Robotica (2014) volume 32, pp. 19–41. © Cambridge University Press 2013
doi:10.1017/S026357471300060X

Two different tools for three-dimensional mapping: DE-based scan matching and feature-based loop detection

Fernando Martín†*, Rudolph Triebel‡, Luis Moreno†
and Roland Siegwart§

†*Systems Engineering and Automation, Carlos III University, Madrid, Spain*
‡*Autonomous Systems Lab, ETH Zurich, Switzerland (currently at Department of Computer Science, Technical University of Munich, Germany)*
§*Autonomous Systems Lab, ETH Zurich, Switzerland*

(Accepted May 23, 2013. First published online: July 19, 2013)

SUMMARY

An autonomous robot must obtain information about its surroundings to accomplish multiple tasks that are greatly improved when this information is efficiently incorporated into a map. Some examples are navigation, manipulation, localization, etc. This mapping problem has been an important research area in mobile robotics during last decades. It does not have a unique solution and can be divided into multiple sub-problems. Two different aspects of the mobile robot mapping problem are addressed in this work. First, we have developed a Differential Evolution-based scan matching algorithm that operates with high accuracy in three-dimensional environments. The map obtained by an autonomous robot must be consistent after registration. It is basic to detect when the robot is navigating around a previously visited place in order to minimize the accumulated error. This phase, which is called loop detection, is the second aspect studied here. We have developed an algorithm that extracts the most important features from two different three-dimensional laser scans in order to obtain a loop indicator that is used to detect when the robot is visiting a known place. This approach allows the introduction of very different characteristics in the descriptor. First, the *surface* features include the geometric forms of the scan (lines, planes, and spheres). Second, the *numerical* features are values that describe several numerical properties of the measurements: volume, average range, curvature, etc. Both algorithms have been tested with real data to demonstrate that these are efficient tools to be used in mapping tasks.

KEYWORDS: Loop detection; Scan matching; 6D SLAM; Differential evolution; Feature descriptor.

1. Introduction

The purpose of an autonomous robot is to coexist with human beings in the same environment. Both humans and robots can utilize a map to accomplish multiple tasks. However, an important skill that should be implemented in a mobile robot is how to interpret the information given by its sensors in order to build or update its representation of the world. The mapping problem, which is one of the most important challenges in mobile robotics, can be included in this tool. The goal is that an autonomous robot can build a map and localize itself in it. The map can be metric or topological. The metric maps consider an n-dimensional space in which the objects are placed with their coordinates relative to the map. The topological approach only considers places and relations between them. The model consists of a graph with nodes (places) and arcs (paths). The metric maps are addressed in this paper.

This mapping concept is closely related to the well-known Simultaneous Localization and Mapping (SLAM) problem, originally developed by Leonard and Durrant-Whyte[14] based on an earlier work

* Corresponding author. E-mail: fmmonar@ing.uc3m.es

by Smith *et al.*[23] This is because map-learning cannot be separated from localization. The SLAM problem for mobile robots consists of building a map of an unknown environment while exploring the environment at the same time, using this map.

Therefore, the main task to address consists of the generation of consistent and robust maps considering the available information. This information is usually provided by two sources: proprioceptive sensors, such as wheel encoders with movement information, and exteroceptive sensors, such as laser range finders receiving environment information. This task can be divided into the following several steps:

1. *Robot's pose estimation*: The robot's pose (robot's position and orientation) with respect to the last one must be computed. The movement information has to be updated.
2. *Pose correction via registration*: The movement error has to be corrected using the exteroceptive sensor information. It is crucial to obtain a good estimate of the metric relation between different scans, which is often called registration or scan matching. Our robot works using a three-dimensional (3D) laser range finder.
3. *Loop detection and loop closure*: The model obtained is not consistent after registration because the accumulated error due to local small errors can be still very important. It is basic to detect when the robot is navigating around a previously visited place to check the global error and to minimize it to give consistency to the global map. The detection task is usually referred to as loop detection, and the global error minimization is called loop closure. There are several algorithms that correct the pose errors after registration.[12,25] However, the loop detection task is still an open problem in mobile robotics.

Our work focuses on two aspects of the mapping process: registration and loop detection.

We have implemented a scan matching algorithm for 3D environments. It is based on the Differential Evolution (DE) method, developed by Storn and Price,[24] which is a particle-based evolutionary algorithm that evolves in time to the solution that yields the cost function lower value. If the cost function is properly chosen, it is possible to solve the scan matching problem using this method. The high accuracy and computational efficiency of the proposed method have been demonstrated with experimental results.

The loop detection problem is also addressed in this paper. We have developed a loop detection method that compares features extracted from two different scans to obtain a loop indicator. It has been called Loop Probability Indicator (LPI). This approach allows the introduction of very different characteristics in the descriptor. First, the *surface* features include the geometric forms of the scan (lines, planes, and spheres). Second, the *numerical* features describe several numerical properties: volume, average range, curvature, and so on. The algorithm has been tested with real data to demonstrate that it is an efficient tool to be used in mapping problems. All true loops are correctly detected and no false detections are appreciated when the mobile robot is covering a long trajectory. The results are similar or even better than those obtained by other research groups. The introduction of different types of features and weights in the comparison formula, and the uncertainty band improve the algorithm performance and make it a more versatile method because it admits different settings.

This paper is organized as follows. The related work is reviewed in Section 2. The mapping process is introduced in Section 3. After that the registration method is detailed in Section 4. In Section 5, the loop detection algorithm is explained. The experimental results are presented in Section 6, and finally, the most important conclusions are summarized in Section 7.

2. Related Work

SLAM or mapping is a very common topic in mobile robotics. Depending on the map dimensions, there are planar and volumetric maps. The interest about 3D scans in mapping has increased recently due to the availability of efficient 3D sensors.

Some research groups have simplified the volumetric mapping by using 2D laser range finders instead of 3D scanners. This idea has been applied by Thrun *et al.*[27] and Früh and Zakhor.[10] Other groups have developed rotating scanners for the same purpose. Wulf *et al.*[31] utilize a scanner that rotates around the vertical axis. A different option consists of utilizing 2D laser scanners and 6D pose estimation algorithms. The Stanford Racing Team[28] has used this technique for high speed terrain classification in the DARPA Grand Challenge.

There are multiple research groups that focus on the mapping problem considering that the robot is in 3D environments with six degree of freedom (6 DOF). Due to available sensors and increase of computer computational capabilities, this procedure must play an important role in SLAM for mobile robots. Nüchter *et al.*[20] have developed a mapping method based on the alignment of 3D laser range scans using the Iterative Closest Points[2] (ICP) scan matching method combined with a heuristic for loop detection and a global relaxation method. Hähnel *et al.*[13] have presented an algorithm "for full 3D shape reconstruction of indoor and outdoor environments with mobile robots." Triebel *et al.*[30] worked with 3D maps building Multi-Level Surface (MLS) maps. An elevation map stores the height of the corresponding area in each cell of a discrete grid. The main shortcoming of this map is that it is not possible to represent vertical structures with multiple levels. The MLS maps allow the storage of multiple surfaces in each cell, making it possible to include bridges or buildings in the map. Cole and Newman[6] demonstrate that the traditional methods used to solve the SLAM problem in planar environments can also be extended to perform 6D SLAM in more difficult conditions, e.g., undulating outdoor areas. Magnusson and Duckett[16] have proposed an alignment method that is based on the Normal Distributions Transform (NDT).[3]

There are different approaches that use information from charge-coupled device (CCD) cameras, which is usually called Visual-SLAM.[22] Their principal disadvantage is that changing light conditions make them difficult to use.

This paper focuses on two particular aspects of the mapping process: scan matching (or registration) and loop detection.

2.1. *Scan matching*

There are different classifications of scan matching methods depending on several factors.

The first one considers the environment dimensionality. The scan matching methods found in the literature can be developed to work with 2D or 3D data. The last case is addressed here.

There are also local[15] and global[29] methods. The local methods match single scans. The disadvantage of the local methods is that the final map is inconsistent because the accumulated error can be important. This inconsistency can be minimized by relaxation mechanisms. The global methods consider the current scan and the global model. Their associated shortcoming is that one single mistake is fatal because a wrong measurement can be included in the model, not being possible to correct this error by additional mechanisms.

It is also possible to distinguish between feature-based, point-based, or mixed approaches. The first case requires a feature extraction before the scan matching. The most common features are line segments, planes, or corners. The point-based approach does not require any distinguishable structure in the environment, and the mixed approach seeks correspondence between points and features. A point-based local method has been developed in this work. Some examples of related approaches are detailed below.

The ICP method[32] is an algorithm that is used to minimize the spatial distance between two scans. The ICP is the most common scan matching method. This method is quite simple and its computational cost makes it possible to use it in real-time. It receives two clouds of points, an initial estimate of the translation and rotation, and the stopping criteria. It iteratively estimates the transformation (translation, rotation) that minimizes the distance between the clouds. Besl and MaKay[2] have developed this method to register 3D shapes.

Many variants of this method have been proposed. A very interesting comparison of several methods depending on different parameters, such as the selection and matching of points and the minimization strategy, can be found in the work by Rusinkiewicz and Levoy.[21] Some mechanisms of the registration method that has been developed in this work are based on the ICP variant proposed by Triebel *et al.*[30] The cost function is changed to match points that belong to similar surfaces. Bosse and Zlot[5] have proposed an improvement based on "the addition of robust optimization techniques to handle outliers and imperfect correspondences between the data."

The Iterative Matching Range Point (IMRP) method, proposed by Lu and Milios,[15] is based on the limitation of maximum translation and rotation. The correspondences are found in the neighborhood of the model, not considering the whole space.

The Iterative Dual Correspondence (IDC) method, also proposed by Lu and Milios,[15] combines ICP and IMRP. The translation is computed by the ICP method and the rotation is estimated by the IMRP method.

The Polar Scan Matching (PSM) method does not need to find correspondences between points. It assumes that model and data are sorted in the same way and only points with the same bearing are matched. Its advantage is that it is not necessary to transform laser measurements into cartesian coordinates (the original distances can be compared). This advantage is the key of the method proposed by Diosi and Kleeman.[8]

Thrun *et al.*[27] consider that the free space in the current model will remain free in the future. They increase the information that is extracted from the laser scan.

In this work, an evolutionary-based scan matching algorithm for 3D environments has been developed. Our method tries to solve the same problem as ICP, which is the minimization of the distance between two point clouds. It also searches corresponding points between scans.

2.2. Loop detection

Several feature-based techniques that have been successfully applied to loop detection will be explained here. An important factor is that these techniques must be rotation-invariant because the locations have different appearances depending on the robot's orientation.

