Incremental Construction of the Saturated-GVG for Multi-Hypothesis Topological SLAM

Tong Tao, Stephen Tully, George Kantor and Howie Choset

Abstract— **The generalized Voronoi graph (GVG) is a topological representation of an environment that can be incrementally constructed with a mobile robot using sensor-based control. However, because of sensor range limitations, the GVG control law will fail when the robot moves into a large open area. This paper discusses an extended GVG approach to topological navigation and mapping: the saturated generalized Voronoi graph (S-GVG), for which the robot employs an additional wall-following behavior to navigate along obstacles at the range limit of the sensor. In this paper, we build upon previous work related to the S-GVG and provide two important contributions: 1) a rigorous discussion of the control laws and algorithm modifications that are necessary for incremental construction of the S-GVG with a mobile robot, and 2) a method for incorporating the S-GVG into a novel multi-hypothesis SLAM algorithm for loop-closing and localization. Experiments with a wheeled mobile robot in an office-like environment validate the effectiveness of the proposed approach.**

I. Introduction

For almost any navigational task, a mobile robot will require a map of the environment in which to plan its path to a goal location. There are different types of maps in which a mobile robot can plan a path: topological maps, grid-based maps, and metric feature maps. The advantage of using topological maps is their ability to concisely represent an environment with a graph, whose vertices are interesting "places" and whose edges are feasible paths between them. Additional advantages of topological maps are their computational efficiency, their reduced memory requirements, and their lack of dependence on metric positioning.

The *generalized Voronoi graph* (GVG) is a specific type of topological map that has successfully been applied to mobile robot navigation and mapping [1]–[4]. The GVG is comprised of meet-points (vertices) that correspond to locations of three-way equidistance (or more) to obstacles in the environment and edges that correspond to paths of two-way equidistance to obstacles that are also feasible paths between meet-points [1] (see Fig. 1-(a)).

The advantage of the GVG as a topological map is its convenience for exploration and obstacle avoidance. Since the embedded edges of the GVG by definition maximize the distance between the robot and its surrounding obstacles, it is inherently safe to follow the GVG edges when navigating in the map. Unfortunately, one drawback of the GVG as a map is its sensor range limitation. When the robot traces an edge of the GVG, it must keep a sufficient number of obstacles inside its sensor range. If the robot moves out of its sensor range from the nearby obstacles and into a large open space, the robot may become lost (see Fig. 1-(b)).

To solve the sensor limitation problem, the *saturated generalized Voronoi graph* (S-GVG) can be constructed instead. The S-GVG has already been discussed in several other works with notable success [5], [6]. The S-GVG includes an additional type of embedded graph edge that corresponds to the locus of points at a saturated distance from an obstacle in the free space of the environment (see Fig. 1-(c)).

In this paper, we introduce a new incremental method to construct the S-GVG of an environment with a robot that has omnidirectional range sensing. Our implementation adapts proven control laws that were designed for the conventional GVG to the mapping process for the S-GVG. We also present a novel implementation of a multi-hypothesis topological simultaneous localization and mapping (SLAM) approach that is customized for the S-GVG. The result is an efficient and robust exploration and mapping method for sensing-limited mobile robots.

II. Related Work

Early research involving Voronoi diagrams relied on methods that required full knowledge of the environment. For example, in [7], an offline retract method is used to compute the Voronoi diagram. Choset et. al. later proposed a sensor-based planning approach with a mobile robot to incrementally construct the GVG using control laws without *a priori* knowledge [1], [3]. Then, to solve the problem of weak or unstable meet-points, Nagatani et. al. introduced in [8] a simplified version of the GVG, called the reduced generalized Voronoi graph (R-GVG).

Lisien et. al in [4] propose a hierarchical map where the GVG serves as a high-level representation of the environment that organizes a collection of lower level feature-based submaps. This hierarchical map was later used for efficient topological localization [9] and multi-hypothesis SLAM [10] in large-scale environments. We note that many of the ideas we present in this paper can be easily extended to a hierarchical approach via the inclusion of additional metric information along the edges in the S-GVG topological graph.

