconds. We expected the error on pre-alignment of the 3D scans to be less than 0.5 m based on the velocity of the platform and the number of time ICP would be computed per second. We used this value to limit the matching distance. We also removed paired points with an angle difference larger than 50° to avoid, for example, points from different side of a wall to match together. This happen often when the robot is moving through different rooms. As for the global map, we maintained a density of 100 points per m^3 every time a new input scan was merged in it. A maximum of 600,000 points were kept in memory to avoid degradation of the computation time performance when exploring a larger environment than expected. The complete list of modules used with their main parameters can be found in Table 3.1.

3.1.1 Indoor Preliminary Tests

To test the range of the platform motions and to demonstrate the need for 3D reconstructions, we ran an experiment in our laboratory leading to the reconstruction of a full staircase (Figure 3.2). The robot started its path in an office located on the E-floor, and was driven down a staircase two floors below (C-floor, in the basement). The robot was controlled using a joystick by an operator following it throughout the path. The robot was then driven six floors up to the I-floor using the same staircase. In this application, the robot acquired scans from a stop-and-go strategy with a scan taken roughly every 2 m.

The complete map was processed onboard the robot. Because the information of the past environment was fused in the global map while the robot went down the stairs, the drift in the localization was considerably reduced on the way up. This experiment comprised two critical moments: (1) when the robot moved out of the office and (2) when the robot entered the basement (C-floor). In both situations the overlap between the new scan and the global map was minimal. This informa-

Table 3.1: Configuration of the ICP chain for the Search & Rescue mapping applications. The definition of the column *Step* follows Section 2. The names used in the column *Module* refer to specific implementation documented in the open source library `libpointmatcher`.

Step	*Module*	*Description*
DF_{read}	`SimpleSensorNoise`	Add uncertainty for the SICK LMS sensor as observed in [Pomerleau et al., 2012a].
	`SamplingSurfaceNormal`	Keep 80 % of the points, while extracting surface normals based on 20 NN.
	`ObservationDirection`	Add vector pointing toward the origin of the sensor.
	`OrientNormals`	Orient surface normals toward the observation direction.
	`MaxDensity`	Subsample to keep point with density of $100\,\mathrm{pts/m^3}$.
DF_{ref}	`SurfaceNormal`	Compute normal and density with 20 NN and an approximation factor $\epsilon = 3.16$.
	`MaxDensity`	Subsample to keep point with density of $100\,\mathrm{pts/m^3}$.
	`MaxPointCount`	Subsample 70 % if there is more than 600,000 points.
MF	`KDTree`	Use an approximate kD-tree with a maximum matching distance of 0.5 m and an approximation factor of $\epsilon = 3.16$.
OF	`TrimmedDist`	Keep 80 % closest paired points.
	`SurfaceNormal`	Remove paired points with normals angle larger than 50°.
EM	`PointToPlane`	Objective function using point-to-plane error.
TC	`Differential`	Stop after a minimum error below 0.01 m and 0.001 rad.
	`Counter`	Stop after the iteration count reached 40.
	`Bound`	Stop if transformation increases beyond 5.0 m and 0.8 rad.

Legend:
DF_{read} = Data Filters for readings, DF_{ref} = Data Filters for references, MF = Matching Function, OF = Outlier Filters, EM = Error Minimizer, TC = Transformation Checker.

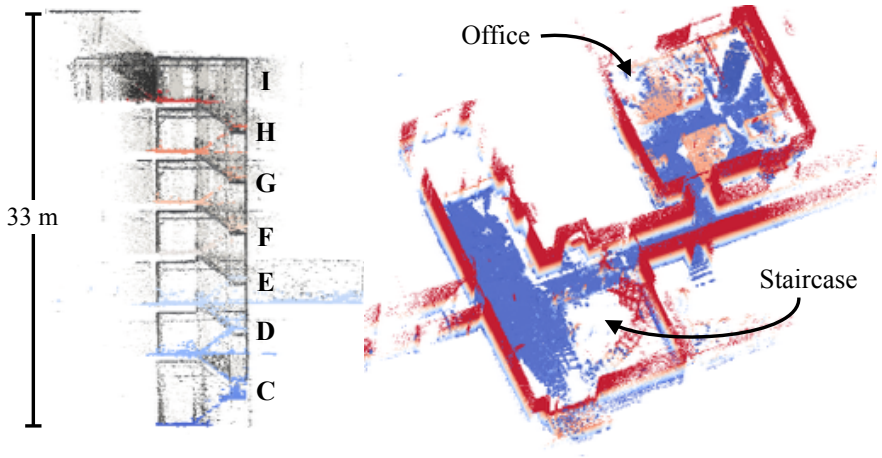

Figure 3.2: Mapping of a 7-floor staircase using a Search & Rescue robot. *Right*: Side view of the resulting map with the floors colored based on elevation. *Left*: Top view of the E-floor with the ceiling removed and the points colored based on elevation, red being higher, blue lower.

tion was known by the operator, so more scans were taken at those moments to avoid large deviations in the global map.

3.1.2 Rail Yard

On two occasions, the NIFTi platform was tested outdoors in a rail yard, with the permission of the *Depo kolejových vozidel Praha* (Prague Depot of Rail Vehicles, Czech Republic). In the first experiment, the robot was also driven in the yard by operators who were following the platform. The 3D reconstruction is shown in Figure 3.3. The robot started its journey at one corner of a railcar, going along the railcar flank to the other corner and then, turned back to the starting position passing through the vegetation located on the other side of the railcar. Even if the path contained a loop, the precision of the registration was accurate enough to properly match the first scan with the ones recorded at the end.

In the second experiment, the robot was driven to explore inside an old railcar where a person was standing still in the shadow to test in parallel the capability of the thermal camera. The robot then went

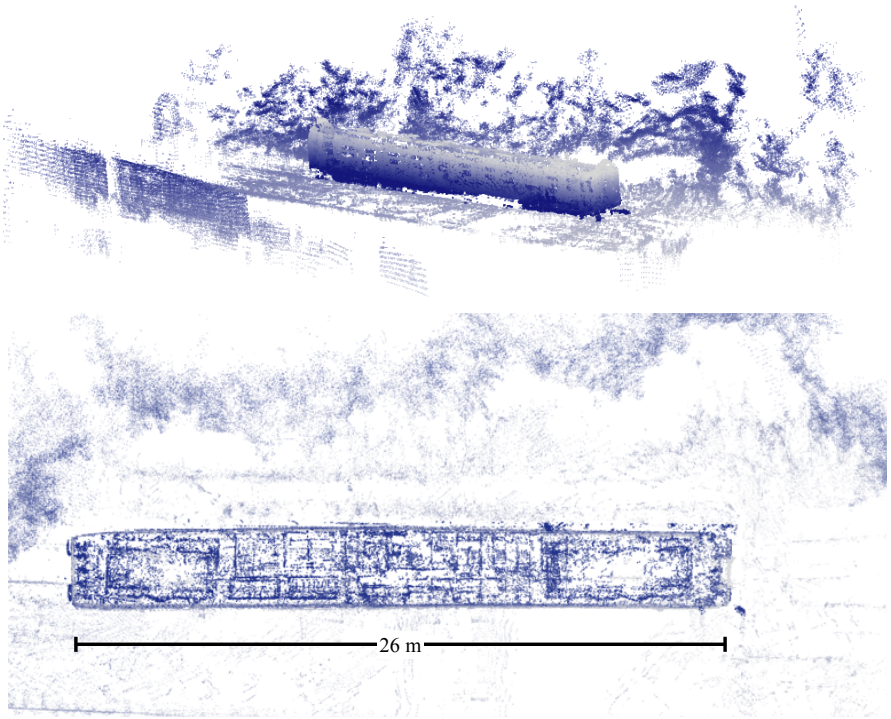

Figure 3.3: Deployment of the NiftiBot in a rail yard with a single railcar and dense vegetation. *Top*: Side view of the reconstructed environment with the railcar in the middle and the vegetation behind. *Bottom*: Top view of the reconstructed environment. For both images, colors of the point clouds were chosen to ease the comprehension of the 3D-scene.

out of the railcar, crossed dense vegetation, followed the side of a more modern railcar and stopped in front of it, where a second operator was captured in the global map.

For both experiments, the robot acquired scans with a stop-and-go strategy. The scans were gathered at uneven distances (up to 8 m apart) by operators without prior knowledge about critical situations. All the 3D scans were processed offline four times faster than the speed at which they were recorded. Those experiments are a good example of semi-structured environments, where trees and dense vegetation share the scene with planar surfaces from trains.

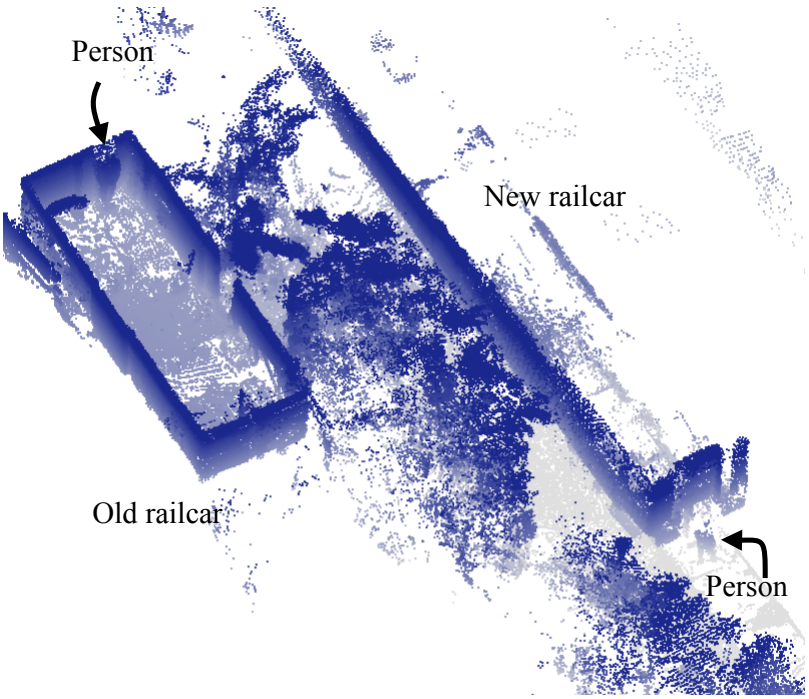

Figure 3.4: Bird's-eye view of the second experiment where the robot explored inside an old wooden railcar, crossed some dense vegetation and finished its path around a more modern railcar.

3.1.3 Collapsed Church

In May 2012 a sequence of earthquakes hit the Emilia-Romagna region, Northern Italy, with a magnitude of 5.8. Three month later, NIFTi partners deployed the platform with the support of the *Vigili del Fuoco* (National Fire-watchers' Corps of Italy) and the *Beni Culturali* (Ministry of Culture of Italy) in Mirandola for a damage assessment mission. One of the visited sites was the *Chiesa di San Francesco d'Assisi*, in which 3D scans were recorded. The robot started outside the church, crossed a door and realized a straight line, navigating on the cluttered floor of the western gallery of the church (Figure 3.5). One can observe on the reconstruction the pillars and arches supporting the remaining roof of the church. The level of damage of the church was quite im-

portant, thus limiting the exploration possibilities of the platform, as depicted in Figure 3.6.

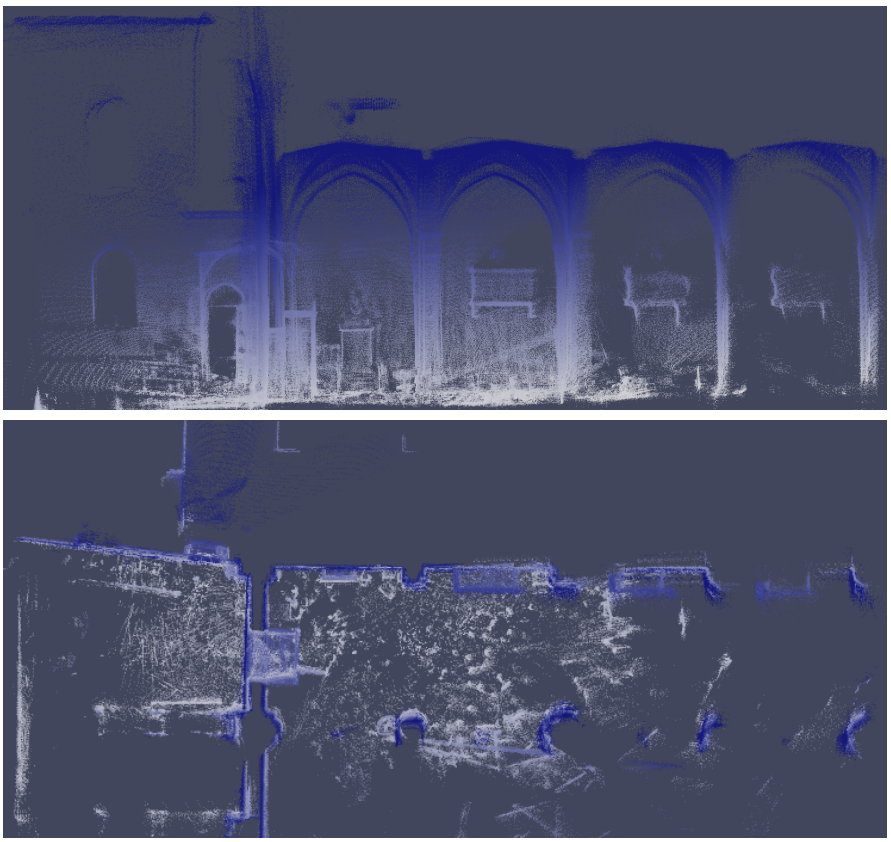

Figure 3.5: Reconstruction of the *Chiesa di San Francesco d'Assisi* with the color following elevation. *Top*: Side view of the reconstruction. *Bottom*: Top view of the church.