A detailed explanation of a geometric descriptor for 3D maps can be found in the work developed by Magnusson *et al.*[17] The environment is divided into cells and each cell is represented by its NDT. Each feature contains a number of cells that belong to a specific surface. The laser range data are described with feature histograms based on the surface orientation and smoothness, and the loops are detected by matching feature histograms.

Granström *et al.*[11] have used a machine learning algorithm called AdaBoost[9] to learn a classifier from previously extracted features. The feature descriptor is composed of 41 different features, all of them invariant to rotation. The first 33 features are numeric values representing different properties such as volume, difference, curvature, centroid, etc. The last nine features are range histograms with different bin sizes.

Bosse and Roberts[4] have developed a histogram-based technique for planar maps with two main components. The first component is an orientation histogram used to compute rotation difference. The second one is a set of projection histograms that determine the distance between scans. The loops are detected by multiplying the peak value of the orientation histogram by the peak value of the projection histograms and comparing the result with a matching threshold. Bosse and Zlot[5] have improved their method in unstructured environments by using entropy sequences of projection histograms instead of orientation histograms.

Cummings and Newman[7] have developed an algorithm called FAB-SLAM which is a probabilistic approach that detects known places based on their appearance. Their maps are represented by a set of visual words that are detected using a SURF descriptor. The previously visited places are detected when a new image contains a word that is already in the set.

3. Mapping Process

The mapping task for mobile robots is an open problem with different solutions and descriptions depending on many factors: available sensors, techniques applied, type of environment, etc. In this section, this problem is described according to the techniques used here. A flow chart of the mapping process presented in the following paragraphs can be observed in Fig. 1. This process is similar to the mapping method proposed by Nüchter *et al.*[20] It will be explained below, but some aspects related to this task are described before.

We have assumed that the mobile robot works in a 3D world. The exteroceptive information is measured by a 3D laser scan and the motion information is computed by wheel encoders. The 3D laser reading can be observed in Fig. 2. These measurements have been obtained by MANFRED-2, which is an experimental platform fully developed by the Robotics Lab of the Carlos III University of Madrid, Spain. It has been recorded inside a lab of the university campus. It is composed of 45,000 points approximately. The DOF in this type of environment is six in the most general case. Therefore, the robot's pose is defined by the following coordinates: position ($\mathbf{Pt} = (x, y, z)^T$) and orientation

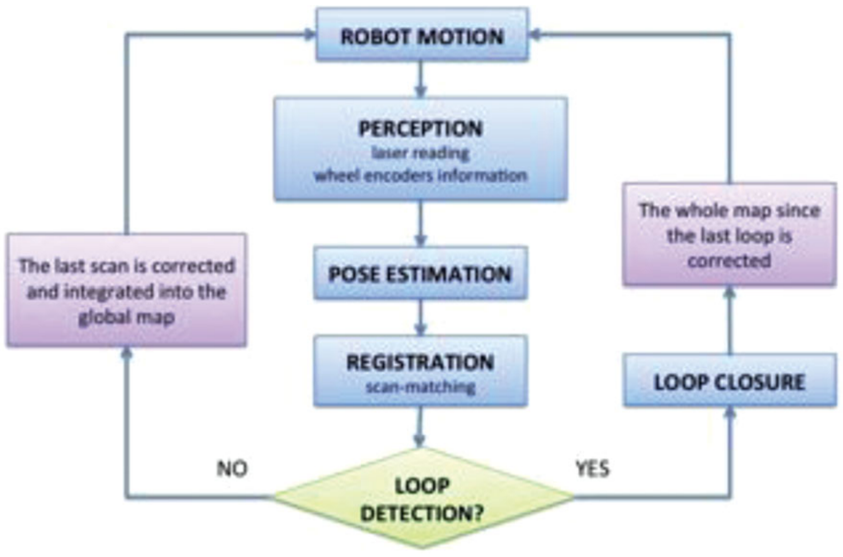

ROBOT MOTION
PERCEPTION
laser reading
wheel encoders information
POSE ESTIMATION
REGISTRATION
scan-matching
LOOP DETECTION?
NO
YES
The last scan is corrected and integrated into the global map
LOOP CLOSURE
The whole map since the last loop is corrected

Fig. 1. (Colour online) General mapping process.

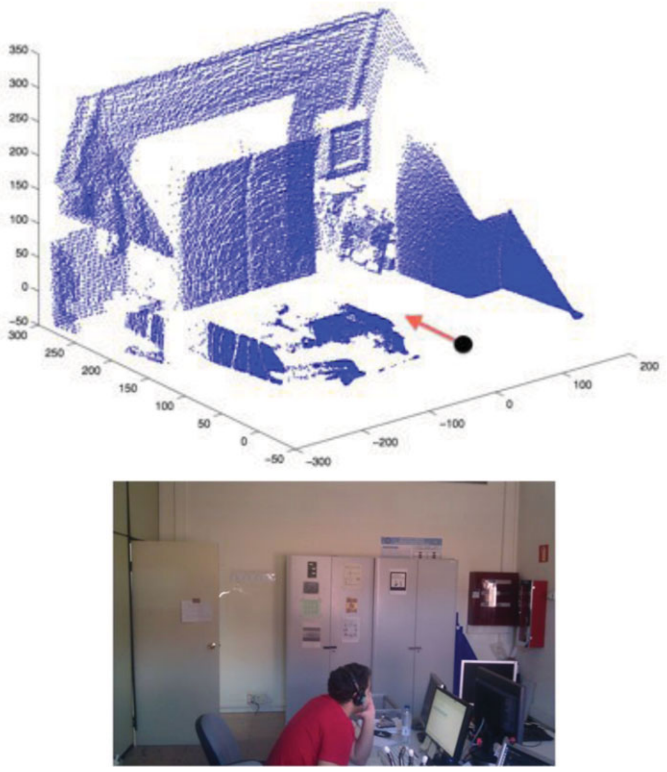


Fig. 2. (Colour online) 3D laser reading obtained by MANFRED-2. All units are in centimeters. Robot's pose: black point and red arrow. Bottom: real photo.

($\mathbf{O} = (\varphi, \theta, \psi)^T$) (roll, pitch, and yaw angles). Each movement from an initial position to a final one can be calculated computing a translation plus a rotation:

$$\mathbf{Pt}_{\text{final}} = \mathbf{Rot} \times \mathbf{Pt}_{\text{initial}} + \mathbf{Tra}.$$

Working directly with 3D data provided by the laser scan is not efficient from a computational point of view. The scan size is reduced in a preprocessing phase to increase computational efficiency, but without losing the capability of obtaining good matching. The computational cost of the scan matching method is decreased approximately by 50 times by introducing data reduction. This value is obtained after comparing the computational cost with the raw laser data to the computational cost with the laser scan after preprocessing.

Since the encoders measure wheel displacements, only 3 DOF representing plane position (x, y) and orientation (θ) can be estimated considering the odometry information. However, our mapping tools have been designed to work in 6 DOF as maximum, and different examples with sensors that receive 6D information are shown in experimental results.

The odometry information is then fed up to the mapping module for pose correction according to the exteroceptive information. This is done by registering the newly acquired laser reading. The current scan pose is corrected until the best overlap with the reference scan is achieved. A 6D DE-based scan matching method has been implemented in this work. A complete explanation of our method is given in Section 4. Besides, the ICP-based scan matching method developed by Triebel *et al.*[30] has also been implemented for comparison.

The models generated after registration are not consistent enough to conclude that the mapping task has been successfully achieved. This consistency will be given by detecting when the robot is situated in a pre-visited place (loop detection) and eliminating the accumulated error when it exists (loop closure). If the pose estimate has a small orientation error at the beginning of the robot's path, the accumulated error after some time can be very important.

Our work focuses on loop detection. Different features that are extracted from a 3D laser scan are the variables used to calculate the similarities between two places. We have developed a feature-based loop detector based on the works by Magnusson *et al.*[17] and Granström *et al.*[11] Our method is detailed in Section 5.

The whole map since the last loop is corrected when the loop is detected. After that the robot continues exploring the environment with the corrected and updated map. If there is no loop, the current scan coordinates obtained after registration and their associated scan are introduced in the global map, thus the model is also updated but no loop is closed. The robot continues its exploration and new measurements will be obtained.

4. Registration: Scan Matching

Since the pose estimation considering only motion information is not accurate enough, it is necessary to develop additional methods to obtain better maps. The first method implemented here that includes information obtained by perceptive sensors is called registration or scan matching. One of the most important aspects of any SLAM algorithm is the registration method. The laser scan matching consists of the current scan pose correction until the best overlap with the reference scan or model is achieved.

A DE-based scan matching algorithm has been implemented to solve the referred problem. It consists of a particle-based evolutionary algorithm that evolves in time to the pose that yields the cost function lower value. The fitness function calculates the correspondences between model and data, and the evolutionary filter minimizes the distances between corresponding points.

The DE algorithm engine has also been used in our recent work to solve the global localization problem, which can be defined as the search of the robot's coordinates in a known environment with no *a priori* information about its location. A complete explanation can be found in our previous work.[18] This scan matching technique is similar to the localization filter.

The following two different ideas have been applied to improve the algorithm performance:

- *Correspondences modification*: The scan matching algorithms are based on the minimization of cost functions that calculate correspondences between points. A modified scan matching algorithm has been implemented which finds correspondences between points of the same type. This means that the cost function considers corresponding points that share any specific characteristic.
- *Acceleration with* k*-dimensional trees*: A k-dimensional tree[1] (k-d tree) is "a space-partitioning data structure for organizing points in a k-d space." The scan matching algorithm is accelerated using k-d trees when looking for correspondences between points. The computational cost is greatly improved when this method is incorporated into the search mechanism.

The fitness function that incorporates these concepts, which is the key element of our method, is introduced in the next section. After that our scan matching method will be explained.

4.1. Fitness function

The starting point of the cost function is the well-known ICP algorithm cost expression proposed by Surmann *et al.*[26] The cost function compares the previously acquired scan (model) to the last acquired one (data). Therefore, the inputs of the fitness function are the following two different sets of 3D points:

- Model set: $M = \{m_1, ..., m_{N_m}\}$,
- Data set: $D = \{d_1, ..., d_{N_d}\}$.

The ICP method "iteratively revises the transformation (**Rot**,**Tra**) needed to minimize the distance between the points of two raw scans" (M and D). The cost function to minimize is the following one:

$$e(\mathbf{Rot}, \mathbf{Tra}) = \sum_{i=1}^{N_m} \sum_{j=1}^{N_d} w_{i,j} \|\mathbf{m}_i - (\mathbf{Rot} \cdot \mathbf{d}_j + \mathbf{Tra})\|^2, \tag{4.1.1}$$

where $w_{i,j}$ is equal to 1 if i nd j are corresponding points. It means that they describe the same point in space. If they are not corresponding points, $w_{i,j}$ is 0.

