J Intell Robot Syst (2013) 72:441–462
DOI 10.1007/s10846-012-9811-5

Critical Rays Scan Match SLAM

Emmanouil Tsardoulias · Loukas Petrou

Received: 25 July 2012 / Accepted: 26 December 2012 / Published online: 9 February 2013

Abstract Scan matching is one of the oldest and simplest methods for occupancy grid based SLAM. The general idea is to find the pose of a robot and update its map simply by calculating the 2-D transformation between a laser scan and its predecessor. Due to its simplicity many solutions were proposed and used in various systems, the vast majority of which are iterative. The fact is, that although scan matching is simple in its implementation, it suffers from accumulative noise. Of course, there is certainly a trade-off between the quality of results and the execution time required. Many algorithms have been introduced, in order to achieve good quality maps in a small iteration time, so that on-line execution would be achievable. The proposed SLAM scheme performs scan matching by implementing a ray-selection method. The main idea is to reduce complexity and time needed for matching by preprocessing the scan and selecting rays that are critical for the matching process. In this paper, several different methods of ray-selection are compared. In addition matching is performed between the current scan and the global robot map, in order to minimize the accumulated errors. RRHC (Random Restart Hill Climbing) is employed for matching the scan to the map, which is a local search optimization procedure that can be easily parameterized and is much faster than a traditional genetic algorithm (GA), largely because of the low complexity of the problem. The general idea is to construct a parameterizable SLAM that can be used in an on-line system that requires low computational cost. The proposed algorithm assumes a structured civil environment, is oriented for use in the RoboCup - RoboRescue competition, and its main purpose is to construct high quality maps.

Keywords SLAM · Scan matching · Random restart hill climbing · Critical rays · Occupancy grid map

E. Tsardoulias · L. Petrou (✉)
Department of Electrical and Computer Engineering, Division of Electronics and Computer Engineering, Faculty of Engineering, Aristotle University of Thessaloniki, 54124 Thessaloniki, Greece
e-mail: loukas@eng.auth.gr

E. Tsardoulias
e-mail: etsardou@gmail.com

1 Introduction

The Simulataneous Localization and Mapping (SLAM) problem is one of the most important aspects of robotics, as it provides a robot with a representation of the environment it is moving in. The most usual way for a robot to perform

SLAM, is via sensors that measure distance, such as sonars and LRFs (Laser Range Finders). In our approach, we assume that the robot is equipped with an LRF, which can measure 270 distinct distances by sending laser pulses (rays), measuring the time needed to reflect back to the sensor. These kind of sensors provide a very important tool for the SLAM problem, as they are highly accurate and extract information from a large part of the environment in the vicinity of the robot.

The proposed SLAM method is called CRSM SLAM, which stands for Critical Rays Scan Match SLAM. Its goal is to suggest a method of using only a part of the LRF's information, but in the same time produce high quality maps. We propose three ways of ray selection, a uniform one, one based on scan density and one that splits the scan in segments and employs their local density. An early implementation of CRSM SLAM was SMG SLAM, which was firstly presented in [1]. CRSM SLAM is an evolution of SMG SLAM, as it uses a hill climbing algorithm instead of a genetic algorithm, in order to increase the execution speed. Also a more sophisticated method of scan selection is employed in comparison with the uniform sampling (which is also described here) performed in SMG SLAM.

CRSM SLAM's main characteristic is that it uses an occupancy grid map [2] in order to depict the environment, i.e. each pixel represents a small fraction of the environment, and its value is proportional to the probability of that space being unoccupied. So, if the map is illustrated as a gray-scale image, the unoccupied space will be white with $p_{\text{free}} = 1$, the obstacles will be black with $p_{\text{free}} = 0$, and the unexplored space will have a gray color with $p_{\text{free}} = 0.5$. An example of an occupancy grid map is depicted in Fig. 1.

Its second characteristic is that the matching is not performed between two successive laser scans, as in the standard implementation, but between the current scan and the map being obtained by that moment. Scan to scan matching, even for trivial errors leads to large cumulative errors as time passes, even if the robot constantly navigates in the same area. In scan to map matching this problem is surpassed with the current scan being matched to the global map having many of the previous scans as potential reference. Therefore, the overall quality of the map is increased and the accumulated errors are decreased.

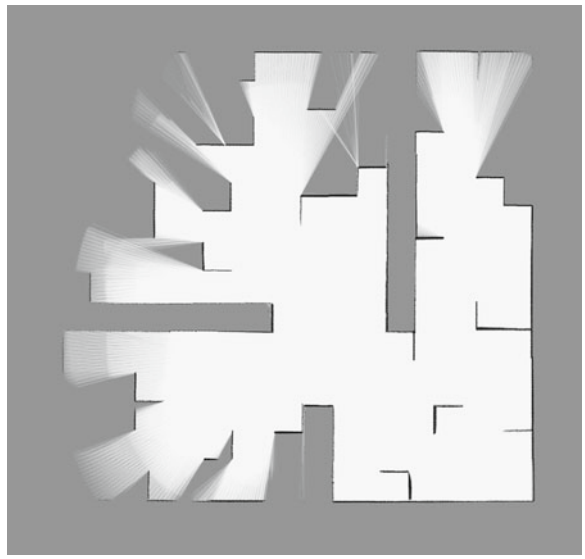

Fig. 1 Occupancy grid map

Another important feature, and the main novelty, of CRSM SLAM is the ray selection. The main idea is to reduce the complexity and the time needed for the matching, by pre-processing the scan and selecting rays that are critical for it. Critical means that the remaining ray information is redundant to the matching process, since critical rays act as features of the scan, even if they are found from heuristics and not from any feature extraction method. This is an advantage of this method as it can work on unstructured and non-static environments. In this paper four different ray selection methods are presented and compared on different environments.

The method used for the matching, is a search algorithm called Random Restart Hill Climbing. This method was chosen, since it is highly flexible and many of its parameters can be altered leading to various SLAM behaviours. This is crucial, as the main problem of SLAM algorithms is that they are not global, that is they cannot be applied to any environmental conditions and produce the same results. Since the algorithm is highly parametrizable, the transition between different environments can become easier.

It must be stated, that the algorithm makes absolutely no use of odometry or robot kinematics,

in order to compute the robot's pose. This decision was taken, as in rescue environments these types of inputs are highly unreliable due to the robot's drifting, because of the existence of debris. In addition, no analysis of the errors brought by a kinematic model is made.

The proposed algorithm's execution flow is illustrated in Fig. 2. First, the algorithm performs initialization of the structures needed to store the map and all acquired scans. Consequently a repetitive procedure follows, by which the Laser Range Finder (LRF) scan is acquired, the redundant ray information is eliminated by the ray selection module, the scan matching is performed via the Hill Climbing algorithm and the final module updates the map's possibilities for each cell to be free or occupied.

In chapter 2, some previous work regarding scan matching is presented. In chapter 3, the performance measures used in this paper, regarding map quality, are presented. In chapter 4, four different methods of ray selection are presented, and their philosophy is explained. Furthermore, their performance is illustrated in custom made environments, created with the USARSim simulator. In chapter 5, the hill climbing method used for the matching process is explained in depth. In chapter 6, map update, the last phase of the SLAM iteration is presented. In chapter 7, the experimental results are presented and analysed. Finally in chapter 8, the proposed algorithm is tested in scan log files hosted in the robotics data set repository (Radish), conclusions are given and future work is presented.

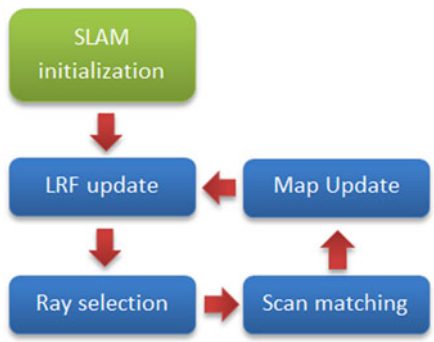

Fig. 2 Diagram of the proposed SLAM

2 Related Work

Due to its simplicity, scan matching has been used in a number of SLAM algorithms, and numerous suggestions have been made on the matching process or the data association between two successive laser scans. A collection of methods is described in [3, 4], such as Classic ICP, Metric Based ICP, Point-to-Line ICP and others. In [5], the Classic Iterative Closest Point (ICP) algorithm is presented for the first time by Besl and Mckay, which is vastly used in many SLAM solutions, because of its computational advantages, as it performs a recursive method of minimizing the mean square errors between two successive LRF scans. ICP is a fundamental algorithm in many scientific areas, such as image processing or 3D modelling. For example, a common use of it, is the matching of 3D cloud points, as Zhang did in [6]. Unfortunately one of its main drawbacks is that it is prone to local minima and as far as SLAM is concerned, to accumulative errors in the produced map. So different approaches and modifications were proposed from the scientific community.

For example, in [7], Lu and Milios proposed the Iterative Dual Correspondence (IDC) method, that uses two algorithms combined. The first is based on matching data points with the use of the curvature of the scans, whereas the second uses point to point correspondence, in order to improve the results produced by the first. IDC resulted in a faster, and more accurate matching than the classical ICP. Another alteration over the classical ICP, is presented in [8] and is called Polar Scan Matching (PSM). The main idea is to use point to point matching approaches and polar transformation of the rays, instead of the traditional Cartesian system used by the ICP. This modification enabled more accurate point to point matching and therefore increased the quality of the resulting map.

In [9], Cai Ze-Su, Hong Bing-Rong and Li Hong, suggested an improved version of the PSM, that uses a genetic algorithm to perform the matching, in order to avoid the step of data association, which is the most tricky in ICP. This suggestion is also made by this paper, where we use Random Restart Hill Climbing instead of

GAs. Hill climbing is used in many scan matching approaches. One of them is the Vasco scan matcher, which performs brute-force search in the solutions space using hill climbing, and is part of the Carnegie Mellon Robot Navigation Toolkit (CARMEN) [10]. This approach was also used in the GMapping proposal by Grisetti, Stachniss and Burgard in [11]. Avoiding data association, in [12], a real-time correlative scan matching is proposed. A 'scan to scan' matching is performed, using the cross-correlation of two successive scans and by searching over the entire space of solutions, in order to avoid local maxima. This method assumes large use of computational power, in order to produce more accurate results. In [13], Martinez, Gonzalez et al, proposed the combination of the ICP method and a genetic algorithm, by applying a rough optimization in the search space and then by using the ICP to refine the result. In that manner they tried to decrease the possibility of ICP falling in local minima, and simultaneously reduce the GA's execution time by not requiring a precise result.

