Sensing Locations Positioning for Multi-robot Inspection Planning

Jan Faigl
Department of Cybernetics (K13133)
Czech Technical University in Prague
Technická 2, Prague 6, 166 27, Czech Republic
xfaigl@labe.felk.cvut.cz

Miroslav Kulich
Center of Applied Cybernetics
Czech Technical University in Prague
Technická 2, Prague 6, 166 27, Czech Republic
kulich@labe.felk.cvut.cz

Abstract

Problems of cooperative multi-robot inspection and exploration play an important role in many practical applications. This paper presents an algorithm for inspection planning based on decomposition of the problem into two subproblems - Art Gallery Problem (AGP) that finds guards (sensing locations) and Multiple Traveling Salesmen Problem (MTSP) that connects the found guards by routes. While standard approaches for Art Gallery Problem try to minimize a number of guards, the proposed method is designed to optimise lengths found by a MTSP solver and therefore to minimise time needed by a team of robots to inspect the working environment. The proposed algorithm has been implemented and tested. Influence of the method to quality of the inspection planning solution and comparison with the Randomized Dual Sampling Schema are discussed.

1. Introduction

Many path-planning tasks including motion planning from the start to goal position, obstacle avoidance, planning with constrains, coordinate planning are studied in mobile robotics domain. If the working environment is a priori known (a map of the environment is available), one of common problems is an inspection task. The problem is to find a route such that a robot surveys whole working space (i.e. sees every point of the environment) while it is moving along the route. Recently, with growing importance of multi-robot systems, the problem has been extented for a team of robots. Using multiple robots can reduce total execution time, but on the other hand, it needs novel algorithms for coordinating robots' movements. Typical application of the inspection task is a search and rescue mission in case of emergencies or catastrophes [4], where the goal is to find (and rescue) insured people or other objects of interest (fire alarms, seats of fire, places, where dangerous materials are placed, etc) as fast as possible .

A typical environment is a building with offices connected by corridors. Nowadays, a map for many such buildings (especially for strategic buildings like hospitals, airports, nuclear plants, etc.) exists in the form of architectonical plans or electronic CAD model. Such a model is easily convertible to a polygonal map, that is one of the common used representations suitable for planning.

The inspection task for a team of mobile robots can be formulated as Multiple Watchmen Routes Problem (MWR). Having a polygonal map of the environment (represented by a polygon with holes P), the aim is to find in an optimal way a route for each of m mobile guards such that each point of P is visible from at least one route. There are two criteria mentioned in the literature evaluating optimality of the found routes. In case of the *MinSum* criterion, the aim is to minimize the sum of lengths of watchmen routes, while the length of the longest watchman route is to be minimized for *MinMax* criterion. Nilsson [5] proved that both problems are NP-hard even for a simple polygon.

Approaches commonly used in robotics solving (i.e. trying to find an optimal solution) MWR problem decompose the original problem into two subproblems: determination of sensing locations (points such that every point of the environment is visible from at least one such point) and planning a route over found locations. The first problem is also known as *Art Gallery Problem* (AGP), while the second one is *Multiple Traveling Salesman Problem* (MTSP). Both problems are known to be NP-hard.

The Art Gallery Problem for a polygon (with holes) P is to find an optimal set of points G (guards) in P such that every point in P is visible from some point of G (we say that two points in polygon P are visible, iff the straight line segment between them entirely lies in P). A polygon with n vertices and h holes can always be guarded with $\lceil \frac{n+h}{3} \rceil$ point guards [1]. If visibility is restricted to distance d two points in P are d-visible if they are visible and their distance is less than d.

The theoretically achieved bound for a polygon with holes is too high in many practical situations, so there is

Proceedings of the IEEE Workshop on Distributed Intelligent Systems: Collective Intelligence and Its Applications (DIS'06)
0-7695-2589-X/06 $20.00 © 2006 IEEE

effort to research algorithms that find a lower number of sensing locations than this theoretical bound. An algorithm based on randomized dual sampling of the environment is described in [3]. Another approach is introduced in [2].

One of the main issues of problem decomposition is that both problems are solved separately without any interaction. Specifically, AGP solvers minimize the number of sensing locations, but an optimal solution for the inspection task might have more points that are compact, i.e. a path connecting them is shorter than for the optimal AGP solution.

