2010 IEEE International Conference on Robotics and Automation
Anchorage Convention District
May 3-8, 2010, Anchorage, Alaska, USA

AntSLAM: Global Map Optimization Using Swarm Intelligence

René Iser and Friedrich M. Wahl

Abstract— **The capability of solving the simultaneous localization and mapping (SLAM) problem is one of the fundamental tasks of mobile robots and many research has focused on this problem over the last decades. In this paper, the SLAM problem is considered as the problem of finding an optimal path through a tree resulting in minimum costs. For this purpose, we apply the *Ant Colony Optimization* meta-heuristic, which belongs to the class of ant algorithms. It has been successfully employed to solve the well known Traveling Salesman Problem with several thousands of cities. We use a simple scan matching technique for generating a rough pre-solution to the SLAM problem. The (inconsistent) map is partitioned into fragments. A new fragment is initialized as soon as the robot has moved several meters. We draw samples from Gaussian distributions representing alignments of consecutive fragments. The resulting set of samples is interpreted as a tree-like data structure with weights assigned to the edges. We use our own variant of an ant algorithm for finding the optimal path through the tree. Real-world experimental results demonstrate the characteristics of our method.**

I. Introduction

One of the fundamental tasks of mobile robots is to build maps of unknown environments. This problem is often named as Simultaneous Localization and Mapping Problem (SLAM) and has been addressed by many researchers in the past, and lots of impressive results are discussed in the open literature. There exist several philosophies of how to interpret the SLAM problem. A well-known class of SLAM approaches are probabilistic methods. One popular map estimation technique is the Rao-Blackwellized particle filter which originally has been introduced by Murphy [19]. An occupancy grid map [18] has been used to represent the map and each particle provides a map hypothesis. The problem of Rao-Blackwellized particle filters is the number of particles required to build an accurate map, resulting in both a high computational complexity and a considerable memory complexity. Grisetti *et al.* [11] significantly reduced the number of required particles. A compact representation method is the feature based technique. Montemerlo *et al.* [27] also employed particle filters for generating map hypotheses, but the map consisted of a set of features. The disadvantage of feature based representations is that they are environment specific. Wurm *et al.* [30] proposed a dual representation of the environment, which combines both techniques. Another popular estimation method is the Extended Kalman filter (EKF) [22], which combines all landmark positions and robot poses into one covariance matrix.

The authors are with the Institut für Robotik und Prozessinformatik, Technische Universität Braunschweig, 38106 Braunschweig, Germany. {r.iser,f.wahl}@tu-bs.de

The approaches described above apply probabilistic techniques for computing a map estimation. On the other hand, the SLAM problem can be interpreted as a graph optimization problem [23]. A very efficient and robust approach to address the graph-based SLAM-Problem has been published by Olson *et al.* [5]. This method applies a stochastic gradient descent to minimize the errors resulting from constraints of the nodes. So far, the work of [5] is the current state-of-the-art technique for the optimization of network constraints. Folkersson *et al.* [7] proposed a graphical SLAM method where the map is represented by a set of so called *energy nodes* resulting from odometry information and feature observation. The energy between the nodes is minimized using a relaxation technique. Grisetti *et al.* [12] adapted the graph-based SLAM approach of [5] by introducing improved parameters to the nodes resulting in a considerable higher performance of the algorithm. In the work of [10], a graph-optimization technique is proposed that allows the computation of accurate three-dimensional maps by distributing the error that results from constraints over a sequence of nodes in all six dimensions. This algorithm requires a considerable adaption compared to the two-dimensional case since rotation in three dimensions is not commutative. In [8], a one million landmark loop is closed in real-time. The map is represented as a tree, and while the robot is moving, a hierarchical tree partition algorithm is applied for optimization.

So far, we have discussed probabilistic techniques as well as graph-based methods. A third interesting class are biologically inspired SLAM methods. Recently, the RatSLAM algorithm has been introduced [16], [17] which simulates the hippocampus of rodents to compute semi-metrical maps.

In this paper, we generate a tree-like data structure, in which the path from the root to a leaf represents a metrical map estimation. An (almost) optimal path is found by employing a swarm intelligence algorithm. Hence, we combine aspects of graph-based methods and biologically inspired techniques. We use a scan matcher for computing a rough pre-solution to the SLAM-problem which is often the core of probabilistic approaches. Of course, the result will exhibit significant errors in general. The map is partitioned into several fragments. A new fragment is initialized whenever the robot has moved several meters. A globally consistent map can be obtained by manipulating the alignment of consecutive fragments. Hence, we draw samples from Gaussian distributions representing possible alignments. A set of samples is assigned to each fragment. Moreover, each fragment employs its own Gaussian distribution in order to draw samples. When closing loops, the knowledge of how the

last fragment is correctly aligned to first one is of significant importance for our approach. Otherwise, the evaluation of the quality of a given path would be impossible. Thus, we use a very efficient enhancement [29] of the well known Random Sample Consensus (RANSAC) approach [6] for computing the correct alignment. Finally, given the tree, we employ an ant algorithm for generating a path resulting in a consistent solution. To the best of our knowledge, nobody has used Ant Colony Optimization (ACO) for solving the SLAM problem before. Instead, ACO has been applied to the path planning and navigation problem of mobile robots several times [9], [15], [20]. The objective of this paper is not to challenge existing SLAM approaches. Instead, we aim to demonstrate an interesting alternative for map optimization which cannot get stuck in local minima. Moreover, only one map hypothesis is required which is very memory efficient. The remainder of this paper is organized as follows. In Sec. II, we will briefly review the general structure of ant algorithms. Our SLAM method is explained in detail in Sec. III. For this purpose, Sec. III-A elucidates how the tree mentioned above is computed in our approach. The ant algorithm employed for generating a globally consistent map is introduced in Sec. III-B. Experimental results demonstrating the characteristics of our method are given in Sec. IV. Finally, Sec. V concludes the paper.

