Efficient Topological Exploration

Ioannis M. Rekleitis Vida Dujmović
Gregory Dudek
Centre for Intelligent Machines,
School of Computer Science, McGill University,
3480 University St., Montreal, Québec, Canada H3A 2A7

Abstract

We consider the robot exploration of a planar graph-like world. The robot's goal is to build a complete map of its environment. The environment is modeled as an arbitrary undirected planar graph which is initially unknown to the robot. The robot cannot distinguish vertices and edges that it has explored from the unexplored ones. The robot is assumed to be able to autonomously traverse graph edges, recognize when it has reached a vertex, and enumerate edges incident upon the current vertex. The robot cannot measure distances nor does it have a compass, but it is equipped with a single marker that it can leave at a vertex and sense if the marker is present at a newly visited vertex. The total number of edges traversed while constructing a map of a graph is used as a measure of performance. We present an efficient algorithm for learning an unknown, undirected planar graph by a robot equipped with one marker. One of the main results of this paper is to show that our strategy leads to performance that is typical linear in the size of the graph. Experimental results obtained by running a large collection of example worlds are also presented.

1 Introduction

In this paper we present a new algorithm for the exploration of graph-like environments by a mobile robot. This builds on previous methods for the exploration and mapping of environments in the absence of metric information. Our approach makes the assumption that the environment can be represented by a planar graph. This assumption permits mapping and exploration to take place much more efficiently than with previously published methods.

In this paper we assume that a robot is able to define a topological representation of its environment; that is, a graph. We assume that the robot can move between the nodes of this graph and do not concern ourselves in this paper with the particulars of how this might be achieved. A purely topological representation is devoid of metric information, including edge lengths or Cartesian coordinates. Techniques that are able to build an environmental map with only such information can be viewed as assuring worst-case performance bounds for environments with noisy metric observations.

In this paper we assume that the robot can move between the nodes of a graph. Any any time, the robot can sense some *signature* associated with its current node, but we cannot be certain that this signature is unique. In practice, such a signature might be the color of the room it is in, or the bounding contour of its current location. Since we assume that the robot can move between nodes, the simplest signature one might postulate is simply the number of edges (e.g doors) incident on the current node. In this paper, since we are concerned primarily with worst-case bounds, we will consider only the degree of the current node as a signature. We also assume that the robot can enumerate the edges incident on its current location in some consistent manner (e.g it can count the doors to the current room in a clockwise fashion). This allows it to take the n'th door out of its current room using the door by which it entered as a reference.

Note that the we do not assume that the robot can associate a unique label with either a node or an edge using sensory information. With only such information, a robot cannot uniquely build a map of an unknown environment. For example, a 3 node cycle is indistinguishable from a 4 node cycle, since in either case the robot simply observes a non-terminating sequence of rooms with two exits. By allowing the robot to sense a movable marker, however, any graph-like environment can be mapped in a number of moves bounded by a polynomial in the number of nodes.

In the next section, we will briefly discuss relevant background research. Section 3 provides necessary definitions and a formal description of the problem. In Section 4 we present our proposed strategy for mapping an unknown planar graph-like world. Section 4.3 presents a counter-example for non planar graphs. Finally, in Section 5 experimental results are presented together with a comparison to previous work.

2 Background

The importance of *topological representations* of the environment has been observed by several researchers in both mobile robotics and human cognitive science. In previous work, Chatila *et al.* [CL85] considered the creation of topological map representations from metric data while Kuipers and Byun [KL88] considered the synthesis of metric maps from topological representations created by servo-like procedures. Several authors have also considered hybrid maps that combine aspects of both metric and topological representations [Ark90, EM92]. Kortenkamp and Weymouth considered the use of multiple sensing modalities to instantiate the nodes of a graph-like (topological) representation [KW94]. Other work has also considered the theoretical issues involved in fully covering an unknown graph using topological exploration [DP90].

