Technical Report

IRI-TR-08-06

Registration of 3D Points Clouds for Urban Robot Mapping

Ernesto Homar Teniente Avilés
Juan Andrade Cetto

Institut de Robòtica i Informàtica Industrial

Abstract

We consider the task of mapping pedestrian urban areas for a robotic guidance and surveillance application. This mapping is performed by registering three-dimensional laser range scans acquired with two different robots.

To solve this task we will use the Iterative Closes Point (ICP) algorithm proposed in [8], but for the minimization step we will use the metric proposed by Biota et al. [10] trying to get advantage of the compensation between translation and rotation they mention. To reduce computational cost in the original ICP during matching, the original data is sampled with the library Aproximate Neares Neighbor (ANN). Finally we propose a hierarchical new correspondence search strategy, using a point-to-plane strategy at the highest level and the point-to-point metric at finer levels. At the highest level the adjust error between a plane and it's n adjacent points describing the plane is computed, if this error is bigger than a threshold then we change the level.

Institut de Robòtica i Informàtica Industrial (IRI)
Consejo Superior de Investigaciones Científicas (CSIC)
Universitat Politècnica de Catalunya (UPC)
Llorens i Artigas 4-6, 08028, Barcelona
Spain
Tel (fax): +34 93 401 5750 (5751)
`http://www-iri.upc.es`

Corresponding author:
Ernesto H. Teniente Avilés
tel: +34 93 401 5780
`ehomar@iri.upc.edu`
`http://www-iri.upc.es/people/ehomar`

Contents

1 Problem Statement

We consider the task of mapping pedestrian urban areas for a robotic guidance and surveillance application. Multiple 3D scans are necessary to build a map with enough environment information. To create a correct and consistent map, the scans must to have a common reference coordinate system. This process is called registration. If the 3D systems were precisely locate with the robots odometry, the registration could be done directly with this information. However, because the uncertainty on the robot sensors, self localization is erroneous, so a method to have a correct overlap of the 3D scans has to be considered. The mapping is performed by registering 3D laser range scans acquired with two different robots. This mapping application is part of the EU URUS Project (Ubiquitous Network Robotics in Urban Settings).

2 State of the art

Scan matching algorithms are often used in mobile robotics to correct the relative motion of a vehicle between two consecutive configurations, by maximizing the overlap between the range measurements obtained at each configuration. The most popular scan matching methods [31] are based on the ICP from Besl and Mckey [8] which is borrowed from the computer vision community. The objective of this algorithm is; to compute the relative motion between two data sets partially overlapped. The algorithm iteratively minimizes the MSE and proceeds as follows: first, for each point in one data set, the closest point in the second one is found or vise versa (correspondence step), then the motion that minimizes the Mean Square Error (MSE) between the correspondences is computed (registration step), finally the data shape is updated (update step).

In the registration proposed by Besl and Mckey a point-to-point metric is used to measure the "closeness" of data, they also suggest an accelerated version of the algorithm by using a linear approximation and a parabolic interpolation with the last three minimization vectors if they are well aligned, which means they have been moving in an approximately constant direction. The use of sampling or tree-based-search to speed up the algorithm are mentioned as future refinements to reduce the computational cost. Chen and Medioni [13] proposed a point-to-plane error metric, which makes the algorithm less susceptible to local minima than the metric proposed by Besl and Mackey [8]. The idea in [13] is, that given a so called control point in the first surface, to compute the distance to the nearest tangent plane in the second cloud. Blais [11] suggests a point-to-projection solution computing the sum of the distances between the two range views in the direction of the rays. This has also been called "reverse calibration". This approach makes registration very fast because it does not involve any search step to find the correspondences. However, one of its disadvantages is that the resulting registration is not as accurate as the one given in the point-to-point and point-to-plane metrics [31]. Turk and Levoy [35] proposed a point-to-triangle correspondence using meshes which they call zippered meshes finding the nearest position on a mesh to each vertex of an other mesh. They find the rigid transformation that minimizes a weighted least-squared distance between correspondences with a value in the range from 0 to 1 called confidence. For the case of structured light scanners, they measure the the confidence of a point on a mesh to be the angle between the mesh normal and the sensor viewing angle. The confidence is a measure of how certain they are of a given range point's position, which helps to eliminate possible wrong matches. Chetverikov et al. [15] presented a robustified extension of the ICP applicable to overlaps under 50%, robust to erroneous and incomplete measurements, and has easy-to-set parameters called Trimmed ICP (TrICP). The algorithm is based on the consistent use of the Least Trimmed Square [30] in all phases of the operation, on the other hand Yamany et al. [1] used genetic algorithms maximizing an objective function, where the genes are formed by concatenating six binary coded parameters, representing the three angles of rotation and the 3 dof for translations. More recently, a new error metric which explores the compensation between the rotation and translation was proposed by Minguez et al. [23, 24] for the 2D space and Biota et al. [9, 10] extended this metric to the 3D space. They used a point-to-projection minimization using triangles as the projection surface, and perform the Nearest Neighbor (NN) correspondences in the space of the new metric space. They did not tackle the computational complexity of the algorithm in any way, so it is understood they use brute force search to do the matching.

The ICP bottleneck in time execution is when searching for point matches. One strategy to reduce the computational complexity is to use tree-based search techniques [28]. Nütcher et al.[25] uses a library called Approximate Nearest Neigborh (ANN), developed by Arya and Mount [5], that uses a balanced kd-tree or box decomposition tree (bd-trees). Also Nütcher et al. [25] proved that kd-trees are faster than bd-trees to the NN problem. Simon et al.[32] used a kd-tree but using a catching points technique, where in the first iteration n neighbors for all the points in the reference cloud are found from the model query and they are cached. It is assumed that after updating the reference cloud, the cached points for each point in the updated cloud should be neighbors. Benjemaa [6] used a double z-buffer structure which provides an explicit space partitioning. Yamany et al. [1] said that the time matching can be significantly reduced by applying a grid closest point (GCP) technique. The GCP is an other sampling scheme which consists of superimposing a 3D fine grid on the 3D space such that the two clouds lie inside the grid, dividing the 3D space on cells, each cell with an index of its closest point in the model set. Greenspan and Godin [17] presented a new solution for the NN search for the matching step which they called Spherical Triangle Constraint. Like Simon et al. [32], they store correspondences at each iteration so that these are available at the next iteration. The Spherical Constraint is applied to determine whether or not the nearest neighbor falls within the neighborhood of each point estimate, and if so, the Triangle Constraint and the Ordering Theorem, are applied to the neighborhood to quickly identify the correspondence. The Ordering Theorem orders a set of points by increasing distance to some point. They shown that after aprox. 20 iterations, their method is more efficient than kd-trees in computational complexity and time execution. More recently Akca and Gruen[2] used a box structure [14] which partitions the search space into boxes, where for a given surface element, the correspondence is searched only in the box containing this element and in the adjacent boxes, the correspondence is searched in the boxing structure during the first few iterations, and in the meantime its evolution is tracked across the iterations. In the end, the searching process is carried out only in an adaptive local neighborhood according to the previous position and change of correspondence. One of the main advantages of the box structure is that it has a faster and easier access mechanism than the tree-based search methods provide.

A another common strategy to accelerate the matching process is to reduce the number of points. Sampling the data reduces the match execution time by a constant factor, but retains linear asymptotic computational complexity. Coarse-to-fine strategies haven been used by Zhang [37] and Turk and Levoy [35]. They start using a less detailed description of the data and as the algorithm approaches the solution, the resolution is hierarchically increased. The techniques used for sampling data vary. Turk and Levoy [35] and Blais [11] used uniform sampling. Masuda et al. [22] used intead, random sampling for each iteration. Moreover, Nütcher et al. [27] and Gutmann [18] used a technique best suited to the nature of data for laser range finders so called, reduction filter, which has been shown to work well on realtime applications .

However, sampled methods are very sensitive to data content, i.e. noise level, occlusion areas, complexity of the range data, etc. If too many points come from outliers due to sensor errors, this may produce too many wrong correspondences, and may cause the solution to converge on a local minimum leading a poor final overlap, or in the worst case, to divergence. We shall remember that the original algorithm from Besl and Mackay considers data sets without outliers. Several approaches to dismiss possibles outliers have been proposed using rejection strategies. Rejections based on thresholds for the maximum tolerable distance between paired points were implemented by Turk and Levoy[35], the threshold is set as twice the maximum tolerable space between range point meshes; and is adaptively changed when building the the points in the mesh. Rejection that use a statistical method based on the distribution of point distances were used by Zhang [37], and Pulli [29], who used two thresholds for the maximum allowed distance between paired points, one of them dynamic. Masuda et al. [22] also rejects pair matches whose point-to-point distance are larger than some multiple of the standar deviation of distances. Pulli as well as Turk and Levoy [35] use not only statistical reasoning, but also topological information to discard matches. They removed matches that occur on mesh boundaries, and Zhang [37] and Pulli [29] used the angle between the normals of the paired points as a constrain to keep matches. When using a laser range data Nütcher et al. [27] use a median filter to remove the Gaussian noise for each scan row.

