Towards Autonomous Topological Place Detection Using the Extended Voronoi Graph*

Patrick Beeson, Nicholas K. Jong, and Benjamin Kuipers
Intelligent Robotics Lab
Department of Computer Sciences
University of Texas at Austin
{*pbeeson, nkj, kuipers*}*@cs.utexas.edu*

***Abstract*— Autonomous *place detection* has long been a major hurdle to topological map-building techniques. Theoretical work on topological mapping has assumed that places can be reliably detected by a robot, resulting in deterministic actions. Whether or not deterministic place detection is always achievable is controversial; however, even topological mapping algorithms that assume non-determinism benefit from highly reliable place detection. Unfortunately, topological map-building implementations often have hand-coded place detection algorithms that are brittle and domain dependent.**

This paper presents an algorithm for reliable autonomous place detection that is sensor and domain independent. A preliminary implementation of this algorithm for an indoor robot has demonstrated reliable place detection in real-world environments, with no *a priori* environmental knowledge. The implementation uses a local, scrolling 2D occupancy grid and a real-time calculated Voronoi graph to find the skeleton of the free space in the local surround. In order to utilize the place detection algorithm in non-corridor environments, we also introduce the *extended Voronoi graph (EVG)*, which seamlessly transitions from a skeleton of a midline in corridors to a skeleton that follows walls in rooms larger than the local scrolling map.

***Index Terms*— Place detection, topological navigation, Voronoi graph, corridor following, coastal navigation.**

I. Introduction

Places are important in many different ways. For spatial reasoning, people use places to define both bounded and unbounded regions of space: "in Texas" or "near the statue." Places are further used for metaphorical high-level reasoning such as mathematical ranking and social ranking: "first place" or "overstepping one's place." Places are even used to refer to normal roles or functions: "the place of the media" or "something is out of place."

The broad use of places is a specific example of a crucial point in the study of intelligent agents—spatial knowledge is a foundation for high-level common-sense knowledge [1]. We believe that by studying the problem of grounding continuous sensory experience to low-level, symbolic spatial constructs, we are developing a foundation for the high-level common-sense knowledge necessary for an intelligent agent to act in environments with people and other animals.

*Research of the Intelligent Robotics lab is supported in part by grants from the National Science Foundation (IIS-0413257), from the National Institutes of Health (EY016089), and by an IBM Faculty Research Award.

This paper investigates the idea of defining places. In previous work, we have described (a) how places, along with paths and regions, can form topological maps [2], [3], and (b) how an agent can learn highly-accurate place recognition (global localization) from experience [4]. Here, we consider the more fundamental problem of *place detection*: how a finite set of discrete places is abstracted from experience in a continuous environment. Spatial knowledge is undoubtedly hierarchical, but we leave the issue of hierarchical spatial abstraction to future work, focusing instead on the detection of the smallest, atomic places necessary for low-level abstraction of experience to symbols.

This paper is organized as follows. In Section II, we outline the two criteria that we use to define basic topological places. Section III summarizes related work in autonomous place detection and demonstrates the failures of these Voronoi-based methods with respect to the criteria for places. Section IV introduces an extension of the Voronoi graph to overcome some of these problems. In Section V, we introduce a way to prune the Voronoi graph using small-scale-space models of the local, immediate surround. We then define an algorithm for reliably detecting places using this new Voronoi graph in Section VI. We conclude with a summary of this work.

II. Defining Topological Places

Topological navigation is a behavior that is used by a variety of different animal species, including humans [5]–[7]. Topological representations discretize the continuous world into a finite set of places connected by paths. This helps facilitate large-scale spatial reasoning, mainly due to the compactness of the representation. The symbolic nature of topological representations allows higher-level reasoning (such as containment, order, connectivity, regions) which can enhance large-scale spatial cognition. Additionally, the abstraction of continuous space into symbols facilitates communication between agents.

All of the above tasks depend on the initial low-level abstraction of atomic places from continuous sensory experience. Here we attempt to define these places using criteria necessary for topological map-building behaviors. We define places at locations in the environment that satisfy the following criteria.

1) *Places occur at qualitative changes along paths.*

The majority of the time, this criterion defines places at intersections. The importance of path intersections in topological representations cannot be overstated. The recent work of Guilford et al. [8], demonstrates that pigeons, long considered to use Earth's magnetic field to fly long distances, follow man-made highway networks in well-known environments, using familiar highway intersections as markers for specific turn actions. It has long been known that humans use intersections as a basis for building spatial representations in unknown environments [9]—preferring first to build structural models of novel environments prior to detailed visual models [10].