The platform was remotely operated from a control station situated outside the church and was able to continuously scan the environment while moving in the environment. Again, all the 3D scans were processed offline four times faster than the speed at which they were recorded. It is another example of semi-structured environment but, in this case, the unstructured part comes from rubble following the partial collapse of the church.

Figure 3.6: Comparison of the point cloud reconstruction with a photograph taken during exploration. *Left*: Photograph of the western gallery with the collapse roof on the right. *Right*: Front view of the reconstruction.

3.1.4 Collaborative Mapping

Within the framework of the European project *sFly* (FP7-ICT-231855), three micro-helicopters (AscTec Firefly) were deployed over a Search & Rescue training site in Zurich, Switzerland. The platform used is shown in Figure 3.7 flying over a collapsed concrete building. The three Fireflies were sent so that each one covered a pre-determined part of the environment and streamed back images to a control station. The collected images were used by the ETH Computer Vision and Geometry Group to reconstruct a 3D representation of the environment explored (Figure 3.8 - *Bottom left*).

Another map was generated using a ground platform (i.e., Nifti-Bot). The robot was tele-operated on a road around the main collapsed building presented in Figure 3.8, for a path totaling 110 m long. The operator had a good prior knowledge of the environment before driving the robot around from a control station. The large road coupled with the awareness of the environment contributed to increase the velocity of the robot while exploring the area. The resulting map of the ground robot and the map of the Fireflies were then fused using a standalone ICP implementation taken from our registration library `libpointmatcher`. Both map were having roughly 300,000 points and were registered using the configuration of Table 3.2. The final map is depicted at the bottom right of Figure 3.8.

Figure 3.7: Photograph of one of the three AscTec Fireflies used to map the environment in collaboration with the NiftiBot.

Table 3.2: Configuration of the ICP chain for the collaborative mapping applications. The definition of the column *Step* follows Section 2. The names used in the column *Module* refer to specific implementation documented in the open source library `libpointmatcher`.

Step	*Module*	*Description*
DF_{read}	`RandomSampling`	Keep randomly 15 % of the points.
DF_{ref}	`SamplingSurfaceNormal`	Keep 15 % of the points, while extracting surface normals based on 50 NN.
MF	`KDTree`	Use a kD-tree with a maximum matching distance of 1.0 m and match a point from the reading to 5 others in the reference.
OF	`TrimmedDist`	Keep 80 % closest points.
EM	`PointToPlane`	Objective function using point-to-plane error.
TC	`Differential`	Stop after a minimum error below 0.01 m and 0.001 rad.
	`Counter`	Stop after the iteration count reached 100

Legend:
DF_{read} = Data Filters for readings, DF_{ref} = Data Filters for references, MF = Matching Function, OF = Outlier Filters, EM = Error Minimizer, TC = Transformation Checker.

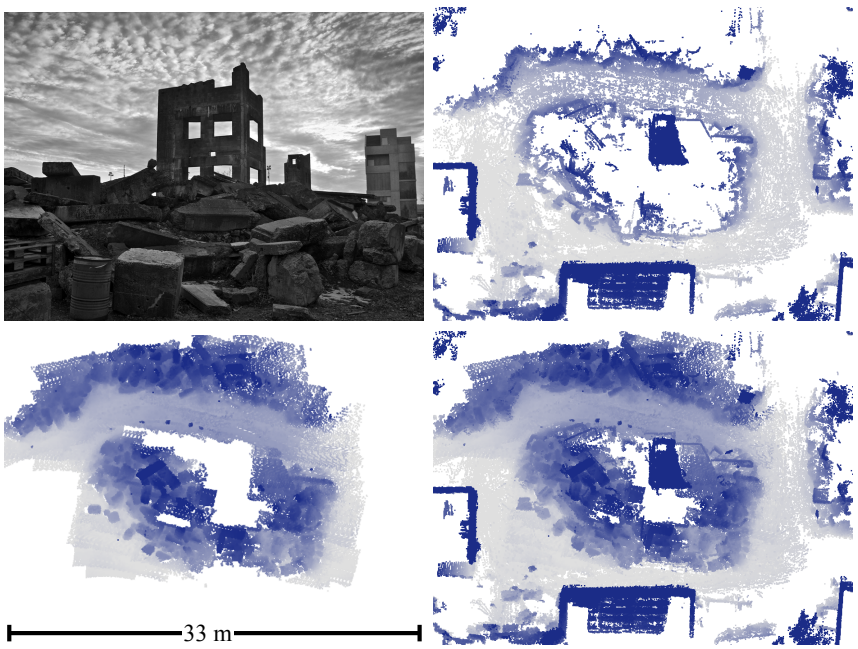

Figure 3.8: Resulting maps of the Zurich firefighter training site. *Top right*: Photograph of the training site with a partially collapsed tower in the middle. *Top left*: Top view of the reconstruction realized with the data from the ground robot with the same tower in the middle of the map. *Bottom left*: Top view of the reconstruction realized with the data from the three Fireflies. *Bottom right*: Top view of the combined map. Note that the color correspond the elevation: light gray is low, dark blue is high.

At that time, the Kalman filter used to fused the odometry with the IMU was not well tuned and bias in the estimation induced drift on yaw estimates. Roughly, a constant drift of 5°/s was estimated visually. The scans were gathered while the ground platform was moving, which generated a larger localization error than prior experiments. A total of four runs were recorded with the ground platform: (1) continuous scanning, turning clockwise around the main building; (2) continuous scanning, counterclockwise; (3) stop-and-go scanning, clockwise; and (4) stop-and-go scanning, counterclockwise. Two experiments out of four closed the loop with a negligible error at the closing point. Surprisingly, the successful runs were the ones turning counterclockwise,

contradicting our first intuition that stop-and-go scanning would be more accurate. This experiment highlighted the importance of correcting IMU drift with an external registration algorithm and also showed that the robot could scan while moving. We selected the resulting representation of the second run to fuse both sources of information (i.e., laser and camera) in a common 3D reconstruction.

3.1.5 Artor - Autonomous Rough Terrain Outdoor Robot

Another platform was developed in parallel to the NiftiBot by the Autonomous Systems Lab (ASL). Modifications were done over a LandShark from Black-I Robotics, in collaboration with RUAG and Armasuisse. The aim of the project was to develop techniques for reliable autonomous navigation of a wheeled robot in rough, outdoor terrain. The robot, named Artor, was much larger than NiftiBot, with a volume of $0.96\,\mathrm{m}^3$ and an approximate weight of $350\,\mathrm{kg}$ (Figure 3.9). Three wheels on each side of the robot gave the same traction as tracked vehicles, while simplifying the maintenance of the locomotion system. Odometry suffered from the same large rotation error problem as NiftiBot because of the unknown friction between the ground and the wheels. The robot can drive at a maximum speed of $4.5\,\mathrm{m/s}$ but was usually driven at around $1.2\,\mathrm{m/s}$ during the presented experiments. The motion of the platform can be smooth on the pavement but in off-road situation, the motion can be more rough, and at high speed the orientation can change quickly. Odometry computation can lead to large error that is mainly caused by the high friction of the wheels on the ground. Given the early development stage of the platform, only the wheel odometry was available as prior information for the registration module. The main sensor used in this experiment is the Velodyne HDL-32E, which produces around 50,000 points per 3D scan (i.e., not all beams always returned a range value) and were recorded at $5\,\mathrm{Hz}$ (i.e, half of the sensor nominal frequency). Other sensors present on the platform were: two SICK LMS-151 (front and rear), high-resolution zoom camera, thermal camera and a GPS-aided IMU.

The critical element for real-time processing is the amount of points that needs to pass at high rate through the registration module. The

Figure 3.9: Photograph of Artor, a Search & Rescue robot specialized for outdoor applications.

tuning evolved from the parameters and filters selected for NiftiBot with the aim of increasing the registration speed for Artor. We first randomly removed 85 % of the points to ensure a stronger data reduction. We also kept a lower density of points because the platform usually covered larger areas than the NifitBot during a typical deployment. The complete list of modules used with their main parameters are listed in Table 3.3. The Artor robot was driven over different types of terrain in a specialized testing facility in Wachtberg, Germany. The path of the platform was 340 m long, following a rectangular shape and no noticeable errors were found at the loop closure. The robot realized the same loop for a second time using the global representation of the first loop without any problems. An overlay of the 3D map top view with an orthogonal projection of an aerial image is provided in Figure 3.10 - *Top right*.

The data were processed offline at the same rate as the recorded one. The operator was driving the platform around to test the mobility capability of the robot, without explicitly considering any registration limitations. On an open terrain, the solution proposed can manage a

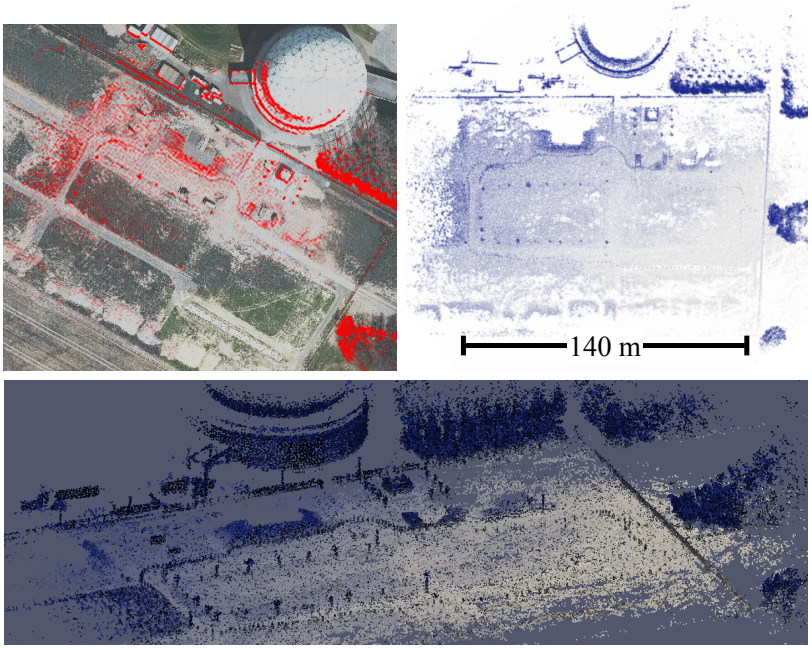

Figure 3.10: Reconstruction of the testing facilities. *Top right*: Overlay of the 3D map with an aerial view. *Top left*: Top view of the reconstruction. *Bottom*: Bird's-eye view of the reconstruction. The color is based on the elevation of the points, light gray being low, dark blue being high. The aerial image was provided by *Bundesamt für Kartographie und Geodäsie*, Frankfurt, Germany - `http://www.bkg.bund.de`.

global and dense representation of the environment, even with the high turn rate of the robot. This might be more challenging for a solution based on visual odometry to avoid drift in those conditions.

3.2 Power Plant Inspection

In collaboration with Alstom Inspection Robotics (AIR), prototypes were developed for the inspection and maintenance of industrial plants. Some inspection tasks need to move inspection tools in environments that are difficult to access by human due to dimensional, temperature or air quality constraints. The use of mobile systems for inspection can not only deal with those constraints, but also can reduce the time and costs of inspections. This would, for example, allow for the inspection of

Table 3.3: Configuration of the ICP chain for the Artor mapping applications. The definition of the column *Step* follows Section 2. The names used in the column *Module* refer to specific implementation documented in the open source library `libpointmatcher`.