In previous work it has been observed that while topological mapping with ambiguous signatures with absolute certainty is infeasible, the use of a single movable *marker* allows efficient mapping. With one marker, a mapping algorithm is possible that uses a number of robot steps which is a low-order polynomial in the number of nodes in the graph that represents the environment [Dud88, DJMW91]. Specifically, for an arbitrary graph, the number of robot moves, referred to as the *mechanical complexity* has a worst-case asymptotic complexity of $6nm$ or $\mathcal{O}(nm)$ where n is the number of nodes in the graph and m is the number of edges in the graph. The key notion is that by moving a marker to incrementally determine the relationship of additional nodes to an expanding *known subgraph*, a complete map of an initially-unknown graph-like environment can be efficiently constructed. For arbitrary (and hence potentially non-planar) graphs, the incremental exploration is substantially complicated by the need to determine how to connect newly discovered nodes to the existing graph.

Relative efficiency of this type of marker-based strategy was examined in comparison to other "footprint" methods in recent definitive work by Deng and Mirzaian [DM96]. In essence, it is difficult to improve on this worst-case complexity bound for marker-based exploration *for arbitrary graphs.*

Other work has considered the representation of possible alternative models of the environment using topological exploration without any markers [DFH92]. This becomes more efficient if even a limited amount of metric data is available, for example constraining local relationships [DMW97]. Given a topological map of an environment, it is also possible to consider the feasibility of verification of the map or the robot's position [DFH93, DJMW97]. Even in purely metric environments, several authors have considered representations that explicitly define locations associated with distinctive sensor signatures [GMR92] or with a geometric tree [Kle94] that facilitate vertex-based positioning or exploration.

3 Problem Specification

In this section we formally specify the mapping problem. The problem is to generate a map in the form of a graph $G = (V, E)$, where V are the nodes (or vertices) and E and the edges, of an unknown environment which is also represented by a graph $G' = (V', E')$. At the completion of the algorithm, the graphs G and G' must be isomorphic to one another.

At any time the robot is located in some node of the graph. It can sense a local non-unique signature of its current node, for example the degree of the node (since the signature is non-unique, we do not require it in the current algorithm and it will not be discussed further). It can also detect the identity of a labelled marked in a room that it enters. It can also implicitly detect the edge by which it entered the current vertex and it can enumerate the (other) edges incident upon the vertex: that is, it can establish a local ordering of the edges with respect to the one it entered by. *Note that the robot cannot sense the actual (absolute) label associated with either a vertex or an edge.*

The robot moves from one vertex to another by selecting an edge incident upon the current vertex and moving along it. This can be expressed by a transition function $\delta(v_j, E_{i,j}, s) = v_k$ where the robot is in vertex v_j, having entered via edge $E_{i,j}$ and s is the index of the edge it chooses to leave by, numbered with respect to $E_{i,j}$. (e.g. "it is in the living room, having entered by the door behind it, and it now leaves by the 3rd door counted clockwise from it's left.") If it finds a marker in its current room, it can elect to pick up the marker and carry it. At any time, it can drop any marker.

4 Approach

In this section we present an algorithm for exploring a planar graph with typically linear cost. The algorithm is incremental and depends on the maintenance of an explored subgraph (subject to verification) upon which newly explored parts of the graph are merged. In contrast to the technique proposed by Dudek *et al* [DJMW91] where a single vertex is added at a time, here a closed path (an *ear* [1]) is added each time. A single marker is used in order to mark the starting node of the explored ear.

Before we continue with a description of the algorithm, we will outline some useful definitions. A vertex in our subgraph is called *fully explored* if every incident edge to it is *fully explored*. An edge of the graph is considered *fully explored* if the robot has traversed it in both directions. The edge that the robot is following on the way out of a vertex is called *outgoing edge*, the edge by which the robot arrives to a vertex is called *incoming edge*.