Another issue on the ICP is the rate of convergence of the minimization step. To accelerate such convergence and to reduce overshoot, some authors propose minor changes to the original extrapolation proposal of Besl and McKay. For instance, Rusinkiewicz and Levoy [31] used the update based on linear zero crossing of the line instead of the extremum of the parabola when quadratic extrapolation is attempted and the parabola opens downwards, and multiply the amount of extrapolation by a dampening factor, arbitrarily set to 0.5 in their implementation. Even when this occasionally reduces the benefit of extrapolation, it also increases stability and eliminates many problems with overshoot. Simon et al. [32] said that while Besl and McKay calculate a single acceleration scale factor for both translation and rotation, they decouple the acceleration of translation and rotation achieving better results.

3 Range image registration

In this section, the Iterative Closest Point (ICP) is described in detail. In the minimization step we will use the metric proposed by Biota et al. [10] trying to get advantage of the compensation between translation and rotation they mentioned. To reduce computational cost in the original ICP during matching, the original data is sampled and the NN search is done with the library Aproximate Neares Neighbor (ANN). Finally we propose a new hierarchical correspondence search strategy, using a point-to-plane metric at the highest level and the point-to-point search at finer levels. At the highest level the adjust error between a plane and its n adjacent points describing the plane is computed, if this error is bigger than an user specified threshold we change the level.

Given a set of 3D points, $S = \{p_1, p_2, ..., p_n\}$ acquired from a reference frame $q = [x, y, z, r_x, r_y, r_z]$, and $S' = \{p'_1, p'_2, ..., p'_m\}$ a set acquired from a new frame $q' = \left[x', y', z', r'_x, r'_y, r'_z\right]$, we need to estimate the relative displacement between the sensor poses at q and q'. The ICP deals with this problem in an iterative process in four steps [8]. At each iteration k, there is a search of correspondences between the points of both scans, then the relative displacement q_k is computed by a minimization process. The model scans are updated with the last computed q_k; this is repeated until convergence.

3.1 Correspondences Search

The basic idea in correspondence search is to find the closest point p_x to a given reference point p_q in a reference set S, and according to a specified metric, usually Euclidean distance. It is known as the Nearest Neighbor (NN) problem, also known as closest point search. A correspondence is established by means of the correspondence operator C, which is the closes point operator defined by:

$$C(p_q, S_{ref,k}) = \underbrace{argmin}_{p_x \in S_{ref,k}} \|(p_x - p_q)\| \tag{1}$$

Specifically our problem is to find for each point p_i in S (in case that exist) the closest point p_{ix} in S', resulting in a subset Y of n correspondences (p_j, p_{jx}). Then for each iteration k a subset Y_k is given by:

$$Y_k = C(S, S'_k)$$

Besl and McKey [8] assume that for each point in reference set must be a correspondence in the data set, in our application this is not the case.

3.2 Registration Step

According with Biota et al [10], the registration problem is solved using a Last Square Minimization (LSM) to compute the q that minimize the realtive displacement between the two points sets. Given two associated points $p_i = (p_{ix}, p_{iy}, p_{iz})$ and $p''_i = (p''_{ix}, p''_{iy}, p''_{iz})$, the next expression needs to be minimized:

$$E_{dist}(q) = \sum_{i=1}^{n} d_p^{ap}(p_i, q(p''_i))^2 \tag{2}$$

The above equation could be expressed as follows

$$E_{dist}(q) = \delta_i^T(q) M \delta_i(q) \tag{3}$$

Where

$$\delta_i(q) = p_i - qp''_i \approx p_i - p''_i + U(p''_i)r - T$$

and

$$U(p''_i) = \begin{pmatrix} 0 & -p''_{iz} & p''_{iy} \\ p''_{iz} & 0 & -p''_{ix} \\ -p''_{iy} & p''_{ix} & 0 \end{pmatrix}$$

$$M = \frac{1}{k} \begin{pmatrix} p_{ix}^2 + L^2 & p_{ix}p_{iy} & p_{ix}p_{iz} \\ p_{ix}p_{iy} & p_{iy}^2 + L^2 & p_{iy}p_{iz} \\ p_{ix}p_{iz} & p_{iy}p_{iz} & p_{iz}^2 + L^2 \end{pmatrix}$$

where $k = \|p_i\|^2 + L^2$. Finally, the q that minimizes equation (3) is given by:

$$q_{min} = \left(\sum_{i=1}^{n} \begin{pmatrix} M & -MU(p_i'') \\ -U^T(p_i'')M & U^T(p_i'')MU(p_i'') \end{pmatrix} \right)^{-1} \cdot \sum_{i=1}^{n} \begin{pmatrix} M\delta \\ UM\delta \end{pmatrix}$$

4 Data association

In the ICP, finding the correspondences is the most computationally expensive step. Kd-trees are suggested in [8], demonstrated in [37] as an alternative, and later implemented in [32, 27] to speed up this step,

4.1 The kd-trees

The kd-trees are a generalization of the binary search trees. The idea behind this data structures (trees) is to extend the notion of a one dimension tree on a recursive subdivision of the space, i.e. for the 2D case the subdivision alternate in using the x or y coordinates to split Fig. 4.1(left) . Therefore we first split on x, next on y, then again on x, and so on. In general dimension, the kd-tree cycles among the various possible splitting dimensions. Each partition (of a point set) is represented by a node containing the two successor nodes or by a bounding box that contains the data points for this node Fig. 4.1(right). The root node represents the whole point cloud and the leafs form a disjunct partition of the set. As long as the number of data points associated with a node is greater than a small quantity, called the bucket size (Friedman et al. [16] proved that a bucket size of 1 is optimal), the box is split into two boxes by an axis-orthogonal hyperplane that intersects this box.

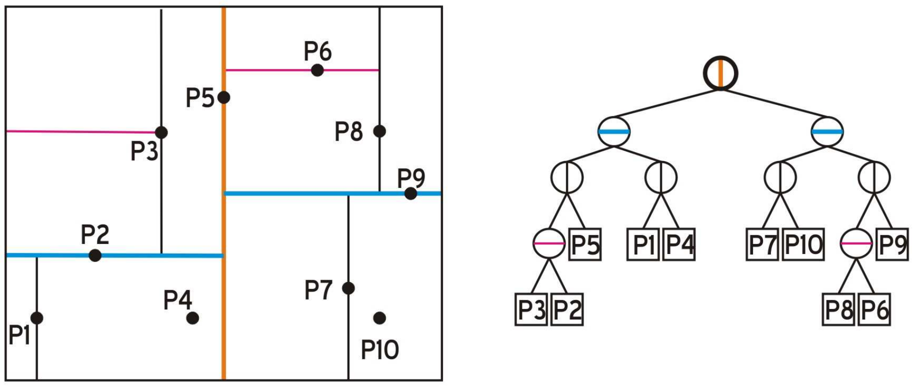


Figure 1: A kd-tree example: subdivided data (left) and the corresponding binary tree (right)

There are different splitting rules which determine how this hyperplane is selected. The choice of the splitting rule affects the shape of cells and the structure of the resulting tree.

- Standard splitting rule: The splitting dimension is the dimension of the maximum spread (difference between the maximum and minimum values), leading to many cells with high aspect ratio Fig. 2(a). The splitting point is the median of the coordinates along this dimension. A median partition of the points is then performed. This rule guarantees that the final tree has height ($log_2 n$), also guarantees that every kd-tree entry has the same probability. Friedman et al. [16] introduced this splitting rule in their definition of the optimized kd-tree.

- Midpoint splitting rule: When splitting the space, to guarantee that the tree is balanced, the most common method is the midpoint splitting rule. The splitting value is the median splitting coordinate Fig. 2(b). As a result, the tree will have $O(logn)$ height.

- Sliding-midpoint splitting rule: First a midpoint split is attempted. If the data points lie on both sides of the splitting plane then the splitting plane remains here. However, if all the data points lie to one side of the splitting plane, then splitting plane "slides" toward the data points until it encounters the first point. One child is a leaf cell containing this single point, and the algorithm recurses on the remaining points Fig. 2(c).

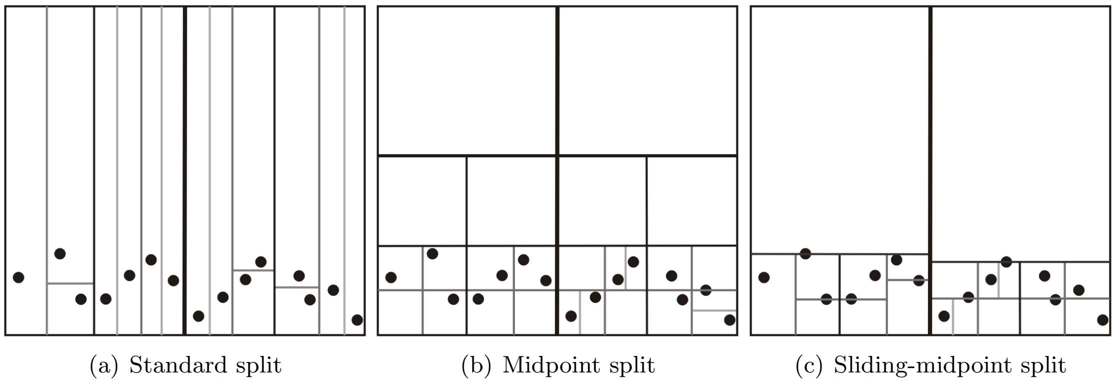

(a) Standard split (b) Midpoint split (c) Sliding-midpoint split

Figure 2: A kd-tree splitting rules example

4.2 Nearest Neighbor Search using kd-trees

In section 3.1, we mentioned the need to solve the NN correspondence problem as one of the step of the ICP algorithm. Now we are going to explain how to tackle this issue using kd-trees. A simple and naive way to approach the NN problem is by using brute force search, where the closest point is found computing the distance (given a metric) between each point in S_{ref} and p_q, this is highly expensive in computation time, $O(n)$ worst case with expected cost $log(n)$. Friedman et al. [16] showed that O(log n) query time is possible in the average case through the use of kd-trees. Their use ensures that the nearest data point to p_q could be found efficiently. High-dimensional (at least three) NN problems arise naturally when complex objects are represented by vectors of d numeric features.