II. Ant Colony Optimization

A very popular family of algorithms that have been successfully applied to several NP-hard combinatorial optimization problems are ant algorithms [1], [2]. The basic idea of this class of optimization techniques is inspired by the behavior of real ant colonies during foraging. While moving, ants are leaving pheromone trails permanently which are used by the colony for an indirect communication amongst the ants. The pheromone is a mean for finding shortest paths between the colony's nest and the forage. A short path contains more pheromone than a longer one because it can be passed more often in the same time. Starting from their nest, ants choose paths with a probability proportional to the strength of the pheromone. The pheromone trails are updated permanently by the ants reflecting the colony's experience [26]. Dorigo *et al.* [25] proposed the Ant Colony Optimization meta-heuristic, which provides a framework for most ant algorithms. Its structure looks as follows [26]:

Algorithm 1 Ant Coloy Optimization

```
1: while termination condition not met do
2:     for i ← 1 to ANTS do
3:         Construct Solution
4:         Apply Local Search                ▷ optional
5:     end for
6:     Update Pheromone Trails
7: end while
```

In the past, this technique has been applied to the famous Traveling Salesman Problem (TSP). The algorithm proposed by Dorigo *et al.* [4] is the first ACO algorithm employed for solving the TSP. A city tour as in Algorithm (1) is constructed by putting each ant on a randomly chosen city at the beginning. Assuming ant k, $k \in [1; M]$, to be at city i, it chooses the next city j with a probability of [26]

$$p_{ij}^k(t) = \frac{[\tau_{ij}(t)]^\alpha \cdot [\eta_{ij}]^\beta}{\sum_{l \in \mathcal{N}_i^k} [\tau_{il}(t)]^\alpha [\eta_{il}]^\beta} , \qquad (1)$$

where $\eta_{ij} = \frac{1}{d_{ij}}$ is a heuristic value. Here, d_{ij} is the distance between city i and city j. $\mathcal{N}_i^k$ is the set of cities which ant k has not yet visited and $\tau_{ij}(t)$ is the amount of pheromone of the edge connecting city i and j. The parameters α and β control the relative influence of the pheromone strength and the heuristic value. In the case α=0 it is a simple greedy algorithm, which is very likely to choose the city with the shortest distance next. Conversely, β=0 leads to a fast stagnation, which means that all ants are following the same path resulting in a highly suboptimal solution in general [4]. Hence, a good trade-off between both variants has to be found. After tour construction, the pheromone trails are updated by

$$\tau_{ij}(t+1) = (1-\rho) \cdot \tau_{ij}(t) + \sum_{k=1}^{M} \Delta\tau_{ij}^k(t) \qquad (2)$$

where $0 < \rho \leq 1$ is an evaporation factor which allows for forgetting bad decisions made previously. $\Delta\tau_{ij}^k(t)$ is the new pheromone added by ant k and is defined as

$$\Delta\tau_{ij}^k(t) = \begin{cases} \frac{1}{L^k(t)} & \text{if edge } (i,j) \text{ has been visited by ant } k \\ 0 & \text{otherwise} , \end{cases} \qquad (3)$$

where $L^k(t)$ is the length of the tour of ant k. Intensive research has shown that Ant System is able to find good solutions only for relatively small TSP instances with 75 cities maximally. Ant System applied to more instances lead to rather poor solutions. Thus, strong efforts have been made to improve the algorithm. In [3] Ant Colony System (ACS) has been proposed. In ACS, only the ant that found the globally best tour is allowed to update the pheromone trail. Furthermore, with a probability of p' ant k moves to the city, for which $[\tau_{ij}(t)]^\alpha \cdot [\eta_{ij}]^\beta$ is maximal when located at city i. With a probability of $(1-p')$ it chooses the next city according to eq. (1). Ant Colony System has proven to give far better results for larger TSP instances than Ant System. Stützle *et al.* [24] introduced the $\mathcal{MAX} - \mathcal{MIN}$ Ant System. In this modification, both the iterative best and the global best ant are allowed to update the pheromone trail. Moreover, lower and upper limits for the pheromone strength are defined restricting the pheromone strength of all edges (i,j) to $\tau_{min} \leq \tau_{ij} \leq \tau_{max}$. Experiments have shown that the upper limit is very important since it avoids an unlimited increase of the pheromone strength on good tours and hence preventing the algorithm from search stagnation much better.

III. Application of Ant Colony Optimization to SLAM

The ACS algorithm and its improvements described in Sec. II are adapted in order to compute a consistent map on the global level. For this purpose, a graph $G = (V, E)$ with vertices V and edges E is necessary, representing possible solutions to the SLAM problem. Furthermore, each edge of the graph is assigned a weight. Given an arbitrary sequence of nodes generated by the ants, the sum of the weights represents the quality of the map. The remainder of this section is organized as follows. In Sec. III-A, the generation of the graph mentioned above is explained in detail. Afterwards, Sec. III-B illustrates how to apply an ant colony for map correction including our modifications of the algorithms explained in Sec. II.