The basis of the algorithm is a recursive depth first exploration. From each vertex, the robot first explores all incident edges and then proceeds to explore all neighboring vertices, as follows:

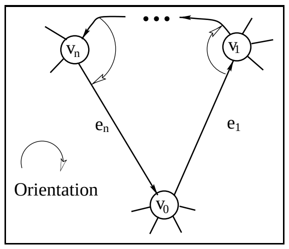
```
While Unexplored Edges Exist
   For Every Unexplored Vertex Do
      For Every Unexplored Edge Do
         Drop Marker
         Explore Ear
         Merge Ear to Existing Subgraph
      Explore Every Adjacent Vertex
```

4.1 Exploring an *Ear*

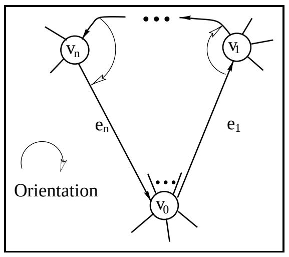
The main component of the exploration algorithm is the exploration of a closed path called an *ear*. Any planar graph can be decomposed into a union of ears. At every vertex where there is an unexplored edge the robot drops the marker in order to mark the starting vertex and then starts the exploration by making only "right turns" until it returns to the marked vertex. Upon arrival at a new vertex, the robot has an orientation set (e.g. clockwise) which determines the next edge to be traversed. For every new vertex visited, a node is instantiated and added to the new *ear*. When the robot arrives at the marked vertex it knows the number of edges traversed p, and the *incoming edge*, but it has no method of knowing the *outgoing edge* by which it started the exploration. In order to connect the newly explored *ear* to the explored subgraph the robot methodically traverses the *ear* in the opposite direction. [2] More specifically, first the robot picks up the marker and backtracks to the previous vertex visited where it drops the marker, then the robot continues backtracking until it either reaches the marked vertex or it has performed p edge traversals. There are three different cases depending on the topology of the explored graph:

Empty *Ear*: After $p+1$ traversals the robot found the marker. In that case the robot has explored an empty *ear* and the *incoming edge* is adjacent to the *outgoing edge* (see Figure 1a). There are no nodes inside such an ear as at every node the immediate neighbor was selected.

Non Empty *Ear*: After $p+1$ traversals the robot has not found the marker. In that case the robot has explored an non empty *ear* and there is a number of edges between the *outgoing edge* and the *incoming edge*. A sequential one-step search of every edge adjacent to the vertex would reveal the number of edges between the *outgoing edge* and the *incoming edge* (see Figure 1b). There are nodes inside an non empty ear but these nodes are completely enclosed by the explored ear, e.g. there are no edges between the vertices inside and any other vertices in the graph except edges to vertex V_0.



(a)



(b)

Figure 1: (a) Exploring an empty *ear*, (b)Exploring a non empty *ear*. In both case from Vertex V_0, *outgoing edge* V_1, *incoming edge* V_n. The order of traversal is clockwise.

Isthmus: The Marker is found after $p-1$ steps. In that case the *incoming edge* is identical to the *outgoing edge* (see Figure 2). The explored path encloses a subgraph that has a single connection to the rest of the graph via the vertex V_0.

[1]Any connected planar graph can be decomposed into a set of cycles which are called ears (Maon et al [MSV88]).

[2]The following step could be eliminated by adding a second marker. The second marker is dropped after following the first outgoing edge thus marking the outgoing edge that started the ear.

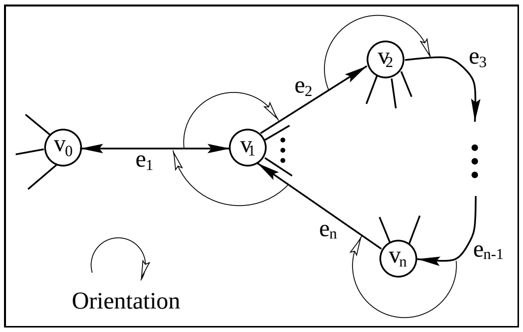


Figure 2: Exploring an *Isthmus*. Returning through the same edge it exited, *incoming* and *outgoing* edges are the same.

4.2 Merging an *Ear*

After a new *ear* is constructed and the *incoming* and *outgoing edges* are known the explored subgraph has to be updated with the new information. A recursive procedure is applied in order to merge the newly explored vertices with the existing ones. When two vertices are merged then their neighbors are also merged according to the following rules:

- If the neighbors along a corresponding edge for node v_i and v_j are different, it means that the neighboring nodes represent the same physical node. Thus we need to merge them.
- If the neighbor along a corresponding edge is unknown for v_i then the neighboring node of v_j becomes the neighbor of v_i.