We present a concept of AGP solver generating a compact set of sensing locations especially for large office environments and a relatively small visibility distance in comparison to dimensions of the environment. The proposed method is compared with a *Randomized Dual Sampling Schema* algorithm. Instead of dual sampling, we use single sampling of a boundary of not covered regions. The second sampling is replaced by a heuristic function that uses already found sensing locations. New locations are placed close to that already found. Although the number of sensing locations can be higher, they are placed in order to minimize lengths of routes travelled by particular robots during inspection.

The rest of this paper is organized as follows. Section 2 contains description of an algorithm based on dual sampling schema 2.1 and proposed algorithm to find sensing locations 2.2.

Related MTSP with MinMax criterion problem is solved by a self-organizing neural network algorithm, which is described in section 3. Experimental results are shown and discussed in section 4 followed by conclusion in section 5.

2 Determination of sensing locations

The first step of an approximative solution of Multiple Watchmen Routes Problem is to find sensing locations ”guarding” whole working environment. Suppose P be a map of the environment represented as a polygon with holes. The aim is to find a set of points (sensing locations) so that every point of the environments is visible from at least one sensing location. Moreover, visibility is limited, which means that only points closer than a specified threshold are mutually visible. This limitation is very practical because of characteristics of current sensors. For example, having camera resolution and a size of objects to be detected, we can easy determine a maximal distance from which the objects are recognizable. Visibility distance is limited by conditions in the scene like darkness or smoke. In the next sections we describe two different algorithms finding sensing location. The first one is our implementation of the *Randomized Dual Sampling Scheme* introduced in [3]. This algorithm is designed to solve the Art Gallery Problem, i.e. it tries to find a minimal set of sensing locations. On the other hand, a novel *Boundary Placement algorithm* described in section 2.2 places sensing locations in order to minimize lengths of routes that connect them.

2.1 Randomized Dual Sampling Scheme

The algorithm was developed by Gonzáles-Bañoz and Latombe. It incrementally adds points to the solution while the volume of not covered region A is larger then zero. The algorithm therefore proceeds as a loop:

1. Denote A an area to be guarded.
2. A random point p lying on the border of the area A is chosen.
3. A polygon V_p is found, which consists of points visible from the point p (this is equivalent to the polygon from which p is visible). All the visibility constrains as defined above are applied.
4. k random samples p_k are placed into the polygon V_p.
5. For each point p_k a visibility polygon (polygon from which p_k is visible) is determined.
6. The guard (point) that can see the most still unguarded area (i.e. the point for which $|A - V_{p_k}|$ is smallest) is chosen as a next guard.
7. Set $A = A - V_{p_k}$.
8. If A is not empty (there exists a point which is not guarded) then go to step 2).

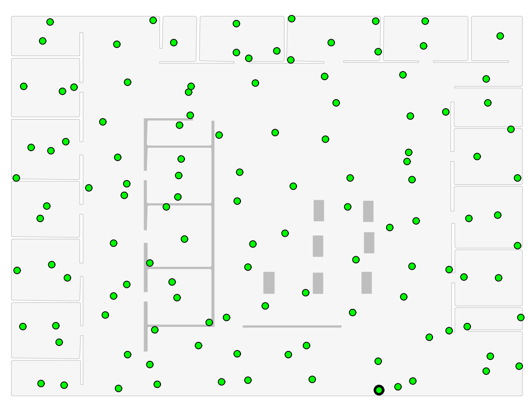

Figure 1. Found guards by RDS algorithm for visibility 200.

2.2 Boundary Placement algorithm

As mentioned above, the main motivation for the Boundary Placement algorithm is to place sensing locations more intelligently, with respect to the inspection task. The idea is not to place the locations close to walls. Instead of this, new locations are placed near to the ones found in the previous steps of the algorithm.

The algorithm consists of three main steps, where each step is an iterative process. First of all, structure of the environment is determined by computation of a cover boundary. The cover boundary B_c is a boundary blown up by a predefined value d_b. More precisely, the cover boundary is a set of points of the environment, which distance to the nearest obstacle boundary is d_b. B_c divides the environment into to sets of regions - exteriors and interiors (see Fig. 2). The internal region is a connected set of points having a distance to a nearest obstacle higher than d_m, while points having this distance smaller than d_m form external regions. In other words, internal regions lie inside the area bounded by the cover boundary, while external regions lie outside.