Finding the NN to a given query point relies on the ability to discard large portions of the tree by performing a simple test. The tree is searched in a depth-first fashion and at each stage it makes an approximation to the nearest distance. When the algorithm decides that there cannot possibly be a closer point it terminates, giving the nearest neighbor.

First, the root node is examined with an initial assumption that the smallest distance to the next point is infinite. The subdomain (right or left), which is a hyperrectangle (in 3D space this is a rectangular prism), containing the target point is searched. This is done recursively until a final minimum region containing the node is found. The algorithm then (through recursion) examines each parent node, seeing if it is possible for the other domain to contain a point that is closer. This is performed by testing for the possibility of intersection between the hyperrectangle and the hypersphere (a plain sphere in 3D) formed by target node and distance to the current best NN estimate. If the rectangle that has not been recursively examined yet does not intersect this sphere, then there is no way that the rectangle can contain a point that is a better nearest neighbor. This is repeated until all domains are either searched or discarded, thus leaving the nearest neighbor as the final result. In addition the algorithm not only provides the NN, but also the square of the distance to the NN. Finding the nearest point is an $O(logN)$ operation.

4.3 A Nearest Neighbor Library

Like in [25] we use the Approximate Nearest Neighbor (ANN) library by Arya et al. [5]. ANN is a library of C++ objects and procedures that supports the NN search and the approximate nearest neighbor search. It is designed for data sets that can be stored in main memory. Points are assumed to be represented as coordinate vectors of reals. The distance between two points can be defined in many ways. ANN assumes that distances are measured using any class of distance functions called Minkowski metrics, including the Euclidean distance, Manhattan distance, and max distance. Preprocessing time and space are both linear in the number of points n and the dimension d. Thus the data structure requires storage that is only moderately larger than the underlying data set. Also it supports kd-trees [16, 7], and box-decomposition trees [5], it is able to use different methods for building these search structures and it also supports two methods for searching these structures: standard tree-ordered search [4] and priority search [5].

4.4 Point-to-point association

The distance between two associated points is normally defined by one of the Minkowski metrics such as the norm L2 (Euclidean distance) defined by :

$$dist(p, p'') = \left(\sum_{0 \le i < d} (p_i - p''_i)^2 \right)^{1/2} \tag{4}$$

where d is the dimension size of the data working with.
An alternative to the euclidean distance, could be defined by the metric proposed bye Biota et al. [9, 10], where a 3D rigid transformation is defined by a vector $q = (x, y, z, \theta n_x, \theta n_y, \theta n_z)$, which represents position and orientation $(-\pi < \theta < \pi)$ of a range finder laser sensor . Therefore, q norm is defined as:

$$\|q\| \; = \; \sqrt{x^2 + y^2 + z^2 + L^2\theta^2} \tag{5}$$

with $L \in \mathbb{R}^+$ for a distance. Given two points $p_1 = (p_{1x}, p_{1y}, p_{1z}) \in S$ and $p_2 = (p_{2x}, p_{2y}, p_{2z}) \in S'$, the distance between both points is:

$$d_p(p_1, p_2) \; = \; min \left\{ \|q\| \; \mid q(p_1) = p_2 \right\} \tag{6}$$

where

$$q(p_1) = R(n, \theta) p_1 + T \tag{7}$$

with T, the translation vector (x, y, z) and $R(n, \theta)$ the matrix of rotation angle θ about the unit vector $n = (n_x, n_y, n_z)$. Unfortunately there is no closed form expression of d_p with respect to the coordinates of the points. However a valid approximation could be computed with small rotations. Linearizing Eq. (7) about $\theta = 0$, we get $cos\theta \approx 1$ and $sin\theta \approx 0$. Developing Eq.(6)we obtain:

$$\begin{aligned} d_p^{ap}(p_1,p_2)^2 &= \delta^T M \delta && (8)\\ &= \|\delta\|^2 - \frac{\|p_1 \times \delta\|^2}{k} && (9)\\ & && (10)\end{aligned}$$

Working out the square value from Eq. (9), we get:

$$d_p^{ap}(p_1,p_2) = \sqrt{\|\delta\|^2 - \frac{\|p_1 \times \delta\|^2}{k}} \quad (11)$$

where $k = \|p_1\|^2 + L^2$, $\delta = p_2 - p_1$ and L is the balance the trade-off between translation and rotation, this is user especified. When $L \to \infty$ the new distance tends to the euclidean distance.

4.5 Point-to-plane association

The objective is to find the closest point in the plane to a given point x_i. The first step is, given a cloud of 3D data points, S_1, containing N individuals points $(p_1, p_2, ...p_N)$, to compute the nearest oriented tangent plane T_p to x_i. The correspondent nearest tangent plane to x_i is computed by finding the set of k nearest points to x_i in S_1, this set is called $S_{T_p,i} = (s_{T_p,1}, s_{T_p,2} ... s_{T_p,k})$. Then the plane could be represented by a point o_i, together with a unit normal vector $\hat{n}_i$, see Fig. (3).

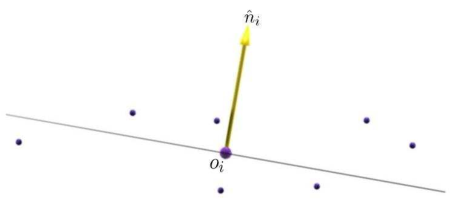


Figure 3: Points defining a plane

The point o_i, called center point is computed finding the media of S_{T_p} and accordingly with Cetto and Villamizar [3]. $\hat{n}_i$ could be obtained by computing an eigenvector of the 3x3 matrix R defined by:

$$R = Q - \frac{qq^T}{k^2}$$

where

$$q = \sum_{i=1}^{n} s_{T_p,i}$$
$$Q = \sum_{i=1}^{n} s_{T_p,i} s_{T_p,i}^T$$

then

$$\hat{n}_i = eig(R)$$

Once the plane is defined, it is necessary to find the minimum distance d_i between $T_{p,i}$ and x_i, where the signed distance is defined to be $d_i = (x_i - o_i) \cdot \hat{n}_i$. Finally with the knowledge of $\hat{n}_i$,o_i and d_i , it is possible to compute the point in the plane associated to d_i. Then $\vec{d_i} = \hat{n}_i \cdot d_i$ and from Fig. 4 :

$$\vec{w} = \vec{x_i} - \vec{o_i}$$
$$\vec{r_i} = \vec{w} - \vec{d_i} \tag{12}$$
$$\vec{s}_{T,i} = \vec{o_i} - \vec{r_i} \tag{13}$$

Substituting Eq. (12) on (13) and developing:

$$\begin{aligned} \vec{s}_{T,i} &= \vec{o_i} + \vec{w} - \vec{d_i} \\ &= \vec{x_i} - \vec{d_i} \end{aligned} \tag{14}$$

Finally the closes point in the plane correspondent to x_i is from Eq.(14), the point associated to $\vec{s}_{T,i}$

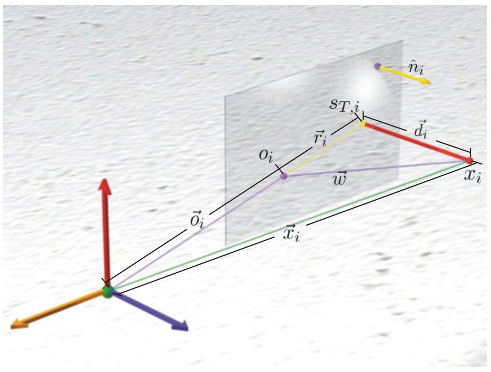


Figure 4: Query point x_i and its closest point $s_{T,i}$ in the plane

4.6 Data Reduction

When a scanned surface is too dense reducing the number of points is an other strategy used to decrease the execution time. It helps to reduce the NN execution time by a constant factor. Sampling the acquired sensor data preserving the geometric nature of the surface, is one of the the strategies we use, and have been proved in [31] to be an effective approach. On the other hand we limit the distances in the x, y an z directions.