There are also other methods that take advantage of the correlation concept of ICP. In [14], Nieto, Bailey and Nebot, proposed the Recursive scan matching SLAM, which is a hybrid of EKF SLAM and scan correlation methods, where the landmarks are formed by patterns of raw data. The contribution of scan correlation methods is that they improve the feature association module of the EKF SLAM algorithm. On a different context in [15] Lakaemper and Alduru extended the matching concept by proposing a map alignment method (Force Field Simulation—FFS), with the use of a gradient descent algorithm, that is motivated by simulation of dynamics of rigid bodies in gravitational fields. There constraints derived from human perception are introduced instead of the laws of physics. The proposed method performs the SLAM in two steps. Initially, a low-precision pre-alignment is performed, but each scan remains independent, i.e. not merged in the map. The second step, is to apply FFS in the pre-aligned scans, in order to produce a more accurate matching. Another method that surpasses the point association problem, is proposed in [16] and is called the Normal Distributions Transform. There, a normal distribution transformation in space is applied. This is a continuous and differentiable probability density, which models the probability of measuring a point. Then, with Newton's algorithm help the matching with the next scan is done without establishment of explicit correspondences.

An approach similar to ours, is presented in [17]. There, Nunez, Vazquez-Martin et al, proposed the procedure of matching between two consecutively acquired scans to be achieved by using their associated curvature based representations, which are extracted from a scan segmentation module. So, instead of performing point to point matching, the characteristic points of the current curvature functions are matched to the ones extracted from the prior scan, in order to obtain the best local alignment between them. This method's concept is similar to ours, as we also try to find "important" points of each scan, albeit not using curvature, but scan density. As far as scan segmentation is concerned, Hee Jin Sohn and Byung Kook Kim in [18], proposed VecSLAM which is based on the extraction of "vectors", another name for scan segments, with the help of a sequential segmentation algorithm. Subsequently the vectors are matched to recent map vectors and stored. The aligned scan vectors are merged into map vectors, that are also used in loop detection and closing. Another concept similar to ours, is presented in [19]. There, the position estimation of the robot Blanche, consists of a matching algorithm, that registers the range data with the map (of line segments), and makes a calibrating estimation based on odometry readings. The similarity to our method, is limited to the scan to map matching, although we don't use odometry or line features. Finally, the concept of "important" points is used in the proposed method (ICE) in [20], which describes the utilization of a different set of scan features than a simple segmentation, such as Intersections, Corners and End of wall. The difference from our method, is that we do not classify the important points but these are computed analytically with just two passes through the LRF scan.

To conclude, in [21], methods of establishing a benchmark evaluation framework for comparing and analysing different SLAM techniques are described.

The contribution of our paper is twofold. First of all, novel techniques are used in order to select the "crucial" points of each LRF scan. This is done to cope with less data, and therefore reduce the computational complexity of the matching algorithm. We also prove, that besides the complexity reduction, the proposed method produces best maps in terms of quality, in comparison with the use of the whole information, as there exists a considerable amount of highly correlated data. The second contribution is the low level modifications inserted in the map update procedure. Specifically, we propose a) a dynamically changing map update intensity, based on the density characteristics of the scan, and b) the increased update intensity of the obstacle cells, in order for the scan matching algorithm to have a better reference. These contributions are demostrated via a set of experiments in different environments.

As far as the technical details are concerned, the matching approach is not scan-to-scan matching, but scan-to-map matching. This results in the elimination of the accumulative errors of classical ICP, as the reference (map) remains almost unchanged during each matching. In comparison between our method and scan to scan matching, the error accumulation is reduced since one error in alignment plays insignificant role to the overall procedure, as the mean of errors is close to zero due to the common reference. Finally, we use a hill climbing algorithm, in order to perform the scan matching, instead of ICP or GAs. The hill climbing algorithm manages to find the solution more accurately than the ICP but faster than GA, as the problem is simple enough.

3 Performance Measures

Before analysing the proposed SLAM method, it is essential to present the performance measures, with which the quality of each map will be decided. These measures are calculated based on a reference map, which is no other than the ground truth. The proposed algorithms were tested in three environments of distinct characteristics, the ground truth of each is presented in Fig. 3. The first environment is characterized as sparse with many features (Environment 1), the second as one with narrow corridors (Environment 2), and the third as a single twisted corridor (Environment 3).

The first metric used, is the mean square error of the resulting map's obstacles, with reference to the obstacles of the ground truth map. If M is the produced map by the proposed method, and M_{GT}

Fig. 3 Ground truth for each testing environment. *Left*: Environment 1, *center*: Environment 2, *right*: Environment 3

the ground truth map, the Obstacle Mean Square Error (OMSE) is defined as

$$OMSE = \frac{1}{|o_i|} \cdot \sum_{\forall o_i \in M} Dist(o_i, o')^2 ,$$

$$o' \in M_{GT} \wedge \text{argmin}_{o'} Dist(o_i, o')$$

where o_i is one cell that represents an obstacle in the resulting map, o' the obstacle cell in the ground truth map that is closest to o_i, and $Dist(o_i.o')$ the Euclidean distance between the two cells. This performance measure is an indicator of how similar is the resulting map to the ground truth.

The second metric is called Corner Mean Square Error (CMSE). A number of distinct corners were selected in each map as features, (Fig. 4) and are used in order to calculate the precision of the SLAM result, as far as topology is concerned. CMSE is calculated as follows.

$$CMSE = \frac{1}{|f_i|} \cdot \sum_{\forall f_i \in M} Dist(f_i, f')^2 ,$$

$$f' \in M_{GT} \wedge f' \rightarrow f_i$$

where f_i is a corner-feature in the resulting map, and f' is the same topological corner as f_i in the ground truth map.

It must be noted, that the first metric measures the obstacle similarity between the result and the ground truth map, whereas the second, is an indicator of the structural similarity between the two. The metrics OMSE and CMSE are measured in square pixels (cells), px^2, as they are sums of square distances.

4 Ray Selection

Ray selection methods were developed, in order to reduce the number of rays, used in the calculation of hill climbing. The initial concept was to keep the critical rays only by means of specific heuristics. A feature extraction method was not chosen, because of its complexity and time requirements in comparison with the ray selection. It must be noted, that the proposed SLAM algorithm consists of three main modules: the ray-selection, the hill-climbing, which performs the scan matching, and the map update module (Fig. 2). In the current chapter, different ways of ray-selection are assumed, whereas the hill-climbing and map update parts remain unchanged, in order for the comparison to be valid.

For safety reasons, rays with a distance greater than 95 % of the laser's nominal maximum distance are discarded. Without loss of generality, it

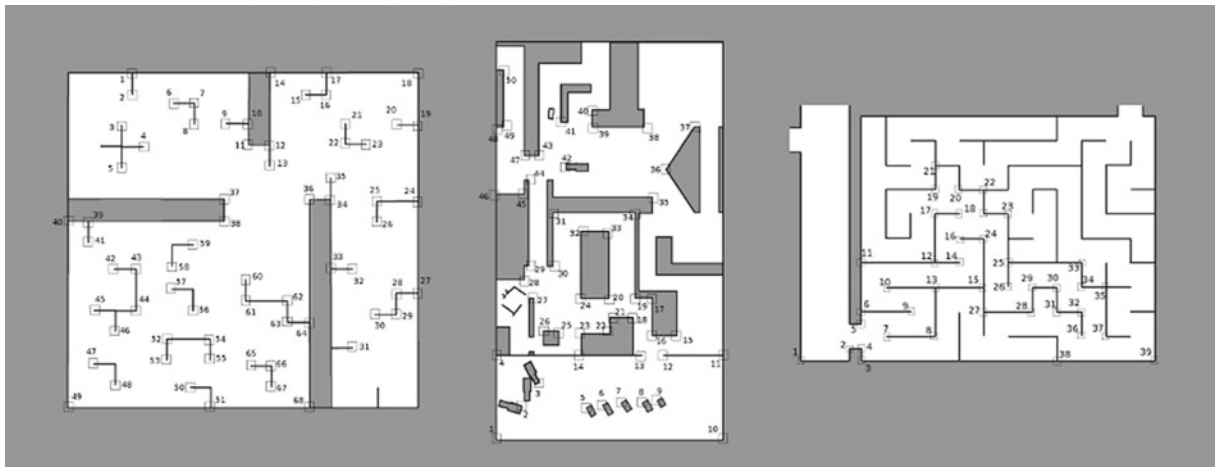

Fig. 4 Selected corners/features in each environment for structural comparison. *Left*: Environment 1, *center*: Environment 2, *right*: Environment 3

is assumed that all the rays have measurements below that threshold.

The experiments using the selection methods, were performed with the help of the open source robot simulator USARSim [22], in cooperation with Player/Stage project [23]. Three different flat maps were used for validation: one that has large rooms with enough obstacles in them (area of 420 m^2), one with narrow and long corridors and junctions (area of 264 m^2), and one with a twisted corridor (area of 170 m^2). The simulated LRF had 270° field of view, 270 rays, and could take measurements up to 9.5 meters. The robot's maximum linear and rotational speed was 0.15 m/sec and 0.12 rad/sec respectively.

The most common practice in scan matching, is to select all rays. In this method, all the scan rays are selected in order to perform the hill climbing. The results of the "all rays selected" experiments are displayed in Figs. 5, 6 and 7.

The figures indicate that good results are obtained, except for the environment with narrow corridors, where SLAM failed. The obvious advantage of this method, is that all information available is used to make an estimation of the robot's translation and rotation. On the other hand, the main drawback is that the hill climbing's iteration time is so long, due to the large number of the rays. Consequently, the method cannot be used for on-line SLAM. Next, the ray selection methods are presented.