The corresponding points computation, which is equivalent to a Nearest Neighbor (NN) search, is accelerated using k-d trees.[1] The objective is to find the corresponding point within the data (represented by a k-d tree) for each point of the model.

Working in a 6-DOF space is a heavy task from a computational point of view, thus it is very common to accelerate scan matching algorithms using different methods. Most matching methods require to search corresponding points between scans. This search is the bottleneck from a computational point of view. The most widely used acceleration approaches modify the correspondence search using trees.

The purpose of the NN algorithm is to find the point in the k-d tree which is closer to a given input point. The search efficiency of the NN algorithm is improved by eliminating large portions of the search space. In general, the NN search-given n points are efficiently accelerated by k-d trees when the following condition is satisfied: $n >> 2^k$. Otherwise, most of the points in the tree are visited and the efficiency is not improved. There are approximate NN methods that can be used in these cases.

The dimension number is three in our particular case, which implies that the number of points of a laser scan must be much larger than eight. This condition is always satisfied.

The correspondences are introduced as weights ($w_{i,j}$) in the error function. This weights have a very important influence on the scan matching process. The method proposed here exploits this idea by modifying the weight selection criterion in order to improve its capabilities.

If the number of correspondences is equal to C and the distance between two corresponding points is denoted by $d(\mathbf{m}_{i_c}, \mathbf{d}_{j_c})$, the cost function 4.1.1 can be rewritten as

$$e(\mathbf{Rot}, \mathbf{Tra}) = \sum_{c=1}^{C} d(\mathbf{m}_{i_c}, \mathbf{d}_{j_c})^2. \tag{4.1.2}$$

The first modification exploits the idea introduced by Triebel *et al.*,[30] which separates the scan in different types of points, only matching points that belong to the same class. The set of points is divided into walls, horizontal planes, and obstacles:

$$e(\mathbf{Rot}, \mathbf{Tra}) = \underbrace{\sum_{c=1}^{C_1} d(\mathbf{m}_{w,i_c}, \mathbf{d}_{w,j_c})^2}_{\text{walls}} + \underbrace{\sum_{c=1}^{C_2} d(\mathbf{m}_{p,i_c}, \mathbf{d}_{p,j_c})^2}_{\text{planes}} + \underbrace{\sum_{c=1}^{C_3} d(\mathbf{m}_{o,i_c}, \mathbf{d}_{o,j_c})^2}_{\text{obstacles}}, \tag{4.1.3}$$

where subindex w is used to express that a point belongs to a wall, p represents horizontal planes (unobstructed floor), and obstacles are denoted by o.

The type of a given point is easily computed following the next steps:

1. KNN search: The nearest neighbors of the point within the scan are calculated. This search is also accelerated by using k-d trees. The number of neighbors is fixed to 10.
2. Plane estimation: The plane that best fits the previously obtained neighbors is calculated. A plane P_i is defined by a point $\vec{p}_i$ and the direction cosines of the normal:

$$P_i = \{\vec{p}_i, \cos\alpha, \cos\beta, \cos\gamma\}.$$

Different types of planes are calculated according to the cosines:
(a) $\gamma > 80^\circ \rightarrow$ wall,
(b) $\gamma < 10^\circ \rightarrow$ horizontal plane,
(c) otherwise $\rightarrow$ obstacle.

The error function only considers points that belong to the same types. Different types can be chosen depending on the environment characteristics. This segmentation is suitable for structured indoor environments, which are mostly composed of walls, horizontal planes, and obstacles.

The next idea consists of choosing adequate weights for each type. In the previous expressions, the weight factor was assigned a value of 1 when corresponding points were found. However, it can be interesting to give higher weights to those points that represent more important characteristics.

Analyzing the point types, the most important one in an indoor environment is the vertical cell. The obstacles are characteristics that uniquely define the environment, but their importance depends on the number of obstacles and their size. In general, there are more vertical points than obstacles points in a frame. It is not always possible to distinguish between places taking into account only traversable zones.

If Eq. (4.1.3) is modified by introducing $w_{i,j} = 2$ for walls and $w_{i,j} = 0.5$ for other types, the error function is now equal to

$$e(\mathbf{Rot}, \mathbf{Tra}) = \underbrace{\sum_{c=1}^{C_1} 2d(\mathbf{m}_{w,i_c}, \mathbf{d}_{w,j_c})^2}_{\text{walls}} + \underbrace{\sum_{c=1}^{C_2} 0.5d(\mathbf{m}_{p,i_c}, \mathbf{d}_{p,j_c})^2}_{\text{planes}} + \underbrace{\sum_{c=1}^{C_3} 0.5d(\mathbf{m}_{o,i_c}, \mathbf{d}_{o,j_c})^2}_{\text{obstacles}}. \quad (4.1.4)$$

An improvement in the distance between corresponding points that belong to walls will have more influence on the error function. It is important to remark that these values have been chosen taking into account the type of environments that have been tested, which are indoor environments mostly composed of walls and horizontal planes. It will be crucial to choose adequate weights in order to obtain good results.

If the obstacles are the most important features, then the matching quality may decrease. First, there are not obstacles enough in this type of environments. Second, dynamic objects have also a negative influence on the matching quality because their influence on the cost function is higher.

If the floor has a higher weight, there are some particular cases where the scan matching can fail to reach the solution. For example, when the robot is located in a long corridor, there are not enough features on the floor to distinguish between places. The distinguishable features are located on the walls (doors, windows).

For all these reasons a higher weight has been given to the walls. However, the ground and the obstacles still have influence on the cost function because the difference between the weights is not too large.

The cost function represented by Eq. (4.1.4) is the engine of the DE-based scan matching method. This method is explained in the following section.

4.2. DE-based scan matching (6 DOF)

Given two independently acquired sets of 3D points (model and data), the DE-based algorithm calculates the rotation and translation that are necessary to apply to the data set in order to maximize matching between them.

Algorithm 1 DE-based Scan Matching

```
 1: for $i = 1 : N_P$ do
 2:     $pop_i^1 \leftarrow init\_pop(data\_initial\_pose)$        ▷ First population generation
 3:     $e_i^0 \leftarrow cost(model, data, pop_i^1)$        ▷ Cost function calculation
 4: end for
 5: for $k = 1 : max$ do
 6:     for $i = 1 : N_P$ do
 7:         $v_i^k = pop_a^k + F(pop_b^k - pop_c^k)$        ▷ Mutation
 8:         for $j = 1 : D$ do
 9:             $u_{i,j}^k = v_{i,j}^k, \forall p_{i,j}^k < \delta$        ▷ Crossover
10:             $u_{i,j}^k = pop_{i,j}^k, \forall p_{i,j}^k \geq \delta$
11:         end for
12:         $e_i^k \leftarrow cost(model, data, pop_i^k)$        ▷ New cost function calculation
13:         if $e_i^k < e_i^{k-1} - \tau$ then        ▷ Selection with thresholding
14:             $pop_i^{k+1} = u_{i,j}^k$
15:         else
16:             $pop_i^{k+1} = pop_i^k$
17:         end if
18:     end for
19:     $pop^k = disc(pop^k)$        ▷ Discarding
20:     $ind\_best \leftarrow min(e^k)$
21:     $bestmem \leftarrow pop^k(ind\_best)$
22:     if $convergence = true$ then        ▷ Execution stops after convergence
23:         $exit(bestmem)$
24:     end if
25: end for        ▷ Return best estimation
```

The stochastic search of the matching pose is done using the DE method for global optimization problems over continuous spaces. The reader can see Algorithm 1 to understand the main concepts. Besides, a complete explanation of the DE algorithm can be found in our previous work.[18,19]

The search starts with a population of N_P candidates. Each candidate is a possible solution. The robot's pose has 6 DOF:

$$pop_i^k = (x_i^k, y_i^k, z_i^k, \varphi_i^k, \theta_i^k, \psi_i^k),$$

where pop_i^k represents element i at iteration k. The position is given by the cartesian coordinates and the orientation is defined by the Euler angles. The initial population will be chosen randomly, but situated in a sphere close to the pose estimate given by the odometry information. The DE-based scan matching is simpler than the DE-based global localization because there is an initial estimation of the robot's pose. In global localization, the initial population was randomly generated covering all the maps because there was no *a priori* information about the robot's location.

For each candidate, its associated cost function is calculated (line 3 of Algorithm 1) according to Eq. (4.1.4).

The main loop starts in line 5 and it is repeated until the maximum number of iterations is reached or one of the convergence conditions is satisfied. Another loop that contains the evolutionary search starts in line 6. It consists of the generation of a new population for the next generation. In a single iteration the algorithm is executed to obtain next candidates, evolving in time to the correct pose.

The initial population is perturbed to generate a variation v_i for each population member:

$$v_i^k = pop_a^k + F(pop_b^k - pop_c^k), \tag{4.2.1}$$

where pop_a^k, pop_b^k, and pop_c^k are three randomly chosen elements at iteration k, and a, b, and c are different from running index i. F is a real and constant factor that controls the amplification of differential variations.

In order to increase the diversity of new generation, the crossover is introduced. The trial vector is denoted by $u_i^k = (u_{i,1}^k, u_{i,2}^k, \ldots, u_{i,D}^k)^T$ and its parameters depend on the crossover probability:

$$u_{i,j}^k = \begin{cases} v_{i,j}^k; & \text{if } p_{i,j}^k < \delta \\ pop_{i,j}^k; & \text{otherwise} \end{cases}, \tag{4.2.2}$$

where $p_{i,j}^k$ is a randomly chosen value from the interval [0, 1] for each parameter j of the population member i at iteration k, and δ is the crossover probability that constitutes the crossover control variable. D, which represents the number of chromosomes (number of components of the population element), is equal to six because the robot's pose has six DOF. The random values $p_{i,j}^k$ are made anew for each trial vector i.

The new candidate u_i^k is compared to pop_i^k to decide which element should become a member of generation $k + 1$. If u_i^k yields a better value for the objective fitness function than pop_i^k then it is replaced by u_i^k, otherwise the current member pop_i^k is retained for the next generation.

The selection mechanism has been modified by adding a thresholding band that avoids premature convergence in noisy optimization problems. The idea of thresholding is to reduce the eagerness of algorithm by rejecting the new solutions that do not improve the previous hypothesis in a pre-specified magnitude τ. In our experiments we have chosen a value that depends on the sensor noise because the thresholding mechanism tries to avoid optimization in the noise band. The threshold τ is equal to $S_N * e_i^k$, where S_N is the sensor noise variance (percentage over the distance weighted) and e_i^k is the fitness function value for the ith population member at iteration k.