A. Computing the SLAM-Graph

We assume that the robot starts its exploration at the origin of a world frame W. It is equipped with a sensor providing range readings, e.g. a laser scanner. Since odometry data is very unreliable, we use a scan matcher in order to align consecutive sensor readings and hence correcting the robot motions on a local level. The scan matcher we apply is described in [14] and [21]. It is well known that errors accumulate over time leading to an inconsistent map even when using a very precise laser scanner. Nevertheless, these maps are consistent on a local level, i.e. small sections of only a few meters can be considered as correct with sufficient small errors. Hence, we subdivide the map computed by the scan matcher into several fragments. More precisely, when the robot has moved a few meters we initialize a new map fragment. In this way, we obtain a set of map fragments $\mathcal{F} = \{f_i \mid i \in [1, N]\}$ where N is the total number of fragments. Each fragment f_i is assigned a local reference frame F_i and an exit point E_i where the robot has left the fragment; E_i is given w.r.t to F_i. In addition, F_{i+1} is initialized w.r.t F_i resulting in a relative homogenous transformation ${}^{F_i}T_{F_{i+1}}$. Currently, we have the following relation:

$$ {}^{F_i}T_{F_{i+1}} = {}^{F_i}T_{E_i} \tag{4} $$

The reason is that the new fragment F_{i+1} is initialized immediately when E_i is reached. The relationship is illustrated in Fig.1 (a).

In order to obtain a consistent map on the global level, the origin of each fragment needs to be modified slightly. Grisetti *et al.* [11] proposed a grid-based SLAM technique based on an highly improved proposal distribution for an Rao-Blackwellized Particle Filter. Each particle represents its own map estimation. The idea is to correct the odometry information by employing scan matching techniques. Samples are generated around this corrected pose which are used to compute a Gaussian distribution. Finally, the particles are sampled according to this distribution. The idea introduced in [11] of generating the Gaussian distribution was the main inspiration of this work for calculating the graph structure described above. Given the robot pose x_t at time t, we correct

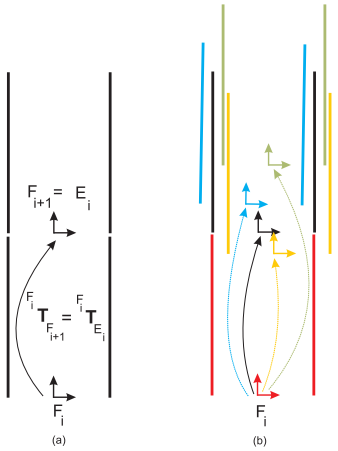

Fig. 1. (a) Given a map fragment F_i, the next fragment F_{i+1} is computed w.r.t F_i. (b) A set of samples representing possible alignments of F_i and F_{i+1} is computed from a Gaussian distribution. The samples are generated around E_i which is marked by the black coordinate system.

the odometry data by applying a scan matching operation resulting in a better estimation $\hat{x}_t$ of the current pose. Let the robot be located at fragment i, then $\hat{x}_t = E_i$ holds. Of course, the latter equation is only true if the robot is leaving fragment i. Then a set of samples $\mathcal{X} = \{x \mid |x - \hat{x}_t| \leq \Delta\}$ is computed from the scan matching result which allows for the calculation of the Gaussian distribution $\mathcal{N}(\mu_{i+1}, \Sigma_{i+1})$ [11]:

$$ \mu_{i+1} = \eta \sum_{k=1}^{|\mathcal{X}|} x_k \cdot p\left(z_t \mid m_{t-1}, x_k\right) \tag{5} $$

$$ \Sigma_{i+1} = \eta \sum_{k=1}^{|\mathcal{X}|} (x_k - \mu_{i+1})(x_k - \mu_{i+1})^T \cdot p\left(z_t \mid m_{t-1}, x_k\right) \tag{6} $$

In both equations, m_{t-1} is the scan matcher map before integrating the current sensor reading z_t by the scan matcher. η is a normalizer. Now, a second set of samples

$$ \mathcal{S}_{i+1} = \{s_j \mid s_j \sim \mathcal{N}(\mu_{i+1}, \Sigma_{i+1}), j \in [1, M]\} \tag{7} $$

is computed, where each sample s_j represents a possible alignment of fragment F_{i+1} to F_i. Hence, s_j constitutes an edge in the graph G. The latter sampling step is depicted in Fig.1 (b). Unfortunately, it is not yet possible to optimize the map because loop closing on a topological level has not been considered so far. Thus, the connection of the last fragment with the first one is still missing. In general, one requirement of our approach is to detect places visited by the robot previously. One example of this problem is depicted in Fig.2 (a). The map consists of six fragments. Starting from the initial frame $F_1 = W$, one can choose a sample from each fragment resulting in a chain of homogenous transformations from F_1 to E_6. The last transformation ${}^{E_6}T_{F_1}$ still cannot be weighted, because we do not know how the optimal transformation ${}^{E_6}T_{F_1,opt}$ looks like. In [14], a method is

proposed how to recognize places visited previously. The general idea is to compute the mean and the covariance of current robot pose after each motion. As soon as the ellipse determined by the covariance matrix intersects an already registered fragment, the Random Sample Matching (RANSAM) operation [29] is triggered; RANSAM is a very time and memory efficient enhancement of the well known Random Sample Consensus (RANSAC) algorithm [6]. It computes a hypothesis ${}^{F_1}H_{E_6}$ representing a correct alignment of the first and the last fragment (cf. Fig. 2 (b)). For details concerning the RANSAM method please consult [29] and [13]. In general, given N map fragments, the transformation ${}^{W}T_{E_N}$ is computed as