This work was supported in part by NSF of China under Grant 60605021, NSF of China under Grant 60805031, NSF of Tianjin under Grant 10JCYBJC07600 and the Ph.D. Programs Foundation of Ministry of Education of China under Grant 200800551015.

T. Tao is a PhD student of the College of Information Technical Science, Nankai University, Tianjin 300071, China, and a visiting student of the Robotics Institute, Carnegie Mellon University, Pittsburgh, PA 15213, USA. TaoTong2006@gmail.com

S. Tully is with the Electrical and Computer Engineering Department and G. Kantor and H. Choset are with the Robotics Institute at Carnegie Mellon University, Pittsburgh, PA 15213, USA. {stully@ece, choset@cs, kantor@ri}.cmu.edu

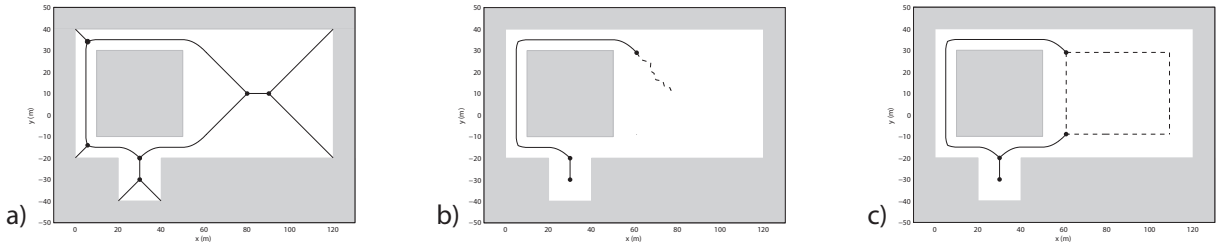

Fig. 1. A comparison between different GVG approaches: (a) GVG with full knowledge, (b) sensor-based incremental GVG construction, and (c) incrementally constructed S-GVG.

Huang et. al. and Beeson et. al. discuss the saturated generalized Voronoi diagram similar to the topology used in this paper in [5] and [6] respectively. The former was one of the first to define the saturated behavior and unsaturated behavior for a mobile robot traversing GVG edges while the latter refers to the S-GVG as the extended Voronoi diagram. Both papers investigate the S-GVG as a solution to a robot's sensor range limitations. While important in many respects, the work in [5] assumes rectilinearity of the environment (which may not always be the case). Both of the papers do not discuss a means for the incremental construction of such a graph with specific control laws necessary to navigate the S-GVG. Additionally, they do not apply the S-GVG to full-fledged SLAM with loop-closing, which is a topic we are addressing in this paper.

For topological SLAM, the challenging task for the robot is to perform vertex matching or loop-closing. The robot must be able to determine whether a vertex in the graph is newly explored or revisited. In [11], Nagatani suggests using several stable features to identify a vertex. The approach, however, does not consider multiple hypotheses and performs loop-closing with simple yes/no heuristics. In [12], a Rao-Blackwellized particle filter is implemented over the space of topologies. In [10], we propose a similar probabilistically grounded multi-hypothesis technique that builds a map/state hypothesis tree for topological SLAM. The SLAM method we are proposing in this paper for the S-GVG is an extension of our previous work and customizes the SLAM algorithm for dealing with a combination of both saturated and conventional GVG regions.

III. Incremental Construction of the S-GVG

While navigating the conventional GVG, the robot traces GVG edges with a control law and homes into the exact location of the meet-point. The robot will explore other GVG edges and detect additional GVG meet-points, recursively, until all meet-points and edges have been explored. Following this procedure, the GVG can be incrementally constructed with a mobile robot using sensor-based control [3] (one caveat being the recognition of loop-closure in an ambiguous environment, which we discuss in Sec. IV).