Fig. 5 Sparse environment with many features—all rays selected method

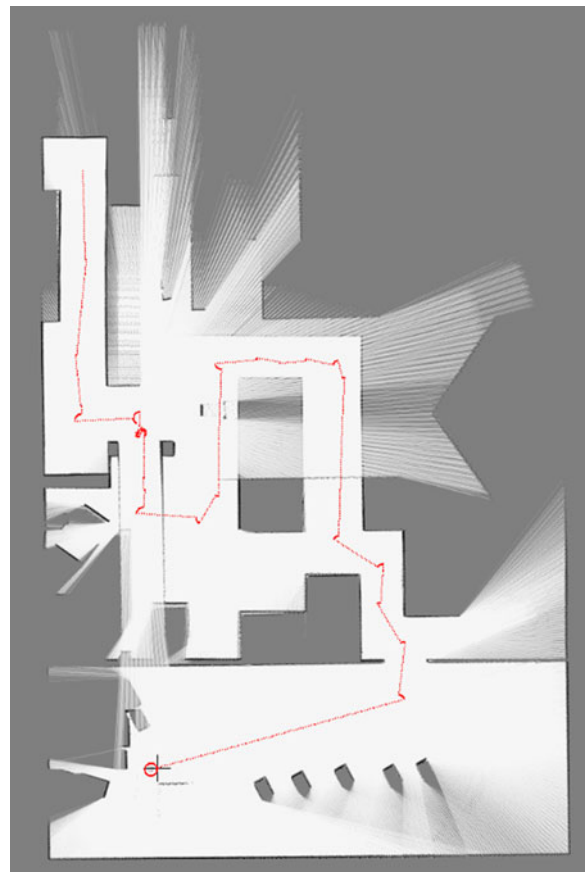

Fig. 6 Environment with narrow corridors—all rays selected method

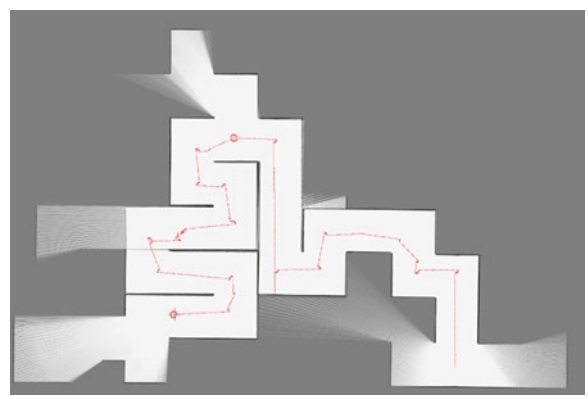

Fig. 7 Single twisted corridor—all rays selected method

4.1 Uniform Sub-sampling

In order to reduce the computation time, a uniform selection of the scans was made, which in fact is a down-sampling of the input data. Here, the number of the selected rays is constant and equal to $N = LaserRays/SkipStep$. Their indices are computed by formula Eq. 1:

$$\mathbf{PickedIndices}_i = SkipStep \cdot i\,, 0 \leq i \leq N \qquad (1)$$

where $LaserRays$ is the number of the LRF rays, and $SkipStep$ is the rate of down-sampling.

This method is $SkipStep$ times faster than the all-rays selection method, as far as the hill climbing method is concerned. The results of the experiments for $SkipStep = 5$ are displayed in Figs. 8, 9 and 10.

The results are of acceptable quality except for the environment with narrow corridors, although the overall quality was decreased in comparison with the all rays selected method. The main defect of this method is that it tends to choose many rays whose measurements are small (meaning that many rays are selected which lay in obstacles close to the robot). This is logical, due to the fact that in nearby obstacles the ray density is large and the displacement between adjacent rays is extremely

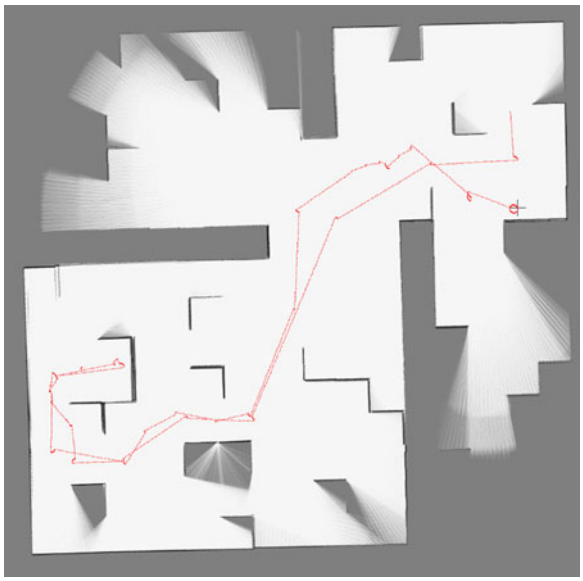

Fig. 8 Sparse environment with many features—uniform sub-sampling method

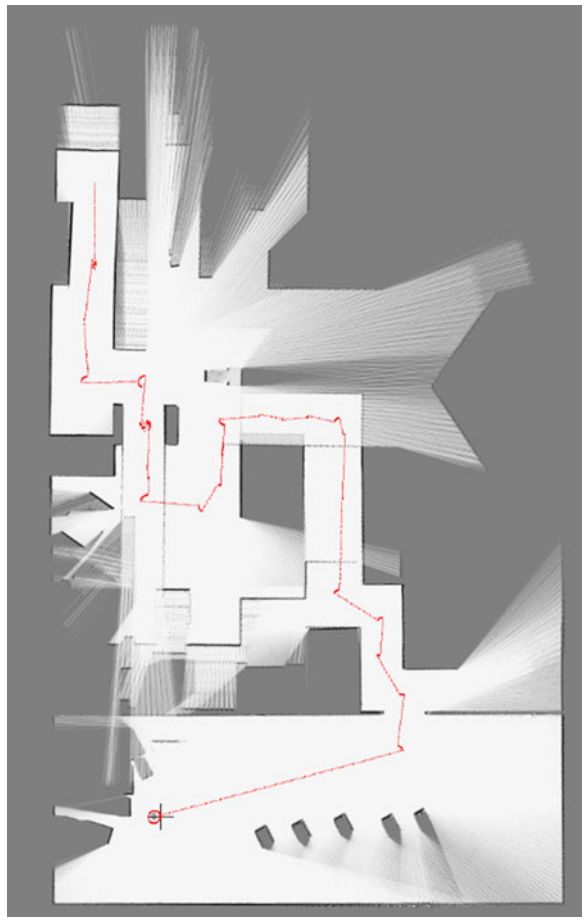

Fig. 9 Environment with narrow corridors—uniform sub-sampling method

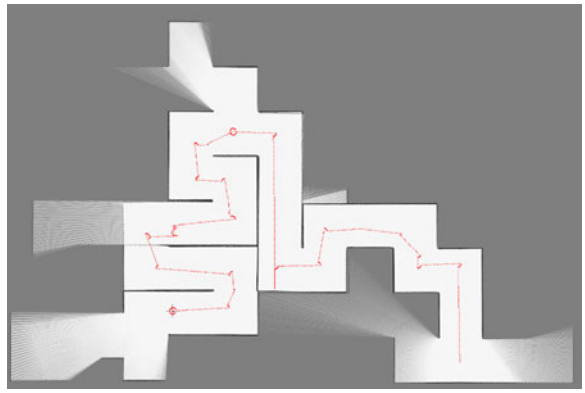

Fig. 10 Single twisted corridor—uniform sub-sampling method

small. An error is introduced in scan matching, in cases where the environment is narrow and has a small number of features (distinct obstacles). In that case, the hill climbing tries to match many similar points, ignoring the more important ones which are the sparser measurements.

4.2 Density Based Selection

In an effort to fix the previous problem, a density based selection method of the rays was implemented. Here an attempt is made in order to keep a fixed space between two successive selections, by means of real distance and not by index distance, as in the uniform sub-sampling method. A matrix is constructed (**RaysDisplacement**) with the displacements between successive LRF rays as elements. Then, the total outline of the current scan is calculated ($Total\,Dist$), which is the sum of the individuals **RaysDisplacement**$[i]$.

Each element of the **RaysDisplacement** matrix is calculated using the cosine law, in the triangles formed by the pairs of successive rays, as can be seen in Fig. 11 with formulas Eqs. 2 and 3.

$$\mathbf{RaysDisplacement}[i] = \sqrt{\mathbf{S}[i]^2 + \mathbf{S}[i+1]^2 - 2 \cdot \mathbf{S}[i] \cdot \mathbf{S}[i+1] \cdot \cos(Laser\,Angle / Laser\,Rays)}\ ,$$

$$0 \le i \le Laser\,Rays - 1 \tag{2}$$

$$Total\,Dist = \sum \mathbf{RaysDisplacement}[i]\ ,$$

$$0 \le i \le Laser\,Rays - 1 \tag{3}$$

Here, $Laser\,Rays$ is the number of the LRF rays, $\mathbf{S}[i]$ an element of the matrix containing the LRF measurements $\mathbf{S}$, and $Laser\,Angle$ the field of view of the LRF. Then, the mean ray distance is calculated by dividing the total scans distance by the number of rays: $Total\,Dist/(Laser\,Rays - 1)$. As in the previous method, the number of selected rays N is constant. To select the correct rays based on density, a uniform sampling of the one dimensional density space was implemented. An illustration of the procedure is in Fig. 12.

The indices of the selected rays are computed with formula Eq. 4:

$$\begin{aligned}\mathbf{PickedIndices}_i &= index\ ,\ (Total\,Dist/N) \cdot (i-1) \\ &\le \mathbf{RaysDisplacement}[\text{index}] \\ &\le (Total\,Dist/N) \cdot (i),\ 1 \le i \le N\end{aligned} \tag{4}$$

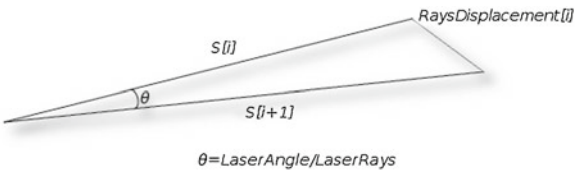

Fig. 11 Cosine law—rays displacement

The result is that the selected rays are uniformly placed in the outline of the LRF scan by means of density. So, the problem of the uniform selection method has been overcome, as a decrement was made in the selected rays lying at close obstacles. On the other hand, the rays that were cast in distant objects were selected more often, because of the greater sparseness.

The experimental results for the density-based selection method can be seen in Figs. 13, 14 and 15.