$$ {}^{W}T_{E_N} = {}^{W}T_{S_{2,j(2)}} \cdot \left(\prod_{i=3}^{N} {}^{S_{i-1,j(i-1)}}T_{S_{i,j(i)}} \right) \cdot {}^{S_{N,j(N)}}T_{E_N}. \tag{8} $$

where $S_{i,j(i)}$ stands for sample $j(i)$ from fragment i. Hence, a necessary constraint for an optimal map is a sequence of samples $\mathfrak{S} = \{S_{i,j(i)} | i \in [2,N], j \in [1,M]\}$ resulting in

$$ {}^{W}T_{E_N} = {}^{F_1}H_{E_N}. \tag{9} $$

This relation is illustrated more detailed in Fig. 2 (a) and (b). In order to weight the transformation ${}^{E_N}T_{F_1}$, we also compute a Gaussian distribution $\mathcal{N}(\mu_H, \Sigma_H)$ as explained above. The samples necessary for the generation of the distribution are gathered around H.

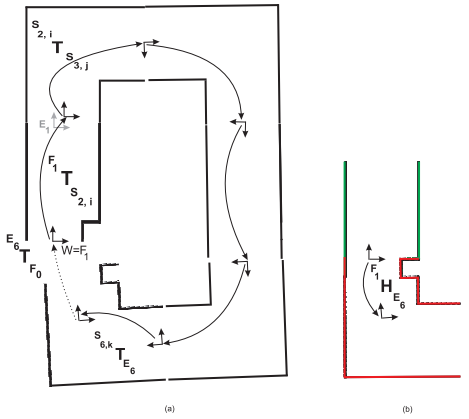

Fig. 2. (a) The world frame W and the reference frame F_i of the first fragment i coincide. Afterwards, one sample is chosen randomly per fragment resulting in an (inconsistent) map estimation. In this example, the transformation from F_1 to F_2 is determined by sample $s_{2,i}$. Moreover, $s_{3,j}$ and $s_{6,k}$ determine the transformation from F_2 to F_3 and from F_5 to F_6. In order to measure the quality of the current solution, the true transformation between F_1 and E_6 has to be generated. (b) One hypothesis H for the alignment of F_1 and F_6 generated by the RANSAM operation.

Given a map consisting of N fragments and a randomly chosen sequence of samples $\mathfrak{S}$ as stated above, the weight for this specific solution is computed as

$$ \mathcal{W}_{\mathfrak{S}} = \sum_{i=2}^{N} (1 - \mathcal{N}(\mu_i, \Sigma_i)(E_{i-1})) + (1 - \mathcal{N}(\mu_H, \Sigma_H)(E_N)) \tag{10} $$

where $\mathcal{N}(\mu, \Sigma)(E)$ is the value of $\mathcal{N}$ at E. The result is subtracted from 1, because we want to minimize the weight of a path through the graph G.

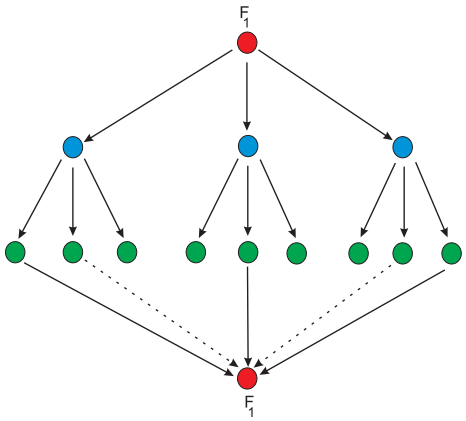

Fig. 3. This example shows the resulting graph for a small map with three fragments and three samples per fragment. Each fragment is encoded by the color of the circles. The third fragment closes the loop. Hence, all edges from the third fragment lead back to the first one. All edges of the graph are assigned a weight according to the Gaussian distribution of eqs. 5 and 6.

One example of G is depicted in Fig. 3. It illustrates the resulting graph for a small map consisting of three fragments where the third fragment closes the loop. G has a tree-like structure. The color of the nodes represents the different fragments. The red node is the initial fragment F_1. In this example, three samples are computed from each Gaussian. Hence, there exist nine possible combinations of how to align the fragments. All edges from the green nodes lead to a red node, because the loop is closed at this point. In general, given N fragments and M samples per fragment, there are M^{N-1} different solutions for the global map. Hence, it is not feasible to establish the graph explicitly from a computational point of view. Please note that the approach described above can be employed to close one single loop. Its application to SLAM instances with several loops requires some modifications and is subject of further research.