Modifying the selection mechanism by the thresholding described above, the following expression is obtained:

$$pop_i^{k+1} = \begin{cases} u_i^k, & \text{if } e_{pop_i}^k - e_{u_i}^k > \tau \\ pop_i^k, & \text{otherwise} \end{cases}, \tag{4.2.3}$$

where $e_{pop_i}^k$ is the fitness function value of the current population member and $e_{u_i}^k$ represents the fitness function value of the trial vector.

In conclusion, if the improvement in the fitness function is larger than the variance (or the standard deviation, depending on the selected units), it can be considered that it is not caused by the noise, and the new member can be introduced in the population.

The previous process (mutation, crossover, and selection with thresholding) is applied to the whole population, obtaining the next generation population $(k + 1)$.

The convergence speed is decreased by the thresholding band. A discarding mechanism has been introduced to increase the algorithm speed while maintaining the thresholding advantages. The idea is to determine the worst fitness individuals of the new population and substitute them by new solutions that are situated close to better ones. In order to do so, a percentage of elements to be discarded is chosen (5%). To avoid concentrating the discarded solutions around the best existing individual, one of the members of the population with its fitness value located in the first half of the fitness ranking is randomly selected. This selected solution plus a relatively small random component is adopted as a new offspring.

Finally, the algorithm returns the best member of the population. This member is the robot's pose that minimizes the difference between model and data. The data set is moved and rotated according to this solution, and the updated data are incorporated into the model.

5. Loop Detection

The loop detection algorithm is detailed in this section. This tool has been developed to detect the situations where the robot is located in a pre-visited place. The robot could be far away from its initial location and the accumulated error must be detected and minimized to give consistency to the global map.

5.1. Surface features

The method developed here is based on several ideas. The first one is extracted from the work by Magnusson *et al.*[17] Their algorithm extracts features from two 3D laser scans and compares them to detect if the loop exists. Their descriptor is composed of the geometric forms of the scan (lines, planes, and spheres). Each component is equal to the number of elements (laser measurements) that belong to a feature (for example, plane with orientation).

Magnusson *et al.*[17] divide the map into a regular grid and calculate the parameters of a normally distributed probability density function describing the local surface shape. This part has been implemented here in a slightly different way.

For each point i of the laser scan, the K nearest neighbors within the scan are efficiently searched by using k-d trees. The covariance matrix of K points is calculated after that. The point i local surface is described by the eigenvectors $(\vec{e}_1, \vec{e}_2, \vec{e}_3)$ and the eigenvalues $(\lambda_1, \lambda_2, \lambda_3)$ of the covariance matrix.

The local surface type is given by the relationship between the eigenvalues $(\lambda_1 > \lambda_2 > \lambda_3)$. Two different variables (t_l and t_p) are defined to distinguish between surfaces:

- Line: $\lambda_1/\lambda_2 > t_l$.
- Plane: It is not a line and $\lambda_2/\lambda_3 > t_p$.
- Sphere: It is neither a line nor a plane.

t_l and t_p are constant thresholds that are used to detect lines and planes.

After that the planes and lines can be sub-divided into different types depending on the orientation, and the different spheres can be defined depending on the surface roughness. Finally, a number N_s of planar, linear, and spherical features is obtained. A good performance is achieved in the experimental results with one line, seven planes, and one sphere ($N_s = 9$). The feature descriptor is $\vec{F} = (f_1, ..., f_{N_s})$, where each element j of $\vec{F}$ represents how many points belong to feature j.

The matching between scans is obtained by comparing feature descriptors. The most common method is to define a difference giving weights to each component of the descriptor. However, our approach here is treated from a different point of view.

If the matching quality is defined by a value that represents the similarities between a pair of scans, the loop detection is converted into a binary process because the loop exists when the difference is lower than the threshold. Nevertheless, the loop detection variable that we have defined is not binary but continuous. We have called it LPI.

The LPI does not literally represent the probability of matching because the feature descriptors are not probabilities at all. But it is possible to define a variable that equals 100% when the scans perfectly match and whose value decreases when the match is poorer. That is why the term *probability* is included. The LPI initial definition is expressed by the following equation:

$$\mathrm{LPI} = \sum_{i=1}^{N_s} \frac{(\max(f_i, g_i) - |f_i - g_i|)}{\max(f_i, g_i)} \times \frac{w_i}{\sum_{i=1}^{N_s} w_i} \times 100, \tag{5.1.1}$$

where f_i and g_i are the components of two different feature descriptors and w_i represents the weight of each component.

The term $(\max(f_i, g_i) - |f_i - g_i|)/\max(f_i, g_i)$ of Eq. (5.1.1) represents similarities between both scans for a single feature. It has been normalized to be 1 when the matching is perfect, and lower when the difference between f_i and g_i increases.

Besides, each feature weight has to be defined and normalized. It has to be higher when the number of points that belong to this feature is larger. The weight given to each feature i could be equal to

$$w_i = \max(f_i, g_i), \tag{5.1.2}$$

choosing the maximum value between f_i and g_i. The weights are normalized by dividing their values by the sum of the maxima of all features. Two different options have also been implemented:

$$w_i = \mathrm{mean}(f_i, g_i).$$

$$w_i = \sum_{j=1}^{T} \max(f_i, g_i)/T. \tag{5.1.3}$$

The first one uses the average value instead of the maximum. The second one not only considers the current frame but also the past information. If the robot has received T laser frames during exploration, each weight is the average of T weights that are obtained for each laser scan according to Eq. (5.1.2).

If the weights are given by Eq. (5.1.2), then Eq. (5.1.1) can be simplified as follows:

$$\mathrm{LPI} = \frac{\sum_{i=1}^{N_s}(\max(f_i, g_i) - |f_i - g_i|)}{\sum_{i=1}^{N_s} \max(f_i, g_i)} \times 100. \tag{5.1.4}$$

This option will be used from now on for simplicity.

The described method considers geometric forms (we have called them *surface* features). Each component holds a number of points that belong to each feature. However, there are different types of features that cannot be added. It can be interesting to add different features (we have called them *numerical* features) and change the LPI formula to introduce them.

5.2. *Numerical features*

There are interesting characteristics that can be represented by a number. Some examples are volume, curvature, average distance between points, and so on. A feature descriptor with numerical characteristics is used by Granström *et al.*[11]

By including these types of features, a new feature descriptor can be defined as

$$\vec{F} = (f_1, \ldots, f_{N_s}, f_{N_s+1}, \ldots, f_{N_s+N_n}),$$

where N_n is the number of *numerical* features.

The LPI value must satisfy the specifications: It has to be equal to 100% in a perfect match and lower when the matching quality decreases. Therefore, it is necessary to define the weights of the new features according to this fact.

Analyzing Eq. (5.1.1), the average value of the weights is equal to

$$W = \frac{\sum_{i=1}^{N_s} \max(f_i, g_i)}{N_s}.$$

A weight equal to W will be given to the new features. Nevertheless, different weights can be calculated depending on the new features to be added. The loop indicator can be now divided into two components:

$$\begin{aligned}
\mathrm{LPI}_s &= \frac{\sum_{i=1}^{N_s}(\max(f_i, g_i) - |f_i - g_i|)}{[\sum_{i=1}^{N_s} \max(f_i, g_i)](1 + \frac{N_n}{N_s})},\\
\mathrm{LPI}_n &= \sum_{i=N_s+1}^{N_s+N_n} \frac{\max(f_i, g_i) - |f_i - g_i|}{\max(f_i, g_i)}\\
&\quad \times \frac{(\sum_{i=1}^{N_s} \max(f_i, g_i))/N_s}{[\sum_{i=1}^{N_s} \max(f_i, g_i)](1 + \frac{N_n}{N_s})},\\
\mathrm{LPI}_n &= \sum_{i=N_s+1}^{N_s+N_n} \frac{\max(f_i, g_i) - |f_i - g_i|}{\max(f_i, g_i)} \times \frac{1}{N_s + N_n},\\
\mathrm{LPI} &= (\mathrm{LPI}_s + \mathrm{LPI}_n) \times 100.
\end{aligned} \tag{5.2.1}$$

The loop indicator still meets the requirements: It is equal to 100% in a perfect match and lower when the matching quality decreases. LPI_s is the component that computes *surface* features. It is calculated following Eq. (5.1.4), but the weights are normalized according to the total number of features $(N_s + N_n)$ by adding the term $1 + N_n/N_s$. LPI_n is used to include *numerical* features. Each component weight is equal to W and the expression is normalized according to the total number of features.

An advantage of the proposed method is that the weights are calculated online, not needing any machine learning algorithm to fix them like the method done by Granström *et al.*[11] The autonomous robot mapping task is not traditionally a machine learning problem, thus the previously explained option that has been developed in this paper is a more suitable approach.

Different *numerical* features have to be considered to be a component of the descriptor. Most of them have been taken from the work of Granström *et al.*[11] The best performance has been obtained with the following ones:

- *Volume*: The volumes of the individual laser beams are added to measure the volume of the scan. Each measurement is the center of a pyramid base with its top in the sensor location (robot's position). If θ_v and θ_h are the laser vertical and horizontal resolutions, the length and width of the pyramid base are $l_i = 2d_i \tan(\theta_v/2)$ and $w_i = 2d_i \tan(\theta_h/2)$. The height is $h_i = d_i$. The volume of the pyramid is $v_i = l_i \times w_i \times h_i/3$. It is possible to obtain a volumetric feature that does not depend on angular resolutions, normalizing the volume of each point with the volume of $d_{\max}$:

$$f_{N_s+1} = \frac{1}{P v_{\max}} \sum_{i=1}^{P} v_i = \frac{1}{P} \sum_{i=1}^{P} \left(\frac{d_i}{d_{\max}} \right)^3 ,$$

  where the range d_i is computed as the distance from point i to the sensor location. The total number of points is P.
- *Distance*: f_{N_s+2} is the sum of the distances between consecutive points: $\delta\vec{p}_i = ||\vec{p}_i - \vec{p}_{i+1}||$, where $\vec{p}_i$ is the 3D position of point i.
- *Curvature*: If A is the area covered by the triangle with vertexes in $\vec{p}_{i-1}$, $\vec{p}_i$, $\vec{p}_{i+1}$, and $d_{i-1,i}$, $d_{i-1,i+1}$, $d_{i,i+1}$ are the pairwise point-to-point distances, the curvature at i is computed as

$$k_i = \frac{4A}{d_{i-1,i} d_{i-1,i+1} d_{i,i+1}}.$$

  f_{N_s+3} is the mean of the vector that contains the curvature for all points.