Occasionally people may define places that are not at intersections. Dead-ends are one example of important places that are not necessarily at an intersection of two or more paths. Less useful places (from a topological mapping perspective) can be defined by using salient landmarks along long and qualitatively uninteresting paths. Such landmarks are useful for occasional relocalization along paths and thus define places based on regions of visibility. Common examples are unusual buildings along highways such as water towers. From the perspective of topological map-building, places that are not at intersections or dead-ends do not yield any additional structural information. In fact, as people do start to remember landmarks, they are biased to first learn landmarks at critical path changes along routes [11], [12].

2) *Places must be reliably detectable.*

Although this criterion seems simple, it is extremely important. Qualitatively interesting locations should be considered as possible places when they exist, but their persistence needs to be taken into account.

Any place detection algorithm needs to provide a stable set of places. Intersections are not only common but extremely stable (they rarely disappear). Thus, this criterion is mostly needed for places that are not at intersections. For example, a bright pink car parked on the side of a highway, while salient, does not define a reliable place. The chances that the car will remain at the location for an extended period of time are very low.

Places that are not at intersections depend on higher-level knowledge about stability, saliency, and informativeness that may not be available for most robotic implementations. For this reason, the algorithms and implementation for mobile robots detailed in the following sections focus exclusively on places at intersections and dead-ends, although future robotic implementations may be able to bootstrap place detection at other interesting locations.

Theoretically, if place detection is deterministic, topological map-building can be viewed as finding the minimal *deterministic finite automaton* that explains the experience of places [2], [13], [14]. When place detection is not deterministic, topological map-building can be viewed as a *partially observable Markov decision process*, which needs a large amount of experience to accurately model even simple environments [15]. A reliably detected set of well-separated but connected places leads to efficient map-building, while unreliable, non-deterministic place detection reduces map-building efficiency. Thus, any autonomous place detection algorithm should limit the amount of false positive and false negative place detections.

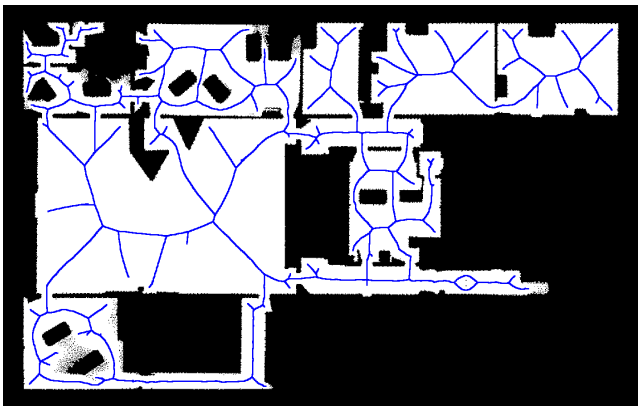


Fig. 1. **The generalized Voronoi graph of a global metrical map.** Using an occupancy grid [19], [20], the Voronoi graph can be drawn by using any non-free cells as obstacles. Junction points include intersections in corridor environments.

Below, we examine a previously published autonomous place detection algorithm. To our knowledge, this is the only such algorithm that is not hand-coded to work for a specific environmental domain. We will discuss both its successes and the conditions where it fails to meet the criteria above. Section VI outlines our own place detection implementation, which is similar in spirit but more robust to environmental variation and sensor noise.

III. Places at Voronoi Junctions

Choset has been a main proponent for the use of Voronoi graphs in topological map-building and navigation [16], [17]. Choset and Nagatani [18] define a *generalized Voronoi graph (GVG)* as a one-dimensional set of points equidistant to the n closest obstacles in n dimensions, where a preset threshold determines whether observations belong to the same "obstacle." When using planar range-finder devices (like common laser and sonar devices) in walled environments, this results in a set of points equidistant from two or more obstacles (Fig. 1).

Choset and Nagatani use the GVG both as a way to discretize the continuous environment into a finite set of places and as a way to define paths. Whenever the robot arrives at a *junction* in the Voronoi graph, it is at a topological place. Junctions are points on the GVG equidistant from $n + 1$ or more closest obstacles: where the graph forks into multiple branches. The branches that emanate from a junction define the paths that the robot can travel along to leave the place. For robotic navigation, we only consider junctions that have more than or less than two emanating branches: pruned graphs may leave junctions with only two emanating branches.

With respect to the first criterion from Section II, junction points on the GVG seem to make reasonable place definitions. As shown in Fig. 2, if a robot has a complete

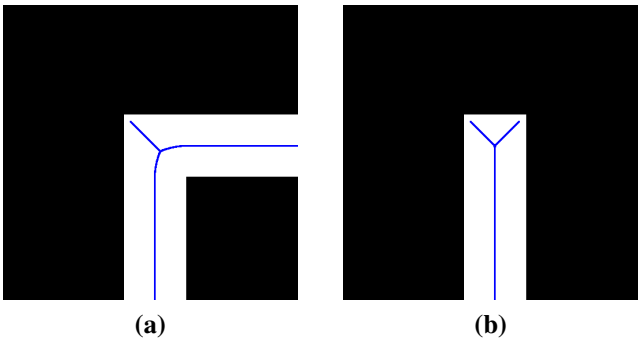


Fig. 2. **The Voronoi graph from the robot's point-of-view.** Given that robots have a limited sensory horizon, the Voronoi graph must be constantly computed in a local, scrolling model centered on the robot. For this scenario, we do not count the edges of the horizon as obstacles, so the branches of the graph can actually touch the edge of the image.

