Fast and accurate map merging for multi-robot systems*

Stefano Carpin

School of Engineering
University of California, Merced
5200 North Lake Rd.
Merced, CA, 95343

Abstract

We present a new algorithm for merging occupancy grid maps produced by multiple robots exploring the same environment. The algorithm produces a set of possible transformations needed to merge two maps, i.e translations and rotations. Each transformation is weighted, thus allowing to distinguish uncertain situations, and enabling to track multiple cases when ambiguities arise. Transformations are produced extracting some *spectral information* from the maps. The approach is deterministic, non-iterative, and fast. The algorithm has been tested on public available datasets, as well as on maps produced by two robots concurrently exploring both indoor and outdoor environments. Throughout the experimental validation stage the technique we propose consistently merged maps exhibiting very different characteristics.
Keywords: Multi-robot systems Mapping Hough Transform

1 Motivation

Research in multi-robot systems is motivated by multiple rationales [17]. Among them is the possibility to build systems that exhibit superior performance in terms of robustness and time needed to complete assigned missions. Despite the fact that these goals drove a lot of research on the topic in the last two decades, the design and implementation of truly robust multi-robot systems is still a challenge. Many difficulties arise while putting together various components. One of the main problems is the integration of information collected by different robots operating in different parts of the environment. Information integration can happen at different levels, but in order to overcome possible communication bottlenecks, an appealing approach consists in merging high level information extracted from raw low level data collected by sensors. In this paper we address one of these problems, namely the integration of occupancy grid maps produced by various robots exploring different parts of the same environment. Simultaneous localization and mapping (SLAM) is a well established field of research that still draws significant attention due its enormous practical importance. Most research, however, focuses, on the problem of building a single map, either from data coming from a single robot, or from multiple robots. In both cases, however, the map being built is unique. In many situations this approach is not practical at all. Besides the problems arising from the increased dimensionality of the composed map, when robots are exploring large environments continuous communication may be unavailable, and robots may be able to exchange data only during sporadic rendezvous. As detailed in section 2, research in map merging is still at its dawn, despite its acknowledged importance.

*To appear in *Autonomous Robots*, accepted in June 2008

In this paper we present a novel algorithm for merging multiple maps that is both fast and accurate. The algorithm can be used to merge maps represented as occupancy grids. This technique advances the current state of the art, including our previous work in the area, in various ways. First, it is extremely fast. Grid maps with more than 250 thousands of cells can be merged in about 500 ms on common desktop PCs. This is orders of magnitude faster than approaches based on iterative stochastic searches. Secondly, the algorithm is deterministic. Therefore its results are repeatable, and its computation time predictable. This later aspect is particularly appealing when the map merging step has to be embedded into a more complex controller with soft or hard real time constraints. The algorithm computes a few functions describing salient features of the maps to be merged, and builds its results from them. Finally, even though the algorithm is deterministic, it can be used in a probabilistic framework. In fact, rather than producing a single solution, it can produce a set of weighted possible solutions. Therefore, when ambiguities arise one can identify and track multiple hypothesis until they are solved. Our technique works particularly well when merging maps produced by multiple robots exploring building interiors. Indoor environments exhibit numerous linear features that can be quickly detected and exploited for map merging. However, the algorithm has been implemented and tested on various maps produced by mobile robots operating in very diverse indoor and outdoor environments, and it has produced consistently good results.

This paper is organized as follows. Section 2 illustrates related literature in the field of map merging. The problem studied in this paper is formalized in section 3, as well as relevant notation. The core of the paper is section 4, where the mathematical foundations of the algorithm are described and the whole approach is presented. Section 5 shows multiple results aimed to test the robustness of the approach and to illustrate its performance when merging very diverse maps. Finally in section 6 we draw the conclusions and outline some possibilities for future work.

2 Related work

In this section we address exclusively related literature concerning map merging. Therefore we do not discuss the SLAM problem. The interested reader is referred to the recent book by Thrun and colleagues for an up to date survey on building maps [20]. In a paper appeared in 2003 Konolige and colleagues state that "[map merging] is an interesting and difficult problem which has not enjoyed the same attention that localization and map building have" [12]. Although some publications on the topic appeared after this statement, the area is still under explored. Most approaches proposed so far assume that maps are represented as occupancy grid maps, and this is the standpoint we assume in this paper. In an earlier stage of this research we have investigated this problem casting it as a stochastic search to solve an optimization problem [2]. A suitable transformation (i.e. a rotation and a translation) aiming to overlap two grid maps is sought in the space of possible transformations, and the functional being optimized measures the overlap between the two maps. In a subsequent refinement [1] we introduced mechanisms to detect failures, and a more sophisticated way to guide the search. Both approaches are guaranteed to find the optimal solution when the number of iterations tends to infinite. Due to their iterative nature, their computational requirements are notable, making them unsuited for real time operation. Howard et al. address the map merging problem exploiting sporadic rendezvous between robots [10]. When two robots meet during the mission, their relative pose is determined and their individual maps are merged into a combined one. While this approach has been demonstrated to work well in practice, one shortcoming is the necessity to compute the relative positions in order to merge maps. If relative localization requires line of sight, for example when it is based on vision, robots will be unable to merge their maps when they can communicate but do not see each other. A similar idea is exploited and refined in a paper

by Fox and colleagues [7]. The main difference is that robots do not meet randomly, but rather actively seek to meet each other to share and combine their maps. Moreover, relative localization is determined exchanging sensor data as soon as two robots can communicate with each other, thus relieving the line of sight requirement. Various other approaches have been proposed that merge maps after having preliminary localized one robot within another robot's map [5, 19, 21].

The map merging problem has a radically different twist when the maps to be merged are topological, rather than based on occupancy grids. A topological map models an environment as a graph, with vertices representing *places* and edges representing *paths* between places [13]. Huang and Beevers [11] recently proposed a method to merge topological maps based on concepts drawn from the maximal subgraph graph problem, a combinatorial problem whose NP-hardness is well known [8]. Given two topological maps, i.e. two graphs, one or more common subgraphs are identified. Each pair of common subgraphs serves as an hypothesis for merging. Hypothesis are later on rejected or accepted based on geometrical aspects endowed with the topological map.

The problem of merging two maps represented as occupancy grids is similar to the *image registration* problem commonly studied in computer vision. Maps can be in fact viewed as pictures, and map merging can be seen as a special case of image registration. Numerous solutions have been proposed to solve this problem. The method we propose in this paper is in close proximity to those based on feature matching. It is however important to note that when occupancy grids are interpreted as images, they offer far less distinctive features than digital images. The reader is referred to [22] for a recent survey on the topic.