The results indicate that the density based method failed in two out of three environments and specifically where corridors were in existence. The main drawback of this method, is that when the robot moves in corridors, where a small

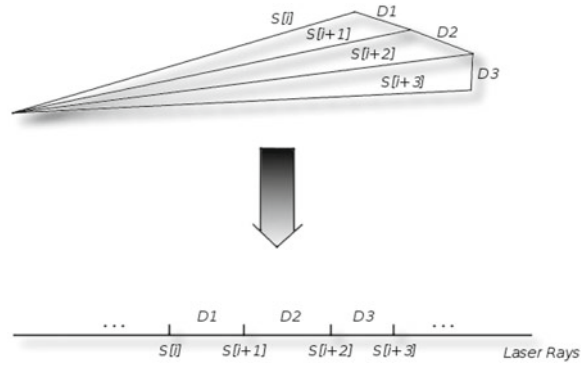

Fig. 12 Construction of 1-D density space out of the LRF outline

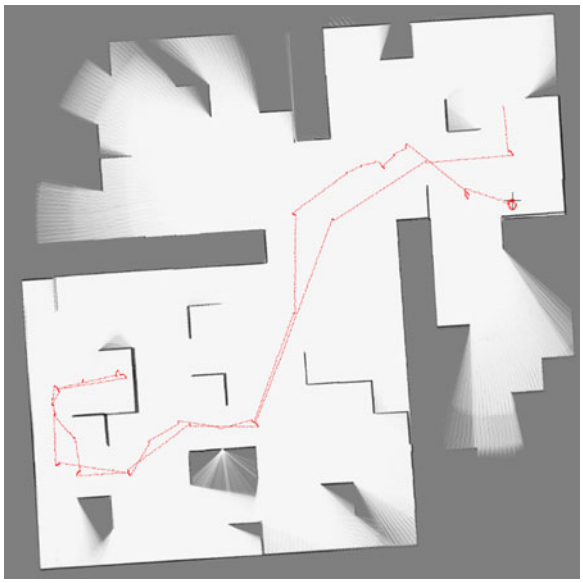

Fig. 13 Sparse environment with many features—density-based selection method

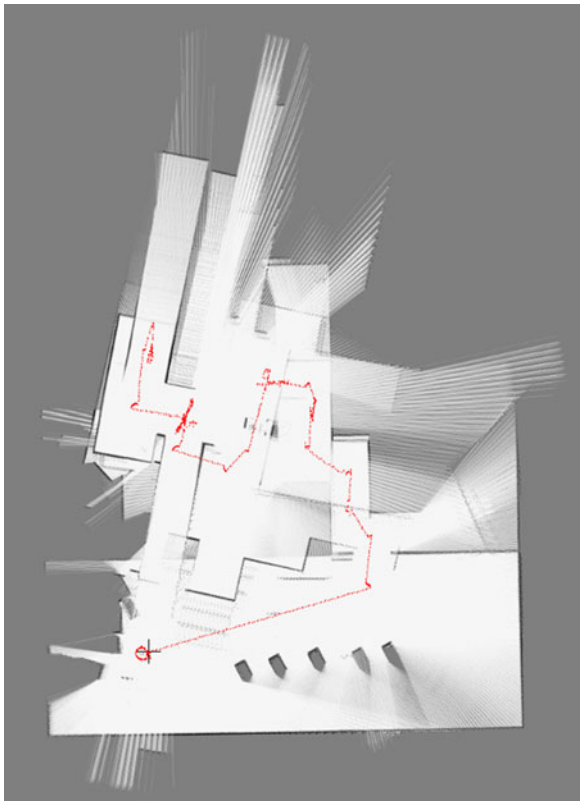

Fig. 14 Environment with narrow corridors—density-based selection method

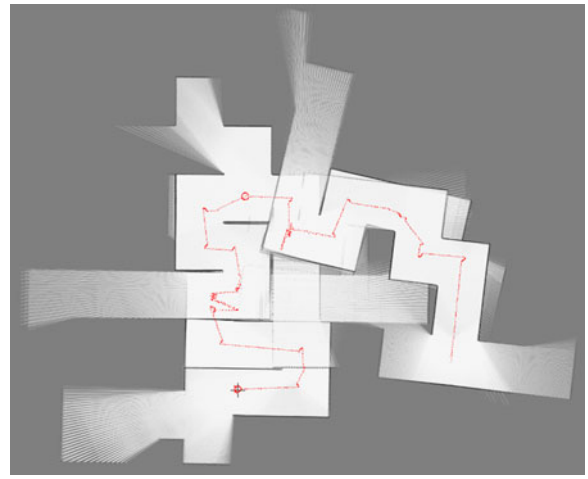

Fig. 15 Single twisted corridor—density-based selection method

number of rays are cast at its end, the vast majority of the selected rays lay at the corridor walls. So because of lack of depth reference SLAM fails.

4.3 Scan Segments and Density Based Selection

In order to fix the deficiency of the density based method, the overall philosophy of the ray selection was reconsidered. The main goal should be to select the critical rays of the laser scan. As critical, are denoted the rays, whose use will be essential to the hill climbing matching. These rays are the distinct features of the laser scan, for example two rays which have a large displacement difference between them. Also, the rays that have larger measurements than others are considered more essential in comparison to the hill climbing procedure, as their local ray density is smaller relatively to the rays correspondent to obstacles close to the robot.

Thus, a segmentation of the laser scan into its parts is performed, according to their detection in relation to distance of successive rays. An example can be seen in Fig. 16.

The first step is the detection of the edges of the scan segments, which are the most essential and easy to detect features of the scan. This is done, simply by checking if the elements in the **RaysDisplacement** matrix are greater than a predefined threshold, which is declared as D_{THRES}.

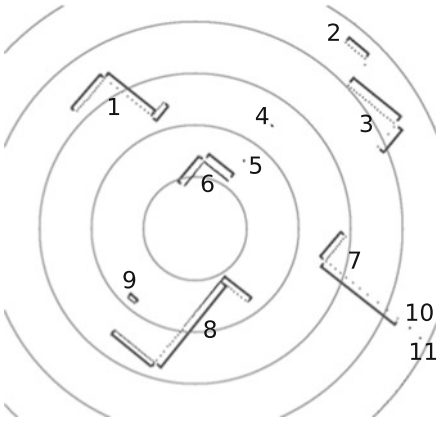

Fig. 16 Segmentation example of a laser scan

So, if **RaysDisplacement**$[i] > D_{\text{THRES}}$, the rays selected are i and $i+1$. An example of the first step in the scan of Fig. 16, is depicted in Fig. 17.

The selected rays are stored in a matrix defined as **FeatureEdges**. A scan segment is the ensemble of the scan rays between two successive elements, stored in **FeatureEdges**. It can be seen that the rays contained at **FeatureEdges** are the limits of the scan segments with the property that their density is extremely small. Usually, these rays are the ones with measurements near $LaserMax$, where $LaserMax$ is the maximum nominal distance of the LRF.

The second step is to perform the density ray selection, described previously, on every scan segment. To apply the density ray selection in a part which begins at $Start =$ **FeatureEdges**$[i]$ and ends at $End =$ **FeatureEdges**$[i+1]$, the mean sparseness S_{spar}, which was used for the sampling in the local density space, must be computed. This is calculated by formula Eq. 5:

$$S_{\text{spar}} = \sum_{j=\text{Start}}^{\text{End}} \textbf{RaysDisplacement}[j]/M \tag{5}$$

where $M = \text{End} - \text{Start}$. Then, the rays are selected by applying the next pseudo-code.

1. $localD = 0$
2. $for\ i = Start : End$
3. $\quad localD = localD +$ **RaysDisplacement**$[i]$
4. $\quad if\ localD \geq \frac{P \cdot S_{\text{spar}}}{\frac{e^{\mathbf{S}[i]}}{\sqrt{\text{Max}}}}$
5. $\quad pick\ ray$
6. $\quad localD = 0$
7. $\quad end\ if$
8. $end\ for$
9. $for\ each\ i$
10. $if\ (\textbf{PickedIndices}_{i+1} - \textbf{PickedIndices}_i) > RAY_{\text{THRES}}$
11. $\quad pick\ ray\ with\ index = \frac{\textbf{PickedIndices}_{i+1} + \textbf{PickedIndices}_i}{2}$

This method is different from the classical density based selection, as it introduces two new concepts in the clause at line 4 of the pseudo-code. Before explaining the necessity of introducing these concepts, we must note that in its simple form, the if clause would be $if\ localD \geq S_{\text{spar}}$. By using this form, the low density (and high sparseness) rays, over the high density ones, would be selected, which is desired, although their number would be close to $End - Start$. As it has been mentioned before, it is desirable to give more credit to scans with measurements close to $End - Start$, and at the same time avoid the rays close to the robot. This is achieved by inserting the ray measurement in the if clause: $if\ localD \geq \frac{S_{\text{spar}}}{e^{\mathbf{S}[i]}}$.

By doing this, if the measurement is large, the second part of the inequality becomes small, allowing more rays to be selected. On the other hand, if $\mathbf{S}[i]$ is small, rays are selected more sparsely. To intensify the results, the exponential clause was inserted. The laser measurements addition, had a good performance to large and rich in features spaces, but led to extremely small number

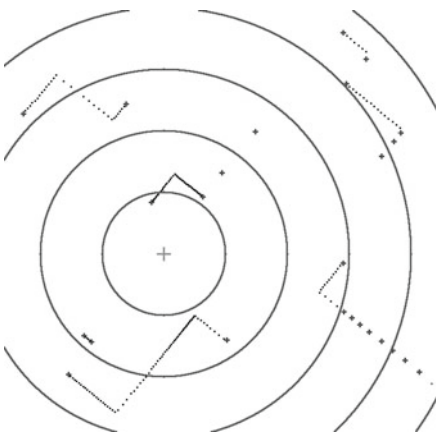

Fig. 17 Elements of **FeatureEdges** matrix

of selected rays in narrow places. That is why the measurement of each ray was replaced by the relative measurement. The latter is the one divided by the square root of the maximum measurement of the scan $(Max) : if\ local D \geq \frac{S_{\text{spar}}}{\frac{e^{S[i]}}{\sqrt{Max}}}$.