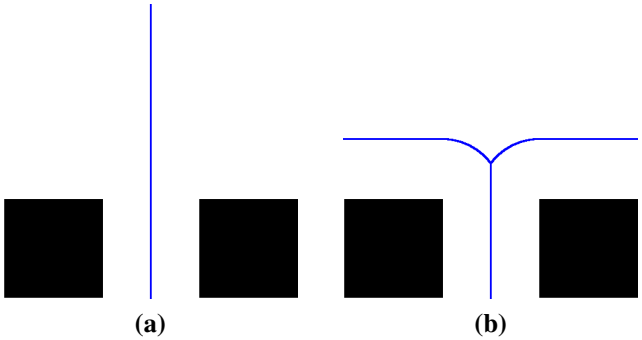


Fig. 3. **Entering a large room.** Voronoi graphs are not useful in non-corridor environments due to the robot's limited sensory horizon. **(a)** Here, the robot should begin to follow a wall, but the path continues into unknown territory without even marking a place at this crucial decision point. **(b)** Here the extended Voronoi graph defines a transition from corridor-following to wall-following, which is what we want.

local model of its immediate surround, it can compute the Voronoi graph (i.e. the skeleton of free space in an occupancy grid). In corridor environments, junctions can be found at most corners, intersections, and dead-ends.

Although the Voronoi graph is well defined in corridor environments, when the robot transitions into non-corridor environments, the Voronoi graph no longer becomes a useful skeleton. This is illustrated in Fig. 3(a). Here, the robot moves into a room that is larger than its sensory horizon. At this location, there exists a large qualitative change in the environment. By the first criterion, a place should be defined at this location; however, no junction point exists.

The appropriate control at this point is coastal navigation, keeping at least one obstacle in sight at all times [21], [22]. Instead of defining paths parallel to the walls, the Voronoi graph defines a path that will quickly bring the robot to a location where no visible obstacles exist. Once this occurs, the Voronoi graph can no longer be computed from local sensor data, and the robot is simply lost. In Section IV, we introduce an extension to the traditional Voronoi graph that seamlessly transitions from a "midline" skeleton to a "wall-following" skeleton (Fig. 3(b)).

Additionally, the GVG may not create junctions at certain intersections. For example, occasionally two paths

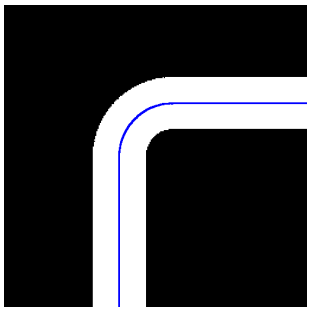


Fig. 4. **A common intersection.** Voronoi graphs do not branch in some commonly found intersections.

intersect in a smooth way, without changing the width of the corridor. The Voronoi graph sees such locations as being on a single path with no nearby places (Fig. 4).

The second criterion for places is that they should be reliably detectable. Choset and Nagatani [18] admit that the generalized Voronoi graph can find many spurious topological places due to range-sensing noise (Fig. 5(a)). They propose using a *reduced generalized Voronoi graph (RGVG)* that contains fewer "spurs" in the graph—branches that end at obstacles (Fig. 5(b)). The RGVG eliminates spurious places because certain junction points become terminal points. However, their pruning algorithm only eliminates the first level of spurious junctions, leaving junctions at non-intersections in noisy environments (Fig. 5(c,d)). In Section V, we discuss a way to prune the Voronoi graph in local small-scale models that eliminates all spurious junctions.

Unfortunately, the RGVG also eliminates topological places at 'L' intersections, which are invaluable for topological inference (Fig. 6). In Section VI, we discuss a way to detect places at 'L' intersections, even at "smooth" intersections (like Fig. 4). The same algorithm also allows complex intersections (those with multiple un-pruned junctions) to be considered as a single place.

IV. The Extended Voronoi Graph (EVG)

The Voronoi graph is a good way to find the "skeleton" of bounded paths such as corridors or streets for use in robot navigation. However, there are scenarios where the Voronoi graph does not work well on a robot with a limited sensory horizon. Fig. 3(a) illustrates how the Voronoi graph acts when the robot moves from a corridor to a room that extends beyond the sensory horizon. In this scenario, the natural and meaningful spatial configuration includes a place at the opening and paths that inform the robot how it can leave the place and continue trajectory-following controls: wall-following or corridor-following.