After the cover boundary as well as internal and external regions are determined, points (guards) covering them are generated for each component separately:

1. Cover Boundary: The cover boundary is represented as a set of linear rings (i.e. borders of polygons). The boundary is covered in an iterative process, where in each step a point lying on it is chosen randomly. Then the region visible from the point is subtracted from the boundary. This process is repeated until the whole boundary is covered.

2. Internal regions: Regions visible from the points covering the cover boundary are subtracted from each internal region (represented by a polygon) which are then covered by an iteration process. If the area of a particular region is small enough then a new guard covering it is placed as close as possible to two nearest guards. In case the region is large, the following steps are performed:

   (a) A random point g at the region boundary is selected and a circle with a center g and a radius $2d$ is constructed.

   (b) The center c of the arc defined as intersection of the circle and the region is determined.

   (c) A new guard is defined in the middle of the line gc.

   Regions can split during the iterative process. If this happens, each of the split parts is covered separately.

3. External regions: Similarly to the previous step, visibility regions of the already found guards are subtracted from all external regions. If the region is large enough (i.e. its area is larger then a predefined value) then the process is similar to covering internal boundaries. Otherwise, the following steps are performed until the whole region is covered:

   (a) Select a random point p on the region boundary, which is not an obstacle (i.e. a boundary created as a subtraction of some visibility polygon).

   (b) Find a closest guard g to p and determine the shortest path from g to p. This path is a polyline $v_1 v_2 v_n$ so that $v_1 = g$ and $v_n = p$.

   (c) Find a point lying on a line $v_{n-1}p$ and covering a largest part of the region. This point is a new guard.

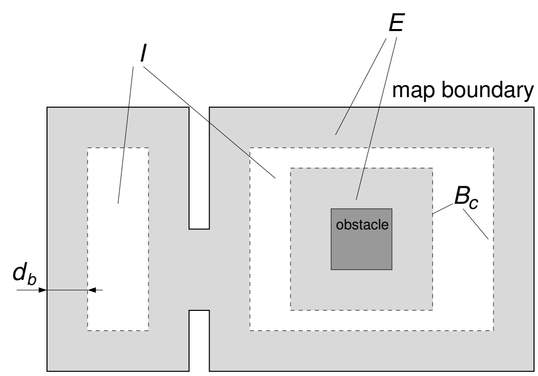

Figure 2. Cover Boundary (B_c) and internal (I) and external (E) regions.

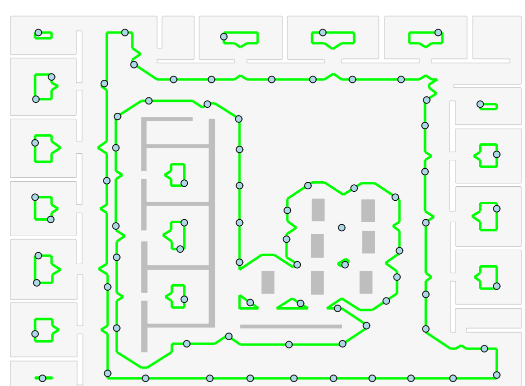

Figure 3. Guards found on the cover boundary.

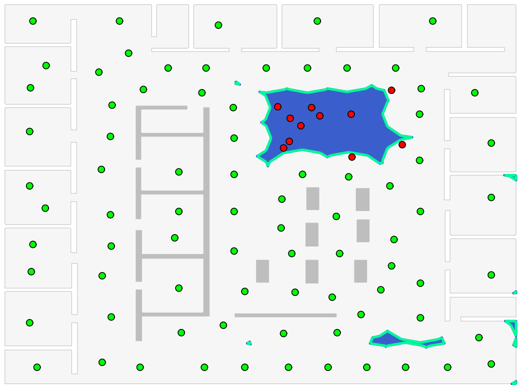

Figure 4. Guards found in internal regions (red dots).