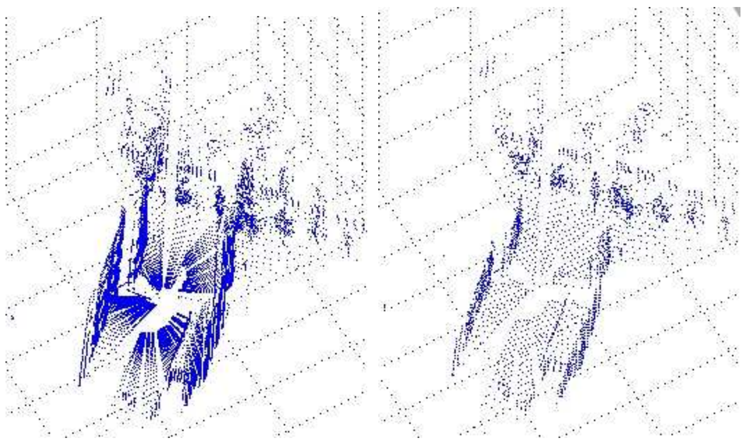
4.6.1 Uniform sampling

To get an uniform sampling, the space is splitted using rectangular boxes (bounding boxes). Initially a major box is created using the maximum an minimum value for each coordinate plus a little δ in all directions, the resulting box is given by $(x_{min} - \delta, y_{min} - \delta, z_{min} - \delta)$ and $(x_{max} + \delta, y_{max} + \delta, z_{max} + \delta)$. Then a user defined value (box_{size}) is given to determine the size for each individual box. Now the boxes maximum number for each axis is computed as $ceil((max{-}min)/box_{size})$. Next for each point the correspondent box index (i_x, i_y, i_z) is obtained by $ceil((data_i - min)/box_{size})$, finally the median of the points stored in each individual box is computed.

4.6.2 Limit axis distance

The other technique to reduce the number of points is done limiting the maximum distance in the x,y and z directions, which also contributes like a filter, because points far from the center do not contribute with important information about the scene and some of them could even be considered as noise. Another reason to apply these limits is to help trade between the points in the vertical and the horizontal surfaces, we have observed that if one of both surface have a grater amount of points it could lead to a final missalignment.

The result of applying the mentioned reduction strategies is shown in Fig. 5(b).



(a) Acquired data (b) Subsampled data

Figure 5: 3D Range laser data acquired with the ETHZ Smartter robot

4.7 Filtering the associated data

Some of the correspondences can eventually come from wrong matches because of noise data or zones with low information. For this reason during the NN search, the corresponding points with a distance bigger than an specified distance threshold (d_{tr}) are rejected, also as ANN could assign a point $p_{i,ref}$ from S_{ref} as the closes point to several query points belonging to the query set S_q. A filter that warranties that any point in S_{ref} and S_q has one and only one correspondence, is applied when using the point-to-point metric.

Also a filter based on Pulli's idea [29] of rejecting the worst $n\%$ of pairs is performed, but we do not use any metric. This is done using histograms and by examinating the histogram we are able to acept a desired $m\%$.

And to deal with the Gaussian noise we present the median filter described in [18].

4.8 Median filter

Scans are noisy and small errors may occur. Two kind of errors mainly occur, Gaussian noise, which occurs for example at the edges, where the laser beam of the scan hits 2 surfaces resulting in a mean and erroneous data value and the error produced for very reflecting areas. To correct

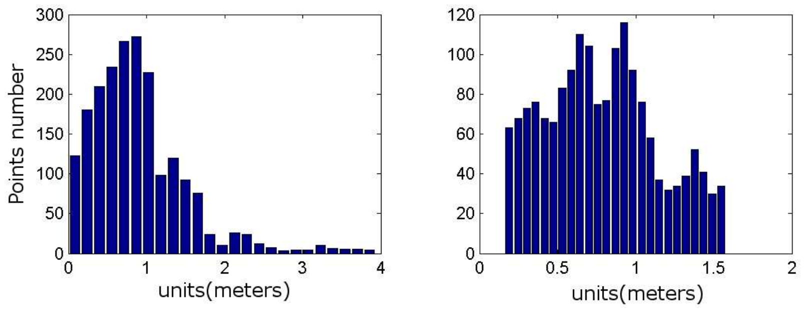
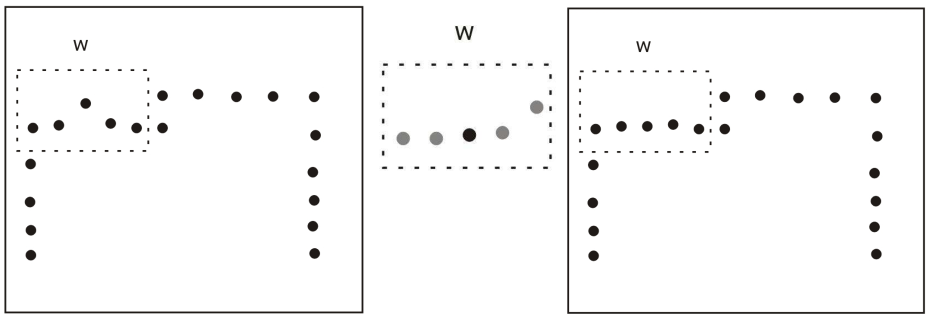


Figure 6: Histogram from the matching (left), filtered histogram (right) with a 92% data acepted

this error [27] used a fast filtering method to smooth the data. The data is stored in the order it was taken, so with knowing of that we applied the median filter [18].The median filter is capable to recognize noise in a scan and to replace with a suitable measurement. To do this, a window becomes for each scan point p_i about the scan point, that contains the last few measurements beside the scan point itself, before and after this point. The scan point then is replaced with a new scan point u_i. The points inside the window are sorted according to their range value, and u_i is the median value in the window Fig. 7(b). The parameter median-number-points determines the window, with which the median filter works. As a large window widths are able to distort the scan, typical values for the windows are rather small, a value of median-number-points = 5 has proved to work well [18].



(a) Selecting the window (b) Point u_i in black (c) Result

Figure 7: Median filter

4.9 A Hybrid Hierarchic Approach to Data Association

When finding the correspondences several techniques have been implemented, point-to-point metric [8, 27, 32], point-to-plane metric [13, 37], and point-to-projection metric with triangular surfaces [9, 10, 11]. Differently from them in our method we propose a hierarchical new correspondence search strategy, using a point-to-plane strategy at the highest level and the point-to-point metric at finer levels. First, the closest plane to a query point is computed with the n NN points from the reference data as in Sec. (4.5), once the plane is defined, we obtain the distance between the plane and it's adjacent n points as an error measurement, called tangent plane error. The error given by:

$$e_{T_p} = \frac{1}{n}\sum_{i=1}^{n}(dist(T_p, p_i)) \tag{15}$$

where $dist(T_p, p_i)$ is defined by:

$$dist(T_p, p_i) = \sqrt{((p_i - o_{T_p}) \cdot \hat{n}_{T_p})^2}$$

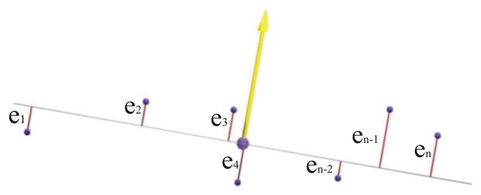


Figure 8: Plane and the errors with its adjacent points

If the tangent plane error is greater than a desired threshold ($e_{T_pT_r}$) then the point-to-point metric is used instead. Once we have the correspondences set Y we use the minimization from Sec. (3.2) to find the displacement between S and S'

5 Experiments

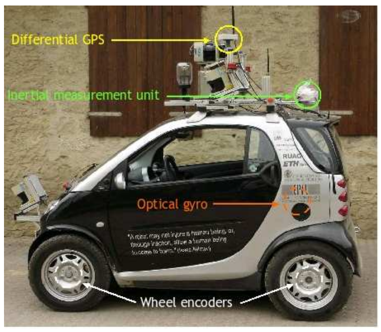
In the next section the developed experiments are described, beginning with the acquisition of the 3D clouds using two different robots. The first robot is able to take scans with a vertical coverage of 360° and in the second robot the vertical coverage is limited by the range finder (190°). Next, some details of the values for the sampling strategies applied are given. Then we talk about ours algorithm implementation and the results for the point-to-point, point-to-plane, and our metric in the Euclidean space and in Biota's metric space in the minimization step. Finally the conclusions are given. The data sets were acquired in the UPC Campus Nord, at the experiments area for the URUS project. This is urban environment with trees, benches, buildings and so on. The ground level in this area varies.

5.1 Data Acquisition

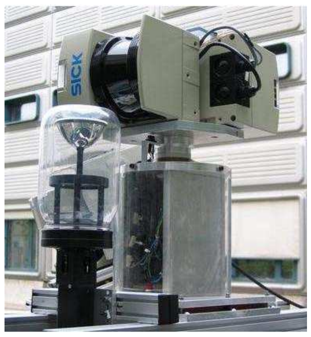
The methods for representing 3D spaces with mobile robots could be divided in two main groups, in the first group they used two 2D range lasers, one in horizontal position and the other in vertical position, Thrun et al. [34] use this system for indoors mounted in a pioneer platform, and Howard et al. [?], perform outdoors scans using a segway platform. In the second group a single laser is mounted moved by a motor (servomotor, stepper motor or DC motor) in pan or tilt configuration, some authors used a tilt mounting for scanning indoors [33, 36] and others [21, 26, 27] for outdoors, in such cases the vertical coverage restricted for the range laser used, usually about 180°,and the horizontal coverage depends on the designed system, while in the mounting pan system it is at the inverse [12]. More recently Lamon et al. [20] presented a rotational system using two laser to get a full 360° vertical scanned scenario with a the Smartter robot.

A 3D data set was taken by the Smartter robot from the "*Eidgenössische Technische Hochschule*" (ETHZ) of Switzerland [20]. The range system in the Samartter consists of two 2D range lasers Sick LMS291-S05 rotating around a vertical axis (pan rotation) Fig. 9(b), delivering point clouds with a 360° vertical coverage. Each second, a full 3D scan of the environment around the vehicle is acquired. Also they used wheel encoders, a differential gps, optical gyro and inertial measurement unit, to get a consistent data set Fig. 9(a), each scan is between 5,000 and 20,000 points.