In addition, the parameter P was inserted in the formula, in order to make the algorithm more adaptive to various environments, as well as to maintain a constant performance. P is a parameter that changes its value autonomously according to the number of selected rays of the previous iteration. If the number of the selected rays is small, it means that the robot is in a featureless environment and is sufficient to select a larger number of rays, by decreasing the value of P. On the other hand, if the number of selected rays is large, there is a need to select less rays, in order to lower the execution time, which is achieved by increasing the value of P. For the experiments presented, the minimum number of rays was 30, the maximum 50 and a typical initial value for P is 100.

The final step of the method, is to patch the results of the uniform sub-sampling method where is needed. Specifically in narrow environments like corridor corners, this method selects only the rays that have large measurements, resulting in a lack

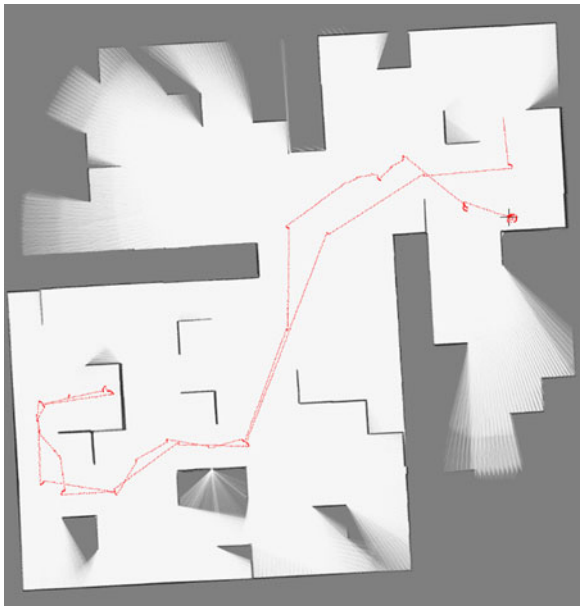

Fig. 18 Sparse environment with many features—density-based with scan segments method

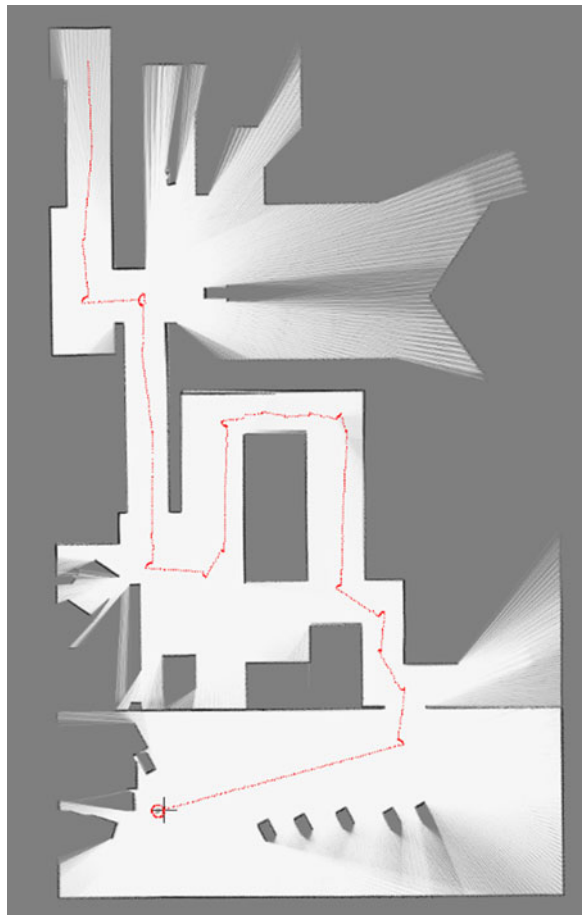

Fig. 19 Environment with narrow corridors—density-based with scan segments method

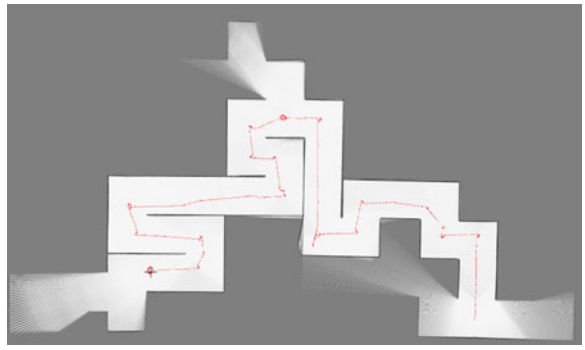

Fig. 20 Single twisted corridor—density-based with scan segments method

of reference of the objects near the robot. So to overcome this problem, we select some extra rays produced by the code in lines 10 and 11, where RAY_{THRES} is a predefined index step.

The experimental results for the scan segments and density-based selection method are illustrated in Figs. 18, 19 and 20.

It is clear that the quality of the maps produced by this method is excellent in all three cases. To conclude, this method fixes the corridor problem that the density method suffered from, as it selects many distant rays along with simultaneous selection of some close-up ones, in order to maintain high reference.

5 Scan Matching-Hill Climbing Method

Hill climbing is a very popular and efficient local search optimization method for finding optimal solutions to complex problems. In principle hill climbing algorithm executes iterations in which the currently known best solution individual is used to produce one offspring. If this new individual is better than its parent, it replaces it and the procedure is performed again. In this sense, it is similar to an evolutionary algorithm with a population size of one. Therefore, in the current chapter, the algorithm will be described using genetic algorithm terminology. The hill climbing method is used to find the correct transformation between the current scan and the global map (which is the inverse of the robot translation and rotation). Next, the genome of the individual, the fitness function and the genetic operators used, are described.

5.1 Genome

The individual's genome is the following: **Genome** = $[Dx, Dy, Dtheta]$, where Dx, Dy, $Dtheta$ are the changes in the robot x, y and angle coordinates. The objective is to find the genome, that fits best the current scan with the global map.

5.2 Fitness Function

The fitness function must be maximized for the correct transformation of the robot, that is for the exact alignment of the current scan to the global map's obstacles. The robot's pose is defined as $[R_X, R_Y, R_{th}]$, the coordinates of a random laser ray in the robot's reference system as $[Ray^i_X, Ray^i_Y]$, the ray's angle to the robot's orientation Ray^i_{Theta}, the global coordinates of the ray as $[GRay^i_X, GRay^i_Y]$,the genome of a random individual as $[Dx, Dy, DTheta]$ and the 2-D map matrix as **Map**. **Map** is a two dimensional matrix of type unsigned character, that holds the possibility of each cell to be free. The value of the cell is 255, 0 or 127, if the cell is free, occupied or unexplored respectively.

The fitness value for a scan is calculated as follows:

1. $fitnessValue = 0$
2. $for\ each$ **PickedIndices** $as\ i$
3. $Ray^i_X = S[i] \cdot cos(Ray^i_{Theta})$
4. $Ray^i_Y = S[i] \cdot sin(Ray^i_{Theta})$
5. $$\begin{bmatrix} GRay^i_X \\ GRay^i_Y \end{bmatrix} = \begin{bmatrix} \cos(R_{th}+DTheta) & -\sin(R_{th}+DTheta) \\ \sin(R_{th}+DTheta) & \cos(R_{th}+DTheta) \end{bmatrix} \cdot \begin{bmatrix} Ray^i_X \\ Ray^i_Y \end{bmatrix} + \begin{bmatrix} R_X + Dx \\ R_Y + Dy \end{bmatrix}$$
6. if **Map**$[GRay^i_X][GRay^i_Y] = 127$
7. $fitnessValue = fitnessValue - 100$
8. $continue\ to\ next\ i$
9. $end\ if$
10. $$\begin{aligned} fitnessValue = {} & fitnessValue \\ & + 10 \cdot (255 - \textbf{Map}[GRay^i_X][GRay^i_Y]) \\ & + (255 - \textbf{Map}[GRay^i_X - 1][GRay^i_Y]) \\ & + (255 - \textbf{Map}[GRay^i_X + 1][GRay^i_Y]) \\ & + (255 - \textbf{Map}[GRay^i_X][GRay^i_Y - 1]) \\ & + (255 - \textbf{Map}[GRay^i_X][GRay^i_Y + 1]) \end{aligned}$$
11. $end\ for$

As it can be seen, the fitness value is calculated by summing the possibilities of occupancy in the selected laser rays according to the transformation of the hill climbing individual. In lines 3 and 4, the local coordinates of the scan are calculated, and in line 5 they are transformed to global coordinates, by applying the transformation stored in the genome of the individual. If this point on the map is unknown (value 127), the algorithm punishes the individual by decreasing its fitness value. If not, the fitness value is increased by the

possibility of that point to be occupied. Also the possibilities of four neighbouring points are taken into consideration, but with a smaller weight than the point itself.

5.3 Genetic Operators and General Functionality

The method selected for hill climbing is RRHC (Random Restart Hill Climbing), or Shotgun Hill Climbing [24]. At random restart hill climbing the individual's genome is initialized at the presumed new robot transformation, based on the speed or the kinematic model, and a mutation is performed. The mutation consists of a random small change in the genome's elements, resulting in a new proposed robot pose. If the fitness value of the individual is higher than the previous one, a similar mutation is applied at the new individual. This procedure continues, until one individual has a lower fitness value than its previous one. Then, the best global result is updated if needed, and the individual is randomly reinitialized. The above process is performed for N_{HC} iterations, or till the best fitness value reaches a satisfying threshold. Finally, when the best transformation is found, the robot pose is updated by formula Eq. 6:

$$\begin{aligned} Rx &= Rx + BestDx \\ Ry &= Ry + BestDy \\ Rtheta &= Rtheta + BestDTheta \end{aligned} \tag{6}$$

where $BestDx$, $BestDy$ and $BestDTheta$ are the output values of the Hill-climbing module.

As far as the algorithm technical details are concerned, we assume that the robot has a maximum linear speed of 0.2 m/sec and a maximum rotational speed of 0.2 rads/sec. Based on this, on each iteration, the robot's new position is located in a circle of maximum radius of 0.2 m. In the experiments, we suppose that one pixel has a side of 0.02 m, so the correct transformation must be searched within a disk of $10px$ radius. The mutations consist of altering the genome's values by a number bounded in $[-5, 5]$ pixels for the x and y coordinates, and $[-0.1, 0.1]$ radians for the orientation of the robot. Also, the random reinitialization takes place in the vicinity of the robot's presumed pose, and the difference from it, is bounded in $[-20, 20]$ pixels for the x,y coordinates and $[-0.25, 0.25]$ for the robot's orientation. These limits were specified after experiments, and are proposed to ensure the correct functionality of the algorithm, as the search space is larger than the expected one. Also, the distribution of sampling in each of these intervals is uniform.