B. Correcting the Map Using Ants

We apply A ants for global map optimization. In the current version of our AntSLAM approach the nodes are chosen by ant k, $k \in [1, A]$ with a probability proportional to

$$ p_{ij}^{k}(t) = \frac{\tau_{ij}(t)}{\sum_{l \in \mathcal{S}_i} \tau_{il}(t)}. \tag{11} $$

Here, we assume ant k to be located at fragment i. $\tau_{ij}(t)$ is the amount of pheromone of sample j at time t. Again, $\mathcal{S}_i$ is set of samples assigned to fragment i (cf. eq. (7)). No

heuristic information as in eq. (1) is used. Thus, each ant constructs its own sample sequence $\mathfrak{S}^k(t)$. The pheromone is updated according to eq. (2) by

$$\Delta\tau_{ij}^k(t) = \begin{cases} \frac{1}{\mathcal{W}_{\mathfrak{S}^k(t)}} & \text{if sample } j \text{ of fragment } i \text{ has been} \\ & \text{chosen by ant } k \\ 0 & \text{otherwise .} \end{cases} \tag{12}$$

In this optimization algorithm, every ant is allowed to update the pheromone strength on the edges of the graph but with a certain frequency, we apply the globally best ant and the iteration best ant for the pheromone update. In order to avoid search stagnation, with a probability of p_0 ant k chooses the next sample according to an equal distribution. With a probability of $1 - p_0$, it chooses its next sample according to eq. (11). Moreover, we apply an upper threshold τ_{max} in order to avoid infinite pheromone increase. The algorithm cannot get stuck in a local minimum since samples can be chosen from an equal distribution. Hence, our method is a combination of Ant System, $\mathcal{MAX} - \mathcal{MIN}$ Ant System, and Ant Colony System, which gave the best results.

The optimization procedure is summarized in Algorithm (2). The parameter set

$$\mathfrak{G} = \{\mathcal{S}_i, \mathcal{N}(\mu_i, \Sigma_i), \mathcal{N}(\mu_H, \Sigma_H), i \in [2, N]\} \tag{13}$$

contains all information necessary for the optimization step. In lines 2 to 6, the globally best weight $\mathcal{W}_{best}$, the globally best sequence of samples $\mathfrak{S}_{best}$, and the pheromone are initialized. The optimization routine is executed until a maximum number of iterations is exceeded (line 8). Now, each ant iterates over all fragments and generates its individual tour, i.e. it chooses its own set of samples resulting in a global map estimation. Afterwards, the current tour quality is calculated and $\mathcal{W}_{best}$ and $\mathfrak{S}_{best}$ is updated if necessary (lines 22 to 26). In lines 29 to 34, the ants update the pheromone strength. A simple modulo operator determines which ants update the pheromone (line 29). Finally, the best sample combination is returned (line 36).

Algorithm 2 Ant SLAM

```
1:  procedure OPTIMIZE(𝔊)
2:      𝔖_best ← ∅
3:      𝒲_best ← ∞
4:      for i = 2 to N do
5:          τ_ij ← τ_init, j ∈ [1, M]
6:      end for
7:      iteration ← 0
8:      while iteration < iter_max do
9:          for k = 1 to A do
10:             𝔖^k ← ∅
11:             𝒲_{𝔖^k} ← ∞
12:             for i = 2 to N do
13:                 Choose random number p
14:                 if p > p_0 then
15:                     Choose next sample
16:                     according to eq. (11)
17:                 else
18:                     Choose next sample
19:                     with equal probability
20:                 end if
21:             end for
22:             Compute 𝒲_{𝔖^k} using eq. (10)
23:             if 𝒲_best < 𝒲_{𝔖^k} then
24:                 𝒲_best = 𝒲_{𝔖^k}
25:                 𝔖_best = 𝔖^k
26:             end if
27:         end for
28:         iteration ← iteration + 1
29:         if iteration mod globBestFreq ≠ 0 then
30:             Let all ants update the pheromone trails
31:         else
32:             Use only the global best ant
33:             and the iteration best ant
34:         end if
35:     end while
36:     return 𝔖_best
37: end procedure
```

IV. Experimental Results

The algorithm proposed in Sec. III has been tested on a system equipped with an AMD 2.2 GHz processor with 1 GB RAM. We employed a Microsoft Windows OS with a Visual Studio 2005 compiler. All sensor data has been gathered using a Sick LMS 200 laser range-finder with a field of view of 180° and an angular resolution of 0.5°. The robot we used to carry out the experiments is a meccanum wheel omnidirectional drive vehicle. Moreover, the optimization algorithm has been configured as follows. A new fragment has been initialized as soon as the robot has traveled a distance of more than 5 m since the last fragment initialization. One example for the partition of the map into fragments is depicted in Fig. 4. The map has been computed by a simple scan matcher. Every second fragment is highlighted by red color. The coordinate systems represent the exit points and the local frames of the fragments as described in Sec. III-A. Note that both coordinate systems coincide, because no optimization has been performed.

We always used 20 ants for optimization. The maximum number of iterations (cf. Algorithm (2)) has been set to 3,000. Moreover, 20 samples have been employed for each example given in this section. The variable p_0 is initialized with 0.3. Thus, about 30 percent of the ants choose the next city according to an equal distribution. Initially, *globBestFreq* is set to 5. Hence, every fifth iteration, only the global best ant and the iterative best ant may update the pheromone trail. τ_{max} is initially set to 10,000. The pheromone is initialized with 100. The algorithm has been tested in three different environments. The results of three different environments are depicted in Fig. 5. It shows the initial solution, the average map after 100 optimization trials, and the evolution of the

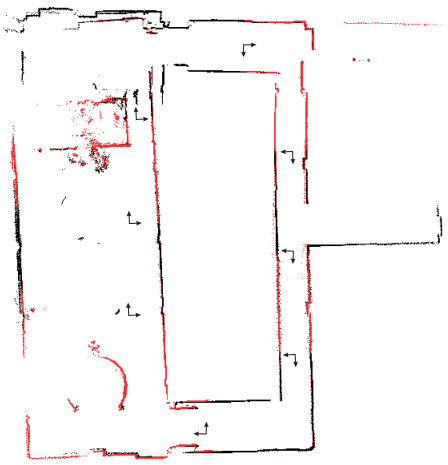

Fig. 4. This map has been generated by a simple scan matcher. Every second fragment is marked by the red color. The coordinate systems represents the exit points and local frame of the next fragment. Both frames coincide.

global best solution. Since the average solution is illustrated, the maps look blurred in some parts.