3 Notation and problem definition

We assume that a grid map M is a matrix with r rows and c columns. Each cell $M(i, j)$ may contain three different values, indicating whether the cell is *free*, *occupied*, or if its status is *unknown*. We assume that these values are encoded as three different integer values. Maps produced by contemporary SLAM algorithms usually encode occupancy beliefs for each cell, and can therefore be easily converted into the representation we require. Each cell in M is also associated with a spatial location that can be assumed to be its center. The spatial location of cell $M(i, j)$ is indicated as $(x^{i,j}, y^{i,j})$. Given two maps, M_1 and M_2, the goal of map merging is to find a rigid transformation T so that the two maps can be overlapped. The transformation T is the combination of a rotation ψ, followed by a translation along the x and y axis of magnitude Δx and Δy, respectively. T can be conveniently represented as a 3×3 matrix, so that homogeneous coordinates can be easily transformed [4].

$$T(\Delta_x, \Delta_y, \psi) = \begin{bmatrix} \cos\psi & -\sin\psi & \Delta_x \\ \sin\psi & \cos\psi & \Delta_y \\ 0 & 0 & 1 \end{bmatrix} \tag{1}$$

When transformation T is applied to map M, we obtain a new map M' indicated as $M' = TM$.

Given two maps M_1 and M_2 there can be multiple transformations overlapping them. Therefore it is necessary to establish a metric to decide which one is better, and to reject possible false positives. In order to rank different transformations we use the the *acceptance index* we formerly introduced [1, 2].

Definition 1 *Let M_1 and M_2 be two maps with r rows and c columns. The* agreement *between M_1 and M_2 (indicated as $agr(M_1, M_2)$) is the number of cells in M_1 and M_2 that are both free or both occupied. The* disagreement *between M_1 and M_2 (indicated as $dis(M_1, M_2)$) is the number of*

cells such that M_1 is free and M_2 is occupied or vice-versa. The acceptance index *between them is defined as*

$$\omega(M_1, M_2) = \begin{cases} 0 & \text{if } agr(M_1, M_2) = 0 \\ \frac{agr(M_1,M_2)}{agr(M_1,M_2)+dis(M_1,M_2)} & \text{if } agr(M_1, M_2) \neq 0 \end{cases} \quad (2)$$

If T_1 and T_2 are two possible transformations to overlap M_1 and M_2, we prefer T_1 if $\omega(M_1, T_1M_2) > \omega(M_1, T_2M_2)$. One instance of the map merging problem therefore consists of two maps, M_1 and M_2, and the goal is to determine a transformation T that maximizes $\omega(M_1, TM_2)$. The reader should note that for the definition of the acceptance index only free or occupied cells are considered, while unknown cells are ignored. This is necessary in order correctly deal with maps including a significant number of unknown cells. According to its definition, ω assumes values between 0 and 1. The two extremes are reached when the maps do not agree in a single grid cell (i.e. $agr(M_1, M_2)$, and then $\omega = 0$), or when the maps are the same (therefore $dis(M_1, M_2) = 0$ and $\omega = 1$). For intermediate situations, values between these two boundaries are obtained. As experimentally studied in [1], values of ω below 0.9 indicate that the two maps do not overlap well, or enough, and the candidate transformation should be discarded.

In the following section we will use some concepts coming from the signal processing domain, like spectra and correlations. In particular we will deal only with discrete time signals. We say that s is a *signal* with sampling period t ($t > 0$) when s is a function from the integer multiples of t to the real numbers[1], i.e. $s : \mathbb{Z}(t) \to \mathbb{R}$ where

$$\mathbb{Z}(t) = \{\ldots, -2t, -t, 0, t, 2t, \ldots\}.$$

Since signals will always be defined over multiple integers of the sampling period, we will write $s(k)$ with $k \in \mathbb{Z}$ to indicate $s(kt)$.

4 Map merging based on spectra

Differently from our previous work, in this approach the overall transformation T is computed in two separate steps. First the rotation ψ is determined, and then the translations Δ_x and Δ_y are deduced. Also, unlike other methods previously discussed, when two robots need to merge their partial maps, we do not entail that they need to localize each other in their respective maps. Finally, we outline that even if our approach inherently deals with just two maps, multiple maps can be merged together in successive merging rounds.

4.1 Orientation estimation based on the Hough spectrum

This step of the algorithm builds upon a recent work by Censi et al. [3], where the concept of Hough Spectrum was introduced for the problem of scan matching. The ideas introduced therein are hereby briefly summarized. Given a binary image, the Hough transform is a well established methodology to detect lines and other geometric curves that can be parameterized with few values (see [6] for a concise introduction). Although Hough transform could be used to detect more complex shapes, like circles and ellipses, we here focus on the detection of lines. This choice is driven by the consideration that while mapping building interiors, numerous linear features arise, due to the walls. Besides their traditional Cartesian coordinates, lines can be represented and

[1] We use the term *sampling period* to be consistent with standard signal processing terminology (see for example [16]). However, none of the discrete signals we will introduce in this manuscript are obtained by sampling a continuous signal.

detected by choosing the polar representation, i.e. $x\cos\theta + y\sin\theta = \rho$. In this representation ρ is the distance of the line from the origin, and θ is the angle between the x axis and the normal from the line to the origin (see figure 1).

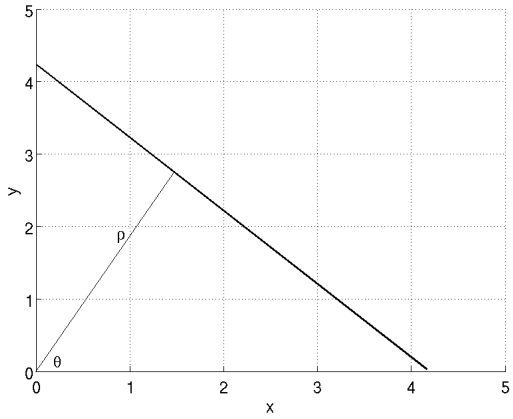

Figure 1: Hough transform uses a polar parameterization to represent lines. The values of ρ and θ uniquely identify the depicted line.

In order to speedup the computation, line detection is usually performed using the Discretized Hough transform (DHT). DHT discretizes the domain for ρ and θ, so that the DHT can therefore be represented by a matrix with ρ_S rows and θ_S columns. In addition it is also necessary to set a bound for ρ, while θ is naturally bounded to the $[0, 2\pi)$ range[2]. The DHT can be applied to detect lines in an occupancy grid map M, by converting it into a binary image. The conversion can be performed setting all occupied cells to black, and all other cells to white, for example. If M is a grid map, we indicate its DHT with $HT_{\mathcal{M}}$. Accordingly to our previous considerations, $HT_{\mathcal{M}}$ is a matrix with ρ_S rows and θ_S columns. Given $HT_{\mathcal{M}}$ we define its associated Hough Spectrum as the following signal with sampling period $\vartheta = 2\pi/\theta_S$:

$$HS_{\mathcal{M}}(k) = \sum_{i=1}^{\rho_S} HT_{\mathcal{M}}(i,k)^2 \quad 1 \leq k \leq \theta_S \tag{3}$$

The signal is extend periodically for values of k outside the range $1 \dots \theta_S$. From an intuitive point of view, $HS_{\mathcal{M}}$ indicates which directions are more frequent among lines detected in M. Figure 2 illustrates a map and its corresponding Hough spectrum.
Hough spectra are unidimensional signals, therefore cross correlation between two such signals can be used to determine similarities. Correlation, in particular outlines translations that will overlap two signals. Since Hough spectra are defined over orientations, given two spectra their cross correlation has to be computed taking the 2π periodicity into account. In other words it is necessary to compute the so called *circular cross correlation*. Formally, if $HS_{\mathcal{M}_1}$ and and $HS_{\mathcal{M}_2}$ are two Hough spectra with the same sampling period ϑ, their circular cross correlation $CC_{\mathcal{M}_1\mathcal{M}_2}$ is a signal with the same sampling period defined as follows:

$$CC_{\mathcal{M}_1\mathcal{M}_2}(k) = \sum_{i=1}^{\theta_S} HT_{\mathcal{M}_1}(i) HT_{\mathcal{M}_2}(i+k) \quad 1 \leq k \leq \theta_S. \tag{4}$$

[2]In certain cases one can allow negative values for ρ and then limit θ to the range $[0, \pi)$. This option will not be used here.