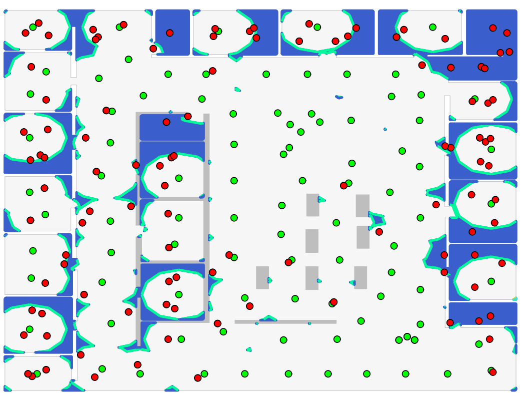

Figure 5. Guards found in external regions (red dots).

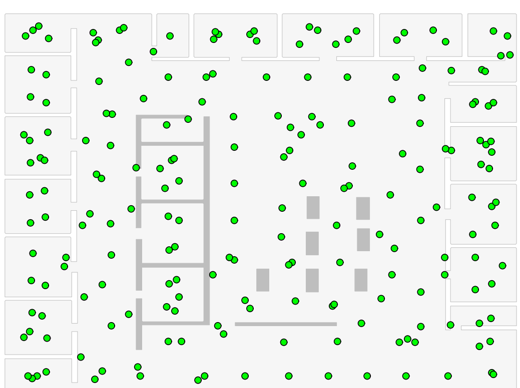

Figure 6. Guards found by BP algorithm for visibility 200.

3 Multiple Traveling Salesmen Problem

The second step of the Multiple Watchmen Route Problem is to connect the guards with m paths in an optimal way in order to form inspection paths. This problem is NP-hard and therefore no polynomial algorithm exists. Our approach extends Somhom's algorithm described in [6]. The idea of the algorithm is to represent a path of each particular robot by a chain of neurons, where neighbouring neurons are connected [4]. At each iteration, a nearest neuron to a randomly selected guard is determined and together with its' neighbours moved closer to the guard. The algorithm can be described as follows:

If we denote n as the number of guards (cities) and m as the number of salesmen, then m chains can be created so that each consists of $M = 2n/m$ neurons. Initially, the on-chain neurons are positioned on a small ring close to a starting point (depot) for each salesman. The next steps then choose random permutations of the guards and existing neuron chains, i.e. C_{p_i} is the i-th city in a permutation, for which the nearest neuron to the C_{p_i} is determined. To select the nearest neuron, the guard-to-neuron distance is defined as their Euclidean geodesic distance weighted by: $weight(r) = ((length(r) - AVG)/AVG)^4$. This suppresses those neurons, which overshoot AVG and prefers those, which are bellow AVG, where AVG stands for the average length of chains in the task.

Whenever determining the minimum distance neuron, the winner and its nearest neighbours on the chain are attracted to the guard. The value of the winner movement towards the guard is proportional to its' distance to the guard and additionally weighted by exponential function of the distance and the iteration number. This ensures rapid changes in the network topology for larger neuron-to-guard distances and more precise and slow convergence at final phases of the iteration process.

The process of the permutation choice, nearest neuron selection and shifting towards proper guard is executed unless a termination criterion is achieved. This can be defined as a maximum distance between a guard and a nearest neuron to this guard being satisfied if the distance is smaller than a certain threshold.

Finally, found paths of particular entities are optimized by 2-opt heuristics, which is commonly used in TSP-like tasks. The heuristics generates so-called 2-optimal tour, i.e. tour in which there is no possibility to shorten the tour by exchanging two arcs.

4 Experimental Results

The proposed Boundary Placement algorithm (BP) as well as Randomised Dual Sampling (RDS) were implemented in order to compare their behaviour. Compari-

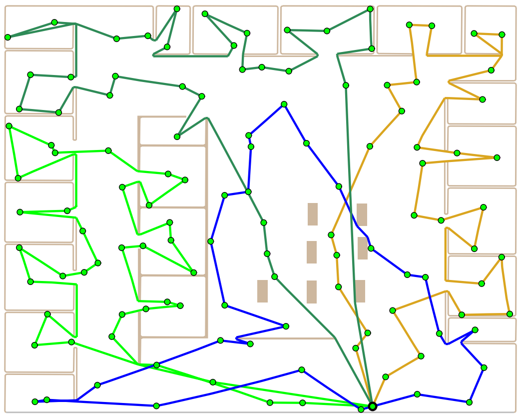

Figure 7. Found routes for 4 salesmen, Incremental Dual Sampling Algorithm.