Figure 3 illustrates this idea in a small graph.

4.3 Non Planar Graphs

The efficiency of this approach derives from the fact that it is specific to planar graphs. This is illustrated in Figure 4. After following a non-planar edge the robot duplicates the map and continues to make copies ad infinitum.

5 Results

Experiments were carried out on random planar graphs in order to obtain a statistical evaluation on the typical cost of our algorithm. In addition the example used by Dudek *et al* [DJMW91] was used for comparisons.

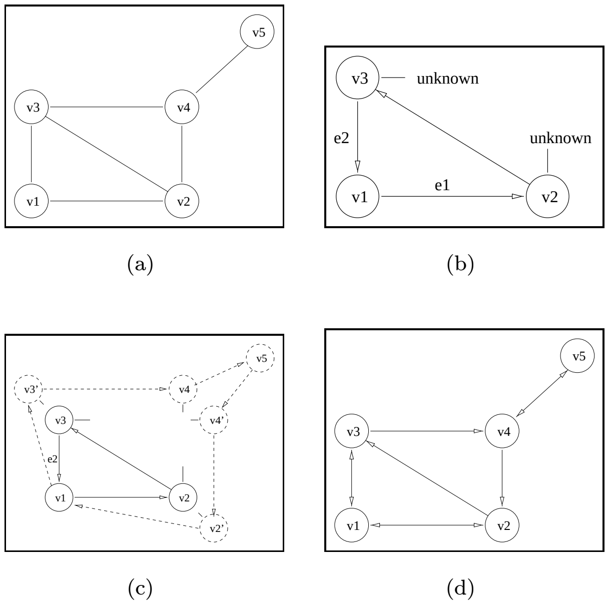


Figure 3: (a) The model world. (b) The first *ear* is explored from vertex V_1 starting from the edge $e_1 = (v_1, v_2)$. (c) The second *ear* is explored starting from the edge $e_2 = (v_1, v_3)$. (d) The second *ear* is merged with the explored subgraph.

5.1 Building Random Planar Graphs

The first task in the experimental work was the robust construction of random planar graphs. In order to build a random but connected planar graph the following process is followed. A set of random points is created in 2D, and the Delaunay triangulation of them is constructed [PS85, For87] (see Figure 5). The Delaunay triangulation leads to a dense planar graph out of which random edges could be deleted in order to create a random graph. In order to eliminate edges randomly, weights are assigned on every edge, the minimal spanning tree (MST) is constructed, and every edge of MST is marked as undeletable. Finally, from a dense graph of V (number of vertices), and E (number of edges) (Figure 5) a random graph could be constructed with the same number of vertices (V), and any number of edges from $V-1$ (keeping just the spanning tree), up to E (the dense Delaunay triangulation). The above procedure ensures that the graph stays planar, and that an ordering of the edges could be obtained for every vertex by radially ordering the edges clockwise. For example typical values for the generated graphs are:

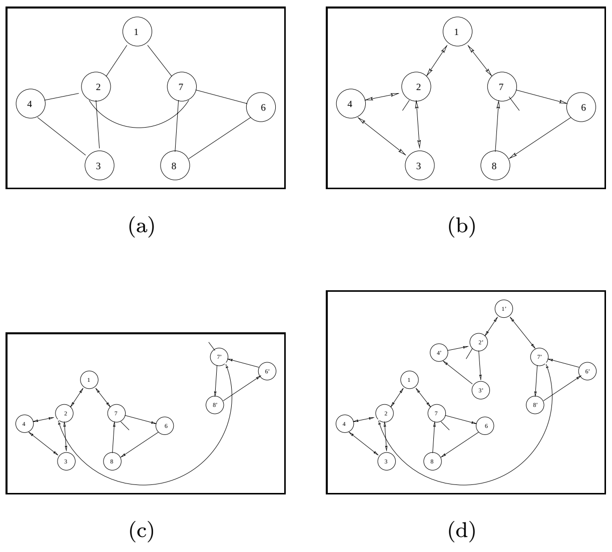


Figure 4: (a) The model world (non-planar graph). (b) The map after traversing edges $1-2$ and $1-7$ (c) The map after traversing edges $2-4$, $2-3$ and $2-7$ (d) The map after traversing edges $\acute{7}$ - $\acute{6}$ and $\acute{7}$ - $\acute{1}$.