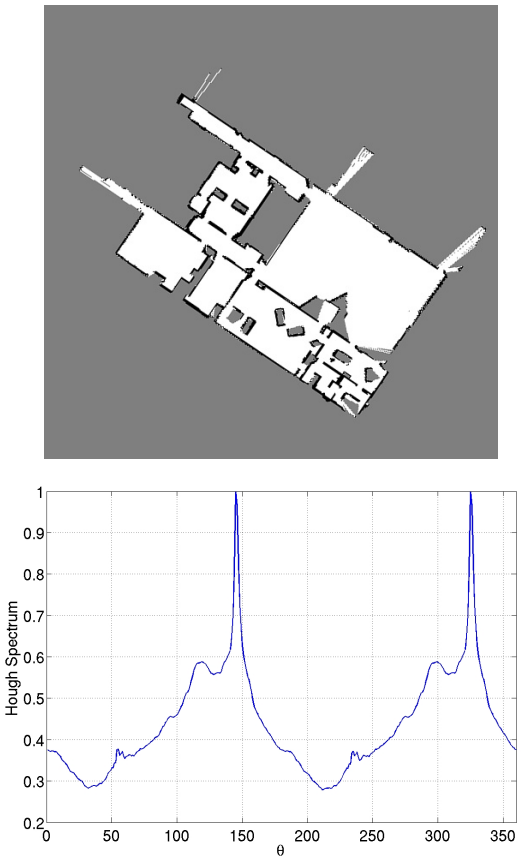

Figure 2: On the top an occupancy grid map M. White cells are free, black cells are occupied and grey cells are unknown. On the bottom, the corresponding Hough spectrum $HS_{\mathcal{M}}$ normalized to the range 0-1. In the depicted case $\theta_S = 360$.

As for Hough spectra, $CC_{\mathcal{M}_1\mathcal{M}_2}$ is extended periodically for k outside the $1 \dots \theta_S$ range. Given two maps M_1 and M_2, the cross correlation between their Hough spectra gives useful indications about how $HS_{\mathcal{M}_2}$ should be translated in order to overlap it to $HS_{\mathcal{M}_1}$. Translating the Hough spectrum corresponds to rotating the associated map. Therefore local maxima in the spectra cross correlation reveal how M_2 should be rotated in order to align it with M_1. Figure 3 shows two Hough spectra and their circular cross correlation. It can be observed that the circular cross correlation displays multiple local maxima. Each of them is associated with a possible rotation ψ_i. The algorithm we propose therefore extracts more than just the global maxima. In fact it extracts a set of n local maxima (n being a specified parameter), and returns n transformations. In this way, if more than one rotation appears promising, they can all be tracked.

4.2 Fast displacement computation

Given a candidate rotation ψ_i, the corresponding translations Δ_x^i, Δ_y^i can be in principle easily determined. Let M_3 be the map obtained rotating M_2 of ψ_i, i.e.

$$M_3 = T(0, 0, \psi_i) M_2.$$

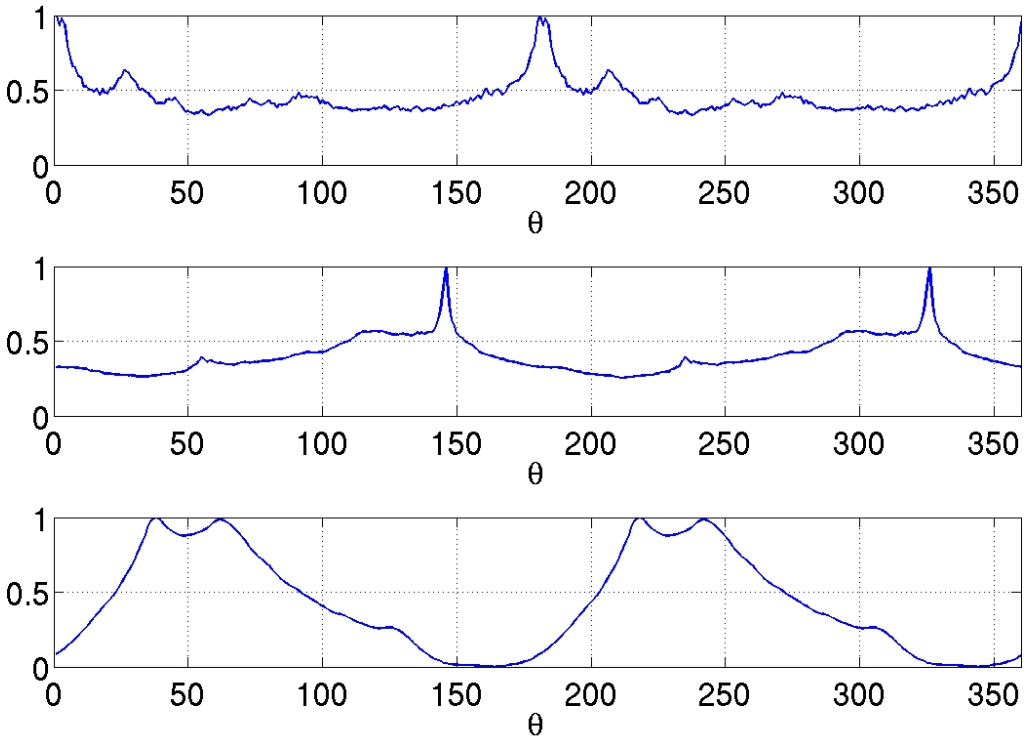

Figure 3: The two top panels show two Hough spectra, and the third one displays their circular cross correlation. Local maxima in the circular cross correlation outline how the second image should be rotated in order to align it with the first one.