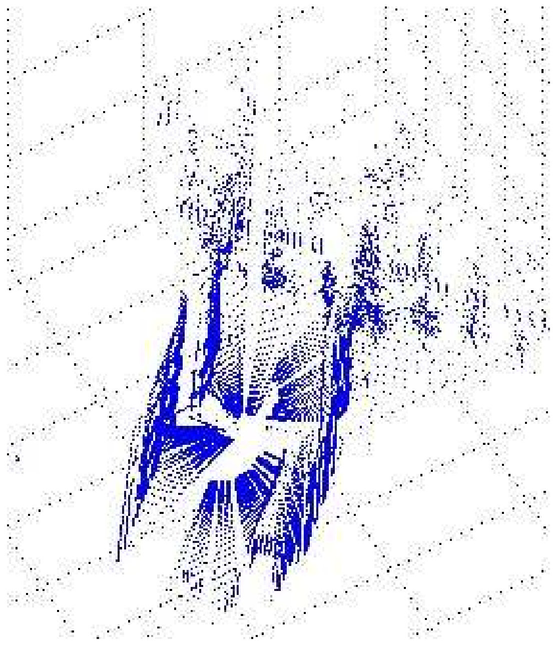
A second data set was obtained using a high definition laser range system designed at the IRI. The IRI 3D range laser [19] is mounted on an Activmedia Pioneer 2AT robotic platform (HELENA) Fig. 10(a). The IRI 3D range laser is based in the idea by [21, 27, 36] using a 2D range laser sensor for pan readings (RS4 Lueze) mounted with a motor giving the tilt movement. The RS4-Leuze has a 528 points maximum resolution per scan line, in a 190.08° ($-5.04° < \theta_{laser} < 185.04°$) amplitude, it means that every 0.36° a point is obtained. Each scan is about 76,000 points. In this case only the odometry coming from the Activmedia platform



(a) Positioning sensors



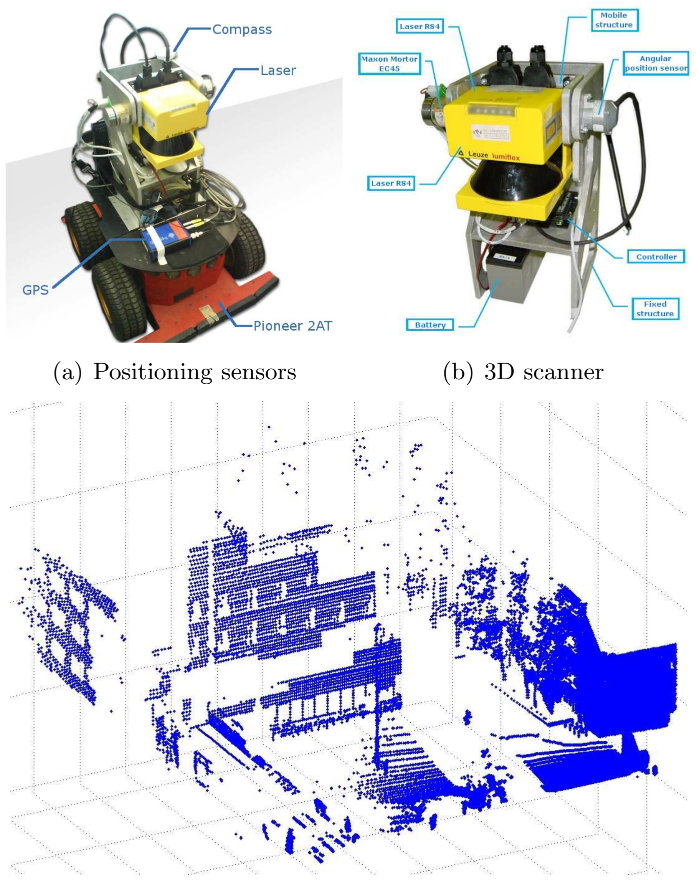
(b) 3D scanner



(c) Single scan

Figure 9: Smartter robot from ETHZ, and a single scan

was used to get a first estimation of the map.



(a) Positioning sensors (b) 3D scanner

(c) Single scan

Figure 10: Helena robot from IRI UPC, and a single scan

5.2 Implementation

In all the previous section the tools for our proposal have been stated. Once we have the data taken we apply the sampling strategies from section 4.6 using for each sampling step and the median filter the values in table (1), all the values were set up experimentally except for the median filter value equal to 5 as suggested in [18]

Data set	Uni. Samp. box size(m)	Max. (x,y) (m)	Max z (m)
Smartter	0.45	± 23	8
Helena	0.35	± 25	9

Table 1: Sampling values for the data

Our ICP algorithm is resumed by the next procedure for each step k:

1. Find the correspondences set Y_k between S and S'

2. Compute the MSE e_k

3. Apply the minimization proposed to compute $q_{min,k}$

4. Recover the transformation matrix H_k from $q_{min,k}$

5. Updating step, transform S'_k with H_k

6. Compute the MSE d_k

7. Apply the above steps until convergence, according with our convergence theorem

The NN search was performed with the ANN library using kd-tree with the sliding-midpoint splitting rule and a unit bucket size. The entire ICP algorithm was implemented and tested over the data sets acquired with both robots (ETHZ's Smartter and IRI's Helena). An empirical comparison was made for the point-to-point, point-to-plane and the hybrid correspondences search using $L = \infty$ and $L = 50$ for the hybrid case. When using the point-to plane method to search for correspondences, local planar planes were fitted using 12 NN for each query point. For ETHZ's data sets we have set experimentally the following parameters; a maximum of 25 iterations, a filtering of 10% of the data, a pairing distance filter threshold of $3.3m$, an 0.1 error threshold between e2 k and d2k for the convergence parameters, a minimum error for $d_k^2 = 0.18m$, and in the hybrid case a maximum tangent plane error $e_{Tp} = 0.18$m. For the IRI data set we decided experimentally also, the following parameters, a maximum of 20 iterations, 92% accepted data, a pairing distance filter threshold of $3.3m$ for the filter parameters, 0.05 error threshold between e_k^2 and d_k^2, a minimum error for $d_k^2 = 0.065m$ for the convergence parameters, and in the hybrid cases a maximum tangent plane error of 0.12m. The parameters for the IRI data sets are smaller because the granularity is finer and the noise levels are smaller than with ETHZ's data.

The ICP algorithm only computes relative transformations between consecutively acquired point clouds. To get a view of the fully corrected map (more strictly, of this new augmented odometry, since no loop closure is being performed at this time) it is necessary to concatenate for each pose the corresponding correction.

Figures 11 to 22 give an empirical comparison between the different methods. Fig. 11(a) shows the MSE error for consecutive paired clouds before the ICP is applied to the ETHZ data set (this is the error induced by odometry only), Fig. 11(b) shows the quadratic error after ICP is applied, and Fig. 11(c) shows the error in orientation. The same is shown for the IRI data set in Fig. 17. The plots of the final revised odometric maps for the various versions of the algorithm are shown, for the ETHZ data set in Figs. 13-16, and for the IRI data set in Figs. 19-22.

6 Conclusions

In this work we proposed a hierarchical new correspondence algorithm that uses a combined point-to-plane and point-to-point correspondence search at different levels of granularity. Our approach is motivated by the work of Biota et al. [10] for the weight rotations must have during data association. In our implementation, the weighting parameter is set slightly higher than as reported previously, producing slightly better registration than what can be obtained with an Euclidean distance only. The setting of the value of parameter L is very sensitive. Its use helps avoid overshoot and consequently divergence during the minimization step of the ICP filtering by enlarging artificially the distance between candidate matches for different orientations. An order of magnitude increase in this parameter to a range between 45 and 70 seemed to work well for our data sets, in contrast to the original work of Biota with values of L in the order of 3 to 5.

It should be noted however, that in our implementation, the computation of fitting local planar patches to the entire point cloud increases the time execution by a constant factor with respect to the point-to-point only strategy.

Data filtering works in most of consecutive cloud pairs in our dataset except for pathological cases. One way to solve this issue would be to heuristically devise a filter that uses not only metric but also topological information [29], or to use rejection strategies for maximum distances [29, 35]. We feel however that this is not a concern since in our future work pathological pairings will be discarded with the use of advanced stochastic loop closing techniques during SLAM. That way we will account for these issues in a more systematic and rigorous way.

7 Acknowledgments

The author thanks Marti Morta for his development of the IRI 3D laser unit.

This work has been partially supported by the Mexican Council of Science and Technology under a PhD Scholarship to Ernesto Teniente; by the Spanish Ministry of Science and Innovation under projects DPI-2008-06022 and MIPRCV Consolider-Ingenio 2010; and by the EU URUS project IST-FP6-STREP-045062.

References

[1] Sameh M . Yamany, Mohamed N. Ahmed, and Aly A. Farag. A new genetic-based technique for matching 3d curves and surfaces. *International Conference on Pattern Recognition*, 32(10):1827–1820, 1999.

[2] Devrim Akca and Armin Gruen. Fast correspondece search for 3d surface matching. *ISPRS Workshop "Laser Scanner "*, pages 186–191, September 2005.