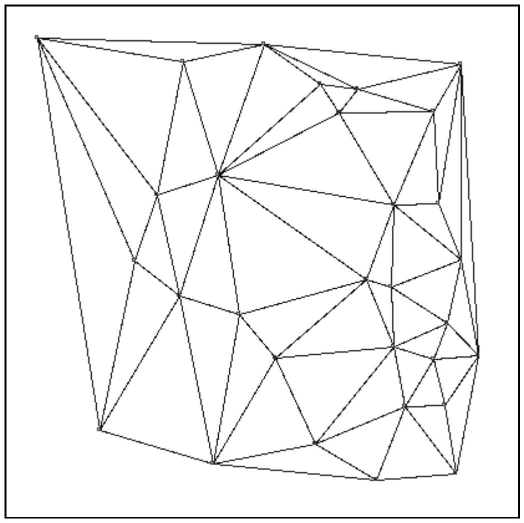


Figure 5: The Delaunay triangulation of a set of random points.

starting with one thousand random points the minimal spanning tree would have 999 edges and the Delaunay triangulation had 2980 edges (for a random sample). Random graphs (V, E) could be constructed with $V = 1000$ and $E \in [999, 2980]$ by following the above procedure.

5.2 Statistical Analysis

Using the above method multiple edge arrangements can be randomly created from a set number of vertices. Consequently, experiments could be performed with different graphs for a set number of vertices (V) and varied number of edges (E), the ratio of the number of edges over the number of vertices $(n = E/V)$ represents the density of the graph. In certain instances of sparse graphs the robot traverses almost the complete graph before making a merge, thus creating duplicate images of the whole graph during the exploration process thus increasing the cost.

The mechanical complexity of the exploration algorithm is measured with the number of moves (edge traversals) In the Figures 7, 8, 9, the results for 100, 500 and 1000 node graphs are displayed, the edge density varies from one to three, and for each pair of (V, E) ten different graphs were created and explored. For every

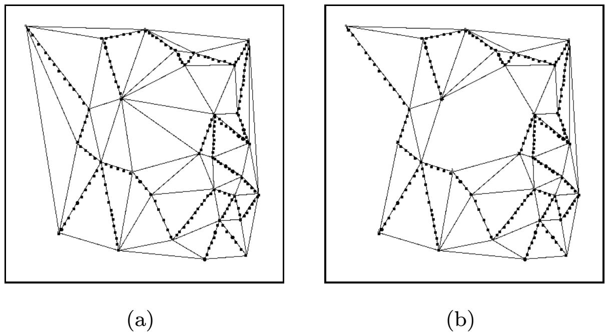


Figure 6: (a) The graph of Figure 5 with an arbitrary spanning tree highlighted. (b) The same graph with five edges (not in the spanning tree) deleted.

edge density value, minimum, maximum and average values are presented.

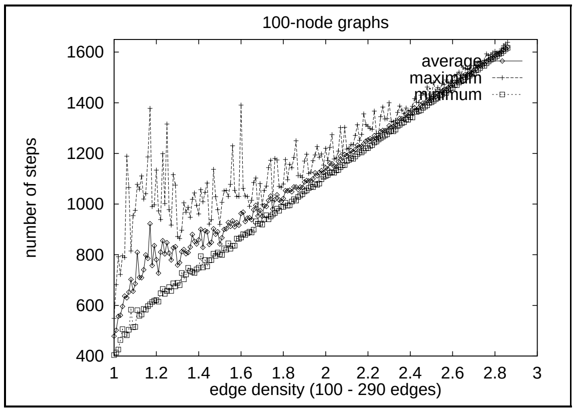


Figure 7: Complexity of the exploration of 100 vertices graphs.