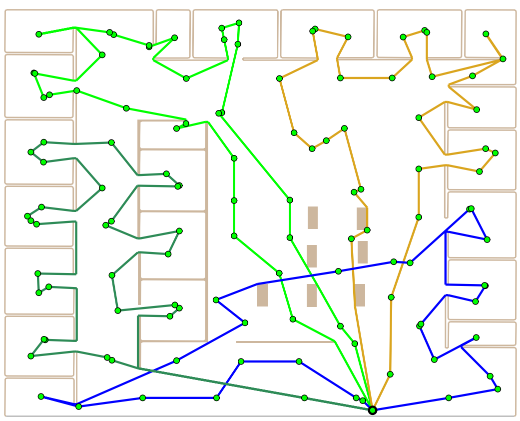

Figure 8. Found routes for 4 salesmen, Boundary Placement Algorithm.

son was performed for three different office-like environments: *A, B, C*. The maps were created semi-automatically from the paper maps of real buildings.

The map A represents a corridor with a free space and many offices (Fig. 6), B stands for a corridor with three large offices (Fig. 9) and C contains several long corridors (Fig. 10).

Both algorithms are randomised, therefore 20 solutions were found for each map. After that the neural network-based algorithm connecting the found guards was performed 20 times for each set of guards and for a different number (2,3, and 4) of robots. d_b of the Boundary Placement was set to 100, while visibility distances varied from 200 to 500. A number of points in the second sampling in RDS was set to 50. [1]

[1]The implementation of Boundary Placement algorithm is written in C++ using CGAL and in absolute precision. Polygon operations are very slow therefore solution cost is not compared.

An average number of found guards with a standard deviation is shown in Table 1. However BP was not designed to minimize a number of guards and in some cases the guards are close to each other, the number of guards is comparable with results obtained by RDS. Results generated by BP for the environment A are even better. Moreover, BP is faster, because many guards are placed on boundary line and interior region that is straightforward, while RDS uses dual sampling.

Name	Visiblity	RDS		BP	
		Avg	Dev	Avg	Dev
A	200	117	2.70	114	3.11
	300	63	3.31	60	2.40
	400	46	2.32	47	1.61
	500	41	1.41	42	1.20
B	200	169	3.32	170	4.92
	300	84	3.13	88	2.32
	400	52	2.87	57	2.45
	500	41	2.17	41	1.94
C	200	349	5.99	334	5.61
	300	177	5.11	179	4.14
	400	121	3.75	124	3.93
	500	91	3.15	96	2.46

Table 1. Comparison of a number of guards found by RDS and BP.

Quality of found solutions is shown in Tables 2, 3 and 4. The first column *d* stands for a visibility distance, the second one for the number of robots, and others describe quality of the algorithms. First of all, the longest tour of each particular solution (remember that *MinMax* criterion is used to evaluate MTSP solutions) is computed and the average length over all 400 MTSP solutions *Avg* as well as their standard deviation *Dev* is presented. The last column *RL* describes percentage of a maximal route length of the proposed algorithm compared to RDS (RDS has 100%).

Although BP is designed especially for small visibilities, experimental results show that found solutions for higher visibilities (comparing to a size of the environment) are not excessively worse. For a visibility distance 200, solutions found by BP are approximately about 5% better than solutions generated by RDS. The best results are generated by BP for the environment A, for which BP gives about 15% shorter lengths than RDS. This results significantly show that guards are placed *more* intelligently.

5 Conclusion and Future Work

The paper deals with a problem of placing sensing locations (guards) for multi-robot inspection planning. A novel

d	n	RDS		BP		
		Avg	**Dev**	**Avg**	**Dev**	**RL %**
200	2	148.28	5.39	124.45	3.27	83.9
	3	108.61	5.59	91.21	3.80	84.0
	4	91.45	4.46	78.88	2.81	86.3
300	2	109.04	6.65	107.54	4.62	98.6
	3	82.12	5.20	79.84	3.94	97.2
	4	73.23	3.84	71.76	2.91	98.0
400	2	95.63	4.52	101.91	3.75	106.6
	3	74.27	4.23	77.22	3.41	104.0
	4	67.33	2.76	70.58	2.26	104.8
500	2	94.61	4.35	99.27	3.54	104.9
	3	73.65	3.43	76.16	2.76	103.4
	4	67.16	2.79	69.81	2.43	104.0