Translations needed to overlap M_3 to M_1 can be obtained computing the bidimensional correlation between M_1 and M_3. This approach however is computationally demanding, and can be avoided. Rather than computing the bidimensional correlation, we extract two more *spectral* structures from the binary images M_1 and M_3. The *X-spectrum* of a binary image M is a signal with sampling period 1 defined as follows:

$$SX_{\mathcal{M}}(j) = \begin{cases} \sum_{i=1}^{r} M(i,j) & 1 \leq j \leq c \\ 0 & \text{otherwise} \end{cases} \tag{5}$$

Similarly, the *Y-spectrum* of image M is defined as the following signal:

$$SY_{\mathcal{M}}(i) = \begin{cases} \sum_{j=1}^{c} M(i,j) & 1 \leq i \leq r \\ 0 & \text{otherwise} \end{cases} \tag{6}$$

These two signals are basically the *projections* along the x and y axis of the two images. Figure 4 shows a map together with its *X-spectrum* and its *Y-spectrum*. Given $SX_{\mathcal{M}_1}$ and $SX_{\mathcal{M}_3}$, Δ_x^i can be easily inferred by looking at the global maximum of the cross correlation between them, defined as follows

$$CCX_{\mathcal{M}_1\mathcal{M}_3}(\tau) = \sum_{k=-\infty}^{+\infty} SX_{\mathcal{M}_1}(k+\tau) SX_{\mathcal{M}_3}(k) \tag{7}$$

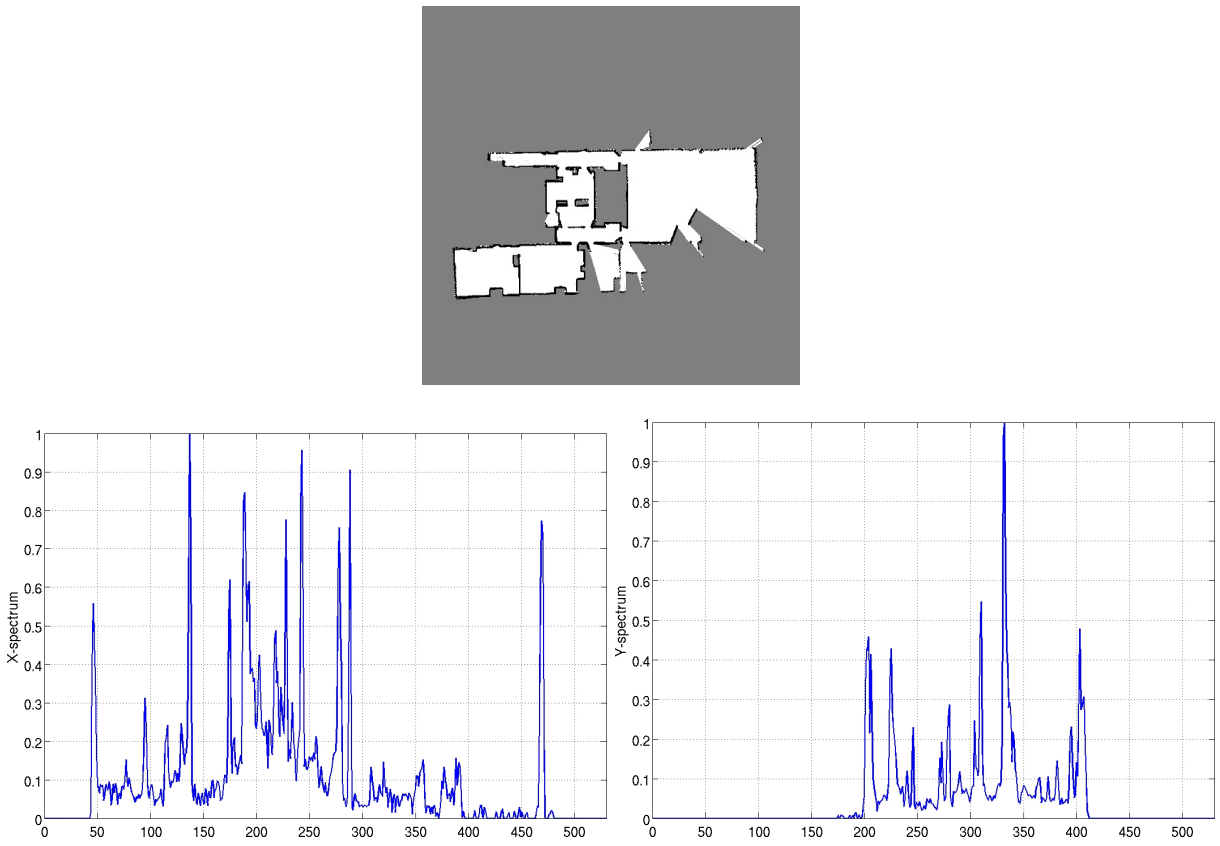

Figure 4: On top an occupancy grid map. On the bottom, its *X-spectrum* on the left and its *Y-spectrum* on the right. Both spectra have been normalized to the range 0-1.

Similarly to the case of correlation between Hough spectra, multiple local maxima may emerge when computing the cross correlation between X-spectra. Each of the maxima is associated with a candidate translation to align the two maps and could be individually tracked. An analogue relationship can be written to derive Δ_y^i. Computing two cross correlations between couples of unidimensional signals is much faster than computing a single correlation between two bidimensional signals. Therefore the above technique yields a great speedup in the computation of the translations. The approach however may be brittle if the map does not provide distinctive projections along the x and y axis, i.e. if the spectra are mostly flat. To overcome this problem, before starting any computation one of the two maps, say M_1 is rotated in order to be *aligned* with the x and y axis. This step is easy to compute as the required rotation is revealed by its Hough spectrum. We have experimentally verified that this simple preprocessing step ensures translations can be reliably computed. For example, the map shown in figure 4 is the result of such preprocessing step. In fact the algorithm started with the top left map displayed in figure 5, and rotated it by the appropriate angle, as computed from its Hough spectrum. As the focus of this paper is on maps of building interiors, it is reasonable to assume that maps will exhibit numerous linear features (i.e. walls), so that such alignment exists. In difficult cases when maps are built for environments that cannot be aligned with the axis, one can revert to the bidimensional correlation, at the price of increased computation time. Section 5 will provide additional details on this aspect.

4.3 Algorithmic details

Algorithm 1 illustrates how the concepts illustrated above can be put together to derive a set of candidate transformations. The algorithm requires two occupancy grid maps, M_1 and M_2 and a number of hypothesis, n. We assume that map M_1 has been preliminary aligned with the axis, as described in the former paragraph.