Step	*Module*	*Description*
DF_{read}	`RandomSampling`	Keep randomly 5 % of the points.
	`SurfaceNormal`	Compute normal with 20 NN and an approximation factor $\epsilon = 3.16$.
	`ObservationDirection`	Add vector pointing toward the origin of the sensor.
	`OrientNormals`	Orient surface normals toward the observation direction.
	`MaxDensity`	Subsample to keep point with density of $50\,\mathrm{pts/m^3}$.
DF_{ref}	`MaxDist`	Keep points within a radius of 70 m from the last sensor pose.
	`SurfaceNormal`	Compute normal and density with 20 NN and an approximation factor $\epsilon = 3.16$.
	`MaxDensity`	Subsample to keep point with density of $10\,\mathrm{pts/m^3}$.
	`MaxPointCount`	Subsample 70 % if there is more than 600,000 points.
MF	`KDTree`	Use an approximate kD-tree with a maximum matching distance of 0.5 m and an approximation factor of $\epsilon = 3.16$.
OF	`TrimmedDist`	Keep 80 % closest paired points.
	`SurfaceNormal`	Remove paired points with normals angle larger than 90°.
EM	`PointToPlane`	Objective function using point-to-plane error.
TC	`Differential`	Stop after a minimum error below 0.01 m and 0.001 rad.
	`Counter`	Stop after the iteration count reached 40.
	`Bound`	Stop if transformation increases beyond 5.0 m and 0.8 rad.

Legend:
DF_{read} = Data Filters for readings, DF_{ref} = Data Filters for references, MF = Matching Function, OF = Outlier Filters, EM = Error Minimizer, TC = Transformation Checker.

critical pieces of equipment on location, without the need to dismantle any structures. Similarly, the installation of scaffolding around a structure becomes unnecessary, thus saving inspection time. The typical environments encountered during inspection procedures are confined spaces (indoors) with well structured, static surfaces.

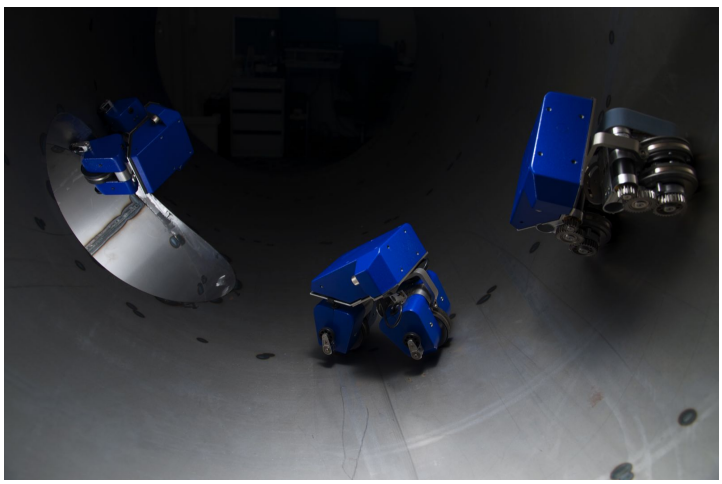

Figure 3.11: Three prototypes of chest inspection robots without the sensors, in a mock-up of a steam chest. The sensor was not installed at the time of the photograph.

For the specific task of steam chest inspection, a robot was developed with high mobility capabilities and a compact size [Tâche et al., 2009]. The robot, named Magnebike, moved around a metallic, cylindric environment by using its two magnetic wheels positioned in the same configuration as a bike (Figure 3.11). Those specialized wheels coupled with small lever arms allowed the platform to move up-side-down, pass 90° edges and navigate in high curvature tube. A considerable amount of effort were invested in reducing the size of the platform, which ultimately gave a small volume of $0.006\,\mathrm{m}^3$ and a weight of $0.34\,\mathrm{kg}$. The platform was the slowest presented in this section, with a maximal velocity value of only 0.045 m/s. The robot gathered high-resolution scans (340,000 points) with a refresh time of $50\,\mathrm{s}$. The 3D scans were assembled from a 2D Hokuyo URG-04LX. Pre-alignment of the scans were ensured by wheel odometry, which displays virtually no slip be-

cause of the magnetic force hold the wheels on the surfaces. An IMU was used in conjunction with the odometry to cope with the 3D nature of the motion [Tâche et al., 2011]. The main sources of pre-alignment errors were: (1) motions perpendicular to the gravity vector that were not observable with the sensor used, and (2) the rapid cumulation of errors by the low-cost IMU used on the slow-moving robot.

During inspection, the robot is intended to be tethered for safety reason, which solved the communication problem between the embedded computer and a faster system used as control station. Because the system must not damage the inspected structure in any case, it traveled slowly in the environment, reducing the pressure on the real-time requirement for the registration. However, the operator might not have a visual contact with the robot at all times during inspection. Therefore, the map resolution must be high enough to detect obstacles and holes during remote operations. The number of points produced by the scanner was significantly larger than strictly necessary for a proper registration. To reduce rapidly this number of points, we randomly removed 90 % of the point as soon as the scan were recorded. We used a maximal density of 20000 points per m^3 to cope with the small size of inspected environments. For the registration, we did not use any pre-alignment to test the worst case scenario (i.e., when the rotation is not observable by the IMU). This forced us to extend the maximal matching distance to 0.5 m, even if a scan was taken roughly at every 0.30 m. Given that scans are taken by an operator at a fix and short interval, we used an outlier ratio of 80 %. The complete list of modules used with their main parameters are listed in Table 3.4.

To test the mapping capability of the platform, a real steam chest was made available by AIR. This part was actually removed from a power plant for reparation purpose. Multiple inspection runs were executed, each run starting from one of the seven entry points (Figure 3.12). We only present here the results from the longest path since it covered the entire environment. The robot started on one side of the steam chest, situated on the left of Figure 3.12. Each 3D scan was taken on a stop-and-go strategy at every 0.1 m. The total path covered a distance of 5.8 m for a total of 59 scans. All runs were registered of-

Table 3.4: Configuration of the ICP chain for the Magnebike mapping applications. The definition of the column *Step* follows Section 2. The names used in the column *Module* refer to specific implementation documented in the open source library `libpointmatcher`.

Step	*Module*	*Description*
DF_{read}	`RandomSampling`	Keep randomly 10 % of the points.
	`SamplingSurfaceNormal`	Keep 80 % of the points, while extracting surface normals based on 20 NN.
	`ObservationDirection`	Add vector pointing toward the origin of the sensor.
	`OrientNormals`	Orient surface normals toward the observation direction.
	`MaxDensity`	Subsample to keep point with density of 20000 pts/m^3.
DF_{ref}	`SurfaceNormal`	Compute normal and density with 20 NN and an approximation factor $\epsilon = 3.16$.
	`MaxDensity`	Subsample to keep point with density of 20000 pts/m^3.
MF	`KDTree`	Use an approximate kD-tree with a maximum matching distance of 0.5 m and an approximation factor of $\epsilon = 3.16$.
OF	`TrimmedDist`	Keep 80 % closest paired points.
	`SurfaceNormal`	Remove paired points with normals angle larger than 50°.
EM	`PointToPlane`	Objective function using point-to-plane error.
TC	`Differential`	Stop after a minimum error below 0.01 m and 0.001 rad.
	`Counter`	Stop after the iteration count reached 100.
	`Bound`	Stop if transformation increases beyond 5.0 m and 0.8 rad.

Legend:
DF_{read} = Data Filters for readings, DF_{ref} = Data Filters for references, MF = Matching Function, OF = Outlier Filters, EM = Error Minimizer, TC = Transformation Checker.

fline approximately 10 times faster than the rate at which they were recorded. Based on this reconstruction, it was possible to have full 3D path planning and navigation in this environment [Stumm et al., 2012].

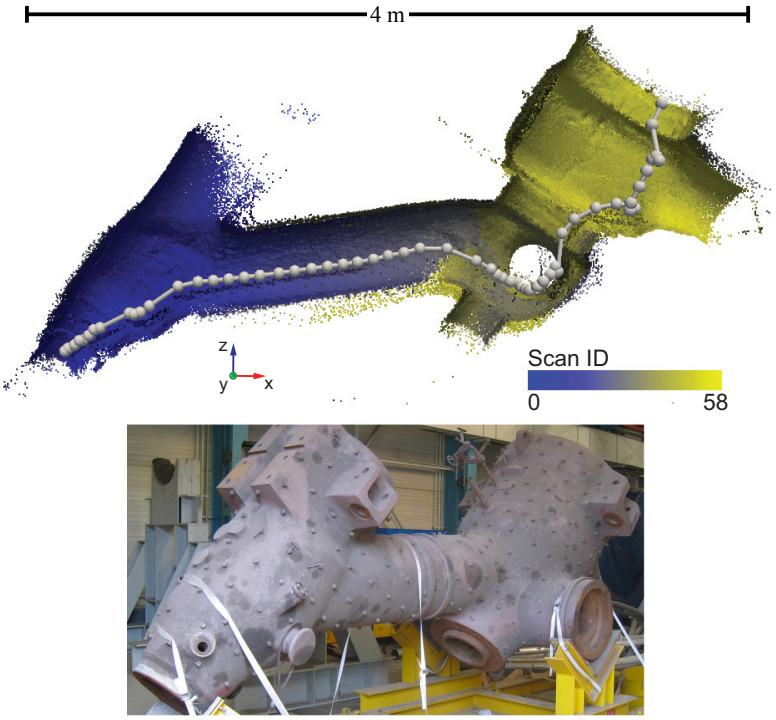

Figure 3.12: Deployment results of Magnebike in a real steam chest. *Top*: Cut view of the reconstructed environment. The light gray line correspond to the path of the robot, with the sphere being the positions where the robot stopped to take a 3D scan. The colors of the map follow the discreet time (from 0 to 58) at which the scans where taken. *Bottom*: The actual steam chest removed for maintenance.

3.3 Shoreline Monitoring

In order to support environmental monitoring of freshwater bodies, an autonomous surface vessel named Lizhbeth was developed at the ASL in collaboration with the Limnological Station of the University of Zurich. Although the vessel was initially developed to deploy biologic sensors in water (see [Hitz et al., 2012, 2014] for details on this

application), a 3D laser was installed on its top to complement the analysis of the ecosystem with geological information. For example, 3D mapping of the shoreline could help to determine the volume of organic material (leaves) falling in a lake, accurately identify inflows of water, quantify coastal erosion, etc. The observation of coastal erosion using rangefinder laser is already an active field in geology [Mitasova et al., 2009], relying mainly on airborne surveys. However, this survey method can not provide a good viewpoint of a cliff, and the deployment costs are quite high. The use of a boat as carrier is comparatively a low-cost method that can give better vantage points in certain situations. Beyond the geological applications, localization on the shore with centimeters precision can increase the autonomy of the system by allowing it to navigate to the sampling point from a parked position in a confined area, such as a boat house, or from its docking recharge station. One of the requirement of such application is to have long range measurements, given that shores imply shallow water, which poses a limit on how close the boat can be without touching the bottom. The type of outdoor environments expected when surveying a water body can vary from structured to unstructured, depending on the intensity of the recreational use by the local people. For example scanned elements can be dams, bridges, houses, beaches, rocky shores, sparse to dense vegetation, etc. Except for other boats, most of the environment is expected to be static with potential for seasonal changes (global motion) monitoring.

The platform was deployed several times in Lake Zurich (Figure 3.13) and once in the alpine Lake Cadagno, both located in Switzerland. It had a volume of $6.75\,\mathrm{m}^3$ and weighted approximately $120\,\mathrm{kg}$. The motion of the robot was ensured by two electrical propellers positioned in the custom-built hulls of the catamaran. This gave differential drive motion capability to the platform, allowing it to turn on spot. The typical velocity of the robot is $0.7\,\mathrm{m/s}$ when surveying away from the shore. The main sensor used for 3D reconstruction was a Velodyne HDL-32E, which produced in average 45,000 points. Compare to the application presented with Artor (Section 3.1.5), the laser return less points in average due to the open horizon of the lac. The point clouds

were recorded at 1.6 Hz. A single-beam underwater sonar was used to produce bathymetric maps. The localization sensors included an IMU, a magnetic compass and a GPS. The GPS was mainly used for offshore navigation because its precision of 5 m made it dangerous for nearshore navigation. The odometry can hardly be computed based on the motor inputs because of the high inertia of the boat in water, and the unknown wind-driven surface currents. Among the main sources of localization perturbations are the waves that may change rapidly the platform orientation, which can be evaluated by the gravity vector measured by the IMU. The smooth motion of the platform rendered difficult to extract reliable translation information without adding any registration algorithms. Nevertheless, a predictive model implying smooth 2D translations on the xy-plane can be used to pre-align scans [Hitz et al., 2014].

Figure 3.13: The autonomous surface vessel, named Lizhbeth, during one of its survey environment: pre-alpine Lake Zurich, Switzerland. The sensor was not installed at the time of the photograph.