[3] Juan Andrade-Cetto and Michael Villamizar. Object recognition. In J. G. Webster, editor, *Wiley Encyclopedia of Electrical and Electronics Engineering*, pages 1–28. John Wiley & Sons, New York, 2007.

[4] Sunil Arya and David M. Mount. Approximate nearest neighbor queries in fixed dimensions. *ACM-SIAM Symposium on Discrete algorithms*, pages 271–280, 1993.

[5] Sunil Arya, David M. Mount, Nathan S. Netanyahu, Ruth Silverman, and Angela Y. Wu. An optimal algorithm for approximate nearest neighbor searching fixed dimensions. *Journal of the ACM*, 46(6):891–923, November 1998.

[6] Raouf Benjemaa and Francis Schmitt. Fast global registration of 3d sampled surfaces using a multi-z-buffer technique. *Image and Vision Computing*, 17:113–123, 1999.

[7] Jon Louis Bentley. K-d trees for semidynamic point sets. *Annual ACM Symposium Computational Geomtrie*, pages 187–197, 1990.

[8] Paul J. Besl and Neil D. McKay. A method for registration of 3d shapes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(2):239–256, February 1992.

[9] Lydia Biota. Algoritmos de scan matchin basados en métricas para estimar el movimiento de robots que se desplazan en espacios tridimensionales. Technical report, Centro Politï¿½cnico Superior de Zaragoza, June 2005.

[10] Lydia Biota, Luis Montesano, Javier Minguez, and Florent Lamiraux. Toward a metric-based scan matching algorithm for displacement estimation in 3d workspaces. *International Conference on Robotics and Automation*, pages 4330–4332, May 2006.

[11] Gérard Blais and Martin D. Levine. Registering multiview range data to create 3d computer objects. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17(8):820–824, August 1995.

[12] Christian Brenneke, Oliver Wuif, and Bernard Wagner. Using 3d laser range data for slam in outdoor environments. *International Conference on Intelligent Robots and Systems*, pages 188–193, October 2003.

[13] Yang Chen and Gérad Medioni. Object modeling by registration os multiples ranges images. *International Conference on Robotics and Automation*, 3:2724–2729, April 1991.

[14] Dmitry Chetverikov. Fast neighborhood search in planar point sets. *Pattern Recognition Letters*, 12(7):409–412, July 1991.

[15] Dmitry Chetverikov, Dmitry Stepanov, and Pavel Krsek. Robust euclidean alignment of 3d point sets: the trimmed iterative closest point algorithm. *Image and Vision Computing*, 23(3):299–309, 2005.

[16] Jerome H. Friedman, Jon Louis Bentley, and Raphael Ari Finkel. An algorithm for finding best matches in logarithmic expected time. *ACM Transaction on Mathematical Software*, 3((3)):209–226, 1977.

[17] Michael Greenspan and Guy Godin. A nearest neighbor method for efficient icp. *International Conference on 3D Digital Imaging and Modeling*, pages 161–168, 2001.

[18] Jens-Steffen Gutmann. *Robuste Navigation autonomer mobile Systeme.* PhD thesis, University of Freiburg (Germany), 2000.

[19] Marti Morta i Garriga. Disseny i construcció d'un laser 3d per al mapejat d'entorns exteriors. Technical report, Escola Técnica Superior d'Enginyería Industrial de Barcelona, June 2008.

[20] Pierr Lamon, Cyrill Stachniss, Rudolph Triebel, Patrick Pfaff, Christian Plagemann, Giorgio Grisetti, Sascha Kolski, Wolfram Burgard, and Roland Siegwart. Mapping with an autonomous car. *In IEEE/RSJ IROS Workshop: Safe Navigation in Open and Dynamic Environments*, 2006.

[21] Kai Lingemanna, Andreas Nüchter, Joachim Hertzberg, and Hartmut Surmann. High-speed laser localization for mobile robots. *Robotics and Autonomous Systems*, 51(4):275–296, 2005.

[22] Takeshi Masuda, Katsuhiko Sakaue, and Naokaza Yokoya. Registration and integration of multiple range images for 3d model construction. *International Conference on Pattern Recognition*, 20(1):879–883, 1996.

[23] Javier Minguez, Florent Lamiraux, and Luis Montesano. Metric-based scan matching algorithms for mobile robot displacement estimation. *International Conference on Robotics and Automation*, pages 3568–3574, April 2005.

[24] Javier Minguez, Luis Montesano, and Florent Lamiraux. Metric-based iterative closest point scan matching for sensor displacement estimation. *IEEE Transaction on Robotics*, 22(5):1047–1054, October 2006.

[25] Andreas Nüchter, Kai Lingemann, Joachim Hertzberg, and Hartmut Surmann. 6d slam with approximate data association. *International Conference on Advanced Robotics*, pages 242–249, 2005.

[26] Andreas Nüchter, Kai Lingemann, Joachim Hertzberg, and Hartmut Surmann. *Heuristic-Based Laser Scan Matching for Outdoor 6D SLAM*, volume 3698. SpringerLink, September 2005. Heuristic-based laser scan matching for outdoor 6D SLAM.

[27] Andreas Nüchter, Kai Lingemann, Joachim Hertzberg, Hartmut Surmann, and Sebastian Thrun. 6d slam with an application in autonomous mine mapping. *International Conference on Robotics and Automation*, 2:1998–2003, May 2004.

[28] Soon-Yong Park and Murali Subbarao. A fast point-to-tangent plane technique for multi-view registration. *International Conference on 3D Digital Imaging and Modeling*, pages 276–283, October 2003.

[29] Kari Pulli. Multiview registration for large data sets. *International Conference on 3D Digital Imaging and Modeling*, 8:160–168, 1999.

[30] Peter J. Rousseeuw. Least median of squares regression. *Journal of the American Statistical Association*, 79(388):871–880, 1984.

[31] Szymon Rusinkiewicz and Mark Levoy. Efficient variants of the icp algorithm. *International Conference on 3D Digital Imaging and Modeling*, pages 145–152, 2001.

[32] David A. Simon, Martial Hebert, and Takeo Kanade. Real-time 3d pose estimation using a high-speed range sensor. *International Conference on Robotics and Automation*, 3:2235–2241, 1994.

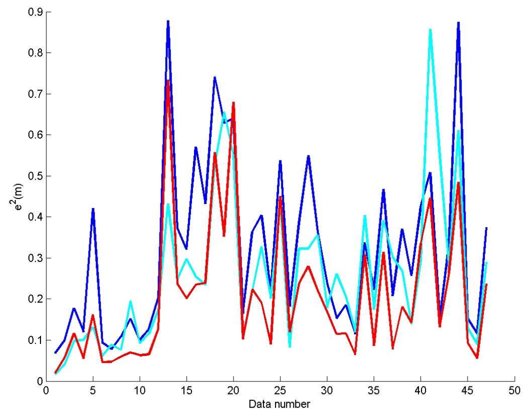
[33] Hartmut Surmann, Andreas Nüchter, and Joachim Hertzberg. An autonomous mobile robot with a 3d laser range finder for 3d exploration and digitalization of indoor environments. *Journal Robotics and Autonomous Systems*, 45(3-4):181–198, 2003.

[34] Sebastian Thrun, Wolfram Burgard, and Dieter Fox. A real-time algorithm for mobile robot mapping with applications to multi-robot and 3d mapping. *International Conference on Robotics and Automation*, 1:321–328, 2000.

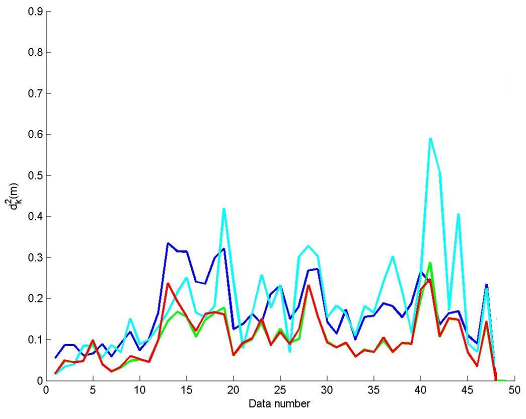
[35] Greg Turk and Mark Levoy. Zippered polygon meshes from range images. *Conference on Computer Graphics and Interactive Techniques*, pages 311–318, July 1994.

[36] Jan Weingarten and Roland Siegwart. Ekf-based 3d slam for structured environment reconstruction. *International Conference on Intelligent Robots and Systems*, pages 3834–3839, 2005.

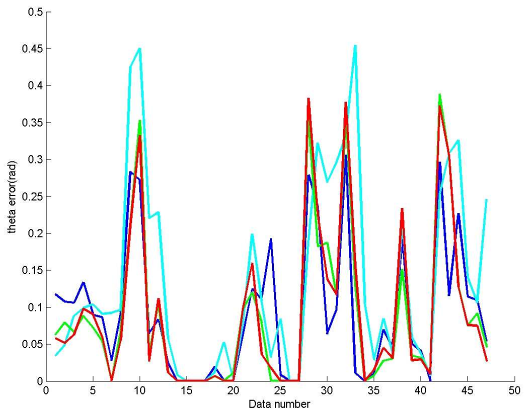
[37] Zhengyou Zhang. Iterative point matching for registration of free-form curves and surfaces. *International Journal of Computer Vision*, 13:119–152, 1994.