To obtain a graph that exhibits this behavior, we introduce an extension to the generalized Voronoi graph. The idea is simple, yet powerful, in that the Voronoi graph can be extended to be useful for a robot in non-enclosed environments.

We can define the *extended Voronoi graph (EVG)* as the subset of all points in the GVG closer than a threshold M units from any obstacle, added to the set of all points

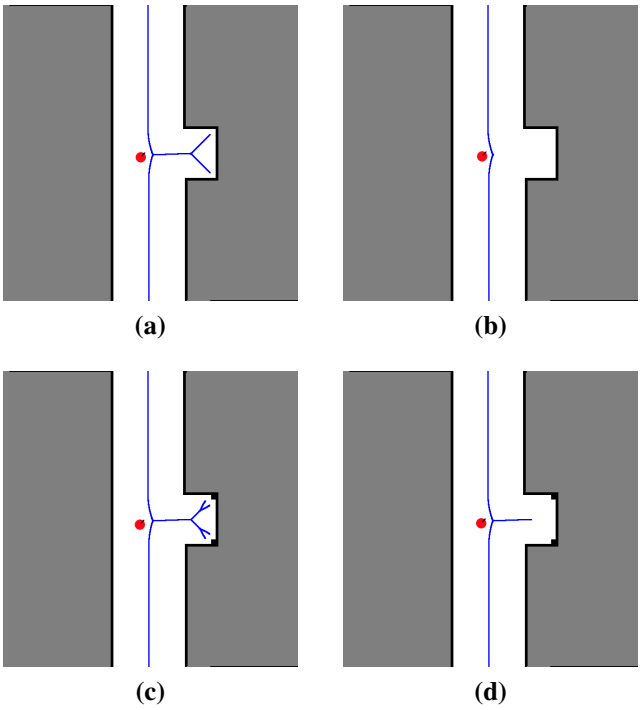


Fig. 5. **Eliminating spurious junctions.** Voronoi graphs are sensitive to noise. This may cause junctions, thus topological places, where they are not wanted. **(a)** Here we see a common example of an alcove along a hallway. **(b)** By reducing the Voronoi graph to the RGVG, "spurious" junctions can be eliminated. **(c,d)** The RGVG proposed by Choset and Nagatani [18] performs fixed depth pruning, leaving spurious junctions in certain noisy environments.

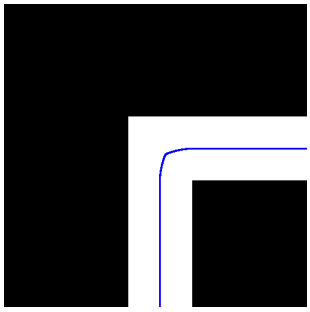


Fig. 6. **The RGVG at an un-noisy 'L' intersection.** In this example of a perfect local model, there is no junction after the spurs are pruned from the graph.

exactly M units away from the closest obstacle. More formally, given that the GVG is the set of all points p equidistant from two or more distinct obstacles O, we define the set of points P in the EVG as follows:

$$P = \{p : \exists o, o' \in O, \text{ where } o \neq o' \wedge dist(p,o) = dist(p,o') \wedge dist(p,o) \leq M \wedge \forall o'' \in O\ dist(p,o'') \geq dist(p,o)\} \cup \{q : \exists o \in O, \text{ where } dist(q,o) = M \wedge \forall o' \in O\ dist(q,o') \geq M\},$$

where M is a preset maximum threshold.

Although this definition is simple, the algorithm for computing the connected graph in a discretized occupancy grid is rather complex due to many special case scenarios. A full description and analysis of this algorithm is planned as future work. Fig. 3(b) shows the EVG at a location

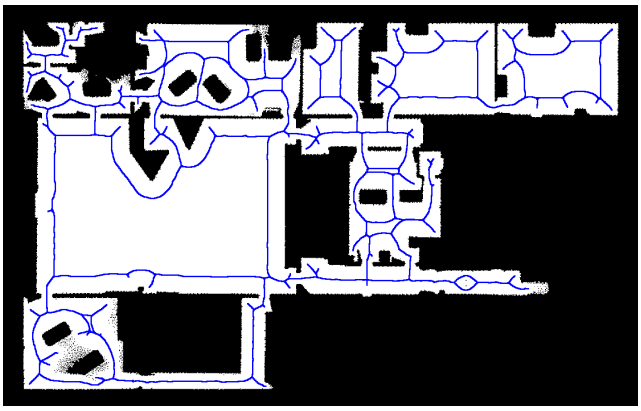


Fig. 7. **The extended Voronoi graph of a global metrical map.** Using an occupancy grid, the extended Voronoi graph can be drawn by using any non-free cells as obstacles. The graph now parallels walls in large rooms. Junction points occur at intersections in corridor environments and at corners in large rooms.

where the robot enters a large room. The EVG of the same pre-computed global metrical map from Fig. 1 is shown in Fig. 7.