One major problem is to determine the value of the maximum number of hill climbing iterations N_{HC}, as the HC algorithm is the major computational bottleneck of the algorithm. So, if a large value is selected, the results will be qualitative

Table 2 Performance for different values of N_{HC} – Environment 2

N_{HC}	MFV	OMSE (px^2)	CMSE (px^2)
100	0.4400	216.67	13012.42
500	0.5713	39.78	4641.84
1000	0.6075	32.67	807.14
5000	0.6541	28.45	1227.52
10000	0.6730	24.23	14.16
20000	0.6770	18.24	11.36
30000	0.6817	16.34	8.08

Table 1 Performance for different values of N_{HC} – Environment 1

N_{HC}	MFV	OMSE (px^2)	CMSE (px^2)
100	0.3668	410.22	12032.44
500	0.576	15.68	1562.26
1000	0.6544	5.53	8.04
5000	0.7079	6.42	10.14
10000	0.7223	5.40	4.48
20000	0.7304	3.87	3.03
30000	0.7374	3.93	4.17

Table 3 Performance for different values of N_{HC} – Environment 3

N_{HC}	MFV	OMSE (px^2)	CMSE (px^2)
100	0.4312	21.27	12639.03
500	0.5398	21.35	6195.25
1000	0.6204	2.22	21.00
5000	0.6788	0.71	6.17
10000	0.6916	0.76	4.82
20000	0.6939	0.70	5.15
30000	0.6949	0.62	5.92

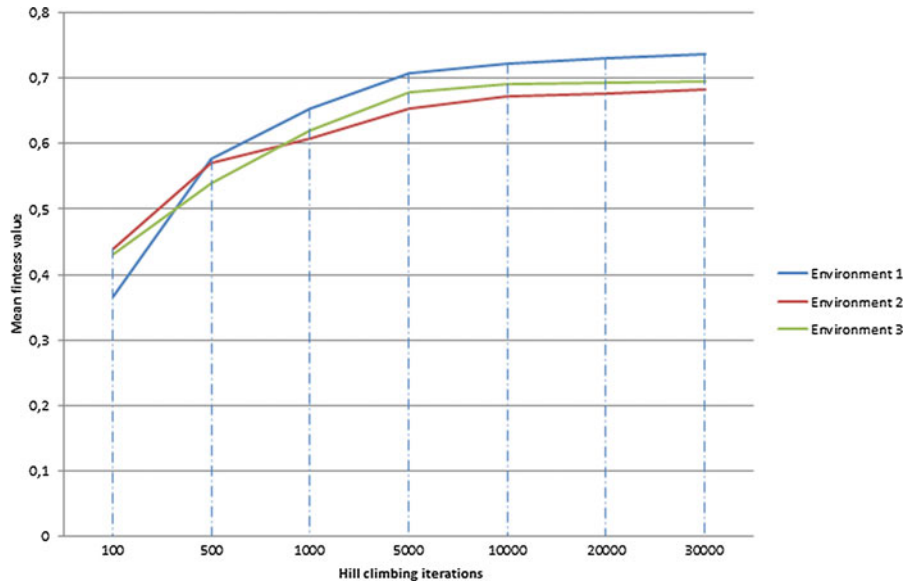

Fig. 21 MFV in the three environments for different values of N_{HC}

but the execution time large, while with a small value the opposite is achieved. To determine the "correct" value of the maximum iterations, a series of experiments were performed in each of the three environments, with different values of N_{HC}, where OMSE and CMSE were measured, as well as the mean fitness value (MFV) of the experiment. It must be clarified that the bigger the MFV, the better the hill climbing algorithm manages to match the scan to the map. The results are illustrated in Tables 1, 2 and 3 and graphically at Figs. 21, 22 and 23.

In addition, experiments were made, in order to determine in which iteration the hill climbing algorithm finds the maximum fitness value. The result is depicted in Fig. 24.

The conclusion which can be reached, is that above the 10000 maximum iterations of the hill

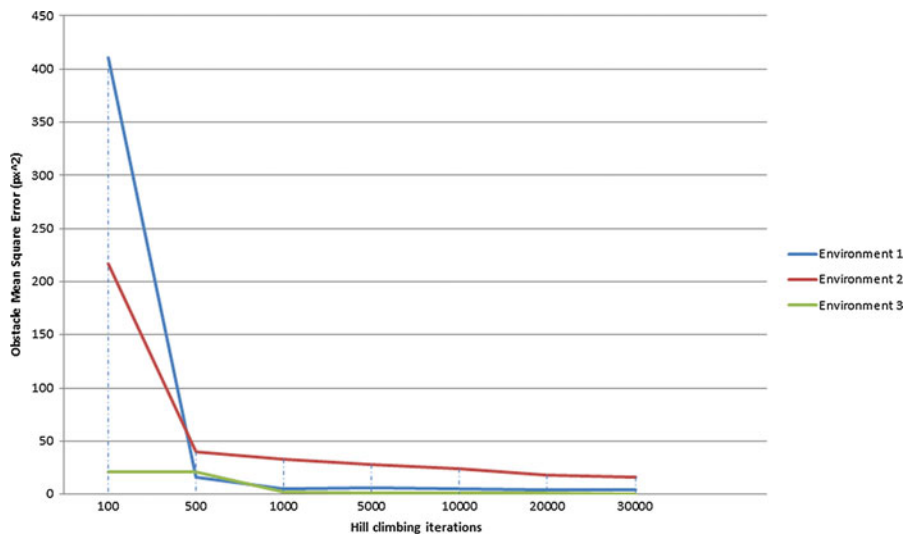

Fig. 22 OMSE in the three environments for different values of N_{HC}

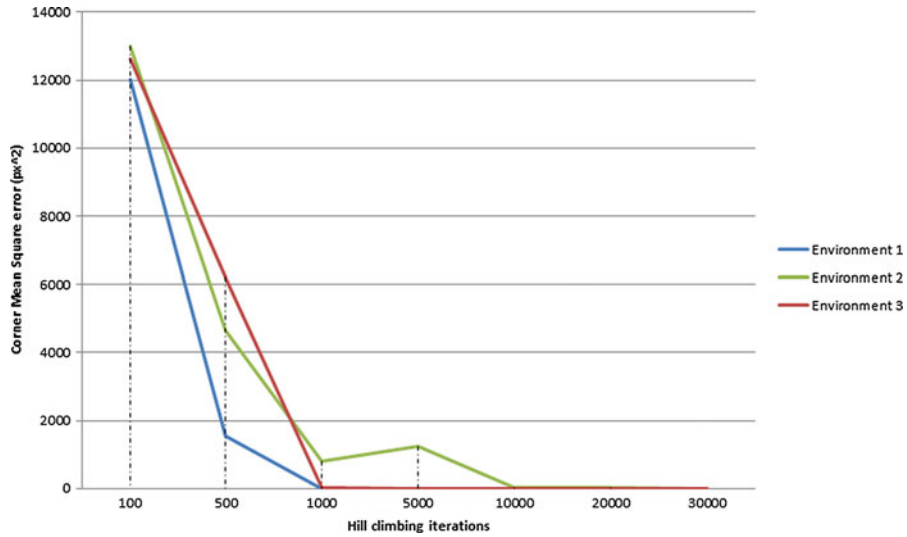

Fig. 23 CMSE in the three environments for different values of N_{HC}

climbing algorithm the performance metrics remain almost the same. Also, the diagram of Fig. 24 indicates that the vast majority of the solutions are computed in less than 5000 HC iterations. Conclusively, and to include a margin of error, the value of maximum HC iterations is chosen to be $N_{HC} = 10000$.

6 Map Update

The last step of the SLAM iteration is the map update. Up to this point, the correct new pose of the robot has been found by performing the ray selection and the hill climbing algorithm. Then the map must be updated based on this robot

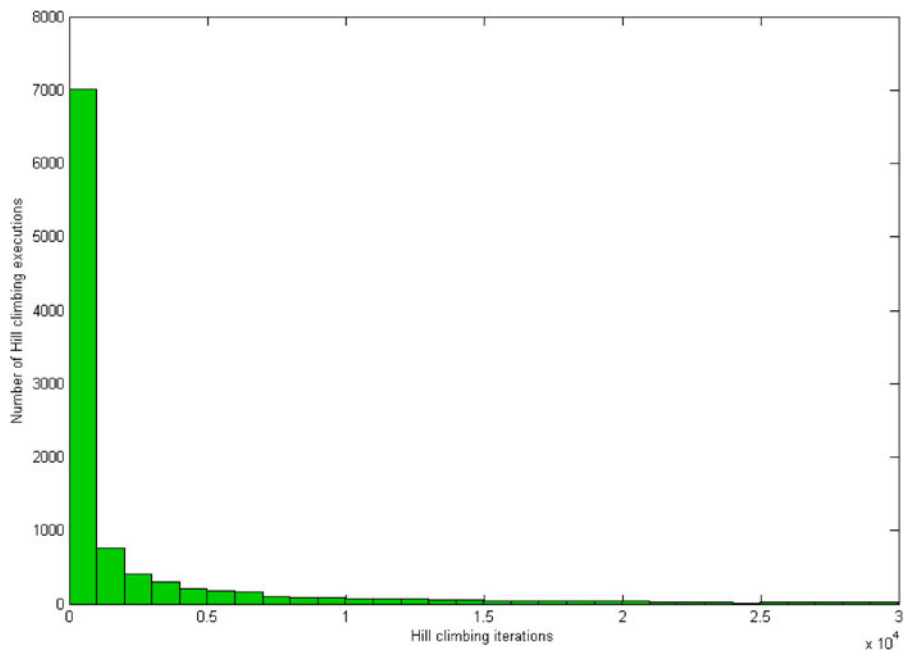

Fig. 24 Distribution of the number of hill climbing iteration number where the best fitness value was discovered i.e. at what iteration the algorithm found the solution

pose, and the current laser scan. The pixel update method of the map chosen is the asymptotic one, that is a specific pixel can never have a possibility of 1 or 0, but if for example the pixel is unoccupied, its possibility will asymptotically increase to 1.