A. Basement of the Computer Science Building

The first result can be seen in Figs. 5 (a) – (c). This example shows the basement of our computer science building, which has an extension of 22 m × 14 m. The loop, which has been closed, has a length of about 53 m. The map after applying a simple scan matcher as mentioned in Sec. III-A is depicted in Fig. 5 (a). The map has a resolution of 4 cm. A large inconsistency at the top of the map can be observed resulting from some rotational errors at the bottom. The map has been partitioned into 9 fragments resulting in 20^8 possible solutions. Fig. 5 (b) shows the average map of 100 optimization trials. As can be verified, the inconsistency caused by the scan matching process has disappeared. The evolution of the best solution is illustrated in Fig. 5 (c). It depicts the mean and the standard deviation. 3,000 iterations require about 0.9 seconds, but the most significant decrease of the weight can be observed within the first 0.1 seconds. The optimization result is quite stable since the standard deviation also decreases very fast.

B. Second Floor of the Robotics Lab

In Figs. 5 (d) – (f), the second floor of our robotics lab has been mapped. This floor has a size of 22 m × 24 m. The according loop has a length of about 82 m. The scan matching result is illustrated in Fig. 5 (d). Again, the map has a resolution of 4 cm and exhibits large inconsistencies. It has been partitioned into 15 fragments. Hence, 20^{14} different solutions are possible. Please note that it is nearly impossible to get the perfect solution due to the high number of combinatorial possibilities. Instead, the algorithm is able to find a good solution. Although this solution is suboptimal, the map looks correct for the human eye on average. The average map of 100 optimization trials is depicted in Fig. 5 (e). The quality of the map is much better compared to the scan matching result because there is no inconsistency left from an optical point of view. The evolution of the map quality can be seen in Fig. 5 (f). Approximately 1.6 seconds are required for 3,000 iterations. As can be verified, after 0.2 seconds there is only a very slight increase of the map quality. Additionally, the standard deviation converges very fast.

C. Third floor of the Computer Science Building

Figs. 5 (g) – (i) show the results when applying our method to the sensor data of the third floor of our computer science building. This environment has a size of 51 m × 24 m. The loop to be closed has a length of about 137 m. The result of the scan matching process is depicted in Fig. 5 (g). Again, the map is not correct due to accumulated errors of the scan matcher. The partition process yields 24 fragments and hence 20^{23} solutions. The average map of 100 trials is illustrated in Fig. 5 (h). The corridor at the top is slightly curved, which is not optimal. Nevertheless, loop closing was successful. The evolution of the map quality as depicted in Fig. 5 (i) shows the same characteristics as in Figs. 5 (c) and (f).

V. Conclusion and Future Work

In this paper, we presented a method for solving the SLAM-Problem by applying an optimization technique based on the Ant Colony Optimization meta-heuristic. ACO has been applied to the well known Traveling Salesman Problem very often. We used a scan matching technique for the partition of the map into several fragments. The alignment of consecutive fragments must be modified slightly in order to obtain a globally consistent map. Hence, a set of samples drawn from a Gaussian distribution is generated for each fragment. When the robot closes a loop, a RANSAC-like matching approach is employed in order to compute the correct alignment of both fragments. This matching is very important in order to be able to measure the quality of a global map hypothesis. Thus, a Gaussian distribution is computed from the matching result. Afterwards, an ant algorithm is employed for map optimization by interpreting the set of samples as a graph with a tree-like structure. For optimization, the ants seek for a path, which has a minimum weight. Although the solution of the ants can be arbitrarily far away from the optimal solution given a constant number of iterations, experimental results have shown that a consistent map of our test environments is obtained on average. Since our experiments include only sensor data from our own building it will be interesting to use other data sets commonly available in the internet. This investigation is important in order to see how the algorithm performs in arbitrary environments. E.g., the algorithm presented in this paper will probably fail if the robot travels through large open spaces since it is hard to compute the map fragments in such a case. Furthermore, it is interesting to apply our method to other low cost sensors like cameras or sonar. Consequently, the sample generation step has to be adapted. Concerning cameras, a stereo vision system is promising since it provides geometrical information about the local