V. Pruning with the Spanning Tree

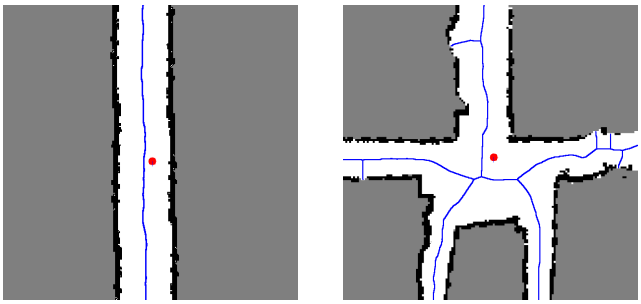
As noted earlier, Voronoi graphs are susceptible to noise, creating spurious junctions and branches. The RGVG [18] was introduced as a way to eliminate some of these "weak" junction points; however, the algorithm proposed cannot handle all unwanted junctions (Fig. 5(c,d)). The problem with pruning the Voronoi graph is that it is often difficult to determine what branches are part of the "core" skeleton and which branches are non-essential.

In order to determine the core skeleton of a Voronoi graph, we assume that the robot is computing a Voronoi graph in a small, local model of its immediate surround. We define a small-scale-space model of the local surround as a *local perceptual map (LPM)*. In this work, the LPM is simply an occupancy grid centered at the robot's location in the world. As information scrolls off the map, it is no longer considered for purposes of place detection (or place recognition or topological map-building [3]).

Given the small, manageable size of the LPM, the robot should never find itself completely enclosed by occupied cells in large-scale environments. Exceptions could be elevators or small rooms where doors get closed. In these instances, our LPM should detect the dynamic doors [23], allowing the algorithm to ignore these as obstacles for the purposes of building the Voronoi graph.

Thus, there is always some region of free cells that touch the edge of the occupancy grid or some cells of "unknown occupancy" (grey cells in the figures) that may provide a way away from the current location (see the occupancy grids in Figs. 5 and 8). We define any terminal point of the Voronoi graph that reaches the edge of the LPM or reaches unexplored cells to be an *exit*.

Given exits, we can define the branches of the Voronoi graph that contain exits to be "core" branches. Instead of actively pruning away branches, a better approach is to include only the union of all shortest paths that connect



(a) (b)

Fig. 8. **The REVG from the robot's point-of-view.** Given that robots have a limited sensory horizon, the Voronoi graph must be constantly computed in a local, scrolling occupancy grid centered on the robot. By eliminating branches that do not connect the edges of the grid, all spurious junctions and branches are eliminated, resulting in the reduced extended Voronoi graph.

each exit to all other exits in the LPM. We define this graph as the *reduced extended Voronoi graph (REVG)*. Fig. 8 shows how the minimal spanning tree eliminates all spurious junctions and branches in these small-scale models.

In the scenario where only a single exit exists (i.e. dead-ends), the REVG computes the spanning tree that connects the exit to the point on the EVG closest to the robot.

VI. Robustly Detecting Places

Above, we demonstrated how we can extend Voronoi graphs to be useful for both corridor and non-corridor navigation. We also showed how to eliminate unimportant information from the graph by focusing on the core subgraph that connects the branches that lead away from the current small-scale frame of reference.

Unfortunately, this does not mean that the robot can always detect places and paths using junction points and incident branches. There are still places and environmental setups where junction points do not exist (see Figs. 4 and 6). Additionally, there are some locations where noise caused by objects or pedestrians can create junction points along otherwise uninteresting paths: Voronoi graphs are not stable and can change due to sensor noise or dynamic worlds. Finally, there are some intersections complex enough to create multiple, nearby junction points (Fig. 8(b)).

Below we outline a new algorithm for place detection based on the Voronoi graph. It allows places to occur even when junction points do not exist in the graph, and places can encompass multiple junction points in certain scenarios. The work below uses the Voronoi graph, grounded in an LPM, to determine *gateways* and *path fragments*, which will lead to paths and places.

A. Gateways

Chown et al. define *gateways* as the locations of major changes in visibility. *"In buildings, these [gateways] are typically doorways.... Therefore, a gateway occurs where there is at least a partial visual separation between two neighboring areas and the gateway itself is a visual opening to a previously obscured area. At such a place, one has the option of entering the new area or staying in the previous area."* [24, page 32]

In this work, we describe an algorithm for gateways that satisfies this definition at places, but the algorithm also works in non-place regions. For a given robot position (always near the center of the LPM), we define a set of gateways that cuts edges on the spanning tree, separating the robot from the exits. Thus, a gateway is a boundary between different regions of the environment. It divides the environment into regions "near the robot" and "away from the robot."