The pixels that are updated are the ones that are in the laser scan's field of view. The update method is based on the ray casting concept, initiating from the robot. The intensity of the update is defined as the rate at which the update is done. If the intensity is big, the change of a pixel's value is bigger than the nominal. In the proposed algorithm, the intensity of the update is proportionate to the mean sparseness of the current scan. If the robot is in a confined space, the update of the map has small intensity, as the ray density is big. This is done, in order to eliminate the accumulative errors due to the lack of features.

Finally at the map update module, the pixels that are occupied are updated in a more intense way than the pixels of the unoccupied space. The goal of this decision is to enhance the obstacles, so that the Hill climbing algorithm can have a better reference to match the scan. The mathematical notation of the update is as follows.

$$
\begin{aligned}
&for\ each\ cell\ in\ each\ LRF\ ray's\ field\ of\ view\\
&\quad if\ cell\ is\ obstacle\ free\\
&\qquad c' = c + (1.0 - c) \cdot S_{\text{spar}}\\
&\quad else\ if\ cell\ is\ occupied\\
&\qquad c' = c - c \cdot S_{\text{spar}} \cdot Dens_{\text{occu}}\ ,\ \ Dens_{\text{occu}} > 1
\end{aligned}
$$

Table 4 Experimental results – OMSE

OMSE (px^2)	Environment 1	Environment 2	Environment 3
All rays selected	4.94	54.41	0.89
Uniform subsampling	4.48	50.29	0.89
Density	7.26	25.74	3.76
Density with segments	5.04	17.56	0.76

Here, c is the previous value of the cell, S_{spar} is the mean sparseness of the scan and $Dens_{\text{occu}}$ a coefficient larger than 1, which increases the intensity of the update of the occupied cells. In Fig. 25, three results are illustrated for different approaches in the update map functionality. The map on the left was produced with the method described above and its performance measures are $OMSE = 17.56px^2$ and $CMSE = 14.16px^2$. The map in the middle is the result of the algorithm, where the update module does not use any intensity adjustments, in which $OMSE = 21.97px^2$ and $CMSE = 31.09px^2$. The resulting map has almost the same structure as the ground truth, but contains larger errors in overall quality as the increase in OMSE indicates. Finally the map in the right was produced by performing the sparseness intensity adjustment, but the occupied and

Fig. 25 Experiments with different map update intensity settings

Table 5 Experimental results – CMSE

CMSE (px^2)	Environment 1	Environment 2	Environment 3
All rays selected	5.35	3994.91	4.50
Uniform subsampling	4.78	4459.27	3.37
Density	6.14	10425.36	5832.5
Density with segments	4.57	10.45	3.75

unoccupied cells were updated with the same intensity. Here, $OMSE = 27.59px^2$ and $CMSE = 411.9px^2$, something that indicates a structural loss. The conclusion is that both of these adjustments are useful in maintaining the map's structural and overall quality.

7 Experiments

In order to determine which of the ray selection methods produces the best results, the performance metrics described in chapter 3 of each of the three testing environments for each method are presented in Tables 4 and 5, as well as the execution time measurements in Table 6.

The proposed SLAM algorithm was also tested with the autonomous robot of the team P.A.N.D.O.R.A. [25], which participates in the RoboCup-RoboRescue competitions.

The P.A.N.D.O.R.A. robot (Fig. 26) is equipped with the URG-04LX laser range finder, which has 726 rays, 240° field of view and a maximum

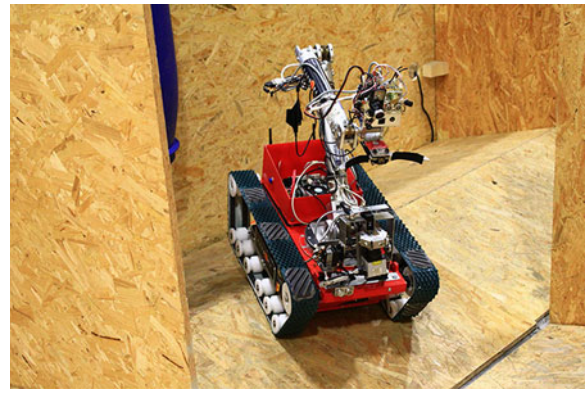

Fig. 26 The P.A.N.D.O.R.A. vehicle at RoboCup-RoboRescue 2011, Turkey, Istanbul

measurement distance of 4 meters. In addition it employs various sensors for victim identification, such as a stereo camera to perform face, skin and motion detection, as well as many other image related algorithms related to RoboCup-RoboRescue, thermal sensors, a CO2 sensor and an array of microphones for sound detection. Also a 5-DOF robotic arm is attached to the robot's main body.

The experiment was performed in the Robotics Lab at Aristotle University of Thessaloniki (AUTH) (Environment 4), which is a complex environment with many small obstacles like chairs, desks etc. The results are illustrated in Figs. 27, 28, 29 and 30.

For an exact comparison between the different ray selection methods a laser scan log was stored from each map and was used off line. The experiments were performed only once, since the input was obtained from the log files, and therefore

Table 6 Experimental results – execution times

	Environment 1		Environment 2		Environment 3		Environment 4	
	Mean time (ms)	Mean selected rays	Mean time (ms)	Mean selected rays	Mean time (ms)	Mean selected rays	Mean time (ms)	Mean selected rays
All rays selected	241,7	247,24	248,54	266,26	253,23	268,4	658,96	555,54
Uniform subsampling	62,14	49,43	61,74	53,26	63,86	53,66	57,65	38,02
Density	44,67	30,85	44,27	32,6	46,53	34,27	55,16	33,82
Density with segments	66,8	49,33	54,2	45,78	55,53	42,57	86,78	54,04

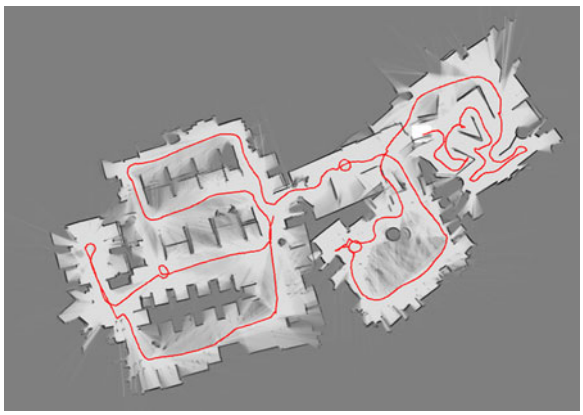

Fig. 27 All rays selected method

no randomness or noise of any kind was introduced. The system on which the algorithms were executed, is a PC with Intel Core i7 CPU at 2.80 GHz, 4 GB RAM, running Ubuntu 9.10 Karmic distribution. The described system is equipped with 4 cores and hyper-threading, but as the proposed algorithms have no parallel procedures, only one core was used.

The results concerning the OMSE, are consistent with the visual feeling of the exported maps. All algorithms achieved a relatively low OMSE in Environment 1, as it is not that challenging, due to the existence of multiple spatial features. In contrast, the Density with segments method had

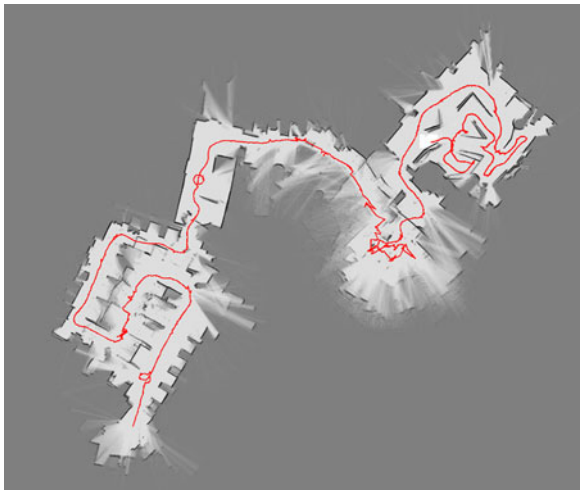

Fig. 28 Uniform sub-sampling method

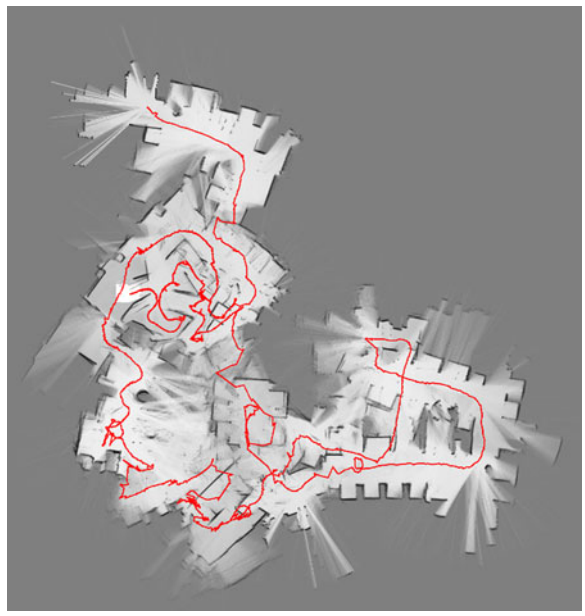

Fig. 29 Density based method

the smallest error in the second environment, as for it maintained the correct environment structure. Finally in the third environment all methods succeeded except for the Density based, which lost reference at the middle of the corridor, producing a poor map.