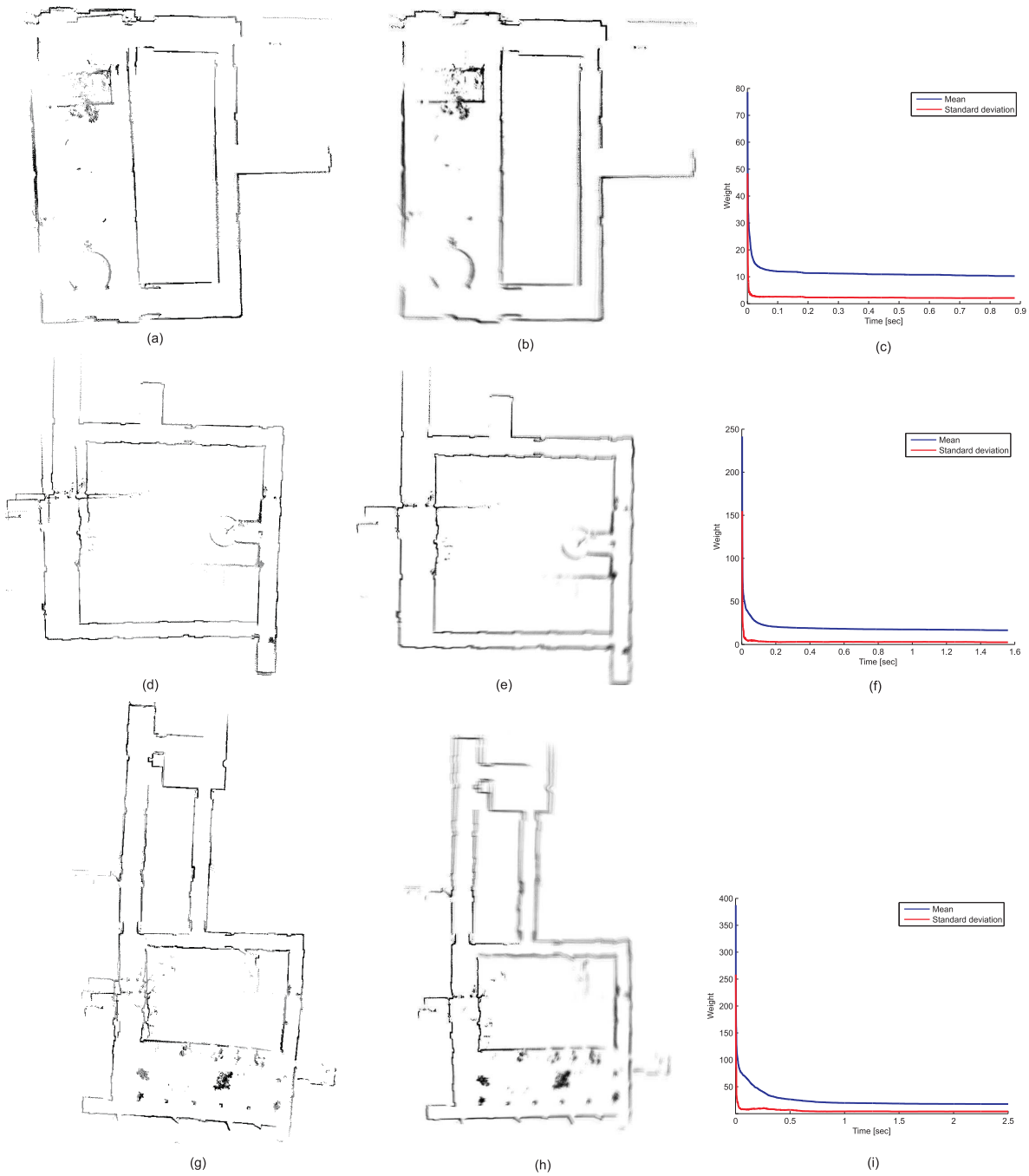

Fig. 5. (a) The map of the basement of our computer science building computed by a simple scan matcher. Note the large inconsistency at the top. (b) The resulting average map after 100 optimization trials. (c) The map quality during the optimization step. The curve depicts the mean and the standard deviation of the weights of 100 optimization trials. (d) The map of the second floor of our computer science building computed by the scan matcher. The map is not correct at the loop closing point at the left hand side. (e) The resulting average map of the second floor after 100 optimization trials. (f) The according map quality during the optimization step. The curve depicts the mean and the standard deviation of the weights of 100 optimization trials. (g) The map of the third floor of our computer science building computed by the scan matcher. Again, the map exhibits large inconsistencies. (h) The resulting average map of the third floor after 100 optimization trials. (i) The curve depicts the mean and the standard deviation of the weights of 100 optimization trials.

environment of the robot which allows for the computation of appropriate sample sets. In contrast to this, a (cheaper) mono SLAM system is appealing, but then the samples needs to represent the back projection error to the image plane. Consequently, a bundle adjustment technique must be applied locally [28]. A comparison with state-of-the-art optimization methods would be interesting.

The algorithm described for generating the graph structure allows for closing only one single loop. Hence, further research will focus on adapting the algorithm for coping with several (nested) loops. To this end, it is necessary to provide trees for each loop encountered during the mapping process. Consequently, given L different loops, L alignment hypotheses H needs to be computed by the RANSAM method (cf. Sec. III-A), which requires a reliable loop closing detection. Subsequently, there exist several options for the optimization procedure as described in Sec. III-B. First, each tree can be processed by the ants in an iterative manner. To be more precise, each loop could be optimized one after the other. I.e., given L loops, the optimization procedure is applied L times. However, this approach might produce suboptimal maps since each loop will contain small errors accumulating over time. Secondly, the individual ants could traverse each loop within the optimization routine. Thus, the weight of the individual solutions depends on the global quality. This method requires higher computational resources but is likely to provide better results on average.

ACKNOWLEDGEMENTS

We would like to thank *QNX Software Systems* for providing free software licenses. Furthermore, we would like to thank the anonymous reviewers for their helpful comments on the draft of this paper.

REFERENCES

[1] B. Bullnheimer, R. F. Hartl, and C. Strauss. An improved ant system algorithm for the vehicle routing problem. *Annals of Operations Research*, 89(0):319–328, 1999.

[2] G. Di Caro and M. Dorigo. Antnet: Distributed stigmergetic control for communications networks. *Journal of Artificial Intelligence Research*, 9:317–365, 1998.