Besides, there are more candidates that could be chosen as *numerical* features: average range, standard deviation of range, range differences (mean and standard deviation of the range difference between consecutive points), etc. Different behaviors will be obtained depending on the descriptor composition.

5.3. *Expression analysis*

The final definition of LPI is given by Eq. (5.2.1). Analyzing its value, an additional mechanism has been implemented to improve method capabilities. Different bands have been defined depending on the loop detection variable value:

- LPI $\geq t_1$: Positive matching. The loop is detected and the accumulated error must be minimized by a relaxation method.
- $t_2 \leq$ LPI $< t_1$: Uncertainty band. There is not enough information to conclude that the loop exists.
- LPI $< t_2$: Negative matching. There is no loop in this case.

t_1 and t_2 are thresholds that are used to distinguish between bands.

An advantage of this approach is that an additional method can be applied when LPI is inside the uncertainty band, thus the loop detection quality is improved. The implemented method consists of analyzing the third scan. If the LPI value between two scans (a and b) is inside the uncertainty band, the third scan is considered and the loop indicator is calculated again using the new scan (a and c). This is done by waiting until the robot receives a new laser reading from the next location that is explored. The loop is now detected depending on the new value that includes more information.

A minimum loop size could be defined to complete the algorithm. If it is too small, consecutive scans will be considered as loops, but the global alignment could be improved. Therefore, a minimum number of scans between two visits to any location could be fixed.

In addition, the LPI definition given by Eq. (5.2.1) makes this method very flexible because it is not necessary to have any previous knowledge about the environment. It has more associated advantages. As said before, different features and values of N_n can be introduced in the formula in order to choose the best combination. It can be considered as an online learning method because it does not need any previous training or a huge available data set.

When the autonomous robot is navigating around the environment, the available information increases, and the LPI weights can be changed to obtain a better performance. The features' average or standard deviation would be a better approximation if there is enough information. For example, the features' average of the past scans can be chosen. This option has been implemented in this work, as can be seen in the second line of Eq. (5.1.3).

After several pair comparisons, an estimation of the most important features in the matching process can be done, and the weights of the most important features can be increased while decreasing those of the least important ones.

The described method is adaptive and its weights will change online while the robot is navigating and acquiring new information. Its adaptation to changes of the environment is automatic. For example, if the robot is working in an indoor environment and it navigates to an outdoor place, the adaptive weights change while it is navigating and the adaptation to the new environment is done online, not being necessary to make any additional calibration.

6. Experimental Results

6.1. Data sets

The $Hannover2$ data set has been recorded at the university campus of Leibniz Universität Hannover, Germany. It contains 924 3D scans (with 360° field of view), covering a trajectory of about 1.24 km. Each scan is composed of 16, 600 points approximately. The maximum range is 30 m. Thanks to Oliver Wulf for making available these data.[1]

The $UC3M$ data set obtained by MANFRED-2 has been recorded at the university campus of Carlos III University of Madrid, Spain. It contains 150 3D scans covering a 900 m^2 indoor environment. Each scan is composed of 45, 000 points approximately. The horizontal (tilt) resolution is 0.25°, and the amplitude of a 2D scan is 190°. The vertical (pan) resolution is 1° and its amplitude is 70°. The maximum range is 15 m. More information is available online.[2]

6.2. Scan matching

Two different experiments have been conducted to verify our DE-based scan matching method. The algorithm has been configured with the following parameters: $F = 0.8$, $\delta = 0.75$, $S_N = 0.02$, $N_P = 20$, and maximum iterations equal to 50.

The $UC3M$ data set is used in the first experiment to study the error in local environments composed of several laser frames. The purpose is to observe the visual appearance of the map that is built after registration.

Some results in different situations are shown in Fig. 3. The distances between consecutive scans are equal to 30 cm. The local map after joining four different scans when the robot is located in a corridor is represented in the upper part. The high accuracy can be observed in the horizontal projection. Two different labs are modeled in the middle and bottom figures. Both figures are composed of four frames. The real photos of the corridor and the bottom lab can be observed at the bottom. The real photo of the middle lab is in Fig. 2.

In general, the high accuracy of our method lets us conclude that the algorithm can be used in manipulation tasks. It is not easy to measure the scan matching error because the robot's true pose is not available in indoor environments. However, the error is equal or lower than a centimeter when analyzing horizontal projections of local maps. Similar errors have been obtained after applying an ICP-based method in the corridor. The ICP method is based on the search of corresponding points. If the rotation between two consecutive scans is very pronounced and there is not an accurate initial estimate about this rotation, the algorithm can fail because the corresponding points do not represent

[1] `http://kos.informatik.uni- osnabrueck.de/3Dscans/.`
[2] `http://roboticslab.uc3m.es/roboticslab/robot.php?id_robot=5`

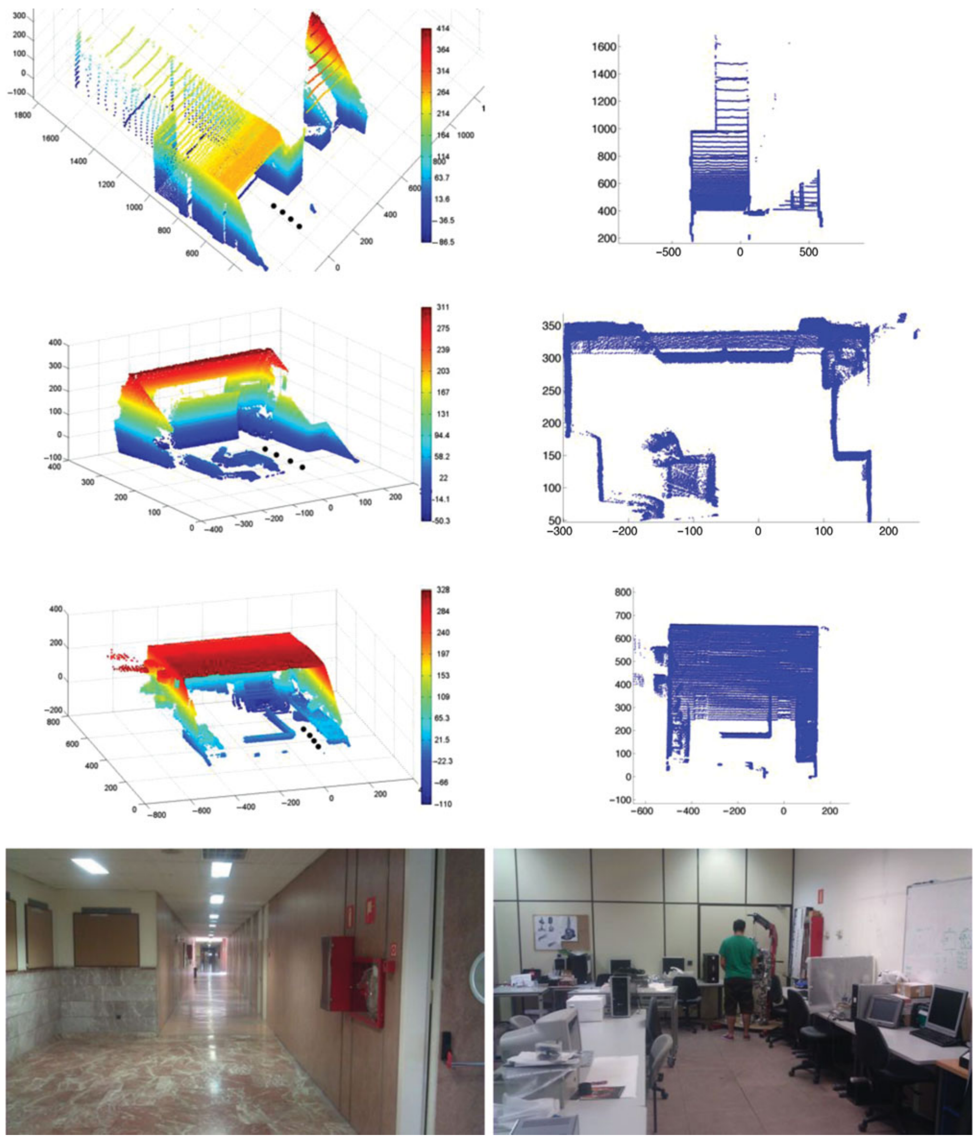


Fig. 3. (Colour online) Local maps after registration. $UC3M$ data set. Left: 3D view. Robot positions in black dots. Right: horizontal projection. All units are in centimeters.

Table I. Scan matching errors for the $Hannover2$ data set given the ground truth. Average error (standard deviation). Distances are in cm. Orientation is in degrees.

	Δx	Δy	Δz	$\Delta\varphi$	$\Delta\theta$	$\Delta\psi$
DE	3.18 (2.26)	6.58 (3.62)	0.90 (0.46)	0.05 (0.04)	0.21 (0.12)	0.97 (0.61)
ICP	7.30 (5.59)	5.80 (2.80)	1.14 (0.58)	0.07 (0.05)	0.22 (0.13)	0.65 (0.34)

the same point in space. The DE method is particle-based and it is less dependent on an accurate initial estimate. The robot follows a straight path in the corridors and it rotates up to 30° between consecutive scans in the labs. Our method accuracy is slightly better than the ICP-based scan matching accuracy in the labs shown in Fig. 3. However, this is a particular experiment and cannot be considered conclusive. Both methods have problems with sharp turns. In our future work, a detailed study about the DE-based scan matching under these conditions must be done.

A map of a large area composed of multiple measurements has been analyzed in the second experiment. The $Hannover2$ data set has been utilized in this case. Figure 4 shows a map built after registration, which is composed of 500 laser scans.