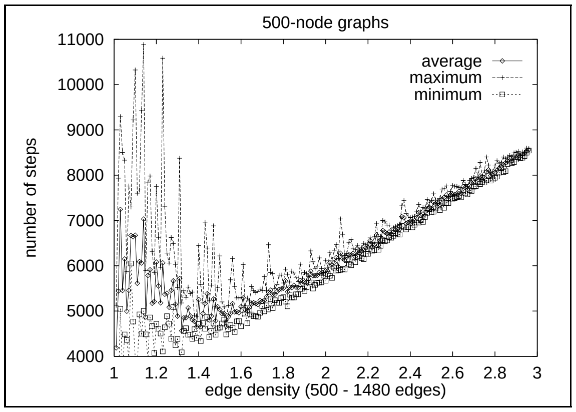


Figure 8: Complexity of the exploration of 500 vertices graphs.

On average the number of translations is linear in the number of edges, this is more evident when for an average edge density of two, random graphs where created starting with 100 nodes up to 1000 nodes. As can be seen in Figure 10 the minimum , maximum and average values all increase linearly to the number of vertices/edges.

5.3 Comparative results

Dudek *et al.* demonstrated their algorithm in a graph representing the Toronto underground complex,

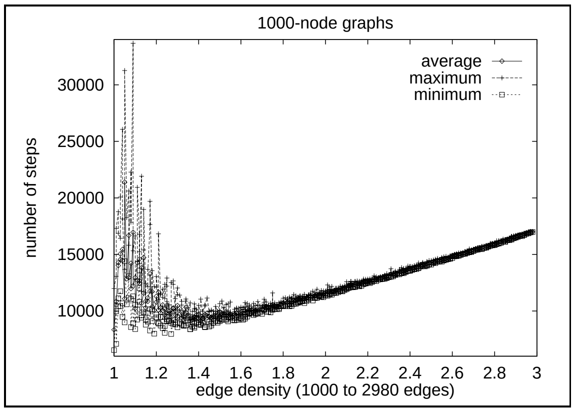


Figure 9: Complexity of the exploration of 1000 vertices graphs.

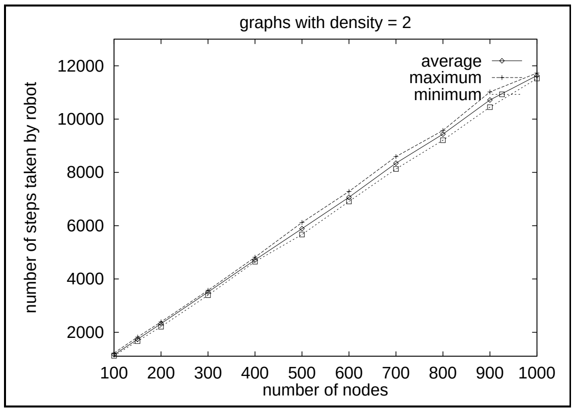


Figure 10: For edge density $n = 2$ and graph sizes from 100 to 1000, the number of translations increases linearly.

(see Figure 11) extending under much of the downtown core of the city [DJMW91]. In their experiments the robot was placed in the upper left vertex and was equipped with a single marker. In their experiments the robot needed 5134 steps to fully explore the graph. Using the method presented here, we conducted the same experiment and the resulting number of steps was reduced to 433.

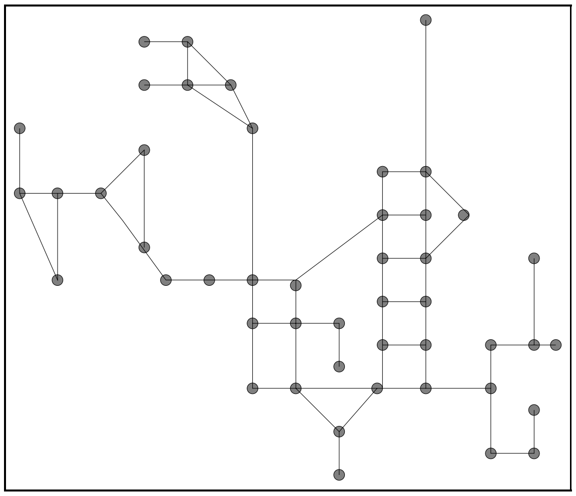


Figure 11: Graph of the Toronto underground complex.