[3] M. Dorigo and L. M. Gambardella. Ant colony system: A cooperative learning approach to the travelling salesman problem. *IEEE Transactions on on Evolutionary Computation*, 1(1):53–66, 1997.

[4] M. Dorigo, V. Maniezzo, and A. Colorni. The ant system: Optimization by a colony of cooperating agents. *IEEE Transactions on Systems, Man, and Cybernetics, Part B*, 26(1):1–13, 1996.

[5] S. Teller E. Olson, J. Leonard. Fast iterative optimization of pose graphs with poor inital estimates. In *Proc. of IEEE International Conference on Robotics and Automation*, pages 2262–2269, 2006.

[6] A. M. Fischler and R. C. Bolles. Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 46(6):381–395, 1981.

[7] J. Folkersson and H. I. Christensen. Closing the loop with graphical SLAM. *IEEE Trans. on Robotics*, 23(4):731–741, August 2007.

[8] U. Frese and L. Schröder. Closing a million landmark loop. In *Proc. of IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 5032–5039, 2006.

[9] M. A. Garcia, O. Montiel, O. Castillo, R. Sepuldveda, and P. Melin. Path planning for autonomous mobile robot navigation with ant colony optimization and fuzzy cost function evaluation. *Applied Soft Computing*, 9(3):1102–1110, 2009.

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

[11] G. Grisetti, C. Stachniss, and W. Burgard. Improved techniques for grid mapping with rao-blackwellized particle filters. *IEEE Trans. on Robotics*, 23(1):364–375, 2007.

[12] G. Grisetti, C. Stachniss, S. Grzonka, and W. Burgard. A tree parametrization for efficiently computing maximum likelihood maps using gradient descent. In *Proc. of Robotics: Science and Systems*, 2007.

[13] R. Iser, D. Kubus, and F. Wahl. An efficent parallel approach to random sample matching (pRANSAM). In *Proc. of IEEE International Conference on Robotics and Automation*, pages 1199–1206, 2009.

[14] R. Iser and F. Wahl. Building local metrical and global topological maps using efficient scan matching approaches. In *Proc. of IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1023–1030, 2008.

[15] M. Kose and A. Acan. Knowledge incorporation into aco-based autonomous mobile robot navigation. In *Lecture Notes in Computer Science*, pages 41–50. Springer, 2004.

[16] M. Milford and G. Wyeth. Mapping a suburb with a single camera using a biologically inspired slam system. In *IEEE Trans. on Robotics*, volume 24, pages 1038–1053, 2008.

[17] M. Milford, G. Wyeth, and D. Prasser. Ratslam on the edge: Revealing a coherent representation from an overloaded rat brain. In *Proc. of IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 4060–4065, 2006.

[18] H. P. Moravec. Sensor fusion in certainty grids for mobile robots. *AI Magazine*, pages 61–74, 1988.

[19] K. Murphey. Baysian map learning in dynamic environments. In *Advances in Neural Information Processing Systems*, pages 1015–1021. MIT-Press, 1999.

[20] C. Shi, Y. Bu, and J. Liu. Mobile robot path planning in three dimensional environment based on acp-pso hybrid algorithm. In *Proc. of IEEE/ASME International Conference on Advanced Intelligent Mechatronics*, pages 252–256, 2008.

[21] D. Silver, D. Bradley, and S. Tayer. Scan matching for flooded subterranean voids. In *IEEE Conference on Robotics Automation and Mechatronics*, pages 422–427, 2004.

[22] R. Smith, M. Self, and P. Cheeseman. Estimating uncertain spatial relationships in robotics. *Autonomous robot vehicles*, pages 167–193, 1990.

[23] B. Steder, G. Grisetti, S. Grzonka, C. Stachniss, A. Rottmann, and W. Burgard. Learning maps in 3d using attitude and noisy vision sensors. In *Proc. of IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 644–649, 2007.

[24] T. Stützle and H. H. Hoos. MAX-MIN ant system and local search for the travelling salesman problem. In *IEEE International Conference on Evolutionary Computation*, pages 309–314, 1997.

[25] T. Stuetzle and M. Dorigo. *The Ant Colony Optimization Meta-Heuristic*, chapter 9, pages 163–179. Wiley, 1999.

[26] T. Stuetzle and M. Dorigo. *Evolutionary algorithms in engineering and computer science: recent advances in genetic algorithms , evolution strategies, evolutionary programming, genetic programming and industrial applications*, chapter 9, pages 163–179. Wiley, 1999.

[27] S. Thrun. Learning metric-topological maps for indoor mobile robot navigation. *Artificial Intelligence*, 99(1):21–71, 1988.

[28] Bill Triggs, P. McLauchlan, Richard Hartley, and A. Fitzgibbon. Bundle adjustment – a modern synthesis. In B. Triggs, A. Zisserman, and R. Szeliski, editors, *Vision Algorithms: Theory and Practice*, volume 1883 of *Lecture Notes in Computer Science*, pages 298–372. Springer-Verlag, 2000.

[29] S. Winkelbach, S. Molkenstruck, and F. M. Wahl. Low-cost laser range scanner and fast surface registration approach. In *Pattern Recognition (DAGM 2006)*, pages 718–728, 2006.

[30] K. M. Wurm, C. Stachniss, G. Grisetti, and W. Burgard. Improved simultaneous localization and mapping using a dual representation of the environment. In *Proc. of the 3rd European Conference on Mobile Robots*, pages 132–137, 2007.