(a) Error e_k^2 at first iteration



(b) Final error d_k^2



(c) Final theta error

Figure 11: ETHZ data sets, error comparison between cloud pairs for the implemented strategies, point-to-point(blue), point-to-plane(cyan), hybrid(gree), hybrid L = 50 (red)

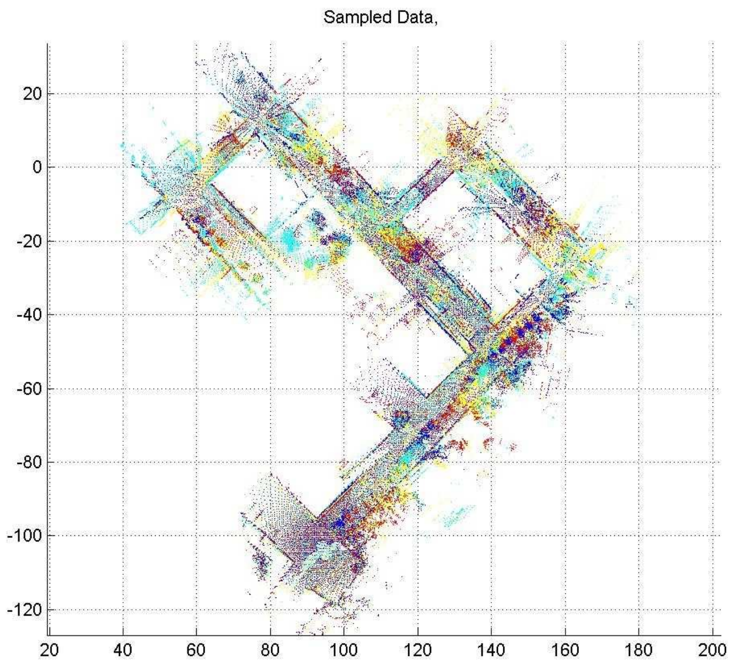


Figure 12: ETHZ, Sampled data set

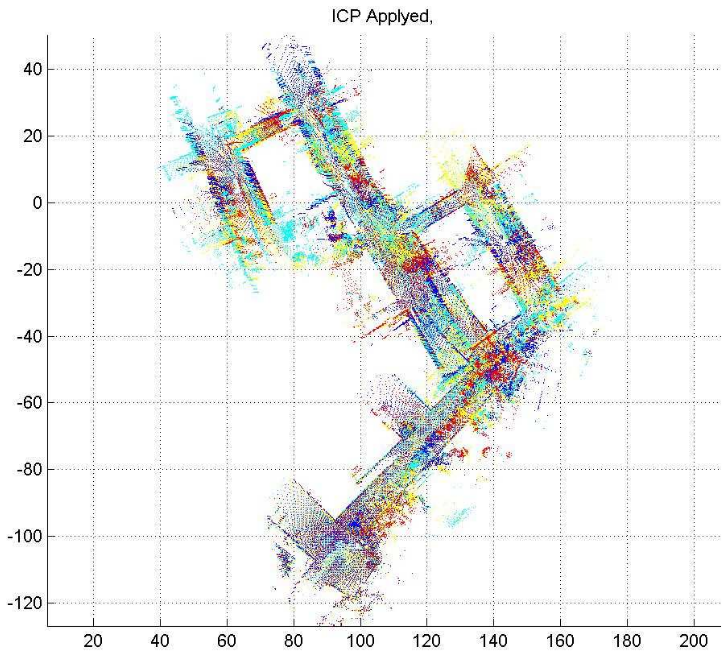


Figure 13: ETHZ data, point-to-point metric

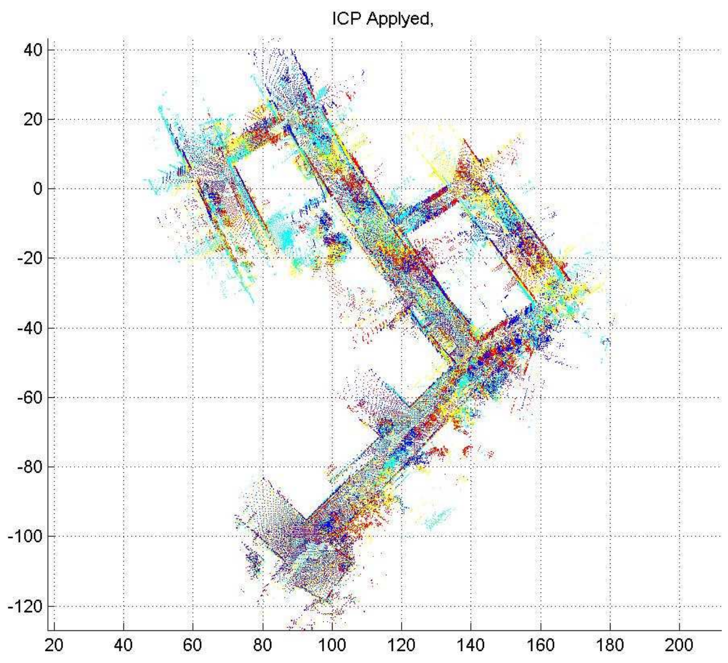


Figure 14: ETHZ data, point-to-plane metric

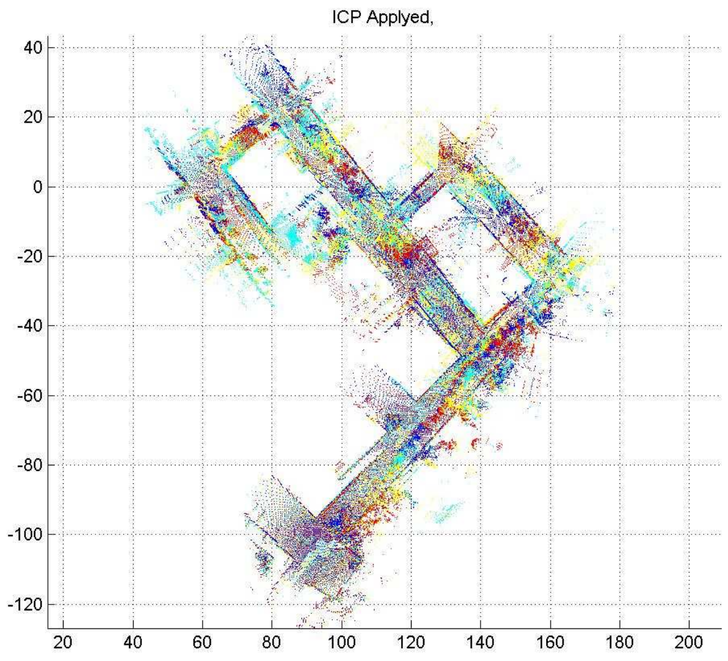


Figure 15: ETHZ data, hybrid metric, euclidean distance

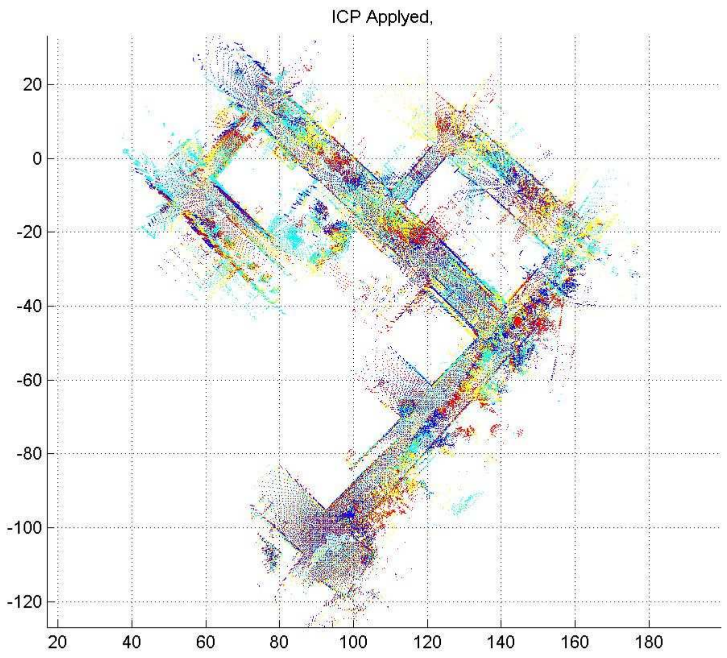
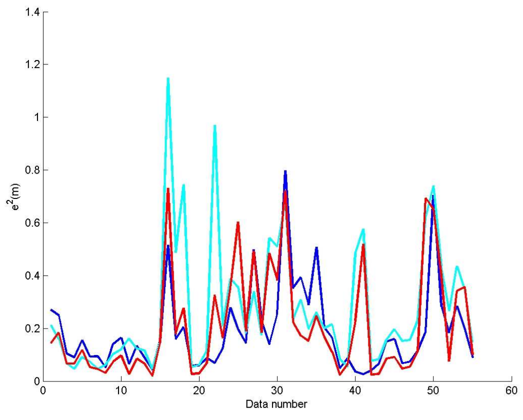
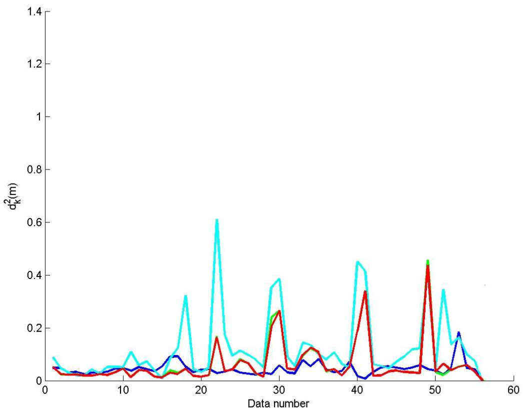