While keeping the constraints of Lizhbeth in mind, we ran preliminary mapping experiments using the Velodyne installed on a small

watercraft (7 m long). The substitute boat was a monohull and thus, was considered less stable on water than Lizhbeth, which is a two-hull vessel. As no external sensors were available, the full solution was tuned to not rely on any pre-alignment of the scans. The input filters applied ensured that the watercraft was removed from the scans, and a fast random subsampling reduced the number of points to ensure registration at every 0.6 s. The watercraft recorded scans while sailing, and its movements depended on water motions. Since waves induced fast changes in the watercraft's orientation, the matching of the scans needed to be fast enough to keep the error on the initial orientation small. The size of the survey area was expected to be large, so a low density of points was forced. When the laser hits the water it is usually not reflected back to the sensor, as opposed to solid ground. Unfortunately, some waves can be detected by the laser because of their variable surface orientations. To reduce wave-reflectance effect, we applied a strict shadow point filter that removes 3D points that display an angle difference larger than 17° between surface normals and the direction of observation. The complete list of modules used with their main parameters are listed in Table 3.5.

The experiment executed with the watercraft was recorded on Lake Zurich, in front of the Limnological Station, the typical operating area of Lizhbeth. The boat started away from the shore moving towards a harbor where multiple boats were parked side by side. This starting position is located on the lower left corner of Figure 3.14. The boat first passed between the harbor and boats anchored on buoys and turned right to continue between the anchored boats and the shore. The boat sailed parallel to the shore up to a boat house situated within an artificial small canal leading to the entrance of the boat house. On the reconstructed environment, one can notice the noise around the anchored boats that was caused by their movements during the experiment, especially around the white one, at the bottom left corner. Also seen in that corner are the noisy light gray points that were generated by the reflection of the laser on the waves. The final map covered an area of 280 by 130 m without displaying any major defects.

Table 3.5: Configuration of the ICP chain for the Lizhbeth mapping applications. The definition of the column *Step* follows Section 2. The names used in the column *Module* refer to specific implementation documented in the open source library `libpointmatcher`.

Step	*Module*	*Description*
DF_{read}	`BoundingBox`	Remove points in a box of 7×7×2 m to avoid self-scanning.
	`RandomSampling`	Keep randomly 20 % of the points.
	`SurfaceNormal`	Compute normal and density with 20 NN and an approximation factor $\epsilon = 3.16$.
	`ObservationDirection`	Add vector pointing toward the origin of the sensor.
	`OrientNormals`	Orient surface normals toward the observation direction.
	`MaxDensity`	Subsample to keep point with density of 50 pts/m^3.
	`Shadow`	Remove points with angle between surface normals and observation direction larger than 17°.
DF_{ref}	`MaxDist`	Keep points within a radius of 70 m from the last sensor pose.
	`SurfaceNormal`	Compute normal and density with 20 NN and an approximation factor $\epsilon = 3.16$.
	`MaxDensity`	Subsample to keep point with density of 50 pts/m^3.
	`MaxPointCount`	Subsample 70 % if there is more than 600,000 points.
MF	`KDTree`	Use an approximate kD-tree with a maximum matching distance of 5.0 m and an approximation factor of $\epsilon = 3.16$.
OF	`TrimmedDist`	Keep 90 % closest paired points.
	`SurfaceNormal`	Remove paired points with normals angle larger than 90°
EM	`PointToPlane`	Objective function using point-to-plane error.
TC	`Differential`	Stop after a minimum error below 0.01 m and 0.001 rad.
	`Counter`	Stop after the iteration count reached 40.
	`Bound`	Stop if transformation increases beyond 5.0 m and 0.8 rad.

Legend:
DF_{read} = Data Filters for readings, DF_{ref} = Data Filters for references, MF = Matching Function, OF = Outlier Filters, EM = Error Minimizer, TC = Transformation Checker.

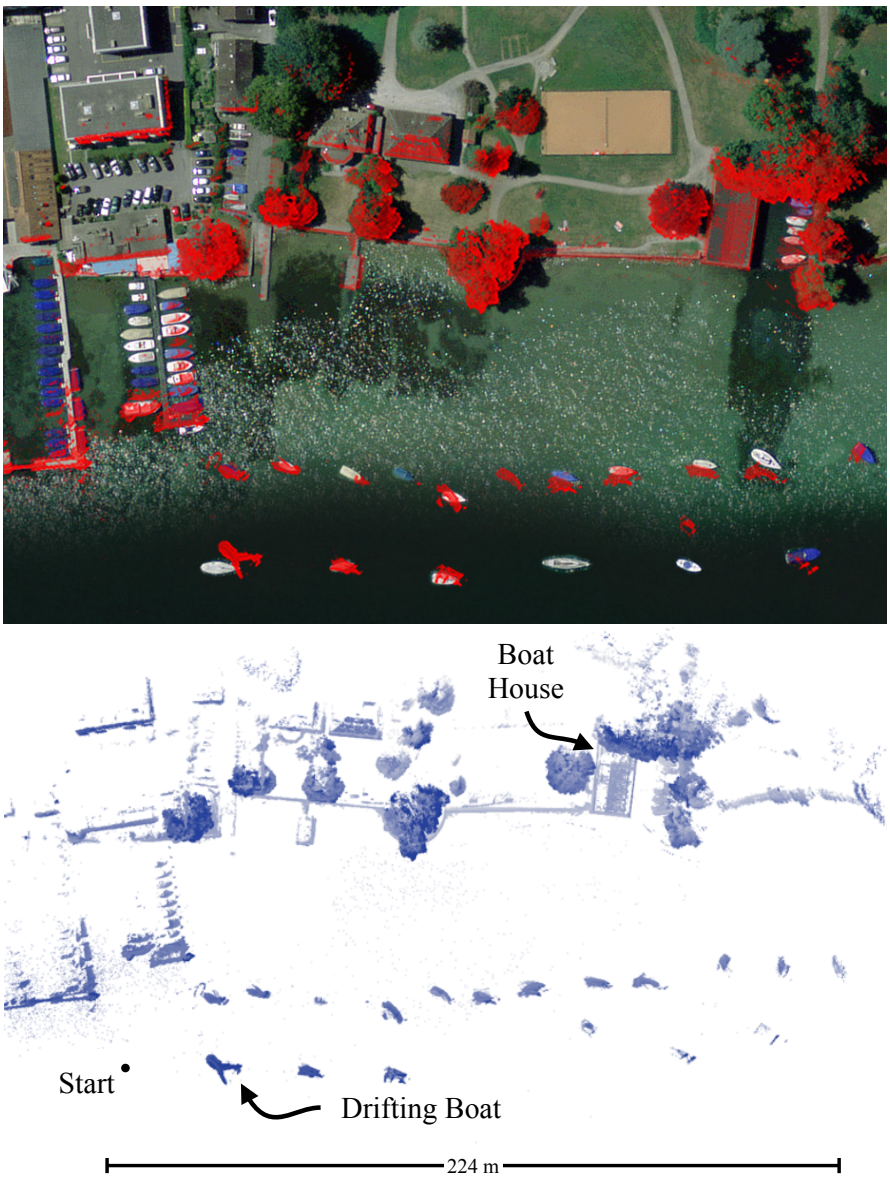

Figure 3.14: Reconstruction of the shoreline from a boat. *Top*: Overlay of the 3D map with an aerial view. *Bottom*: Top view of the 3D map with point colors based on elevation, light gray being low and dark blue being high. The orthogonal projection of the aerial image was provided by the Bundesamt für Landestopografie swisstopo (Art. 30 GeoIV): 5704 000 000

The final solution must take into consideration that elements located offshore can have multiple possible positions. By keeping the global map updated at every uses, those multiple positions will be retained in the map, thus reducing the chances of large drifts in cases where, for example, only the boats on the buoys were scanned.

3.4 Autonomous Driving

The recent expansion of the autonomous driving field have been pushed forwards by car companies that teamed up with research partners, like Volkswagen in the EU project V-Charge, or by large company like Google hiring roboticists to develop new car prototypes. The American state of Nevada even officially delivered its first license for driverless car in May 2012. Although photometric registration is more attractive from the industry point of view, due to the potential low cost of cameras, geometric dense maps can support other activities. For example, road inspection is a tedious task that is mainly realized visually by operators driving on the roads. Large-scale road construction sites can also profit from a fast monitoring of the work progress. Applications usually target urban environments, which are mostly structured (e.g., road, buildings) or semi-structured when the vegetation is more prominent. The environment is predominantly static, but a large part of the field of view can be occupied by other cars such as in dense traffic situations.

The SmartTer (Figure 3.15) was a modified version of a Smart Fortwo. The car was developed by the ASL and served in 2006 has a technological demonstrator in the European SPARC project realized in collaboration with Daimler Chrysler. The Smart is one of the most compact car with a volume of $6.38\,\mathrm{m}^3$ and a weight of 730 kg. Two SICK LMS-291 laser rangefinders were mounted on a vertical rotating axis, each of them providing 14,000 points every second. Motion compensations were applied to the 3D scans to cope with the high speed of the vehicle (15 km/h). Other sensors included navigation laser, omnidirectional camera, monocular camera, GPS and IMU. The overall motion of the vehicle is expected to be smooth, with a strong assumption of

translation on the xy-plane. For a deeper description of the vehicle, we refer the reader to the publication of Lamon et al. [2006].

Figure 3.15: The autonomous car, named SmartTer, with its suite of five SICK LMS-291 laser rangefinders. Larger wheels were installed to allow off-road driving.

The solution selected for this application had to deal with a large scanned volume and noises caused by the velocity of the vehicle. We drastically reduced the number of inputted points by keeping a maximal density of 0.5 points per m^3. For the registration, we cut points beyond the maximal reach of the sensor to reduce the NN searching space. The pre-alignment of the scans was fairly accurate, but we kept a maximum matching distance of 1.5 m to recover from loop-closing error. Many large trees along the road were having their surface normal wrongly estimated due to their unstructured nature, which increased the point-to-plane alignment error. With that in mind, we used a point-to-point error metric for the minimization, as opposed to other solutions presented in this section. A denser global representation was maintained to ensure for more stability of the registration, especially on the ground where the density of a single scan was dropping rapidly.

The complete ICP solution ran at 3 Hz, which permits the registration to run in real-time with a bit of margin. As showed in Figure 3.16, the SmartTer started and ended its path in a street situated at the top

Table 3.6: Configuration of the ICP chain for the SmartTer mapping applications. The definition of the column *Step* follows Section 2. The names used in the column *Module* refer to specific implementation documented in the open source library `libpointmatcher`.

Step	*Module*	*Description*
DF_{read}	`SurfaceNormal`	Compute normal with 20 NN and an approximation factor $\epsilon = 3.16$.
	`MaxDensity`	Subsample to keep point with density of $5\,pts/m^3$.
DF_{ref}	`MaxDist`	Keep points within a radius of 40 m from the last sensor pose.
	`SurfaceNormal`	Compute normal and density with 20 NN and an approximation factor $\epsilon = 3.16$.
	`MaxDensity`	Subsample to keep point with density of $10\,pts/m^3$.
	`MaxPointCount`	Subsample 70 % if there is more than 900,000 points.
MF	`KDTree`	Use a kD-tree with a maximum matching distance of 1.5 m.
OF	`TrimmedDist`	Keep 70 % closest paired points.
EM	`PointToPlane`	Objective function using point-to-point error.
TC	`Differential`	Stop after a minimum error below 0.01 m and 0.001 rad.
	`Counter`	Stop after the iteration count reached 100.
	`Bound`	Stop if transformation increases beyond 5.0 m and 0.8 rad.

Legend:
DF_{read} = Data Filters for readings, DF_{ref} = Data Filters for references, MF = Matching Function, OF = Outlier Filters, EM = Error Minimizer, TC = Transformation Checker.

left corner of the aerial view. During the drive, a total of four loops were realized: two loops were made counterclockwise around the Eidgenössische Technische Hochschule (ETH) main building (*Hauptgebäude*), and two other loops clockwise around the Hospital (*Universitätsspital Zürich*). Even with the low density of point used, a reasonable amount of details could be preserved, as depicted in Figure 3.17, where even wires powering trams are visible over the street junction at the bottom of the image. At the first loop-closing point after 660 m, ICP-based odometry had accumulated an alignment error of 4.5 m on the z-axis. At the second loop-closing point after 920 m, the error on the z-axis was 14.3 m. In both cases, the error was recovered and registration could

proceed further without any problem It is interesting to note that, at those two closing points, errors on z-axis were the most predominant of the 6 DoF possible. When looking at each scan separately, we observed that the ground area displaying a usable density of points had a radius of 20 m around the car. Also, the distance between two scans was often 8 m, going as high as 17 m when accelerating. Urban environment are often referred as canyon-shaped, which means that the sides are well constrained by the buildings, but lacking points on the ground can lead to drift on pitch angle or/and on z-axis. The fast motion of the car coupled with the short range measurements gave little overlap to stabilize the elevation or the pitch angle and the error cumulated during each loop. When the car was passing again in streets previously explored, the global map was reused successfully by recovering the large offset at loop closure points. Although the 3D map mainly follows the aerial map, the combined impact of scanning rate and velocity of the platform brings the utility of laser odometry to its limit. This application results are the largest presented in this section, with a total path length of 3.8 km.