Gateways can be grounded in an LPM by using the Voronoi graph. Given a pruned Voronoi graph (an RGVG or REVG), the method for finding gateways is as follows:

- l is the location of the physical robot w.r.t the LPM.
- c is the Voronoi point closest to l.
- J is the set of Voronoi junctions j.
- r_p is the distance from Voronoi point p to the closest obstacle.
- $K = \{j \in J : dist(j, l) \leq r_j\}$ (the robot is within the "radius" of the junction point).
- Two junctions j and j' are neighbors if $dist(j, j') <= max(r_j, r_{j'})$.
- F is the transitive closure of neighboring junctions, using K as the starting set of junctions. This defines the "core" of the place neighborhood.
- If $F = \emptyset$ then $F = \{c\}$.
- Q is the set of all Voronoi points q, where $\exists f \in F$ s.t. $dist(f, q) = r_f \wedge \forall f' \in F\ dist(f', q) \geq r_{f'}$. This selects the set of points on the border of the "core" of the place.
- For each $q \in Q$, walk along the branch that contains q in the direction away from the "core" of the place (i.e. F). Look for a point p that corresponds to a *constriction*.
  In Figs. 9 and 10, a constriction was defined as a local minimum in $R = \{r_p : p \in P\}$. Due to noise in the LPM, this local minimum search should account for minute variations in R, e.g. by utilizing a smoothing filter. (More recently, we have switched to using the minimum absolute value over the derivatives of $r_p \in R$ to define constrictions.)
- At each of these constrictions m, g is defined as a line segment of length $2 \cdot r_m$, centered on m, oriented normal to the branch at point m.

A recursive algorithm was implemented that corresponds to the formalism above. The algorithm runs quickly enough to recompute gateways in real-time. This algorithm was used to produce Figs. 9(b-d). Not only does it define gateways at non-intersections, but gateways are also defined to conglomerate locations in complex intersections that may include more than one Voronoi junction point (Fig. 9(d)).

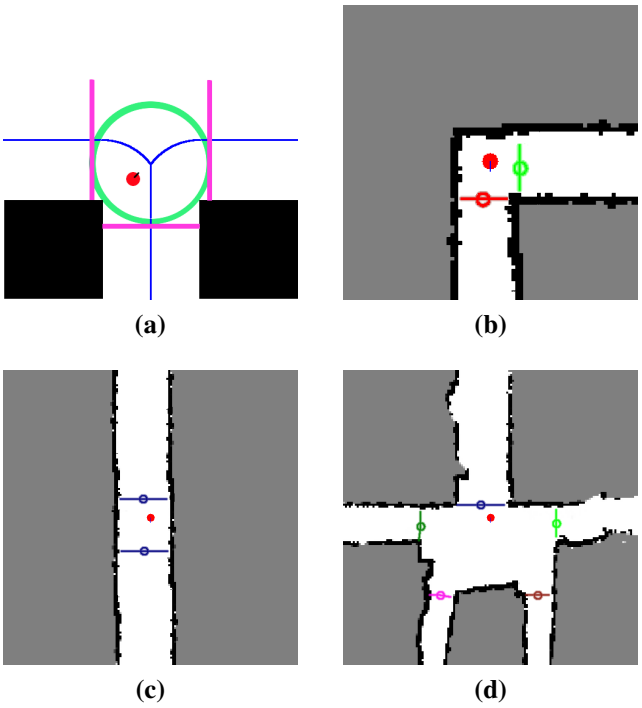


Fig. 9. **Real-world gateways.** The gateway algorithm uses the Voronoi graph to find the "boundaries" of a location. **(a)** At simple places, the gateways are tangents to the inscribing circle centered on the junction point. **(b,c)** Even at locations with no junction points, the gateway algorithm works. **(d)** At complex places, gateways may bound several close junction points.

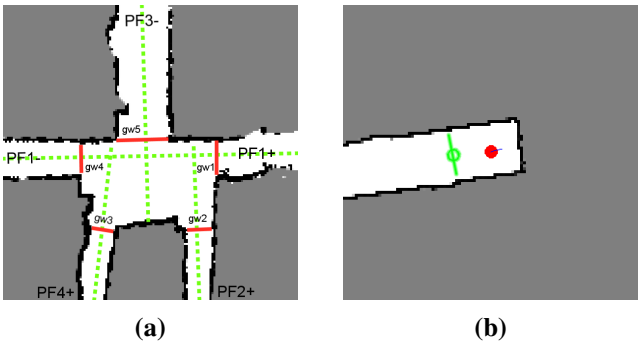


Fig. 10. **Path fragments. (a)** The robot has a criterion to decide whether two gateways belong to the same path fragment. A path passes through the small-scale map of the local surround if its path fragment has two gateways. **(b)** At dead-ends, there is only one gateway, thus only one path fragment.