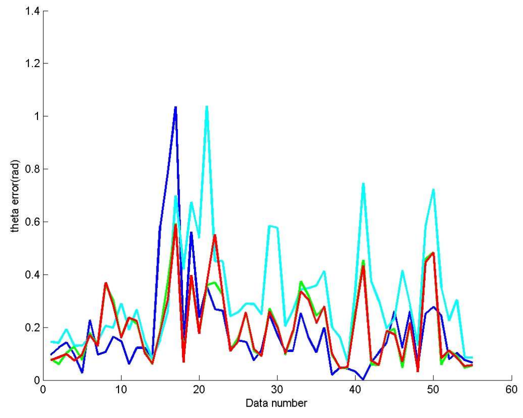
Figure 16: ETHZ data, hybrid metric, Biota's metric, L= 50



(a) Error e_k^2 at first iteration



(b) Final error d_k^2



(c) Final theta error

Figure 17: IRI data sets, error comparison between cloud pairs for the implemented strategies, point-to-point(blue), point-to-plane(cyan), hybrid(gree), hybrid L = 50 (red)

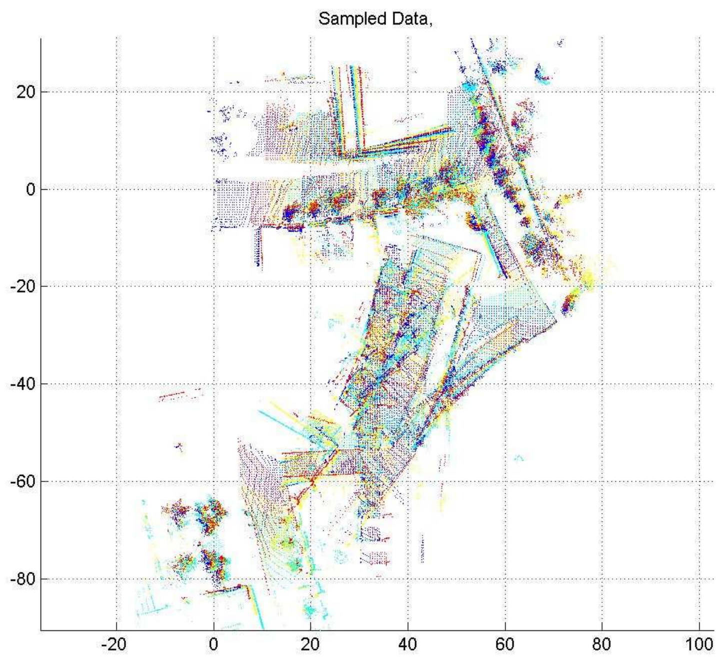


Figure 18: IRI, Sampled data set

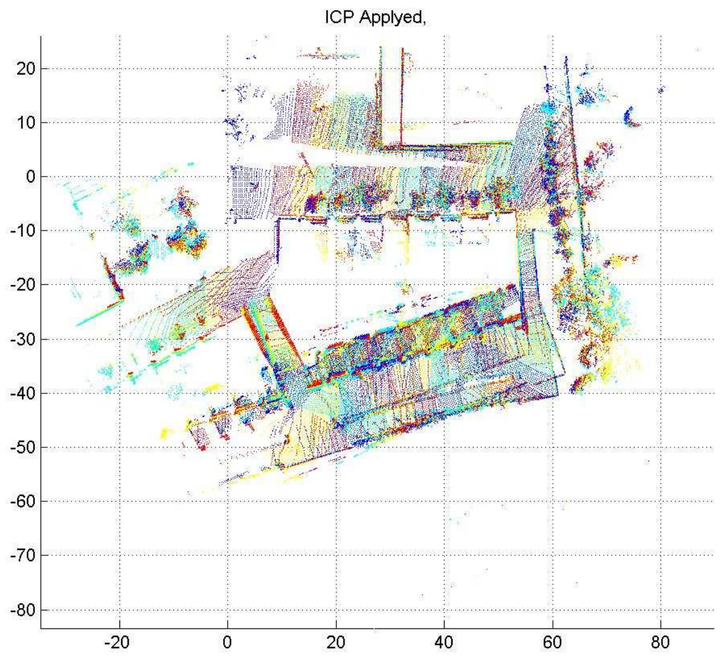


Figure 19: IRI data, point-to-point metric

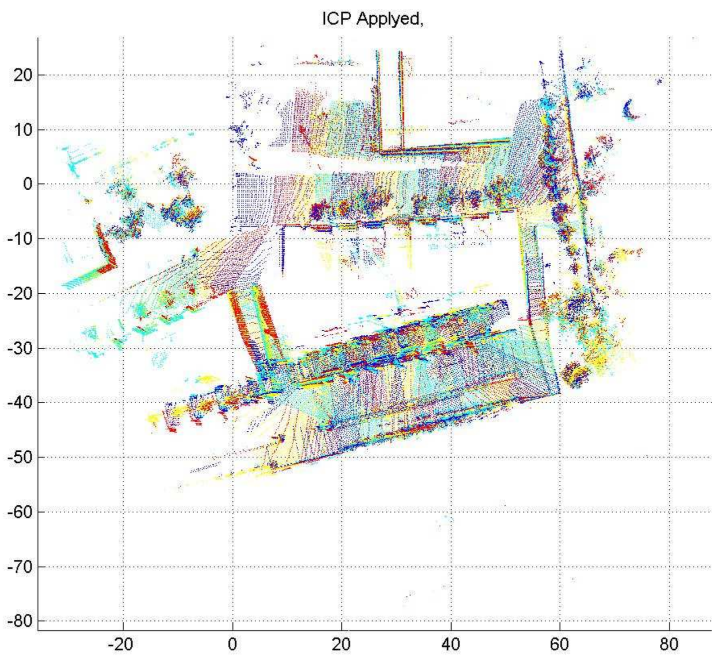


Figure 20: IRI, point-to-plane metric

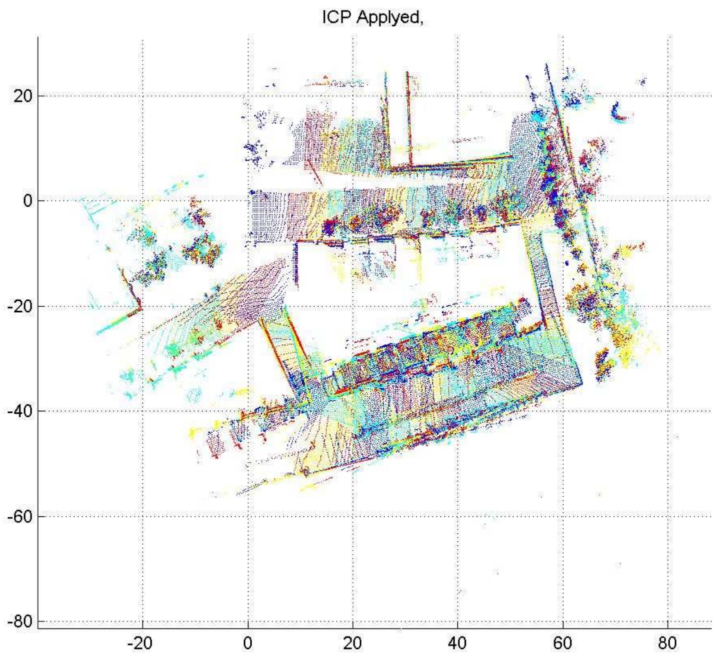


Figure 21: IRI, hybrid metric, euclidean distance

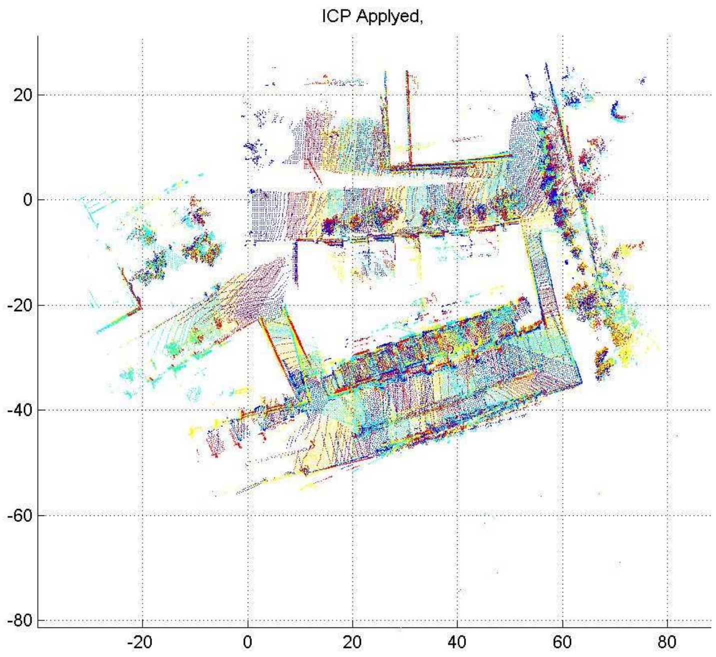


Figure 22: ETHZ, hybrid metric, Biota's metric, L= 50