3.5 Summary

This section covered a wide range of applications that were performed by different types of robotic platforms. The key characteristics of the robots employed in those applications are recapitulated in Table 3.7. We presented registration utilizations in situation awareness for Search & Rescue activities. The feasibility of real-time mapping deployments was demonstrated in a confined staircase, in open outdoor areas and in a collapsed church. We also demonstrated that surveying images recorded by an aerial vehicle and scans from ground vehicle can be gathered to enhance a scene reconstruction of a heavily damaged deployment site. A solution based on compact inspection systems was also suggested for mapping unreachable components in power plants. Such solutions could help reducing costs, time and dangers for the operators by bringing 3D informations from inside the part inspected, without the need for complex structures to support the operators. Finally, large-

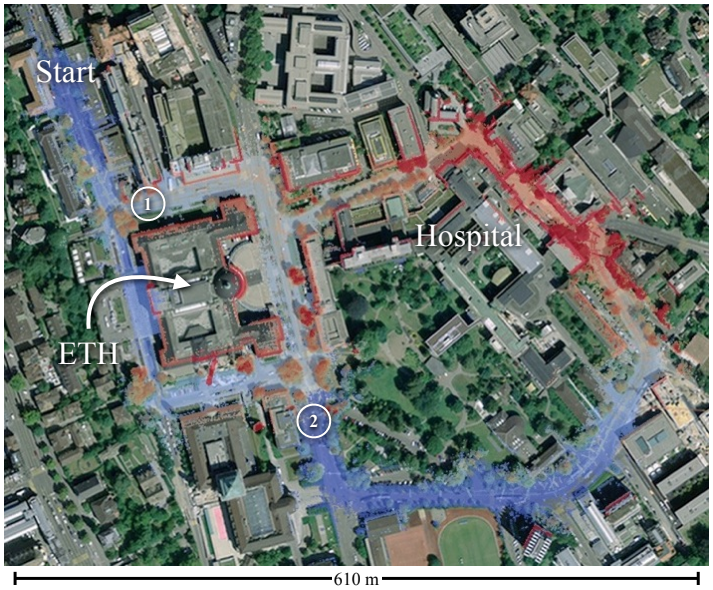

Figure 3.16: Overlay of a large scale reconstruction of the ETH main building and its surroundings with an aerial view. Colors represent the elevation, dark blue being low and dark red being high. Loop closing points are marked by numbers in circle, (1) being the small loop, (2) being the larger loop. The orthogonal projection of the aerial image was provided by the *Bundesamt für Landestopografie* swisstopo (Art. 30 GeoIV): 5704 000 000

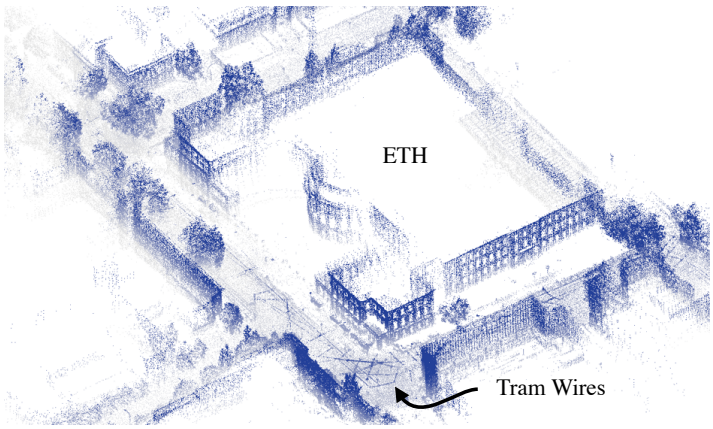

Figure 3.17: Bird's-eye view of the ETH main building. Colors represent elevation, light gray being low, dark blue being high.

scale environmental surveys were shown to be successful without the need for specific loop closure algorithms, whether the survey is on water or roads.

Table 3.7: Summary of the robot key characteristics influencing the proposed registration solutions.

	Magnebike	*NiftiBot*	*Lizhbeth*	*Artor*	*SmartTer*
Weight (kg)	0.34	20	120	300	730
Volume (m^3)	0.006	0.17	0.96	6.75	6.38
Speed (m/s)	0.045	0.3	0.7	1.2	4.17
Depth Sensor	URG-04LX	LMS-151	HDL-32E	HDL-32E	LMS-291
Nb of Points	340,000	55,000	45,000	50,000	28,000
Scan Rate (Hz)	0.02	0.35	1.6	5	1
Point Rate (Hz)	6,800	19,250	72,000	250,000	28,000

Based on the realized Search & Rescue deployments, we observed that most of the exploration activities are linear or expend following a star shape. Most of the tasks imply: (1) going somewhere where no direct sight is possible from a safe zone, (2) assessing the situation and damages, and (3) backtracking the robot's path to the control station. Those tasks rarely imply loop closing and a coherent representation will be enough to bring the robot back to the control station even with some drift. Although none of the applications described in this section used loop closure, the street survey with the SmartTer would not be the appropriate solution in its current form. It is a good example of the utility of error relaxation and loop closing. Nevertheless, having more accurate local registrations relaxes the pressure on loop closing, thus extending its reach to even larger loop.

Some lessons were learned while tuning the registration solutions for each one of the applications. One of the main parameter to tune first would be the maximal density required by the application. It can rapidly resolve real-time issues and remove local minima from the registration minimization. The maximal density and the maximum number of points in the global map depend on the expected size of the explored zone. For the specific case of exploration using NiftiBot, the fact that the sensor is 0.21 m above the ground greatly reduces the

range at which the density of the points is sufficient for motion planning on a cluttered floor. Also, speed reduction should automatically be applied when the elements around the robot get too close, e.g., when crossing doorways. This would avoid the rapid decrease of the overlap between scans, which would drag down the registration quality. As for Artor, its fast motion capability required the handling of real-time processing in priority. During tuning, the fact that 3D scans jumped in critical situations (e.g., quick rotations) was enough to break the chain of registrations. In the case of inspection using the Magnebike, our evaluations provided better specifications for the construction of a new compact rotating laser. This new sensor will rotate faster (10 rpm) and will read from a UTM-30LX instead of the more noisy URG-04LX. This should bring the quality of inspection closer to the goal of 0.01 m precision that is required for defect detection. The preliminary work on the autonomous surface vessel, Lizhbeth, lead to a prioritization on the orientation estimation. For next applications, we decided to replace the Velodyne by a cheaper UTM-30LX moved by a custom tilting mechanism, allowing variable vertical scanning angles. This would give faster scanning rates when the obstacles are far away, while having the possibility to scan overhead vegetation when close to the shore. Those advantages come at the expense of range. The lessons learned with this use case lead to the monitoring of lake shore over seasonal changes based on laser points and motivated a more accurate state estimation solution for autonomous surface vessel [Hitz et al., 2014]. The SmartTer project finished a few years ago, but it was nevertheless interesting to push the capabilities of local registration to its limit. One of main observation is that even though the SICK LMS-291 specifications mentioned a maximal range of 70 m, the usable range remains under 20 m on concrete roads. Velodyne sensors can exceed this reach by three to four times, which can lower the cumulated error over large distances. Although more powerful, Velodyne sensors produce dense reading that are shaped like concentric disks, which can create local minima when registered. Matching against a group of past scans, instead of only the last one, helps to overcome this problem. In our case, we matched against a global map (i.e., all the past scans), but the use of a tempo-

ral window or spacial window can keep the computation time constant instead of growing with the number of points [Pomerleau et al., 2014].

Based on the multiple use cases presented in this section, a number of similar registration challenges emerged. First, solutions can be specialized for a given *environment type*. Indoor environments are most of the time man-made and is constituted of many large surfaces easy to model. On the other side, environment with a large variety of structures or with complex and detailed surfaces (e.g., leafs on a tree) will lead to reconstruction errors. Material properties of a given environment will also be challenging when it comes to high reflectivity values. Certain type of applications need to be deployed in busy environment (i.e., *dynamic scene*). Pedestrians, cars and even wind on trees may alter the scene slightly generating larger registration error. Also, the *scene size* depend of the application. It is often limited the duration of a phenomena of interest, coverage speed of the robot and its power autonomy. The overlap between point clouds (i.e., the information relevant to the error minimization) is challenging to asses and to control in many applications. Occlusions caused obstacles in the environment can cause *variable overlaps* and in the worst cases, too *small overlaps* won't have enough information to ensure a proper registration. Key components of the overlap ratio are also the velocity of the robot, the scanning rate of the sensor and the sensor FoV. Applications guide also the *real-time* need for a registration algorithm. Surface reconstruction applications, where the visual quality or high precision is required, are often run offline. When it comes to user situation awareness or to robot motion influenced by its environment, a trade-off between accuracy and computation speed must be made. The sensor modality (e.g., triangulation, time-of-flight, phase-shift) selected influences the range, the uncertainty on depth measurements and other characteristics influencing the registration outcome. Applications limiting the robot size and power or the environment conditions (e.g., water, dust) reduce the sensor selection forcing registration solutions to cope with *sensor noise* larger than wanted. Finally, the quality of the *initial transformations* provided is highly influenced by the interaction between the environment and the robot. Forces lost can be challenging to estimate during navigation and

depend of the availability and accuracy of other sensors to roughly measure the pose of the robot. Table 3.8 relates all those challenges with the use cases presented in this section. This overview provides support to new applications by dressing a list of concrete elements one should be careful and linking them to real deployment scenarios.

All examples demonstrate the added value of a modular ICP chain as each application has a specific set of requirements, that can still be fulfilled with the same open-source ICP library. The text-based parameter configurations combined with visual debugging tools allowed us to rapidly tune and understand limitations of configurations in order to achieved fast and accurate solutions.

Table 3.8: Relation between the use cases presented in this section and common registration challenges. When applicable, elements listed after (+) improve the situation with respect to a given challenge, while elements after (−) degrade it. The color of a cell represents the qualitative level of registration difficulty, with red being critical, yellow being important and green being low.

Challenges	*Registration Use Cases*					
	Section 3.1	Section 3.1.4	Section 3.1.5	Section 3.2	Section 3.3	Section 3.4
	Search & Rescue	Collaborative Mapping	Artor	Inspection	Shoreline Monitoring	Autonomous Driving
Environment type	(−) thin walls, multiple ground elevation, large range of structures		(+) buildings, large open space; (−) vegetation	(+) planar structures, rusted surfaces (old pipes); (−) shiny surfaces (new pipes)	(+) buildings, large open space; (−) vegetation, water reflexions	(+) buildings, roads; (−) trees
Dynamic scene	(−) operators moving around		(−) operators moving around		(−) moored boats, water waves	(+) no traffic
Scene size	(+) limited by the navigation complexity	(−) large	(−) large	(+) small volumes to inspect	(−) large	(−) multiple roads
Variable overlap	(+) low robot speed; (−) turning around sharp corners, concealed areas, low scanning rate		(+) high scanning rate, open space	(+) scanned position manually controlled	(+) high scanning rate, open space	(−) large range of acceleration

Continued on next page...

Table 3.8: continued.

Challenges	*Registration Use Cases*					
	Section 3.1	Section 3.1.4	Section 3.1.5	Section 3.2	Section 3.3	Section 3.4
	Search & Rescue	Collaborative Mapping	Artor	Inspection	Shoreline Monitoring	Autonomous Driving
Small overlap	(−) turning sharp corners, concealed area, teleoperator focussing on safety instead of the robot mapping capability	(−) different types of locomotion, different areas covered	(+) high scanning rate	(+) driver focussing on the robot mapping capability	(+) high scanning rate	(−) fast vehicle coupled with low scanning rate
Real-time	(+) low scanning rate; (−) critical for teleoperation		(−) high scanning rate, quick attitude change	(+) slow robot motion	(−) high scanning rate, no prior on transformation	(+) low scanning rate, good prior on transformation; (−) large map
Sensor noise		(−) high noise on map from cameras		(−) compact sensor with high noise		
Initial tranformation noise	(−) high ground friction	(+) manually pre-aligned	(−) high ground friction, quick pitch motion	(−) no prior, full 3D path	(−) no prior	(+) precise wheel odometry
Point distribution	(−) sensor low on the ground		(−) sparse rings		(−) sparse rings	

4

Conclusion

In this review, we presented the problem of geometric registration and the classical ICP algorithm. They are common to a few research field but we focus on mobile robotics. Even then, the problem is still complex and presents multiple facets. There is a large number of publications with a lack of a general comparison methodology. This is understandable given the diversity of applications, characteristics of the sensors, motion capabilities of the robotic platform, characteristics of the environments, but it hinders the selection process of an appropriate instance of the algorithm. This review tackles this challenge by three means: (1) a wide literature review with a historical perspective, (2) the elaboration of a theoretical descriptive framework of geometric registration grounded in the literature review, and (3) the analysis of practical use cases covering a wide diversity of mobile robotics applications.