Algorithm 1 Details

ComputeHypothesis(M_1, M_2, n)
$HS_{\mathcal{M}_1} \leftarrow HoughSpectrum(M_1)$
$HS_{\mathcal{M}_2} \leftarrow HoughSpectrum(M_2)$
$CC_{\mathcal{M}_1\mathcal{M}_2} \leftarrow CircularCrossCorrelation(HS_{\mathcal{M}_1}, HS_{\mathcal{M}_2})$
$\psi_1 \dots \psi_n \leftarrow LocalMaxima(CC_{\mathcal{M}_1\mathcal{M}_2}, n)$
$SX_{\mathcal{M}_1} \leftarrow XSpectrum(M_1)$
$SY_{\mathcal{M}_1} \leftarrow YSpectrum(M_1)$
for $i \leftarrow 1$ **to** n **do**
  $M_3 \leftarrow T(0, 0, \psi_i)M_2$
  $SX_{\mathcal{M}_3} \leftarrow XSpectrum(M_3)$
  $SY_{\mathcal{M}_3} \leftarrow YSpectrum(M_3)$
  $\Delta_x^i \leftarrow \arg\max_\tau CCX_{\mathcal{M}_1\mathcal{M}_3}(\tau)$
  $\Delta_y^i \leftarrow \arg\max_\tau CCY_{\mathcal{M}_1\mathcal{M}_3}(\tau)$
  $T_i \leftarrow \Delta_x^i, \Delta_y^i, \psi_i$
  $\omega_i \leftarrow (M_1, TM_2)$
return $T_1 \dots T_n$, $\omega_1, \dots \omega_n$

The algorithm first computes the circular cross spectra between M_1 and M_2 and then extracts the n local maxima associated with the highest values. Each of them corresponds to a candidate rotation ψ_i. For each candidate rotation the algorithm computes the corresponding translations Δ_x^i and Δ_y^i. Finally, the algorithm computes the ω value associated with each candidate transformation returned by the procedure. In this way it is possible to track multiple hypothesis and to rank them. Depending on whether one is more interested in speed or accuracy, the key parameter to play with is the number of hypothesis n to be created and evaluated. We have implemented two different versions of the algorithm. The first one implements the procedure just described, and will be referred to as *basic version* while describing the experimental results. The second one instead creates three hypothesis for each local maxima detected in the cross correlation between the Hough spectra. To be specific, if ψ_i is a local maxima, two additional hypothesis for rotation are created, namely $\psi_i + \varepsilon$ and $\psi_i - \varepsilon$. This variant of the algorithm is indicated as *robust version*, and turns out to be more appropriate to use when it is necessary to merge maps whose occupied cells do not mainly lie along straight lines. It is clear that when using this improved version there is a tradeoff between accuracy and speed, due to the necessity to perform additional computations for each supplemental hypothesis being evaluated.

5 Experimental results

Two different sets of experiments have been performed to measure the effectiveness of the proposed technique[3]. The first aims to determine the robustness of the algorithm, i.e. the ability to consis-

[3]The code implementing the algorithm presented herein and the datasets are available for download on `http://robotics.ucmerced.edu/Software`

tently recover a set of transformations close to the optimal one. These experiments were performed using public available data sets and maps. In the second set of experiments we built maps with two robots exploring different parts of the same environment. Some experiments were conducted indoor, i.e. in the ideal situation for the depicted technique. However, some experiments were also performed outdoor, in order to verify how the algorithm performs in situations that deviate from its ideal operating conditions. All the computations took place on a computer equipped with an Intel dual core processor running at 2.14 GHz with 2 Gb of RAM. The machine runs Linux and the code is written in C++.

5.1 Robustness

If $M_1 = M_2$, then by definition $\omega(M_1, M_2) = 1$. Starting from this fact we can get a first indication of the algorithm performance by applying a random transformation to a map, and verifying whether the algorithm is capable to recover the inverse transformation. More precisely, let M be grid map, and let $M' = TM$, with T being a random transformation. Let T_B be the best transformation produced by *ComputeHypothesis*. The value $\omega(M, T_B M')$ should be 1 if the algorithm is capable to recover the inverse transformation. In order to perform this test we have used the four maps displayed in figure 5. These maps have been built using one of the datasets available on the Radish repository [18]. The SLAM algorithm used to produce the maps is GMapping by Grisetti et al. [9], whose implementation is available on the OpenSlam web-site [15]. These maps have been already used as benchmarks in some previous publications about map merging, and therefore constitute a good reference point. We performed 1000 trials. At each iteration one of the four maps was randomly chosen, and then a random transformation T was applied. Results are illustrated in figure 6. The figure plots $\omega(M, T_B M')$ as a function of the random rotation[4]. When the algorithm perfectly recovers the inverse transformation this value is 1. The average value of $\omega(M, T_B M')$ is 0.9930, and its standard deviation is 0.0048. These results were produced by the basic version of the algorithm, setting the number of hypothesis to be tracked to $n = 4$. The average time to compute a set of 4 hypothesis was 558 ms. Each of the maps used for this experiment had 530 rows and 530 columns.

The maps used in the first experiment nicely fit the optimal hypothesis for the algorithm we propose, i.e. they exhibit many straight lines. In order to test how the algorithm performs when dealing with maps that feature also curved walls, we have used another dataset available on Radish that produces the map with curved walls displayed in figure 7. This is significantly bigger than those used in the previous test: it has 1000 rows and 1000 columns. Figure 8 compares the results of the value obtained by two versions of the merging algorithm. Red dots illustrate the basic version, while blue dots show the performance of the robust version. It is clear that for this specific map the use of the robust version pays off. For what concerns speed, the average time to compute 12 hypothesis using the robust version is about 5 seconds. The average value of ω was 0.9904 with a standard deviation of 0.106.

Finally, in figure 9 we show a comparison between the ω value associated with the 12 hypothesis produced while solving one instance of the map merging problem with the later dataset using the robust version of the algorithm. In this case it is possible to see how the ω value discriminates good transformations from bad ones. The first transformation has a ω value of 0.9821. All the other transformations have significantly lower values. In this case it is therefore possible to rule out ambiguities, and commit to the transformation with the highest ω value.

[4]The random transformation T includes both a random rotation and random translations. We here display the trend of ω as a function of the rotation, as this component of the transformation turns out to be the hardest to recover.

5.1.1 Considerations on the computational speed

The two set of experiments illustrated above give an idea about the speed of the algorithm being proposed. In the first set of experiments the maps being merged have average size and feature plenty of linear pieces. In such case the basic version of the algorithm can be applied and the speed is remarkable. In the second set of experiments 12 hypothesis need to be tracked, and the maps are significantly bigger. A corresponding slowdown is observed. The size of the maps being merged plays of course a crucial role. In order to estimate T, the size of the map is not the only key parameter. The number of occupied cells, in fact, plays also a key role, and these two numbers may be significantly different. This is a consequence of the fact that the grid map is preliminary cast into a black and white image, and only occupied cells are considered during spectral processing. However, when T has been determined and ω is evaluated, the size of the map matters. In fact, to evaluate ω it is necessary to construct $M' = TM$. Such operation considers all the cells in the map (free, occupied or unknown). In the results presented here we have used the transformation routines provided with the OpenCv library [14], so the computational time we report is negatively affected by the necessity to transform the occupancy maps into a format that can be processed by OpenCv. If for a certain application speed is of utmost importance, one may consider implementing these operations from scratch in order to avoid this overhead. Alternatively one can directly represent maps in a format that can be used by OpenCv.

5.2 Performance on different maps

In the second set of experiments we have collected data with two pioneer P3AT mobile robots exploring different parts of the same building or an outside area. Both robots are equipped with a SICK range finder (see figure 10), sonars and odometry.