Table 2. Comparison of BP and RDS for the environment A.

d	n	RDS		BP		
		Avg	**Dev**	**Avg**	**Dev**	**RL %**
200	2	215.70	9.91	202.84	8.63	94.0
	3	158.95	11.45	149.09	9.99	93.8
	4	132.99	5.78	126.32	4.94	95.0
300	2	176.11	9.58	170.97	8.77	97.1
	3	131.34	6.49	127.62	5.02	97.2
	4	117.69	3.74	114.87	3.10	97.6
400	2	158.86	9.80	155.29	8.63	97.8
	3	123.03	7.02	119.62	4.66	97.2
	4	112.38	3.67	110.00	2.61	97.9
500	2	152.31	9.55	149.07	8.63	97.9
	3	117.93	5.40	114.96	6.32	97.5
	4	110.27	4.13	108.50	3.43	98.4

Table 3. Comparison of BP and RDS for the environment B.

d	n	RDS		BP		
		Avg	**Dev**	**Avg**	**Dev**	**RL %**
300	2	496.48	16.74	486.00	18.66	97.9
	3	396.09	18.18	385.89	19.02	97.4
	4	347.01	18.40	338.07	16.73	97.4
400	2	469.61	20.26	465.83	19.39	99.2
	3	383.42	20.79	376.61	19.52	98.2
	4	336.45	16.16	329.48	15.10	97.9
500	2	448.73	22.77	447.25	22.12	99.7
	3	373.05	23.06	369.36	21.31	99.0
	4	327.70	15.26	325.36	13.73	99.3

Table 4. Comparison of BP and RDS for the environment C.

algorithm designed especially for visibilities significantly smaller than a size of the working environment is presented. The method has been implemented and experimentally compared with a standard Randomized Dual Sampling algorithm. The experiments show that the proposed algorithm gives better results for small visibilities.

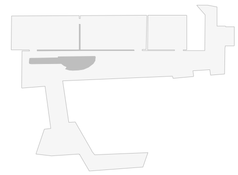

Figure 9. Map B.

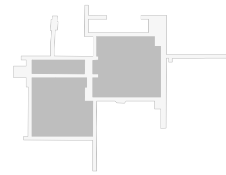

Figure 10. Map C.

The presented planning approach was integrated into the PeLoTe system [5] and as its part it was experimentally tested in many simulated rescue missions and demonstrated in the Firefighter Training Facility in Wuerzburg. The tests and demonstrations showed feasibility of the planning system in real life.

In the future work we would like to focus on setting the constant d_b determining a distance between the cover boundary and obstacles. This constant should be set automatically taking into account a shape of the environment and robot's visibility distance. Other stream will focus on improvement of the method and more comprehensive comparison with other approaches.

Acknowledgement

This work has been supported by the Ministry of Education of the Czech Republic under Project No. 1M0567.

The work has been partially supported with the IST project No. 506958 "ECOLEAD".

References

[1] I. Bjorling-Sachs and D. Souvaine. A tight bound for guarding polygons with holes. Technical report, Rutgers University, 1991. Report LCSR-TR-165.

[2] T. Danner and L. Kavraki. Randomized planning for short inspection paths. In *Proceedings of The IEEE International Conference on Robotics and Automation (ICRA)*, pages 971–976, San Fransisco, CA, April 2000. IEEE Press.

[3] H. González-Baños and J.-C. Latombe. Planning robot motions for range-image acquisition and automatic 3d model construction. In *AAAI Fall Symposium*, 1998.

[4] M. Kulich, J. Faigl, and L. Přeučil. Cooperative planning for heterogeneous teams in rescue operations. In *IEEE International Workshop on Safety, Security and Rescue Robotics*, 2005.

[5] B. J. Nilsson. *Guarding Art Galleries: Methods for Mobile Guards*. PhD thesis, Lund University, 1995.

[6] S. Somhom, A. Modares, and T. Enkawa. Competition-based neural network for the multiple travelling salesmen problem with minmax objective. *Computers and Operations Research*, 26(4):395–407, 1999.