Literature review

The first contribution of this review is then an extensive literature review about registration problems. This problem is common to several research fields but as been separated in their own communities. The respective developments have focused on different constraints and fol-

lowed more or less independent pathways. For instance, it is manifest that medical imaging reported their advancements in a more structured manner than in robotics We believe that, after the exploratory bloom of new developments, robotics is now ripe for a consolidation of the algorithms.

This work proposes a formalization of geometric registration primary based on the literature of robotics. However, although we covered a considerable mount of publications, many more remain unlisted.

Theoretical framework

Our second contribution is a theoretical framework into which variations of geometric registration algorithms can be cast. On the one hand, it helps structure the analysis of a specific solution by providing a systematic grid to be filled from the description of algorithm. On the other hand, for a researcher facing a problem, it provides a directory of modules to choose from in order to assemble a specific solution to this problem. Instead of the complex process of designing ad-hoc variations of the ICP algorithm, we propose to select data filters, matching function, outlier filters, and error minimization according to the problem at hand.

This approach in itself does not solve the problem of the comparison of different modules. But instead of comparing full algorithms with several differences, we can assess the merit of each independent variation. In past works, we have proposed a methodology for this assessment [Pomerleau et al., 2013] based on real-world datasets [Pomerleau et al., 2012b]. Those works attempt to consolidate the advantages and disadvantages of different algorithms. It is expected that as the field matures more works will utilize this type of analysis.

Robotics application

As choosing from many modules still requires a good understanding and intuition of the algorithm, we have presented the design of solutions to several scenarios. The applications included search and rescue, power plant inspection, shoreline monitoring, and autonomous driving

with robots as diverse as a tracked or wheeled rovers, a magnetic two-wheeled mini-robot, a boat, and a car. The diversity of applications, velocities, environments, locomotion types, sensors, etc. provides a wide sample of use cases to serve as a guide for the reader own needs.

All those solutions were implemented using the same modular open source library[1], designed according to our theoretical framework. Many modules, defined in this review, have already been implemented. Despite its modular design, this library can be seen to provide strong computational performance in our applications.

Guidelines for solution implementations

Bringing the proposed framework of solutions presented in Section 2 with the main challenges highlighted in Section 3, Table 4.1 proposes a high level guideline based the experience developed over several years implementing, tuning and testing registration solutions. It should be use as an investigation starting point when problems arise from a given application. For a more quantitative interpretation of the impact of a specific module and its parameters on registration quality, we direct readers toward [Pomerleau et al., 2013]. Three modules from Section 2 are missing in the table as their selections mainly depends more on the context of an application than on registration tuning. *Reading and reference sources* follow the sensors at hand and can be upgraded based on their specifications but hardly simply for tuning a registration solution. *Transformation parameters* follow the data acquisition process and the deformation resulting from moving a given sensor in the environment. Lastly, *transformation checkers* depend on transformation parameters used and the application. Detecting convergence, divergence or timeout of the algorithm can be implemented in a generic way, but the actions resulting of such events (e.g., discarding the result, restarting with other parameters or another solution) depend on the application. Moreover, it worth noting that some suggestions are contradictory when addressing a given challenge. For example, the row *Real-time* highlights the fact that point-to-plane error has a faster convergence rate, but at the

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

same time, highlight that surface reconstruction is expensive to compute. The general registration strategy selected for an application may require to balance advantages and disadvantages of the proposed solutions. Table 4.1 is also an opening for future works, where the knowledge on a given cell can be enhanced by more researches on a specific sub-registration problem. Also, grey cells represent configurations that were not encountered in this work, which could be an opportunity for innovation.

With this work, we hope to provide a clear overview of geometric registration for mobile robotics, as well as a method to solve it in specific instances. We invite fellow researchers to describe and compare their algorithms following our framework, and even, when possible to contribute code to the open source library.

Table 4.1: Relations between the different framework modules presented in Section 2 and common challenges highlighted with the applications described in Section 3. Cell information proposes guidelines for readers facing similar challenges. Element *emphasized* correspond to specific cases. Shaded areas represent either no link or that more investigation is required.

Challenges	*Framework Modules*			
	Section 2.3	Section 2.4	Section 2.5	Section 2.6
	Data Filters	Association Solver	Outlier Filters	Error Minimization
Environment type	*Double sided surfaces*: extract normal vectors and orient them.	Augment the number of matches to cover noise in surface reconstruction.	Weight wrong surface normal associations.	*Unstructured environments*: point-to-point relies on less reconstruction assumptions.
Dynamic scene	Removed identified dynamic points (e.g., scanner self-reading). Use algorithms to identify dynamic points based on prior registrations.	Augment the number of matches to augment the change of matching with a static structure.	Weight association using an identified dynamic point.	Use weighted error metric emphasizing static associations.
Scene size	Down sample large point clouds. Roughly extract the zone of interest.			
Variable overlap		Avoid using distance approximations.	*Hard rejection*: fix threshold using adaptive techniques. *Quartile based solutions*: use the lowest expected overlap as a threshold.	
Small overlap	Apply filters with leading to a uniform sampling in Euclidean space.	Avoid using distance approximations.	*Hard rejection*: use a strict threshold.	

Continued on next page...

Table 4.1: continued.

Challenges	*Registration Use Cases*			
	Section 2.3	Section 2.4	Section 2.5	Section 2.6
Challenges	Data Filters	Association Solver	Outlier Filters	Error Minimization
Real-time	Reduce the number of points with random sampling (low computation footprint). Reduce the number of needed filters. Reduce the use of surface reconstruction filters based on NN search. Use filters relying on the assumption that the ordering of the data is known.	Use kD-tree or other space partitioning techniques. Use distance approximation. Limit the maximal search range. Use NN search relying on the assumption that the ordering of the data is known. Limit the search to one NN per points.		Remove any point with zero weight before minimizing the error. Point-to-plane error converge in fewer iterations than point-to-point.
Sensor noise	Compute point uncertainty based on specific sensor noise model. Handle shadow points.		Weight association based on the combination of both points uncertainties.	Use weighted error metric using sensor noise information.
Initial tranformation noise		Use distance metric relying on spaces independent from the transformation parameters (i.e., descriptors). Increase the number of matches per points.	Weight associations based more on descriptors than features.	Point-to-plane has a larger convergence space than point-to-point. Error metrics rely on entropy have a larger convergence space in general. Use weighted error metric dealing with match uncertainties.
Point distribution	Apply filters with leading to a uniform sampling in Euclidean space. Avoid using filters relying on fix distance thresholds.	Avoid search relying on a maximal search radius.	Avoid using threshold based on fix distances of associated points.	

Acknowledgements

This research was supported by the Natural Sciences and Engineering Research Council of Canada for the Postdoctoral Fellowships grant and the European Union FP7 program under the NIFTi (247870) and TRADR (609763) projects. We acknowledge the help of Andreas Breitenmoser, Gregory Hitz, Philipp Krüsi, Ming Liu, Luciano Spinello, Fabien Tâche, Rudolph Triebel from the Autonomous Systems Lab (ETH Zurich), our NIFTi partners in CMP (CVUT, Prague) and Alcor (La Sapienza, Rome), and our colleagues of the sFly project for the experimental data.

Appendices

A

Derivation for Point-to-Plane Error

This appendix presents a solution for minimizing the point-to-plane error in 3D. We first define our transformation parameter set $\mathcal{T}$ as a 6D vector:

$$\mathcal{T} = \boldsymbol{\tau} = \begin{bmatrix} \boldsymbol{r} \\ \boldsymbol{t} \end{bmatrix} = \begin{bmatrix} \alpha \\ \beta \\ \gamma \\ t_x \\ t_y \\ t_z \end{bmatrix}, \tag{A.1}$$

where α, β and γ are the rotational components, while t_x, t_y and t_z are the translation components. We also define the objective function for point-to-plane:

$$e_{\mathrm{p}\Phi} = \sum_{k=1}^{K} \left\| [(\boldsymbol{R}\boldsymbol{p}_k + \boldsymbol{t}) - \boldsymbol{q}_k] \cdot \boldsymbol{n}_k \right\|_2, \tag{A.2}$$

where $\boldsymbol{n}_k$ is the normal vector representing the surface at the point $\boldsymbol{q}_k$ and the index k represents paired points. The method presented here rely on rotation matrix linearization. This linearization can be achieved

using the small-angle approximation:

$$\boldsymbol{R} = R(\alpha, \beta, \gamma) \approx \begin{bmatrix} 1 & -\gamma & \beta \\ \gamma & 1 & -\alpha \\ -\beta & \alpha & 1 \end{bmatrix} = [\boldsymbol{r}]_\times + \boldsymbol{I}, \tag{A.3}$$

where $[\boldsymbol{r}]_\times$ is a cross-product operator transforming the vector $\boldsymbol{r}$ to a 3×3 skew-symmetric matrix. In the context of ICP, the impact of linearization is reduced through the iterative process of the whole registration algorithm. Combining Equation A.2 with Equation A.3, we can approximate the objective function as

$$\begin{aligned} e_{\mathrm{p}\Phi} &\approx \sum_{k=1}^{K} \|[([r]_\times + \boldsymbol{I})\boldsymbol{p}_k + \boldsymbol{t} - \boldsymbol{q}_k] \cdot \boldsymbol{n}_k\|_2 \\ &\approx \sum_{k=1}^{K} \|(\boldsymbol{r} \times \boldsymbol{p}_k) \cdot \boldsymbol{n}_k + \boldsymbol{p}_k \cdot \boldsymbol{n}_k + \boldsymbol{t} \cdot \boldsymbol{n}_k - \boldsymbol{q}_k \cdot \boldsymbol{n}_k\|_2 , \end{aligned}$$

which can be rewritten using the *scalar triple product* and by reorganizing the terms

$$\begin{aligned} e_{\mathrm{p}\Phi} &\approx \sum_{k=1}^{K} \left\| \boldsymbol{r} \cdot \underbrace{(\boldsymbol{p}_k \times \boldsymbol{n}_k)}_{\boldsymbol{c}_k} + \boldsymbol{t} \cdot \boldsymbol{n}_k - \underbrace{(\boldsymbol{q}_k - \boldsymbol{p}_k)}_{\boldsymbol{d}_k} \cdot \boldsymbol{n}_k \right\|_2 \\ &\approx \sum_{k=1}^{K} \|\boldsymbol{r} \cdot \boldsymbol{c}_k + \boldsymbol{t} \cdot \boldsymbol{n}_k - \boldsymbol{d}_k \cdot \boldsymbol{n}_k\|_2 , \end{aligned}$$

We can then minimize the error $e_{\mathrm{p}\Phi}$ with respect to $\boldsymbol{r}$ and $\boldsymbol{t}$ and setting the partial derivatives to zero

$$\begin{aligned} \frac{\partial e_{\mathrm{p}\Phi}}{\partial \boldsymbol{r}} &= \sum_{k=1}^{K} 2\boldsymbol{c}_k(\boldsymbol{r} \cdot \boldsymbol{c}_k + \boldsymbol{t} \cdot \boldsymbol{n}_k - \boldsymbol{d}_k \cdot \boldsymbol{n}_k) = \boldsymbol{0} \\ \frac{\partial e_{\mathrm{p}\Phi}}{\partial \boldsymbol{t}} &= \sum_{k=1}^{K} 2\boldsymbol{n}_k(\boldsymbol{r} \cdot \boldsymbol{c}_k + \boldsymbol{t} \cdot \boldsymbol{n}_k - \boldsymbol{d}_k \cdot \boldsymbol{n}_k) = \boldsymbol{0} \end{aligned}$$