Maps were built using the Gmapping algorithm described above. Since the sake of this experiment is to verify how the algorithm works when merging challenging maps, robots were remotely controlled in order to drive the exploration towards difficult spots in the environment. Whether robots explore the environment autonomously, or are remotely operated by a human makes no difference for the algorithm we propose. In the first scenario robots explore the inside part of the science and engineering building at UC Merced. The building features long corridors with offices and lab spaces on both sides. Figure 11 shows the result of one the experiments performed. The two robots started at the lower right corner of the map (close to the spot indicated by the letter S in the lower panel). The first robot went up, then came back, turned right, explored the long corridor down to the hall displayed on the left of the map, and went finally back to the initial point. The map produced by the first robot is displayed in the top panel. The second robot first moved down the corridor on the right, then moved back and entered the room appearing on the top of the map. The corresponding map is displayed in the middle panel. The merged maps are displayed in the bottom panel. Similar results were produced in the other tests executed inside the building. It can be observed that the merged map perfectly integrates the two partial ones produced by the individual robots.

Finally we executed some tests outside, in the campus green (see the left picture in figure 10). The outdoor environment obviously offers less linear features and poses significant challenges to the mapping algorithm we used. In addition to lack of walls, an additional problem arises since the robots move on an uneven terrain and many spurious readings returned by the SICK sensor are not due to real obstacles, but rather to the robot moving down a slope. Figure 12 shows the two maps produced by the individual robots, and the merged map. As long as the maps being merged feature at least one wall of significant length the merging algorithm is capable to determine the

needed rotation. Other features present in the map (like pillars, trees and benches) usually provide enough data to compute the needed translation. In cases like these, however, it may make sense to track not only multiple rotations but also multiple translations. This extension can be easily added by extracting multiple local maxima from the cross correlations of the X and Y spectra.

6 Conclusions, limitations, and future work

We presented a novel algorithm to merge occupancy grid maps produced by two or more robots operating in the same environment. The presented algorithm merges two maps, but can deal with more than two by repeatedly merging couple of maps. The algorithm produces a set of candidate transformations, i.e. rotations and translations, that can be used to overlap two maps. Each candidate transformation is weighted using the acceptance index indicator we introduced in a former stage of this research. The weight of each transformation indicates its quality, and the relative weights allow to distinguish cases where more than one transformation appear appropriate. Our technique has been tested to verify its performance on various maps. We have experimentally assessed that when merging maps predominantly composed of straight lines the algorithm quickly and reliably finds a suitable transformation by just tracking few hypothesis. Moreover, we have tested the performance on maps built by robots exploring outside areas. In such case the algorithm still manages to find meaningful transformations leading to the correct merging, but it needs to track more hypothesis, due to the augmented uncertainty.
Three limitations affect the algorithm we propose. First, it is necessary to assume that the two maps being merged have been built using the same scale. The algorithm is not capable to determine whether one of the two needs to be magnified in order to be matched with the other, and this possible extension appears not easy to include. Second, in order for the merging to be successful, it is necessary that the two maps being merged exhibit a certain degree overlapping. If this is not the case the algorithm is unlikely to find the appropriate transformation, although the acceptance index will indicate that the produced merging should be discarded. It should however be noted that most algorithms for map merging proposed so far share these limitations as well. Finally, when robots map large environments, produced maps may end up being affected by a remarkable degree of distortion, so that a rigid transformation would not produce a satisfactory alignment. In that case if would be necessary to look for a transformation with local characteristics, i.e. a transformation capable of modifying the two maps based on the local distortion. This problem has also been scarcely addressed in the literature and appears to be beyond the range of problems solvable by the proposed technique. However, throughout the tests we performed such problem never appeared, thanks to the robustness of the mapping algorithm used.
The proposed technique could be extended in various directions. Firstly, when robots meet often, and therefore merge their respective maps often, it could make sense to solve the merging task not from scratch, but rather taking into account the prior partial results. In addition, considering that Hough transform can be extended also to detect three dimensional shapes, we will investigate whether the algorithm we propose could be extended to merge three dimensional maps produced by tilting range sensors or three dimensional proximity finders.

References

[1] A. Birk and S. Carpin. Merging occupancy grids from multiple robots. *Proceedings of the IEEE*, 94(7):1384–1397, 2006.

[2] S. Carpin, A. Birk, and V. Jucikas. On map merging. *Robotics and autonomous systems*, 53(1):1–14, 2005.

[3] A. Censi, L. Iocchi, and G. Grisetti. Scan matching in the Hough domain. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 2739–2744, 2005.

[4] J. J. Craig. *Introduction to robotics – Mechanics and control*. Prentice Hall, 2005.

[5] G. Dedeoglu and G.S. Sukhatme. Landmark-based matching algorithm for cooperative mapping by autonomous robots. In *Distributed Autonomous Robotic Systems 4*, pages 251–260. Springer, 2000.

[6] R. Duda and P. Hart. Use of the hough transform to detect lines and curves in the pictures. *Communications of the ACM*, 15(1):11–15, 1972.

[7] D. Fox, J. Ko, K. Konolige, B. Limketai, D. Schulz, and B. Stewart. Distributed multirobot exploration and mapping. *Proceedings of the IEEE*, 94(7):1325–1339, 2006.

[8] M.R Garey and D.S. Johnson. *Computers and Intractability. A guide to the theory of NP-Completeness*. W.H. Freeman and Company, 1979.

[9] S. Grisetti, C. Stachniss, and W. Burgard. Improving grid-based slam with rao-blackwellized particle filters by adaptive proposals and selective resampling. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 2432 – 2437, 2005.

[10] A. Howard, L.E. Parker, and G.S. Sukhatme. Experiments with a large heterogeneous mobile robot team: exploration, mapping, deployment and detection. *International Journal of Robotics Research*, 25(5-6):431–447, 2006.

[11] W.H. Huang and K.R. Beevers. Topological map merging. *International Journal of Robotics Research*, 24(8):601–613, 2005.

[12] K. Konolige, D. Fox, B. Limketkai, J. Ko, and B. Steward. Map merging for distributed robot navigation. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 212–217, 2003.

[13] B. Kuipers. Modeling spatial knowledge. *Cognitive science*, 2:129–153, 1978.

[14] Open Computer Vision Library. http://sourceforge.net/projects/opencvlibrary/.

[15] Openslam. http://www.openslam.org, 2007.

[16] A.V. Oppenheim and A.S. Willsky. *Signal and Systems*. Prentice Hall, 1997.

[17] L.E. Parker. Current state of the art in distributed autonomous mobile robots. In L.E. Parker, G. Bekey, and J.Barhen, editors, *Distributed Autonomous Robotic Systems 4*, pages 3–12. Springer, 2000.

[18] Radish. The robotics data set repository. http://radish.sourceforge.net, 2007.

[19] S. Thrun. A probabilistic online mapping algorithm for teams of mobile robots. *International Journal of Robotics Research*, 20(5):335–363, 2001.

[20] S. Thrun, W. Burgard, and D. Fox. *Probabilistic Robotics*. MIT Press, 2006.

[21] S.B. Williams, G. Dissanyake, and H. Durrant-Whyte. Towards multi-wehicle simultaneous localisation and mapping. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 2743–2748, 2002.