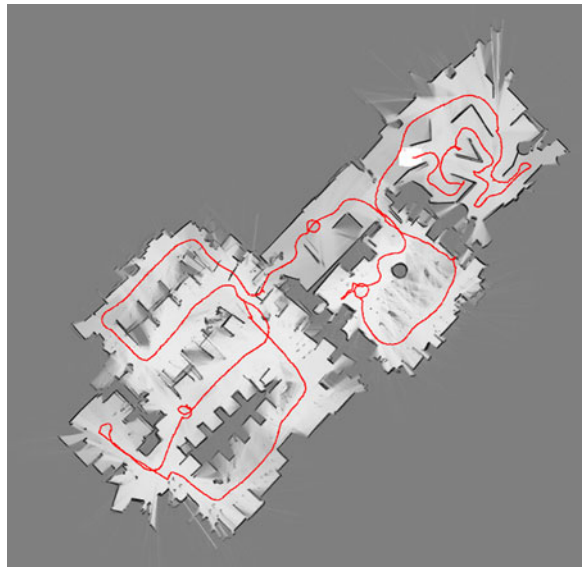

Fig. 30 Density based with scan segments method

CMSE measurements are in concurrence with the prior results, with the exception that the erroneous maps receive a much bigger mean square error than the OMSE metric. This is expected, as the CMSE takes into account selected features of the environment and compares them with the same structural features, having one to one correspondence. On the other hand OMSE has an obstacle correspondence based on a proximity measure. So even if a map is deformed, there is a chance that OMSE returns a low error, something that CMSE does not. To conclude, as far as map quality is concerned, the "density-based and scan segments" method has the best performance compared to the other three proposals, since it creates a more precise map in comparison to the ground truth, as it has achieved the lowest errors in all three environments. It must be noted that unfortunately OMSE and CMSE were not computed for the Environment 4, as a ground truth map could not be acquired.

Regarding execution times the results indicate that the mean iteration times of the All rays selected method are considerable higher than the desired one, which is approximately 100 ms (since the refresh rate of URG-04LX is 10 Hz). In conclusion a ray selection method is encouraged, as it outperforms the "all rays selected" method by means of execution time and computational resources. Also it is obvious, that the ray selection must be done in a clever way, in order to make a good use of the critical rays and discard the redundant information. Although the average number of selected rays of the density and scan segments method is slightly larger than the density based method, the experimental results indicate that the difference in performance is in favor of the scan segments density method, both in simple and complex environments, a crucial feature for on-line SLAM applications.

8 Conclusions—Future work

In this paper we proposed a novel approach for the SLAM problem in a structured indoors environment. This SLAM scheme performs scan matching by implementing a ray-selection method, in order to reduce the computational cost of the algorithm. Also a scan-to-map matching technique was used in order to reduce the accumulative errors and an RRHC algorithm was applied for computation of the proper robot transformation between two iterations of the algorithm. Finally, several novel modifications were created, such as dynamic intensity map updates, aiming at increasing the map's quality.

To conclude the proposed CRSM SLAM produces high quality maps while requiring low execution time and small computational power. The interesting result of this paper is that we proved, based on the experimental results, that using the full amount of information available does not always improve the results, but depending on the problem it can lead to reduction of quality. That's because in the specific problem, the LRF's information contains redundant data, as many adjacent rays measure the same distance.

It must be noted, that this method was used to produce the maps of the autonomous robot of the team P.A.N.D.O.R.A., in the RoboCup-RoboRescue competition in Istanbul, where the resulting maps were described as "crisp and of good quality". The proposed method was also tested on two LRF data sets from Radish: The Robotics Data Set Repository [26]. The first set was recorded in the Interior of the Intel Research Lab in Seattle, and the second in the University of Freiburg, building 079, AIS-Lab. The results are depicted in Figs. 31 and 32 respectively.

The obvious problem of the proposed method is that it is inferior in the loop closure domain, as it failed (although not in a large degree) to close the loop successfully in the Intel Research Lab data. On the other hand the performance in multi-featured environments, such as the building 079, AIS-Lab in University of Freigburg, can be characterized as more than satisfying.

Finally a comment about scalability, is that the proposed algorithm is highly scalable, as its parts make use of information in the vicinity of the robot, regardless of the map's size. Of course the same does not apply for the memory needed, as the occupancy grid map grows linearly to the explored environment's area. Conclusively, if enough memory is provided, the SLAM algorithm has constant execution time.

Fig. 31 Intel lab in Seattle

Various alterations can be made in the future towards the improvement of the proposed SLAM method. The most crucial of them, is to employ a method aiming to solve the problem of loop-closure, which is a major field of research in robotic intelligent systems. The solution can be chosen from a variety of algorithms such as particle filters, classification of ray scans by extracting parameters, insertion of features (lines, corners) and use of Kalman filters. Finally one of the future plans is to increase the method's efficiency and quality of the map by periodically processing the occupancy grid using filters, in order to eliminate the obstacles pixel noise.

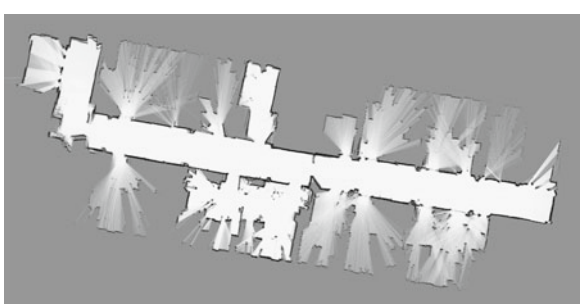

Fig. 32 University of Freiburg, building 079, AIS-Lab

References

1. Mingas, G. Tsardoulias, E. Petrou, L.: An FPGA implementation of the SMG-SLAM algorithm. Microprocess. Microsyst. **36**(3), 190–204 (2012)
2. Milstein, A.: Occupancy grid maps for localization and mapping: motion planning. In: Jing, X.-J. (ed.) InTech, pp. 381–408 (2008)
3. Rusinkiewicz, S., Levoy, M.: Efficient variants of the ICP algorithm. In: Proceedings of the 3rd Intl. Conf. on 3D Digital Imaging and Modeling, p. 145152 (2001)
4. Brenna, M.: Scan matching, covariance estimation and SLAM: models and solutions for the scanSLAM algorithm. Master Thesis, Politecnico di Milano (2008)
5. Besl, P.J., Mckay, N.D.: A method for registration of 3-D shapes. IEEE Trans. Pattern Anal. Mach. Intell. **14**(2), 239–256 (1992)
6. Zhang, Z.: Iterative point matching for registration of free-form curves and surfaces. Int. J. Comput. Vis. **13**(2), 119–152 (1994)
7. Lu, F., Milios, E.: Robot pose estimation in unknown environments by matching 2D range scans. J. Intell. Robot. Syst. **18**(3), 249–275 (1997)
8. Diosi, A., Kleeman, L.: Laser scan matching in polar coordinates with application to SLAM. In: IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2005), pp. 3317–3322 (2005)
9. Ze-Su, C., Bing-Rong, H., Hong, L.: An improved polar scan matching using genetic algorithm. Inf. Technol. J. **6**, 89–95 (2007)
10. Montemerlo, M., Roy, N., Thrun, S., Hahnel, D., Stachniss, C., Glover, J.: CARMEN the carnegie mellon robot navigation toolkit. http://carmen.sourceforge.net (2002). Accessed 29 Oct 2011
11. Grisetti, G., Stachniss, C., Burgard, W.: Improved techniques for grid mapping with Rao-Blackwellized particle filters. IEEE T. Robot. **23**(1), 34–46 (2007)
12. Olson, E.B.: Real-time correlative scan matching. In: IEEE International Conference on Robotics and Automation ICRA 09, pp. 4387–4393 (2009)
13. Martinez, J.L., Gonzalez, J., Morales, J., Mandow, A., Garcia-Cerezo, A.J.: Mobile robot motion estimation by 2D scan matching with genetic and iterative closest point algorithms. J. Field Robot. **23**(1), 21–34 (2006)
14. Nieto, J., Bailey, T., Nebot, E.: Recursive scan-matching SLAM. Robot. Auton. Syst. **55**(1), 39–49 (2007)
15. Lakaemper, R., Adluru, N.: Force field simulation based laser scan alignment. In: Lazinica, A. (ed.) Recent Advances in Multi Robot Systems, InTech, pp. 326 (2008)
16. Biber, P., Straber, W.: The normal distributions transform: a new approach to laser scan matching. In: IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2003), vol. 3, pp. 2743–2748. (2003)
17. Nunez, P., Vazquez-Martin, R., Bandera, A., Sandoval, F.: Fast laser scan matching approach based on

adaptive curvature estimation for mobile robots. Robotica **27**, 469–479 (2009)
18. Sohn, H., Kim, B.: VecSLAM: an efficient vector-based SLAM algorithm for indoor environments. J. Intell. Robot. Syst. **56**(3), 301–318 (2009)
19. Cox, I.J.: Blanche-an experiment in guidance and navigation of an autonomous robot vehicle. IEEE Trans. Robot. Autom. **7**(2), 193–204 (1991)
20. Taleghani, S. Sharbafi, M.A., Haghighat, A.T., Esmaeili, E.: ICE matching, a robust mobile robot localization with application to SLAM. Proc. ICTAI **1**, 186–192 (2010)
21. Burgard, W., Stachniss, C., Grisetti, G., Steder, B., Kmmerle, R., Dornhege, C., Ruhnke, M., Kleiner, A., Tards, J.D.: A comparison of SLAM algorithms based on a graph of relations. In: Proceedings of IEEE/RSJ International Conference on Intelligent robots and systems, pp. 2089–2095. St. Louis, MO, USA (2009)
22. Carpin, S., Lewis, M., Wang, J., Balakirsky, S., Scrapper, C.: USARSim: a robot simulator for research and education. IEEE International Conference on Robotics and Automation, pp. 1400–1405 (2007)
23. Gerkey, B.P., Vaughan, R.T., Howard, A.: The player/stage project: tools for multi-robot and distributed sensor systems. In: Proceedings of the International Conference on Advanced Robotics (ICAR 2003), pp. 317–323 (2003)
24. Russell S., Norvig, P.: Artificial intelligence: a modern approach 2nd edn. Ser. Prentice Hall series in Artificial Intelligence, Prentice Hall (2002)
25. Antaris, S., Doulgeri, Z., Nikolaidis, G., Papadopoulos, C., Papanikas, G., Papazoglou, A., Petridis, V., Petrou, L., Serenis, C., Skolarikis, M., Tsalidis, P., Tsardoulias, E., Zolotas, C.: Program for the advancement of non-directed operating robotic agents. In: 19th International Conference on Automated Planning and Scheduling (ICAPS). Thessaloniki, Greece (2009)
26. Howard A., Roy, N.: The robotics data set repository (radish). http://radish.sourceforge.net/ (2003). Accessed 22 May 2009