We can assemble those derivative under the linear form $\boldsymbol{A\tau} = \boldsymbol{b}$, by bringing the independent variables on the right side of the equation

$$\sum_{k=1}^{K} \begin{bmatrix} \boldsymbol{c}_k(\boldsymbol{r}\cdot\boldsymbol{c}_k) + \boldsymbol{c}_k(\boldsymbol{t}\cdot\boldsymbol{n}_k) \\ \boldsymbol{n}_k(\boldsymbol{r}\cdot\boldsymbol{c}_k) + \boldsymbol{n}_k(\boldsymbol{t}\cdot\boldsymbol{n}_k) \end{bmatrix} = \sum_{k=1}^{K} \begin{bmatrix} \boldsymbol{c}_k(\boldsymbol{d}_k\cdot\boldsymbol{n}_k) \\ \boldsymbol{n}_k(\boldsymbol{d}_k\cdot\boldsymbol{n}_k) \end{bmatrix}$$

$$\sum_{k=1}^{K} \begin{bmatrix} \boldsymbol{c}_k\boldsymbol{c}_k^\top\boldsymbol{r} + \boldsymbol{c}_k\boldsymbol{n}_k^\top\boldsymbol{t} \\ \boldsymbol{n}_k\boldsymbol{c}_k^\top\boldsymbol{r} + \boldsymbol{n}_k\boldsymbol{n}_k^\top\boldsymbol{t} \end{bmatrix} = \sum_{k=1}^{K} \begin{bmatrix} \boldsymbol{c}_k(\boldsymbol{d}_k\cdot\boldsymbol{n}_k) \\ \boldsymbol{n}_k(\boldsymbol{d}_k\cdot\boldsymbol{n}_k) \end{bmatrix}$$

$$\sum_{k=1}^{K} \begin{bmatrix} \boldsymbol{c}_k\boldsymbol{c}_k^\top & \boldsymbol{c}_k\boldsymbol{n}_k^\top \\ \boldsymbol{n}_k\boldsymbol{c}_k^\top & \boldsymbol{n}_k\boldsymbol{n}_k^\top \end{bmatrix} \begin{bmatrix} \boldsymbol{r} \\ \boldsymbol{t} \end{bmatrix} = \sum_{k=1}^{K} \begin{bmatrix} \boldsymbol{c}_k \\ \boldsymbol{n}_k \end{bmatrix} (\boldsymbol{d}_k\cdot\boldsymbol{n}_k)$$

which brings us to the linear system of equations that we were looking for

$$\underbrace{\sum_{k=1}^{K} \begin{bmatrix} \boldsymbol{c}_k \\ \boldsymbol{n}_k \end{bmatrix} \begin{bmatrix} \boldsymbol{c}_k^\top & \boldsymbol{n}_k^\top \end{bmatrix}}_{\boldsymbol{A}_{6\times 6}} \boldsymbol{\tau} = \underbrace{\sum_{k=1}^{K} \begin{bmatrix} \boldsymbol{c}_k \\ \boldsymbol{n}_k \end{bmatrix} (\boldsymbol{d}_k\cdot\boldsymbol{n}_k)}_{\boldsymbol{b}_{6\times 1}} \tag{A.4}$$

Once the matrix $\boldsymbol{A}$ and the vector $\boldsymbol{b}$ can be constructed, the linear system of Equation A.4 can be resolved for $\boldsymbol{\tau}$ using the Cholesky decomposition. Implementing such solution will require a loop for the summations over K to build $\boldsymbol{A}$ and $\boldsymbol{b}$. An alternative formulation relying on dense matrix multiplication can be computed by assembling

$$\boldsymbol{G} = \underbrace{\begin{bmatrix} \cdots & \boldsymbol{p}_k \times \boldsymbol{n}_k & \cdots \\ & \boldsymbol{n}_k & \end{bmatrix}}_{6\times K}$$

and

$$\boldsymbol{h} = \underbrace{\begin{bmatrix} \vdots \\ (\boldsymbol{q}_k - \boldsymbol{p}_k)\cdot\boldsymbol{n}_k \\ \vdots \end{bmatrix}}_{K\times 1}$$

leading to

$$\boldsymbol{A\tau} = \boldsymbol{b}$$
$$\Updownarrow$$
$$\boldsymbol{G}\boldsymbol{G}^\top\boldsymbol{\tau} = \boldsymbol{G}\boldsymbol{h},$$

which is the same formulation as proposed in Section 2.6.2 □

References

Francesco Amigoni, Monica Reggiani, and Viola Schiaffonati. An insightful comparison between experiments in mobile robotics and in science. *Autonomous Robots*, 27(4):313–325, August 2009.

Leopoldo Armesto, Javier Minguez, and Luis Montesano. A generalization of the metric-based Iterative Closest Point technique for 3D scan matching. In *Robotics and Automation, 2010. Proceedings of the IEEE International Conference on*, pages 1367–1372, 2010.

K S Arun, T S Huang, and S D Blostein. Least-Squares Fitting of Two 3-D Point Sets. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 9(5):698–700, 1987.

Sunil Arya and David Mount. Approximate nearest neighbor queries in fixed dimensions. In *Discrete Algorithms, 1993. Proceedings of the 4th Annual ACM-SIAM Symposium on*, pages 271–280. Department of Computer Science, University of Maryland, College Park, Maryland, 20742 Department of Computer Science and Institute for Advanced Computer Studies, University of Maryland, College Park, Mary- land, 20742, 1993.

R Benjemaa and F Schmitt. Fast global registration of 3D sampled surfaces using a multi-z-buffer technique. In *3-D Digital Imaging and Modeling, 1997. Proceedings of the International Conference on Recent Advances in*, pages 113–120, 1997.

R Bergevin, M Soucy, H Gagnon, and D Laurendeau. Towards a general multiview registration technique. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 18(5):540–547, 1996.

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

P J Besl. Geometric modeling and computer vision. In *Proceedings of the IEEE*, pages 936–958, 1988.

Michael Bosse and R Zlot. Map Matching and Data Association for Large-Scale Two-dimensional Laser Scan-based SLAM. *The International Journal of Robotics Research*, 27(6):667–691, June 2008.

Michael Bosse and R Zlot. Keypoint design and evaluation for place recognition in 2D lidar maps. *Robotics and Autonomous Systems*, 57(12):1211–1224, 2009a.

Michael Bosse and Robert Zlot. Continuous 3D scan-matching with a spinning 2D laser. In *Robotics and Automation, 2009. Proceedings of the IEEE International Conference on*, pages 4312–4319, 2009b.

Michael Bosse, Paul Newman, John Leonard, and Seth Teller. Simultaneous Localization and Map Building in Large-Scale Cyclic Environments Using the Atlas Framework. *The International Journal of Robotics Research*, 23 (12):1113–1139, December 2004.

Kevin W Bowyer, Kyong Chang, and Patrick Flynn. A survey of approaches and challenges in 3D and multi-modal 3D+2D face recognition. *Computer Vision And Image Understanding*, 101(1):1–15, January 2006.

A Censi. Scan matching in a probabilistic framework. In *Robotics and Automation, 2006. Proceedings of the IEEE International Conference on*, pages 2291–2296, 2006.

A Censi. An ICP variant using a point-to-line metric. In *Robotics and Automation, 2008. Proceedings of the IEEE International Conference on*, pages 19–25, 2008.

G Champleboux, S Lavallee, R Szeliski, and L Brunie. From accurate range imaging sensor calibration to accurate model-based 3D object localization. In *Computer Vision and Pattern Recognition, 1992. Proceedings of the 1992 IEEE Computer Society Conference on*, pages 83–89, 1992.

Y Chen and G Medioni. Object modeling by registration of multiple range images. In *Robotics and Automation, 1991. Proceedings of the IEEE International Conference on*, pages 2724–2729. IEEE Comput. Soc. Press, April 1991.

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

D Chetverikov, D Svirko, D Stepanov, and P Krsek. The Trimmed Iterative Closest Point algorithm. In *Pattern Recognition, 2002. Proceedings of the 16th International Conference on*, pages 545–548, 2002.

Francis Colas, Srivatsa Mahesh, François Pomerleau, Ming Liu, and Roland Siegwart. 3D path planning and execution for search and rescue ground robots. In *Intelligent Robots and Systems (IROS), 2013 IEEE/RSJ International Conference on*. IEEE, 2013.

J Diebel, K Reutersward, S Thrun, J Davis, and R Gupta. Simultaneous localization and mapping with active stereo vision. In *Intelligent Robots and Systems, 2004. Proceedings of the IEEE/RSJ International Conference on*, pages 3436–3443 vol.4, 2004.

S Druon, M J Aldon, and A Crosnier. Color constrained ICP for registration of large unstructured 3d color data sets. In *Information Acquisition, 2006. Proceedings of the IEEE International Conference on*, pages 249–255. IEEE, 2006.

David W Eggert, Adele Lorusso, and Robert B Fisher. Estimating 3-D rigid body transformations: A comparison of four major algorithms. *Machine Vision and Applications*, 9(5-6):272–290, 1997.

David W Eggert, Andrew W Fitzgibbon, and Robert B Fisher. Simultaneous Registration of Multiple Range Views for Use in Reverse Engineering of CAD Models. *Computer Vision And Image Understanding*, 69(3):253–272, February 1998.

N Fairfield and D Wettergreen. Evidence grid-based methods for 3D map matching. In *Robotics and Automation, 2009. Proceedings of the IEEE International Conference on*, pages 1637–1642, May 2009.

O D Faugeras and M Hebert. The Representation, Recognition, and Locating of 3-D Objects. *The International Journal of Robotics Research*, 5(3):27–52, September 1986.

J Feldmar and N Ayache. Locally affine registration of free-form surfaces. In *Computer Vision and Pattern Recognition, 1994. Proceedings of the IEEE Computer Society Conference on*, pages 496–501, 1994.

Jacques Feldmar and Nicholas Ayache. Rigid, affine and locally affine registration of free-form surfaces. *International Journal of Computer Vision*, 18 (2):99–119, May 1996.

O Fluck, C Vetter, W Wein, A Kamen, B Preim, and R Westermann. A survey of medical image registration on graphics hardware. *Computer Methods and Programs in Biomedicine*, 104(3):e45–e57, December 2011.

H Gagnon, M Soucy, R Bergevin, and D Laurendeau. Registration of multiple range views for automatic 3-D model building. In *Computer Vision and Pattern Recognition, 1994. Proceedings of the IEEE Computer Society Conference on*, pages 581–586, 1994.

N Gelfand, L Ikemoto, S Rusinkiewicz, and M Levoy. Geometrically stable sampling for the ICP algorithm. In *3-D Digital Imaging and Modeling, 2003. Proceedings of the Fourth International Conference on*, pages 260–267, 2003.

G Godin, M Rioux, and R Baribeau. Three-dimensional registration using range and intensity information. In *Videometric III. Proceedings of SPIE Conference on*, pages 279–290, 1994.

Steven Gold, Anand Rangarajan, Chien-Ping Lu, Suguna Pappu, and Eric Mjolsness. New algorithms for 2D and 3D point matching: pose estimation and correspondence. *Pattern Recognition*, 31(8):1019–1031, August 1998.

Giorgio Grisetti, C Stachniss, and W Burgard. Improving Grid-based SLAM with Rao-Blackwellized Particle Filters by Adaptive Proposals and Selective Resampling. In *Robotics and Automation, 2005. Proceedings of the IEEE International Conference on*, pages 2432–2437, 2005.

Giorgio Grisetti, S Grzonka, C Stachniss, P Pfaff, and W Burgard. Efficient estimation of accurate maximum likelihood maps in 3D. In *Intelligent Robots and Systems, 2007. Proceedings of the IEEE/RSJ International Conference on*, pages 3472–3478, 2007.

P Henry, M Krainin, E Herbst, X Ren, and D Fox. RGB-D mapping: Using Kinect-style depth cameras for dense 3D modeling of indoor environments. *The International Journal of Robotics Research*, 31(5):647–663, April 2012.

Gregory Hitz, François Pomerleau, Marie-Eve Garneau, Cedric Pradalier, Thomas Posch, Jakob Pernthaler, and Ronald Siegwart. Autonomous Inland Water Monitoring: Design and Application of a Surface Vessel. *Robotics & Automation Magazine, IEEE*, 19(1):62–72, March 2012.

Gregory Hitz, François Pomerleau, Francis Colas, and Roland Siegwart. State Estimation for Shore Monitoring Using an Autonomous Surface Vessel. In *International Symposium on Experimental Robotics (ISER)*, pages 1–15, Marrakech/Essaouira, June 2014.

Berthold K P Horn. Closed-form solution of absolute orientation using unit quaternions. *Journal of the Optical Society of America A*, 4(4):629, 1987.

Berthold K P Horn, Hugh M Hilden, and Shahriar Negahdaripour. Closed-form solution of absolute orientation using orthonormal matrices. *Journal of the Optical Society of America A*, 5(7):1127–1135, 1988.

John R Hurley and Raymond B Cattell. The Procrustes program: Producing direct rotation to test a hypothesized factor structure. *Behavioral Science*, 7(2):258–262, April 1962.

A E Johnson and Sing Bing Kang. Registration and integration of textured 3-D data. In *3-D Digital Imaging and Modeling, 1997. Proceedings of the International Conference on Recent Advances in*, pages 234–241, 1997.