[22] B. Zitová and J. Flusser. Image registration methods: a survey. *Image and vision computing*, 21:977–1000, 2003.

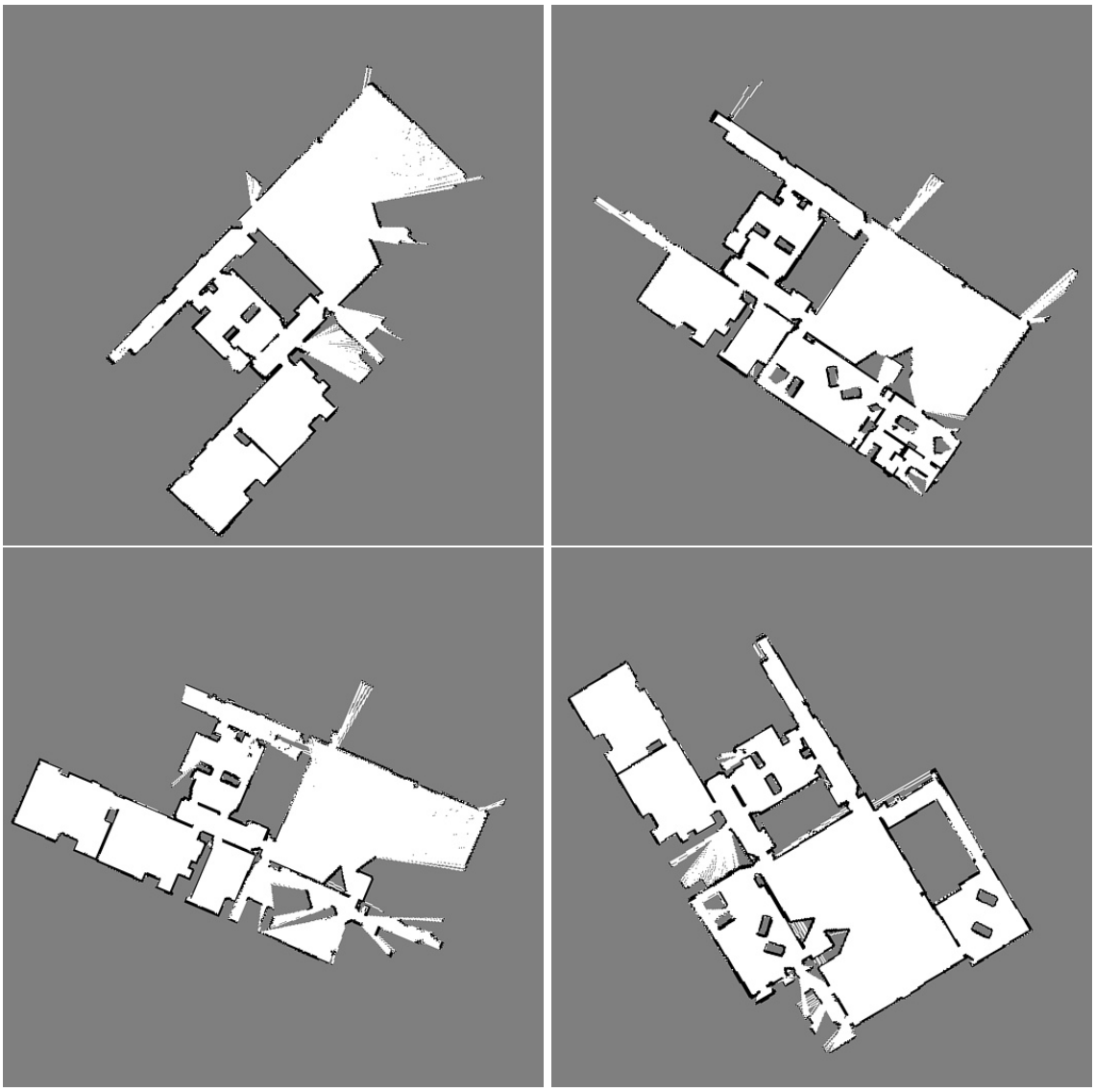

Figure 5: The four maps used to run the first robustness test. Each the map has been preliminary rotated so that its walls are not aligned with the x and y axis. White cells are free, black cells are occupied and gray cells are unknown.

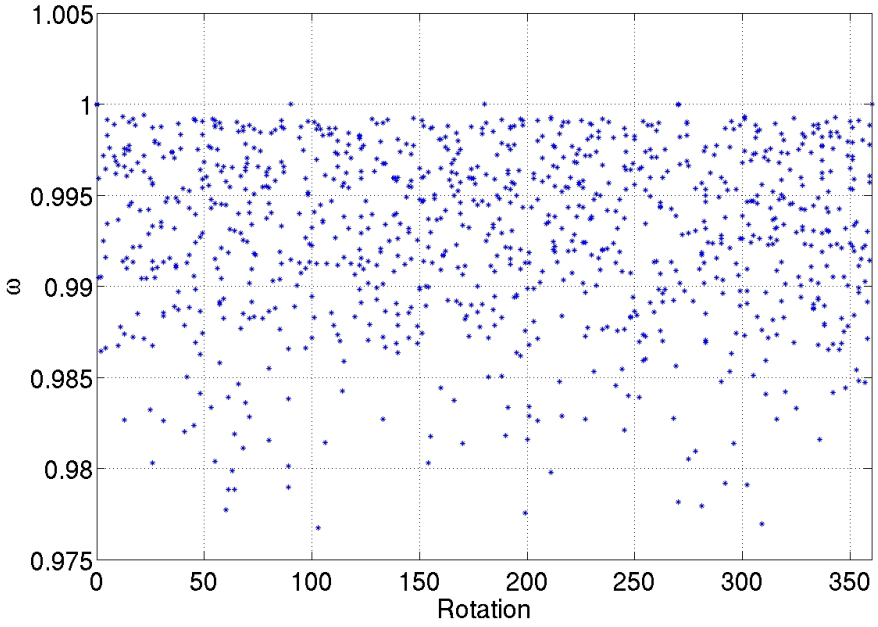

Figure 6: Trend of ω as a function of the random rotation for the first dataset.

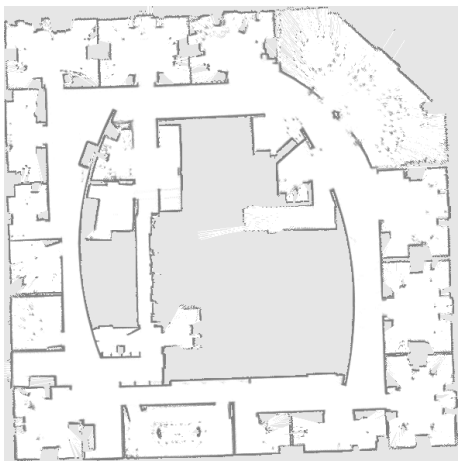

Figure 7: The map featuring curved and straight walls used for the second set of experiments.