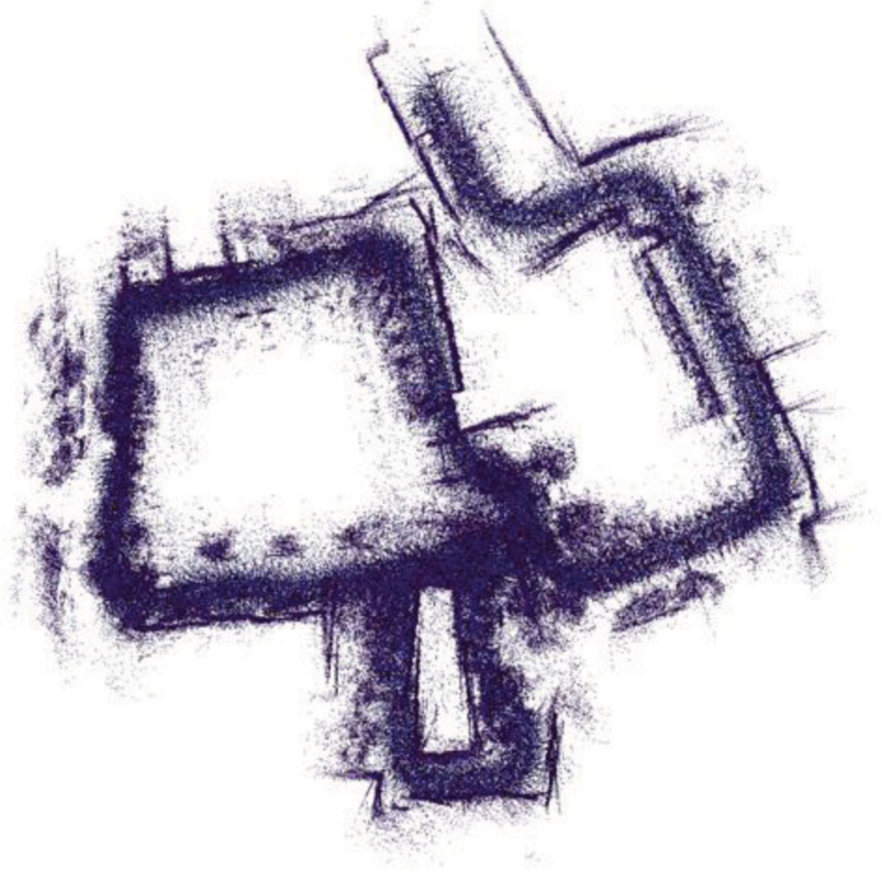


Fig. 4. (Colour online) Map of a large area composed of 500 laser scans.

The robot's trajectory starts at a point of the rectangle that is located in the bottom part of the figure and ends in the upper part of the figure. It can be observed that the map has an acceptable appearance, but the accumulated error is specially important in the last part or the robot's path. Although accurate results have been obtained after registration of scan pairs, the accumulated error makes the development of a loop detection and closure algorithm necessary.

As the ground truth is available for the $Hannover2$ data set, the scan matching average error using multiple successful cases has been computed, and it is presented in Table I. An example of successful matching between two frames with higher resolution is presented in Fig. 5. It contains the 3D view and the horizontal projection. The accuracy of the DE-based method is about a few centimetres using frames with ranges equal to 30 m. When comparing our method to ICP, the error with DE is slightly lower. Diosi and Kleeman[8] have estimated an average error equal to 3.8 cm and 0.86° using PSM in planar maps. Bosse and Zlot[5] have obtained an error equal to 11.3 cm, also in planar maps.

The average time needed to register a newly acquired scan and incorporate it into the model is equal to 1.83 s when using the DE-based scan matching. This time is equal to 1.73 s for the ICP-based algorithm. The preprocessing step is included in these times. Both times are low enough to use the algorithms online, taking care of the minimum time between two consecutive scans.

The algorithm complexity is $O(iter \cdot N_P \cdot n \cdot \log n)$, which means that it grows linearly with the population size (N_P), the iterations ($iter$), and the sensor size after preprocessing (n). The term $\log n$ corresponds to the average of the NN search accelerated with k-d trees. It does not depend on the DOF, which can be easily expanded if necessary.

6.3. Loop detection

The algorithm has been tested using the $Hannover2$ data set. The configuration parameters are: $K = 10$, $t_l = t_p = 0.1$, $N_s = 9$. Different values of N_n and configurations of *numerical* features have been tested. Thresholds t_1 and t_2 have been also studied.

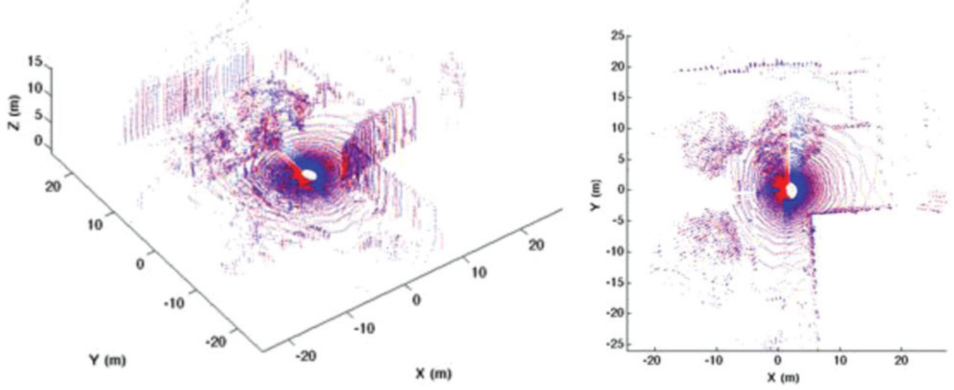


Fig. 5. (Colour online) Scan matching between a pair of scans of the $Hannover2$ data set. Each frame is in different color. Left: 3D view. Right: horizontal projection. Units are in meters.

The loop detection matrix between all possibilities, given a robot's trajectory composed of 375 consecutive scans, is shown in Fig. 6. The thresholds are set to $t_1 = 93\%$ and $t_2 = 90\%$. The positive matching is represented by white points, gray means uncertainty band, and black is negative matching. The LPI values between all pairs have been calculated. It is a symmetric matrix where the axis indicates the scan number (which are consecutively numbered starting by the first reading). The robot's path can be observed at the bottom of the figure. This type of experiment has been also shown in the works by Bosse and Zlot,[5] Granstrom *et al.*,[11] and Magnusson *et al.*[17]

Analyzing the robot's path, several loops must be detected. The first loop detected is located around the coordinates (110, 0) of the matrix (or (0, 110) because the matrix is symmetrical). There is not only one white point because the robot's path after the first loop detection coincides with the previous one. The loop size variable is reset after detection in a real application, therefore no more loops around these coordinates will be detected after the first positive. It is the first time that the robot is navigating around a pre-visited place and is correctly detected.

After that the largest close trajectory is described by the robot and the starting point is visited again. This second visit that is also classified as positive matching has approximate coordinates equal to (340, 0). Finally, the third loop is detected around coordinates (110, 340). It is a logical detection because in both cases the robot is visiting the starting point. No more loops are appreciated in the matrix, which is a correct result because there are no more coincident locations in the robot's path.

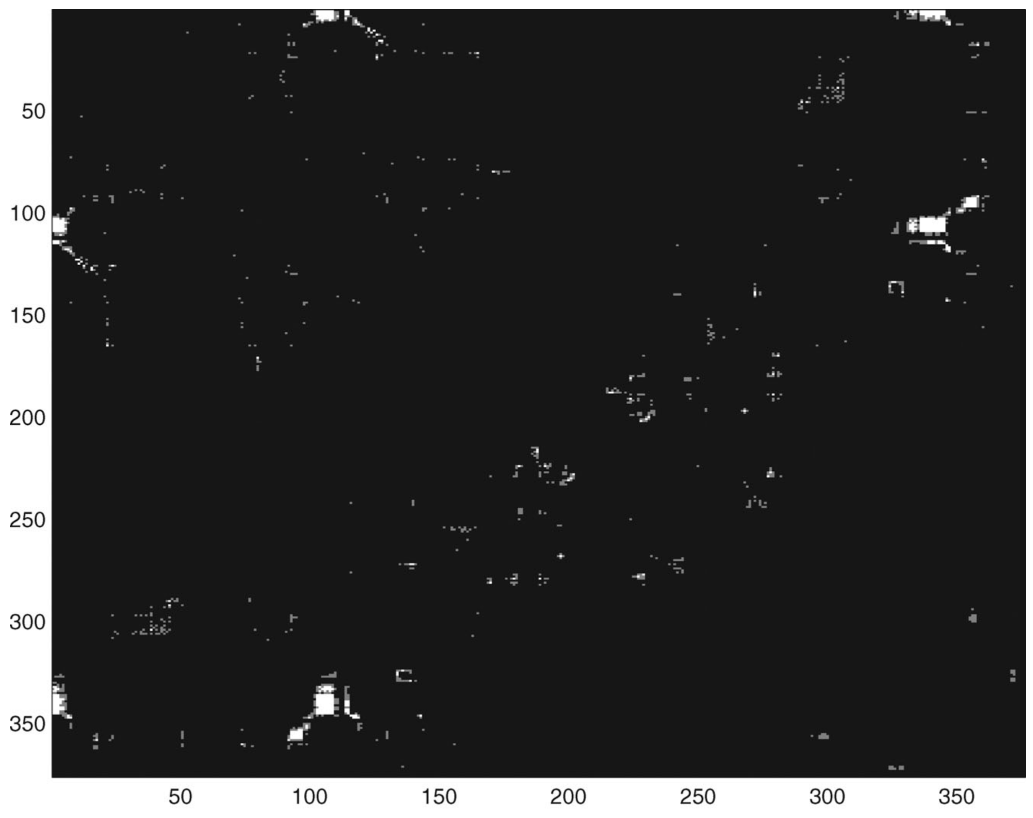
Since all loops are correctly detected and no false detection is appreciated, it can be concluded that the loop detection algorithm presents an optimal performance.

An additional question that needs to be answered is how to fix thresholds t_1 and t_2. A preliminary heuristic study has been developed in this paper. Figure 7 shows the probability histogram between 150 different scans of the $Hannover2$ data set. All possible combinations have been computed. The LPI descriptor is composed of three *numerical* features (volume, distance, and curvature) and nine *surface* features.

The histogram is composed of two Gaussian-shaped distributions. If the loops (positive matching) are contained in the right one, the thresholds can be chosen depending on right distribution properties. The distribution average is equal to 95.04% and the standard deviation is 1.66%. A conservative choice for t_1 could be the lower limit of probability distribution, which is around 90%. But this will be discussed in the next experiment.

A different experiment has been designed to quantify the loop detection performance. The ground truth distance of the $Hannover2$ data set is available, and a distance threshold (d_{gt}) has been defined to calculate two probabilities:

- Probability of Detection (PD): Number of pairs classified as positive matching and a distance between them lower than d_{gt} divided by the total number of pairs located closer than d_{gt}.
- Probability of False Alarm (PFA): Number of pairs classified as positive matching and a distance between them greater than d_{gt} divided by the total number of pairs located farther than d_{gt}.