T Jost and H Hügli. Fast ICP algorithms for shape registration. In *Pattern Recognition, 2002. Proceedings of the 24th DAGM German Symposium on*, pages 91–99. Pattern Recognition Group, Institute of Microtechnology, University of Neuchâtel, Breguet 2, CH-2000 Neuchâtel, Switzerland, 2002.

T Jost and H Hugli. A multi-resolution scheme ICP algorithm for fast shape registration. In *3D Data Processing Visualization and Transmission, 2002. Proceedings of the First International Symposium on*, pages 540–543, 2002.

T Jost and H Hugli. A multi-resolution ICP with heuristic closest point search for fast and robust 3D registration of range images. In *3-D Digital Imaging and Modeling, 2003. Proceedings of the Fourth International Conference on*, pages 427–433, 2003.

D Kim. A Fast ICP Algorithm for 3-D Human Body Motion Tracking. *Signal Processing Letters, IEEE*, 17(4):402–405, 2010.

G J M Kruijff, M Janicek, S Keshavdas, B Larochelle, H Zender, N J J M Smets, T Mioch, M A Neerincx, J van Diggelen, Francis Colas, M Liu, F Pomerleau, Roland Siegwart, V Hlavac, T Svoboda, T Petricek, M Reinstein, K Zimmerman, F Pirri, M Gianni, P Papadakis, A Sinha, P Balmer, N Tomatis, R Worst, T Linder, H Surmann, V Tretyakov, H Surmann, S Corrao, S Pratzler-Wanczura, and M Sulk. Experience in System Design for Human-Robot Teaming in Urban Search and Rescue. In *Field and Service Robotics*, pages 1–14, Matsushima, Japan, July 2012.

Vladimír Kubelka, Lorenz Oswald, François Pomerleau, Francis Colas, Tomáš Svoboda, and Michal Reinstein. Robust data fusion of multi-modal sensory information for mobile robots. *Journal of Field Robotics*, in press, 2014.

P Kumari, R Shrestha, and B Carter. Registration of LiDAR data through stable surface matching. In *Geoinformatics, 2009. Proceedings of the 17th International Conference on*, pages 1–5, 2009.

P Lamon, S Kolski, and Roland Siegwart. The SmartTer-a vehicle for fully autonomous navigation and mapping in outdoor environments. In *Proceedings of CLAWAR*, 2006.

Ce Li, Jianru Xue, Shaoyi Y Du, and Nanning Zheng. A Fast Multi-Resolution Iterative Closest Point Algorithm. In *Pattern Recognition, 2010. Proceedings of the Chinese Conference on*, pages 1–5, 2010.

Yonghuai Liu. Automatic Range Image Registration in the Markov Chain. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 32(1): 12–29, 2010.

D Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110, 2004.

F Lu and E Milios. Globally Consistent Range Scan Alignment for Environment Mapping. *Autonomous Robots*, 4(4):333–349, 1997.

M Magnusson, Andreas Nüchter, C Lorken, A Lilienthal, and J Hertzberg. Evaluation of 3D registration reliability and speed - A comparison of ICP and NDT. In *Robotics and Automation, 2009. Proceedings of the IEEE International Conference on*, pages 3907–3912, 2009.

J B Antoine Maintz and Max A Viergever. A survey of medical image registration. *Medical Image Analysis*, 2(1):1–36, March 1998.

P Markelj, D Tomaževič, B Likar, and F Pernuš. A review of 3D/2D registration methods for image-guided interventions. *Medical Image Analysis*, 16(3):642–661, April 2012.

T Masuda. Generation of geometric model by registration and integration of multiple range images. In *3-D Digital Imaging and Modeling, 2001. Proceedings of the Third International Conference on*, pages 254–261, 2001.

T Masuda, K Sakaue, and N Yokoya. Registration and integration of multiple range images for 3-D model construction. In *Pattern Recognition, 1996. Proceedings of the 13th International Conference on*, pages 879–883 vol.1, 1996.

C R Jr Maurer, G B Aboutanos, B M Dawant, R J Maciunas, and J M Fitzpatrick. Registration of 3-D images using weighted geometrical features. *Medical Imaging, IEEE Transactions on*, 15(6):836–849, 1996.

G Medioni, C K Tang, and M S Lee. Tensor voting: Theory and applications. In *Reconnaissance des formes et Intelligence Artificielle, 2000. Proceedings of the Conference on*, 2000.

Helena Mitasova, Margery F Overton, Juan José Recalde, David J Bernstein, and Christopher W Freeman. Raster-Based Analysis of Coastal Terrain Dynamics from Multitemporal Lidar Data. *Journal of Coastal Research*, 252:507–514, March 2009.

E Mortensen, Hongli Deng, and L Shapiro. A SIFT descriptor with global context. In *Computer Vision and Pattern Recognition, 2005. Proceedings of the IEEE Computer Society Conference on*, pages 184–190 vol. 1, 2005.

Andreas Nüchter, H Surmann, K Lingemann, J Hertzberg, and S Thrun. 6D SLAM with an application in autonomous mine mapping. In *Robotics and Automation, 2004. Proceedings of the IEEE International Conference on*, pages 1998–2003 Vol.2, 2004.

Andreas Nüchter, K Lingemann, J Hertzberg, and H Surmann. 6D SLAM with approximate data association. In *Advanced Robotics, 2005. Proceedings of the 12th International Conference on*, pages 242–249, 2005.

Andreas Nüchter, K Lingemann, and J Hertzberg. Cached k-d tree search for ICP algorithms. In *3-D Digital Imaging and Modeling, 2007. Proceeding of the Sixth International Conference on*, pages 419–426, 2007.

Ye Pan, Bo Dai, and Qicong Peng. Fast and robust 3D face matching approach. In *Image Analysis and Signal Processing, 2010. Proceedings of the Second International Conference on*, pages 195–198, 2010.

S T Pfister, K L Kriechbaum, S I Roumeliotis, and J W Burdick. Weighted range sensor matching algorithms for mobile robot displacement estimation. In *Robotics and Automation, 2002. Proceedings of the IEEE International Conference on*, 2002.

J P W Pluim, J B A Maintz, and M A Viergever. Mutual-information-based registration of medical images: a survey. *Medical Imaging, IEEE Transactions on*, 22(8):986–1004, 2003.

François Pomerleau, Francis Colas, François Ferland, and F Michaud. Relative motion threshold for rejection in ICP registration. *Field and Service Robotics*, pages 229–238, 2010.

François Pomerleau, Stéphane Magnenat, Francis Colas, Ming Liu, and Roland Siegwart. Tracking a depth camera: Parameter exploration for fast ICP. In *Intelligent Robots and Systems, 2011. Proceedings of the IEEE/RSJ International Conference on*, pages 3824–3829, 2011.

François Pomerleau, Andreas Breitenmoser, Ming Liu, Francis Colas, and Roland Siegwart. Noise Characterization of Depth Sensors for Surface Inspections. In *Applied Robotics for the Power Industry, 2012. Proceedings of the Second International Conference on*, pages 1–6, Zurich, Switzerland, August 2012a.

François Pomerleau, M. Liu, Francis Colas, and Roland Siegwart. Challenging Data Sets for Point Cloud Registration Algorithms. *The International Journal of Robotics Research*, September 2012b.

François Pomerleau, Francis Colas, Roland Siegwart, and Stéphane Magnenat. Comparing ICP Variants on Real-World Data Sets. *Autonomous Robots*, 34(3):133–148, February 2013.

François Pomerleau, Philipp Krüsi, Paul Furgale, and Roland Siegwart. Long-term 3D map maintenance in dynamic environments. In *2014 IEEE International Conference on Robotics and Automation (ICRA 2014)*, pages 1–8, February 2014.

K Pulli. Multiview registration for large data sets. In *3-D Digital Imaging and Modeling, 1999. Proceedings of the Second International Conference on*, pages 160–168, October 1999.

L Reyes, G Medioni, and E Bayro. Registration of 3D points using geometric algebra and tensor voting. *International Journal of Computer Vision*, 2007.

S Rusinkiewicz and M Levoy. Efficient variants of the ICP algorithm. In *3-D Digital Imaging and Modeling, 2001. Proceedings of the Third International Conference on*, pages 145–152, 2001.

J Salvi, C Matabosch, D Fofi, and J Forest. A review of recent range image registration methods with accuracy evaluation. *Image And Vision Computing*, 2007.

C Schutz, T Jost, and H Hugli. Multi-feature matching algorithm for free-form 3D surface registration. In *Pattern Recognition, 1998. Proceedings of the 14th International Conference on*, pages 982–984 vol.2, 1998.

Aleksandr Segal, Dirk Haehnel, and Sebastian Thrun. Generalized-ICP. In *Robotics: Science and Systems V. Proceedings of the Conference on*, page 21. Stanford University, 2009.

L Silva, O Bellon, and K Boyer. Precision range image registration using a robust surface interpenetration measure and enhanced genetic algorithms. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 27(5): 762–776, 2005.

C Stewart, Chia-Ling Tsai, and B Roysam. The dual-bootstrap iterative closest point algorithm with application to retinal image registration. *Medical Imaging, IEEE Transactions on*, 22(11):1379–1394, 2003.

E Stumm, A Breitenmoser, François Pomerleau, C Pradalier, and Roland Siegwart. Tensor-voting-based navigation for robotic inspection of 3D surfaces using lidar point clouds. *The International Journal of Robotics Research*, 31(12):1465–1488, November 2012.

Fabien Tâche, Wolfgang Fischer, Gilles Caprari, Roland Siegwart, Roland Moser, and Francesco Mondada. Magnebike: A magnetic wheeled robot with high mobility for inspecting complex-shaped structures. *Journal of Field Robotics*, 26(5), May 2009.

Fabien Tâche, François Pomerleau, W Fischer, Gilles Caprari, F Mondada, R Moser, and Roland Siegwart. MagneBike: Compact magnetic wheeled robot for power plant inspection. In *Applied Robotics for the Power Industry, 2010. Proceedings of the First International Conference on*, pages 1–2, 2010.

Fabien Tâche, François Pomerleau, Gilles Caprari, Roland Siegwart, Michael Bosse, and Roland Moser. Three-Dimensional Localization for the MagneBike Inspection Robot. *Journal of Field Robotics*, 28(2):180–203, 2011.

Sebastian Thrun, Wolfram Burgard, and Dieter Fox. A Probabilistic Approach to Concurrent Mapping and Localization for Mobile Robots. *Machine Learning*, 31(1/3):29–53, 1998.

C Tsai, C Li, G Yang, and K Lin. The Edge-Driven Dual-Bootstrap Iterative Closest Point Algorithm for Registration of Multimodal Fluorescein Angiogram Sequence. *Medical Imaging, IEEE Transactions on*, 29(3):636–649, 2010.

T Tuytelaars and K Mikolajczyk. A survey on local invariant features. *Foundations and Trends in Computer Graphics and Vision*, 2008.

Michael W Walker, Lejun Shao, and Richard A Volz. Estimating 3-D location parameters using dual number quaternions. *CVGIP: Image Understanding*, 54(3):358–367, November 1991.

O Wulf, Andreas Nüchter, J Hertzberg, and B Wagner. Benchmarking urban six-degree-of-freedom simultaneous localization and mapping. *Journal of Field Robotics*, 25(3):148–163, 2008.

Kai M Wurm, Armin Hornung, Maren Bennewitz, Cyrill Stachniss, and Wolfram Burgard. OctoMap: A Probabilistic, Flexible, and Compact 3D Map Representation for Robotic Systems. In *Workshop on Best Practice in 3D Perception and Modeling for Mobile Manipulation, 2010 (ICRA)*, 2010.

H Yoshitaka, K Hirohiko, O Akihisa, and Y Shin'ichi. Mobile Robot Localization and Mapping by Scan Matching using Laser Reflection Intensity of the SOKUIKI Sensor. In *Industrial Electronics, 2006. Proceedings of the IEEE 32nd Annual Conference on*, pages 3018–3023, 2006.

Zhengyou Zhang. Point matching for registration of free-form surfaces. *Lecture Notes in Computer Science*, 719:460–467, May 1993.

Zhengyou Zhang. Iterative point matching for registration of free-form curves and surfaces. *International Journal of Computer Vision*, 13(2):119–152, October 1994.

T Zinsser, J Schmidt, and H Niemann. A refined ICP algorithm for robust 3-D correspondence estimation. In *Image Processing, 2003. Proceedings of the IEEE International Conference on*, pages II–695–8 vol.3, 2003.

Barbara Zitová and Jan Flusser. Image registration methods: a survey. *Image And Vision Computing*, 21(11):977–1000, 2003.

R Zlot and Michael Bosse. Place recognition using keypoint similarities in 2D lidar maps. *Experimental Robotics*, 54:363–372, 2009.