6 Conclusions

In this paper we proposed a new strategy for exploring unknown planar graph-like environments with help of a single marker. Though applicable only to planar graphs, it is a significant improvement over previous methods. Thousand of experiments were carried on random environments verifying a linear average cost of the exploration, based on the number of edges.

This approach can also be extended to environments that include both metric and topological data. We are currently exploring this using vision-based sensing.

References

[Ark90] R. Arkin. Integrating behavioural, perceptual, and world knowledge in reactive navigation. *Robotics and Autonomous Systems*, 6:105–122, 1990.

[CL85] R. Chatila and J. Laumond. Position referencing and consistent world modelling for mobile robots. In *IEEE International Conference on Robotics and Automation*, pages 138–170, 1985.

[DFH92] Gregory Dudek, Paul Freedman, and Souad Hadjres. Algorithms for active exploration of unknown environments: Using uncertain sensing data to create a reliable map. In *Proc. Conf. on Mobile Robotics VII*, Boston, MA, Nov. 1992. International Society for Optical Engineering.

[DFH93] Gregory Dudek, Paul Freedman, and Souad Hadjres. Using local information in a non-local way for mapping graph-like worlds. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence*, pages 1639–1645, Chambery, France, August 1993. Internation Joint Conf. on Artificial Intelligence Inc.

[DJMW91] Gregory Dudek, Michael Jenkin, Evangelos Milios, and David Wilkes. Robotic exploration as graph construction. *Transactions on Robotics and Automation*, 7(6):859–865, December 1991.

[DJMW97] G. Dudek, M. Jenkin, E. Milios, and D. Wilkes. Map validation and robot self-location in a graph-like world. *Robotics and Autonomous Systems*, 22(2):159–178, November 1997.

[DM96] Xiaotie Deng and Andranik Mirzaian. Competitive robot mapping with homogeneous markers. *IEEE Trans. on Robotics and Automation*, 12(4):532–542, August 1996.

[DMW97] G. Dudek, M. Jenkin. E. Milios, and D. Wilkes. On building and navigating with a globally topological but locally metric map. In *Proc. 3rd ECPD Int. Conf. on Advanced Robotics, Intelligent Automation and Active Systems*, pages 132–136, Bremen, Germany, 1997.

[DP90] Xiaotie Deng and Christos Papadimitriou. Exploring an unknown graph. In *Annual Symposium on the Foundations of Computer Science*, pages 335–361, 1990.

[Dud88] Gregory Dudek. *Segmentation using information from multiple features.* Laboratory for biological and computational vision, University of Toronto, September 1988.

[EM92] Sean P. Engelson and Drew V. McDermott. Error correction in mobile robot map learning. In *Proc of the IEEE International Conference on Robotics and Automation*, pages 2555–2560, Nice, France, May 1992.

[For87] Steve J. Fortune. A sweepline algorithm for voronoi diagrams. *Algorithmica*, 2:153–174, 1987.

[GMR92] Leonidas J. Guibas, Rajeev Motwani, and Prabhakar Raghavan. The Robot Localization Problem in Two Dimensions. In *Proceedings of the Third Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 259–268, Orlando, FL, January 27–29 1992.

[KL88] B. Kuipers and T. Levitt. Navigation and mapping in large-scale space. *AI Magazine*, pages 25–43, summer 1988.

[Kle94] Jon M. Kleinberg. The localization problem for mobile robots. In *Proc. 35th IEEE Conference on Foundations of Computer Science*, Santa Fe, NM, Nov. 1994. IEEE Computer Society Press.

[KW94] David Kortenkamp and Terry Weymouth. Topological mapping for mobile robots using a combination of sonar and vision sensing. pages 979–984, 1994.

[MSV88] Yael Maon, Baruch Schieber, and Uzi Vishkin. Parallel ear decomposition search (eds) and st-numbering in graphs, 1988.

[PS85] F. P. Preparata and M. I. Shamos. *Computational Geometry: An Introduction.* Springer-Verlag, New York, NY, 1985.