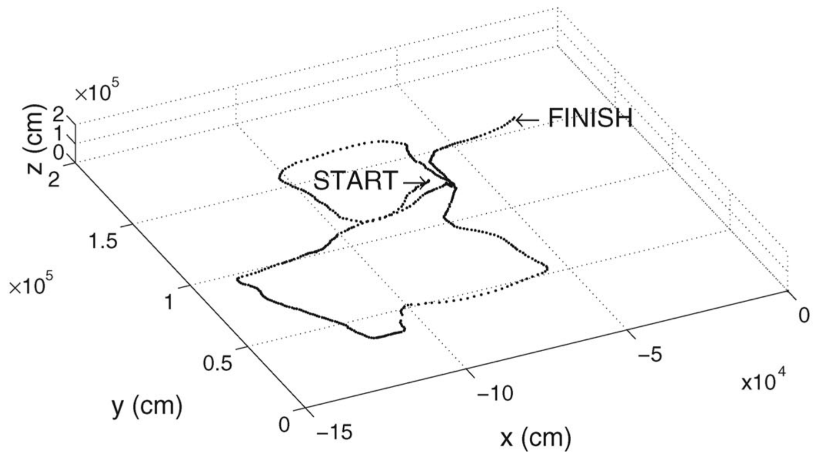


Fig. 6. Loop detection matrix between 375 consecutive scans ($t_1 = 93\%$, $t_2 = 90\%$): white represents positive matching, gray means uncertainty band, and black is negative matching. Bottom: robot's path.

If the ground truth distance between a pair of scans is lower than the threshold, the loop detection algorithm must classify this pair as positive. The PD must be the highest possible, but high values for the PFA are critical, understanding the PFA as the probability of detecting a loop when it does not exist. In addition, one single false detection makes the whole mapping process collapse. Therefore, the threshold t_1 will be the highest PD value that keeps the PFA at minimum.

The influence of the most important variables on the PD and the PFA has been analyzed. The first aspect to be studied are the weights of the LPI function. Since three different weights have been considered (Eqs. (5.1.2) and (5.1.3)), each one will be represented in a different table.

The second factor is the number of *numerical* features (N_n). The feature descriptor is composed of nine *surface* features, and different possibilities for N_n have been tested: $N_n = 0$, $N_n = 3$ (volume,

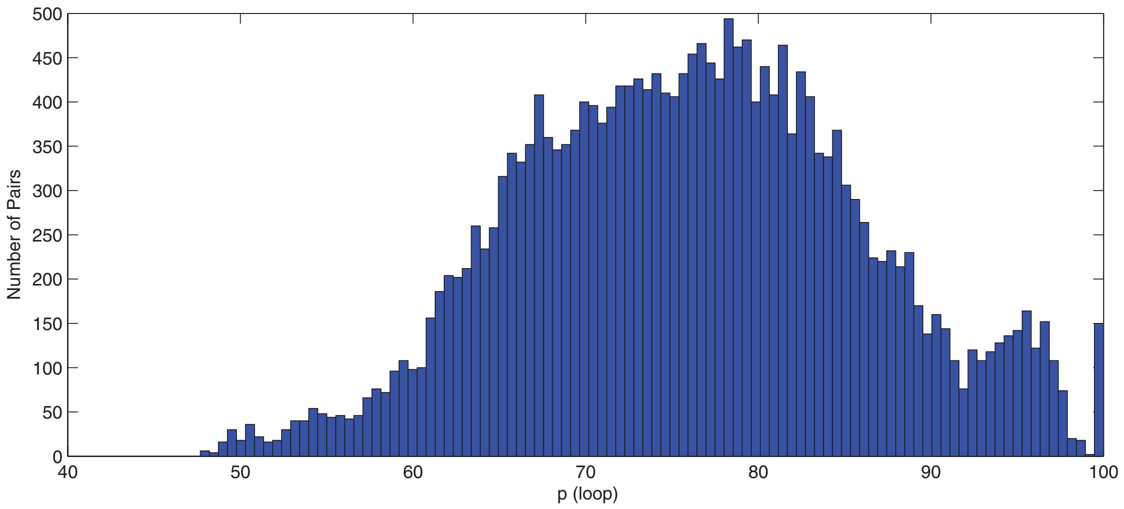


Fig. 7. (Colour online) Probability histogram between 150 different scans of the $Hannover2$ data set.

distance, and curvature mean), and $N_n = 12$ (volume, distance, curvature mean, curvature standard deviation, range standard deviation, range mean, range kurtosis, centroid, distance to centroid mean, distance to centroid standard deviation, range differences mean, and range differences standard deviation).

The third factor is the uncertainty band. If the LPI value between a pair of scans is inside the uncertainty band, then the third scan is considered. The loop is now detected depending on the new LPI value using the new acquired scan (LPI $\geq t_1$). It means that when the robot is checking if it is navigating around a previously visited place and the LPI value is inside the uncertainty band, it waits for a new laser reading and uses it to improve its loop detection capabilities. Three different options have been considered: no uncertainty band ($t_1 = t_2$), narrow band ($t_1 = t_2 + 3$), and wide band ($t_1 = t_2 + 10$).

The highest PD values that keep the PFA at the minimum depending on the previously explained factors have been represented in Tables II, III, and IV. The experiment consists of decreasing t_1 (and t_2, because this value is a function of t_1) while checking the PFA. The value of t_1 that is chosen as critical is the minimum value that keeps the PFA at the minimum. The PD is calculated for this critical value of t_1. Only these critical values are represented for simplicity.

The improvement introduced by the uncertainty band can be observed in the tables. The results with a wide band are slightly better than those when a narrow band is chosen. The algorithm performance is also improved by the introduction of *numerical* features. However, it is not possible to conclude if a feature descriptor with $N_n = 12$ is better than a feature descriptor with $N_n = 3$. Finally, there are no important differences when comparing the results using different weights.

The best results are obtained using the average between features as weight, three *numerical* features, and a wide uncertainty band. In this case, the probability of detection is equal to 38.67%. This probability is high enough to be successfully applied to loop detection. The results are similar or even better than those obtained by other authors. Magnusson *et al.*[17] show a PD = 35.3% with the same data set; Granstrom *et al.*[11] obtain a PD = 63% with no false positives, but their algorithm is included in a machine learning scope. It is a completely different focus and both algorithms cannot be compared because their method needs a large amount of initial data to build the classifier. The loop detection problem cannot be included in this scope because an autonomous robot has to be able to build a robust model even with no initial information.

Besides, d_{gt} is equal to 6 m, which is not a very restrictive assumption. The information received from two different locations with distance between them equal to 6 m can be very different. If this value is reduced to 3 m, the PD is increased to 65.2% while keeping the PFA at the minimum. This is a very promising result.

The PD and the PFA are represented as functions of t_1 for the best option of the previous experiment in Fig. 8. There are two graphics that correspond to $d_{gt} = 6$ m and 3 m. We have built this graphic

Table II. The highest probability of loop detection with the minimum probability of false alarm depending on N_n and the uncertainty band. Both values are in percentage. PD(PFA) $w_i = \max(f_i, g_i)$.

N_n	$t_1 = t_2$	$t_1 = t_2 + 3$	$t_1 = t_2 + 10$
0	23.53 (0.66)	27.13 (1.19)	22.93 (0.86)
3	26.53 (0.86)	29.73 (1.26)	32.80 (1.20)
12	28.00 (0.60)	33.20 (1.07)	**35.53** (0.80)

Table III. The highest probability of loop detection with the minimum probability of false alarm depending on N_n and the uncertainty band. Both values are in percentage. PD(PFA) $w_i = \text{mean}(f_i, g_i)$.

N_n	$t_1 = t_2$	$t_1 = t_2 + 3$	$t_1 = t_2 + 10$
0	26.27 (1.19)	29.67 (1.20)	32.4 (1.13)
3	31.80 (0.66)	32.60 (1.13)	**38.67** (1.00)
12	28.20 (1.20)	30.67 (1.00)	33.53 (1.06)

Table IV. The Highest probability of loop detection with the minimum probability of false alarm depending on N_n and the uncertainty band. Both values are in percentage. PD(PFA) $w_i = \sum_{j=1}^{T} \max(f_i, g_i)/T$.

N_n	$t_1 = t_2$	$t_1 = t_2 + 3$	$t_1 = t_2 + 10$
0	22.93 (1.07)	25.80 (0.93)	25.20 (0.66)
3	26.53 (0.60)	29.93 (0.93)	30.20 (0.80)
12	28.47 (0.86)	32.80 (1.00)	**34.73** (0.60)

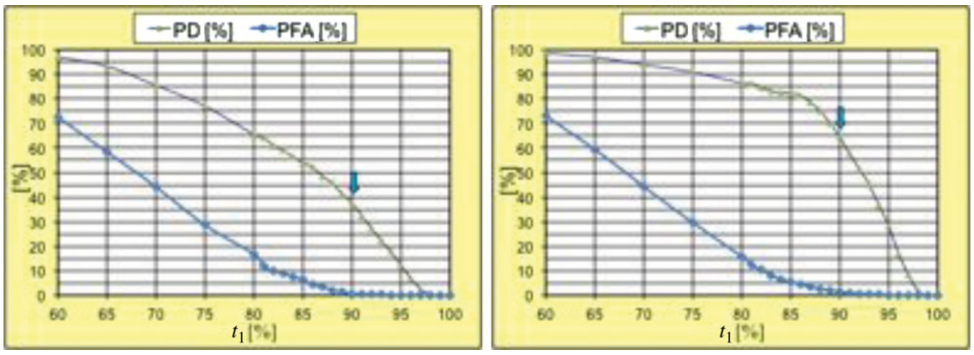


Fig. 8. (Colour online) PD and PFA as functions of t_1. Settings: $w_i = \text{mean}(f_i, g_i)$, $N_n = 3$, $t_1 = t_2 + 10$. Left: $d_{gt} = 6$ m. Right: $d_{gt} = 3$ m.

to calculate the value that is shown in Table III. It can be observed in the left part of the figure that the highest PD that keeps the PFA at the minimum is the same that is shown in Table III (38.67%). The optimal value of t_1 is around 90%, which is similar to the heuristic value deduced above in this section. The increase of the PFA starts when t_1 is below this value.

A final experiment has been developed to test the LPI and its thresholds in a different environment. A different data set recorded at the University Campus of Koblenz, Germany, has been used. It contains 333 3D scans (360° field of view), covering a closed path of two laps around the university campus. This data set is available online.[3] The experiment is similar to the first one detailed in this section. The loop detection matrix between all possibilities is computed. The algorithm has been configured with the parameters other than the mentioned experiment.

[3] `http://kos.informatik.uni- osnabrueck.de/3Dscans/`. Thanks to Johannes Pellenz and Dagmar Lang for making available this data set.