B. Path Fragments

Once gateways are found, the robot can then determine the local *path fragments*: portions of large-scale topological paths that are grounded in the LPM. Each gateway is associated with exactly one path fragment, while each path fragment is associated with either one or two gateways. If a path fragment terminates at a place, it will have only one gateway. If it passes through the small-scale-space map, it will have two. Path fragments are illustrated in Fig. 10.

The robot is required to have a procedure for determining whether a path fragment passes through the local area or terminates near the robot. In our implementation, the criterion for path continuity is that for each of two gateways, after arrival at one, the clear unique default travel

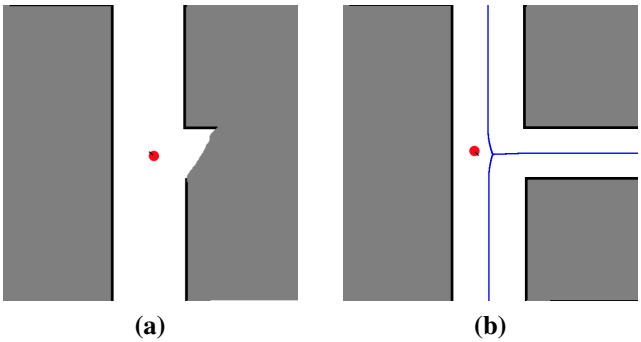


Fig. 11. **Ensuring reliable place detection. (a)** Due to a limited field of view, a robot may not have a complete model of the surrounding environment. This can create gateways and path fragments that may not exist in a fully explored LPM. **(b)** By moving to investigate unexplained space within the LPM, the robot can "fill out" the occupancy grid, ensuring that structurally unimportant alcoves (as seen in Fig. 5(b)) are distinguished from actual paths.

continuation is the other. The "clear unique default" used to make Fig. 10 and used in previous experiments [3] requires that some ray normal to one gateway intersects only the other and vice versa.

C. Defining Places

Using the notions of gateways and path fragments, we can formulate a robust criterion for defining places. As the robot moves along a path, there are exactly two gateways, and, according to the path continuity criterion, one path fragment (Fig. 9(c)). If either of these conditions ever changes, the robot has entered a place.

At intersections, the number of path fragments is strictly greater than one. This is true even for the 'L' intersection (Fig. 9(b)), which was ignored by the RGVG place detection method of Choset and Nagatani [18]. At dead-ends there is only one path fragment, but only one gateway (the one which leads away from the dead-end), thus dead-ends are seen as places (Fig. 10(b)).

Initial experiments show that this place detection method works very well when the LPM is fully explored. Occasionally, the robot may not observe specific pieces of the local surround (e.g. when using a 180° laser). This is illustrated in Fig. 11(a). Since Voronoi branches that touch unknown space are not pruned from the REVG, this causes gateways to appear along these branches. In our initial implementation, our robot simply spins around when it sees more than one path fragment. This allows the robot, which has only a 180° field of view, to better observe the local surround, recalculate the Voronoi graph, and check whether there actually is a place at the location (Fig. 11(b)) or if the environment is qualitatively a simple path (Fig. 5(b)).

VII. Conclusion

Abstraction from the continuous world into a finite set of easily detectable places is not only important for topological map-building but is also a crucial step in building up the foundations of common-sense reasoning. Previously, any mobile robot performing autonomous topological navigation used environment/sensor specific algorithms to detect places based on ad hoc criteria. An exception to this is the

work done by Choset and colleagues, which uses Voronoi graphs to ground paths and places in the robot's sensory inputs.

We showed how to overcome problems of previous methods using a set of simple, yet powerful mechanisms that lead to robust, autonomous place detection with no *a priori* knowledge of the environmental characteristics. The innovation is as follows:

- Maintain a scrolling metrical model of the local surround in order to build a reliable local Voronoi graph. We call this a local perceptual map (LPM).
- Extend the Voronoi graph to have upper distance limits so that the graph stays close to obstacles. (Lower limits are easily added in order to keep the robot from considering exits too small for it to pass through.)
- Find the minimal spanning tree that connects the exits of the Voronoi graph. Exits are points that touch the edge of the LPM or touch unexplored portions of the LPM.
- For a given robot position (always near the center of the LPM), define a set of gateways that cuts edges on the spanning tree, separating the robot from the exits.
- Define path fragments given these gateways.
- Define a place to be a region of reachable space around the robot that is bounded by gateways and obstacles, unless there are exactly two gateways and exactly one path-fragment. (In the latter case, it is just a location along a path.)

ACKNOWLEDGMENT

We would like to thank Joseph Modayil for his contributions to the research and implementation of gateways and path fragments.

REFERENCES

[1] G. Lakoff and M. Johnson, *Metaphors We Live By*. Chicago: The Univ. of Chicago Press, 1980.