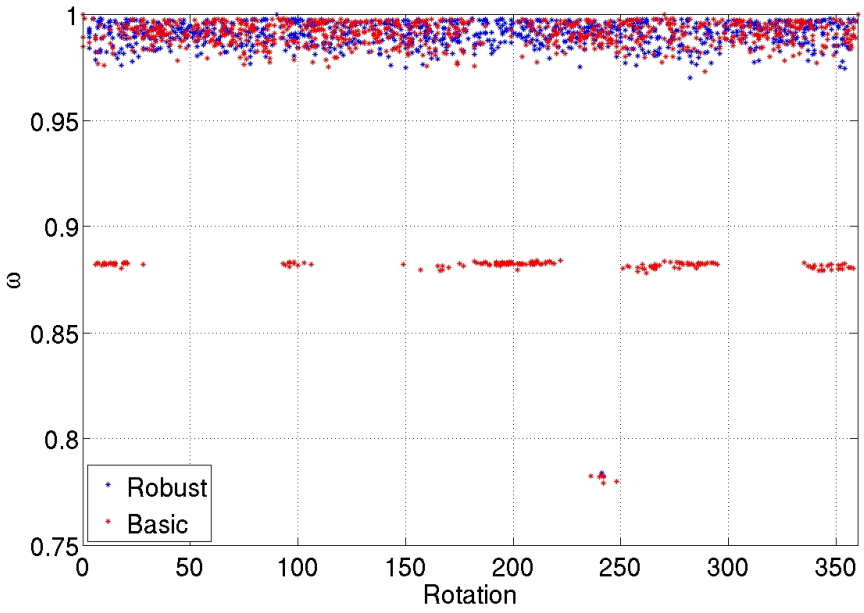

Figure 8: Trend of ω as a function of the random rotation for the first dataset.

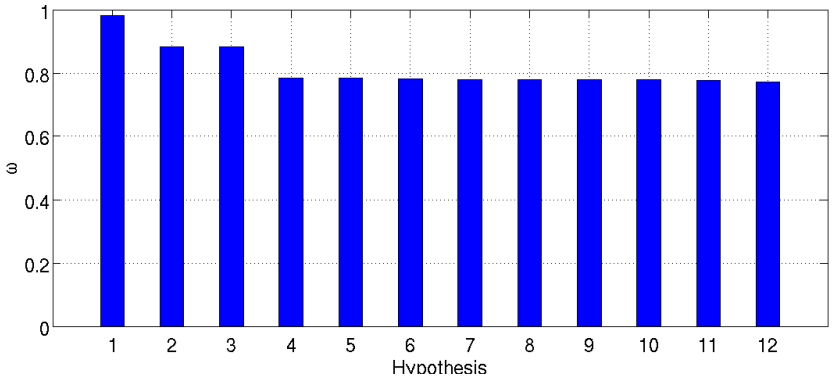

Figure 9: Comparison between the ω values associated with a set of 12 candidate transformations produced in the second set of experiments. In this case the algorithm correctly retrieved the inverse transformation. This is outlined by the transformation associated with a ω value close to 1. Hypothesis have been sorted according to their ω value for displaying purposes only.

Figure 10: On the left the two robots exploring the interior of the engineering and science building. On the right, one of the robots exploring the outside green quad at UC Merced.

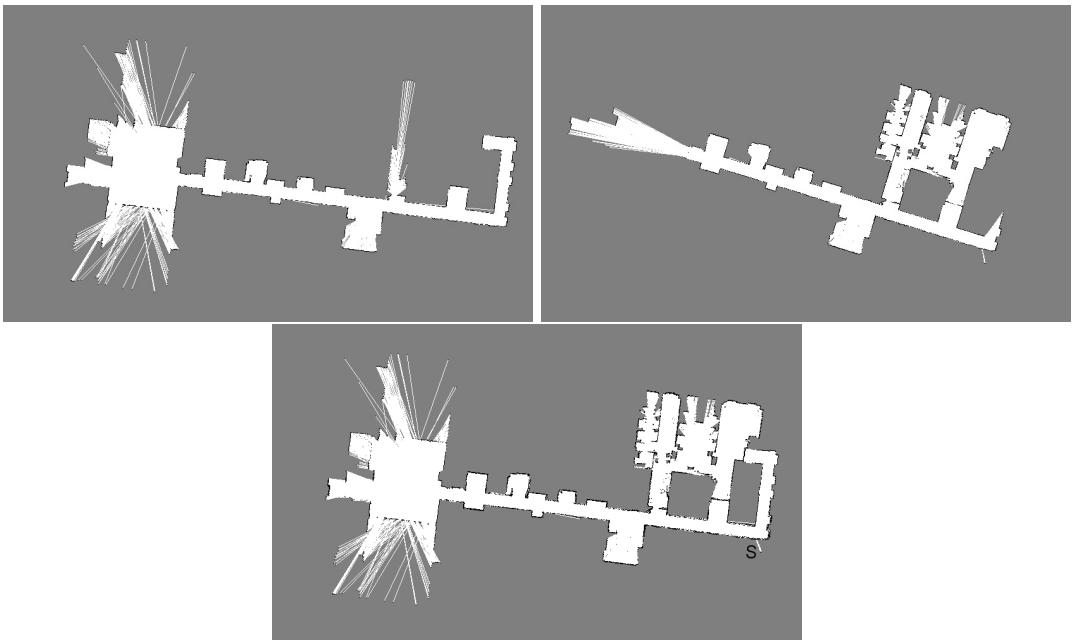

Figure 11: The top two panels show the two maps built individually by the two robots. The bottom panel illustrates the map obtained by merging the top ones.

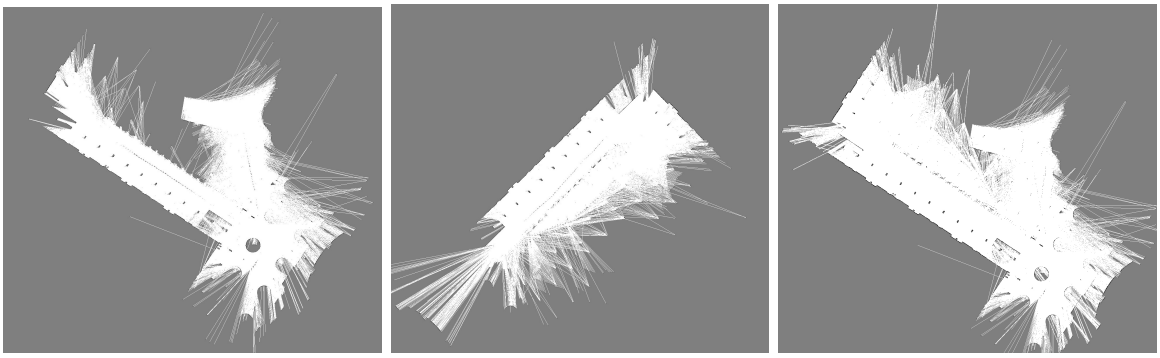

Figure 12: On top, the two maps produced by the two robots while exploring the campus green. On the bottom the merged map.