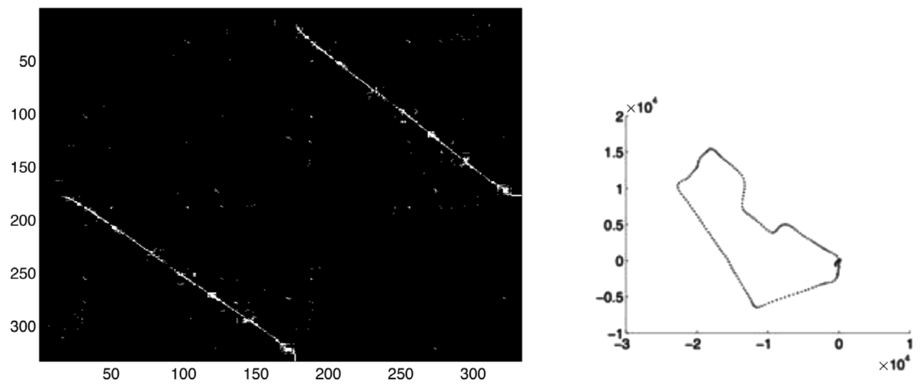


Fig. 9. Left: Loop detection matrix, Koblenz data set, 333 scans ($t_1 = 93\%$, $t_2 = 90\%$). Right: First lap of the robot's path. Units are in centimeters.

The results are shown in Fig. 9. In the left part, the loop detection matrix is represented. The first lap of the vehicle (horizontal projection) is in the right part. The same trajectory is repeated in the second lap. As observed in the figure, the first loop is detected around coordinates (175, 0). This loop occurs when the starting point is visited for the second time. After that the same trajectory is covered in the second lap. This corresponds to the diagonal white line that appears in the matrix. It is possible to conclude that all loops are correctly detected and there are no false detections for this data set without changing the configuration parameters.

The computational cost of the implemented method is an important aspect because it is necessary to handle a large amount of information provided by 3D sensors. The time complexity of a pairwise comparison is $O(n)$, where n is the laser scan size. The computational cost can be separated into three components: data reduction (0.016 s for a scan), features extraction (0.090 s for a scan), and loop detection (0.002 s for a pairwise comparison). Average times are given in parentheses. The feature vectors are stored. When a new scan is obtained and the number of scans already in the map is M, the computational cost will be equal to $0.106 + 0.002 \times M$ s, in average, when checking all possible loops.

7. Conclusions

Two different tools related to the mobile robot mapping problem have been developed in this paper. The first one is a DE-based scan matching algorithm that works in 3D environments (6 DOF). We have also designed a loop detection method that extracts the most important features from two different 3D laser scans in order to obtain an indicator that is used as a threshold to detect when the robot is visiting a known place.

The DE-based method's local accuracy is high enough to apply this algorithm in manipulation tasks. It is a logical result because a high accuracy has been obtained in our previous work by a similar method for robot global localization.

The algorithm could be used online depending on the time needed between two consecutive readings. The computational cost is the most important shortcoming of the 3D mapping and our algorithm presents a good performance in this aspect because it does not depend on the DOF but on the perceptive sensor number of measurements and the population size.

The DE method has many other characteristics that make its use very interesting: It can deal with nonlinear state space dynamics and noise distributions, it does not require any assumptions on the shape of the posterior density, the computational resources are focused on the most relevant areas, the algorithm is able to cope with a high level of sensor noise with low degradation of the estimation results, etc.

The LPI has been demonstrated to be an efficient tool to be used in mapping problems. All true loops are correctly detected and no false detections are appreciated when the mobile robot is covering a long trajectory and there is one place that is visited several times.

The results are similar or even better than those obtained by other research groups when analyzing the probability of detection. The introduction of different types of features and the uncertainty band improve the algorithm performance and make it a more versatile method because it admits different settings.

Since the LPI formula does not depend on any previous state and its weights are updated online, the algorithm can be classified as an online learning method, which is a more appropriate approach for an autonomous robot when exploring the environment. The mobile robots do not have a wide amount of perceptive data at the beginning, which makes it difficult to apply machine learning algorithms to build loop detection classifiers.

In addition, a wide variety of scan properties can be contained in this descriptor and different weights have been tested. A detailed study of the descriptor and the influence of different factors have been shown in experimental results.

Finally, the computational cost of the proposed method makes it possible to use the algorithm in real-time applications.

References

1. J. L. Bentley, "Multidimensional binary search trees used for associative searching," *Commun. ACM* **18**, 509–517 (1975).
2. P. J. Besl and N. D. McKay, "A method for registration of 3D shapes," *IEEE Trans. Pattern Anal. Mach. Intell.* **14**(2), 239–256 (1992).
3. P. Biber and W. Straβer, "The Normal Distributions Transform: A New Approach to Laser Scan Matching," *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS'03)*, Las Vegas, USA (2003).
4. M. Bosse and J. Roberts, "Historgram Matching and Global Initialization for Laser-Only SLAM in Large Unstructured Environments," *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA'07)*, Roma, Italy (Apr. 10–14, 2007).
5. M. Bosse and R. Zlot, "Map matching and data association for large-scale 2D laser scan-based SLAM," *Int. J. Robot. Res.* **27**, 667–691 (2008).
6. D. M. Cole and P. M. Newman, "Using Laser Range Data for 3D SLAM in Outdoor Environments," *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA'06)*, Orlando, USA (2006).
7. M. Cummings and P. Newman, "FAB-MAP: Probabilistic localization and mapping in the space of appearance," *Int. J. Robot. Res.* **27**(6), 647–665 (Jun. 2008).
8. A. Diosi and L. Kleeman, "Laser Scan Matching in Polar Coordinates with Application to SLAM," *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS'05)*, Edmonton, Canada (2005).
9. Y. Freund and R. E Schaphire, "A decision-theoretic generalization of on-line learning and an application to boosting," *J. Comput. Syst. Sci.* **55**, 119–139 (1997).
10. C. Früh and A. Zakhor, "3D Model Generation for Cities Using Aerial Photographs and Ground Level Laser Scans. *Proceedings of the Computer Vision and Pattern Recognition Conference (CVPR'01)*, Kauai, Hawaii, USA (2001).
11. K. Granström, T. B Schön, J. I Nieto, and F. T Ramos, "Learning to close loops from range data," *The International Journal of Robotics Research*, **30**(14), 1728–1754 (2011).
12. G. Grisetti, C. Stachniss, S. Grzonka and W. Burgard, "A Tree Parameterization for Efficiently Computing Maximum Likelihood Maps Using Gradient Descent," *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA'08)*, Pasadena, CA, USA (2008).
13. D. Hähnel, W. Burgard and S. Thrun, "Learning compact 3D models of indoor and outdoor environments with a mobile robot," *Robot. Auton. Syst.* **44**, 15–27 (2003).
14. J. J. Leonard and H. Durrant-Whyte, "Mobile robot localization by tracking geometric beacons," *IEEE Trans. Robot. Autom.* **7**, 376–382 (1991).
15. F. Lu and E. Milios, "Robot pose estimation in unknown environments by matching 2D range scans," *J. Intell. Robot. Syst.* **20**, 249–275 (1997).
16. M. Magnusson and T. Ducket, "A Comparison of 3D Registration Algorithms for Autonomous Underground Mining Vehicles," *Proceedings of the Second European Conference on Mobile Robotics*, Ancona, Italy (2005).
17. M. Magnusson, H. Andreassonand, A. Nüchter and Achim J. Lilienthal, "Automatic appearance-based loop detection from three-dimensional laser data using the normal distributions transform," *J. Field Robot.* **26**, 892–914 (2009).
18. F. Martín, L. Moreno, S. Garrido and D. Blanco, "High-accuracy global localization filter for three-dimensional environments," *Robotica* **30**, 363–378 (2011).
19. L. Moreno, S. Garrido and M. L. Muñoz, "Evolutionary filter for robust mobile robot localization," *Robot. Auton. Syst.* **54**(7), 590–600 (2006).
20. A. Nüchter, K. Lingemann and J. Hertzberg, "6D SLAM-3D mapping outdoor environments," *J. Field Robot.* **24**, 699–722 (2007).

21. S. Rusinkiewicz and M. Levoy, “Efficient Variants of the ICP Algorithm,” *Proceedings of the Third International Conference on 3D Digital Imaging and Modeling*, Quebec City, Canada (2001).
22. S. Se, D. G. Lowe and J. J. Little, “Vision-based global localization and mapping for mobile robots,” *IEEE Trans. Robot.* **21**, 3 (2005).
23. R. Smith, M. Self and P. Cheeseman, “Estimating Uncertain Spatial Relationships in Robotics,” **In**: *Proceedings of the Second Conference Annual Conference on Uncertainty in Artificial Intelligence (UAI’86)* (New York, NY: Elsevier Science, 1986) pp. 267–288.
24. R. Storn and K. Price, “Differential evolution - A simple and efficient heuristic for global optimization over continuous spaces,” *J. Glob. Optim.* 11, 341–359 (Dec. 1997).
25. H. Surmann, A. Nüchter, K. Lingemann and J. Hertzberg, “6D SLAM – Preliminary Report on Closing The Loop in Six Dimensions,” *Proceedings of the 5th IFAC Symposium on Intelligent Autonomous Vehicles (IAV’04)*, Lisbon (2004).
26. H. Surmann, A. Nüchter and J. Hertzberg, “An autonomous mobile robot with a 3D laser range finder for 3D exploration and digitalization of indoor environments,” *Robot. Auton. Syst.* **45**(3–4), 181–198 (2003).
27. S. Thrun, W. Burgard and D. Fox, “A Real-Time Algorithm for Mobile Robot Mapping with Applications to Multi-Robot and 3D Mapping,” *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA’00)*, San Francisco, USA (2000).
28. S. Thrun, M. Montemerlo and A. Aron, “Probabilistic Terrain Analysis for High-Speed Desert Driving,” In: *Robotics: Science and Systems* II, University of Pennsylvania, Philadelphia, Pennsylvania, USA (Aug. 16–19, 2006) (Cambridge, MA, USA).
29. M. Tomono, “A Scan Matching Method Using Euclidean Invariant Signature for Global Localization and Map Building,” *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA’04)*, New Orleans, USA (2004).
30. R. Triebel, P. Pfaff and W. Burgard, “Multi-Level Surface Maps for Outdoor Terrain Mapping and Loop Closing,” *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS’06)*, Beijing, China (2006).
31. O. Wulf, K. O. Arras, H. I. Christensen and B. A. Wagner, “2D Mapping of Cluttered Indoor Environments by Means of 3D Perception,” *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA’04)*, New Orleans, USA (Apr. 2004).
32. Z. Zhang, “Iterative point matching for registration of free-form curves and surfaces,” *Int. J. Comput. Vis.* **13**, 119–152 (1994).