[2] E. Remolina and B. Kuipers, "Towards a general theory of topological maps," *Artificial Intelligence*, vol. 152, pp. 47–104, 2004.

[3] B. Kuipers, J. Modayil, P. Beeson, M. MacMahon, and F. Savelli, "Local metrical and global topological maps in the hybrid Spatial Semantic Hierarchy," in *Proc. IEEE Int. Conf. on Robotics and Automation (ICRA)*, 2004.

[4] B. Kuipers and P. Beeson, "Bootstrap learning for place recognition," in *Proc. of the 18th National Conf. on Artificial Intelligence (AAAI-2002)*, Edmonton, Canada, 2002.

[5] K. Lynch, *The Image of the City*. Cambridge, MA: MIT Press, 1960.

[6] J. L. Gould, "Honey bee cognition," *Cognition*, vol. 37, pp. 83–103, 1990.

[7] P. Stopka and D. W. Macdonald, "Way-marking behaviour: an aid to spatial navigation in the wood mouse (Apodemus sylvaticus)," *BMC Ecology*, vol. 3, no. 3, 2003, http://www.biomedcentral.com/1472-6785/3/3.

[8] T. Guilford, S. Roberts, D. Biro, and I. Rezek, "Positional entropy during pigeon homing II: navigational interpretation of bayesian latent state models," *Theoretical Biology*, vol. 227, no. 1, pp. 25–38, March 2004.

[9] A. W. Siegel and S. H. White, "The development of spatial representations of large-scale environments," in *Advances in Child Development and Behavior*, H. W. Reese, Ed. New York: Academic Press, 1975, vol. 10.

[10] A. A. Kalia and B. J. Stankiewicz, "Acquisition of object and structural landmark knowledge in virtual environment navigation," in *Object Perception and Memory*, 2002.

[11] V. Aginsky, C. Harris, R. Rensink, and J. Beusmans, "Two strategies for learning a route in a driving simulator," *Journal of Environmental Psychology*, vol. 17, pp. 317–331, 1997.

[12] H. Heft, "The role of environmental features in route learning: Two exploratory studies of way finding," *Environmental Psychology and Nonverbal Behavior*, vol. 3, pp. 172–185, 1979.

[13] R. L. Rivest and R. E. Schapire, "Inference of finite automata using homing sequences," in *Proceedings of the 21st Annual ACM Symposium on Theoretical Computing*. ACM, 1989, pp. 411–420.

[14] K. Basye, T. Dean, and J. S. Vitter, "Coping with uncertainty in map learning," *Machine Learning*, vol. 29, 1997.

[15] H. Shatkay and L. P. Kaelbling, "Learning topological maps with weak local odometric information," in *Proc. of the 15th Int. Joint Conf. on Artificial Intelligence (IJCAI-97)*. San Mateo, CA: Morgan Kaufmann, 1997, pp. 920–927.

[16] H. Choset and J. Burdick, "Sensor-based exploration: the hierarchical generalized Voronoi graph," *Int. Journ. of Robotics Research*, vol. 19, no. 2, pp. 96–125, 2000.

[17] H. Choset, S. Walker, K. Eiamsa-Ard, and J. Burdick, "Sensor-based exploration: incremental construction of the hierarchical generalized Voronoi graph," *Int. Journ. of Robotics Research*, vol. 19, no. 2, pp. 125–148, 2000.

[18] H. Choset and K. Nagatani, "Topological simultaneous localization and mapping (SLAM): toward exact localization without explicit localization," *IEEE Transactions on Robotics and Automation*, vol. 17, no. 2, pp. 125–137, April 2001.

[19] H. P. Moravec, "Sensor fusion in certainty grids for mobile robots," *AI Magazine*, Summer 1988.

[20] A. Elfes, "Occupancy grids: A probabilistic framework for robot perception and navigation," Ph.D. dissertation, Carnegie Mellon Univ. 1989.

[21] B. J. Kuipers and Y.-T. Byun, "A robot exploration and mapping strategy based on a semantic hierarchy of spatial representations," *Journ. of Robotics and Autonomous Systems*, vol. 8, pp. 47–63, 1991.

[22] N. Roy, W. Burgard, D. Fox, and S. Thrun, "Coastal navigation – mobile robot navigation with uncertainty in dynamic environments," in *Proc. IEEE Int. Conf. on Robotics and Automation (ICRA)*, Detroit, MI, May 1999.

[23] J. Modayil and B. Kuipers, "Bootstrap learning for object discovery," in *Proc. of the Conf. on Intelligent Robots and Systems (IROS)*, 2004.

[24] E. Chown, S. Kaplan, and D. Kortenkamp, "Prototypes, location, and associative networks (PLAN): Towards a unified theory of cognitive mapping," *Cognitive Science*, vol. 19, no. 1, pp. 1–51, 1995.