The S-GVG, on the other hand, will contain regions in which the edges follow two-way equidistance as per usual (conventional GVG edges) and regions in which the edges follow an obstacle at a saturated distance (saturated GVG edges). We refer to a meet-point whose associated edges are all conventional GVG edges as a conventional GVG meet-point. Likewise, we refer to a meet-point whose edges include at least one saturated GVG edge as a saturated GVG meet-point. Fig. 1-(c) shows the saturated edges (the dashed lines) and the conventional edges (solid lines).

Incrementally constructing the S-GVG is similar to creating the conventional GVG. The difference is when a robot is traversing a hallway that leads into a large open space. In this case, a different control law must be used to home into a meet-point with two-way equidistance instead of three-way equidistance. Also, when departing a saturated meet-point onto a saturated edge, a new control law must be used to follow the obstacle boundary at the saturated distance.

A. Control Law

Choset et. al. introduced a control law for generating the GVG [3], [13]. The control law performs sensor-based planning by controlling the heading of the robot in a way that merges the prediction and correction phases so as to avoid the jagged path generated by traditional control methods. According to his work, at a point x in the neighborhood of the interior of a GVG edge, the robot steps in the direction,

$$\dot{x} = \alpha \text{Null}(\nabla G(x)) + \beta (\nabla G(x))^{\dagger} G(x), \tag{1}$$

where $G(x)$ is the difference of the distances to nearby obstacles. In the planar case, $G(x)$ can be written as,

$$G(x) = [d_1(x) - d_2(x)]. \tag{2}$$

In this formulation, the solutions of $G(x) = 0$ form the GVG path, the distances to nearby obstacles ($d_1(x)$ and $d_2(x)$) are the local minima of the range sensor, and $(\nabla G(x))^{\dagger}$ is the Penrose pseudo-inverse of the Jacobian $\nabla G(x)$, i.e.,

$$(\nabla G(x))^{\dagger} = (\nabla G(x))^T (\nabla G(x)(\nabla G(x))^T)^{-1}. \tag{3}$$

In [3], Choset proves that (1) produces a heading direction for the robot that converges onto the GVG path when $\beta < 0$.

The conventional GVG edge maximizes the distance to each obstacle. For saturated GVG edge tracing, the distance of the robot to the obstacle should equal the sensor range r. This choice maximizes the distance of the robot to the obstacle up to the limit where the boundary of the obstacle can still be sensed. This is demonstrated in Fig.1-(c): the distance between the dashed line and the wall will be r.

In order to take advantage of the control law in (1), we introduce a *virtual minimum*, which simulates the presence of another obstacle on the other side of the robot when the robot

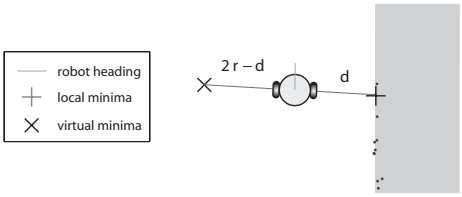

Fig. 2. A virtual minimum for tracing the S-GVG.

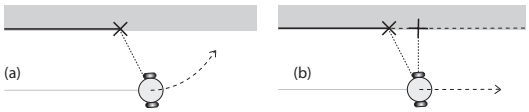

Fig. 3. Line fitting: × is sensed minimum and + is corrected minimum.

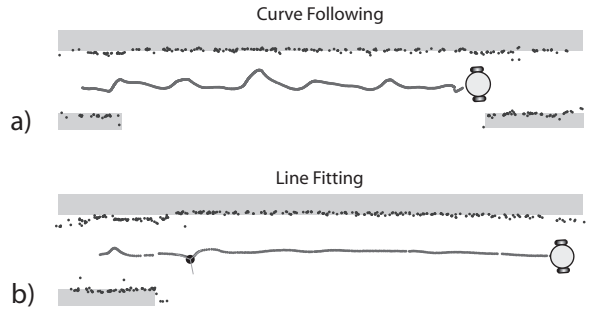

Fig. 4. The effect of line-fitting for the S-GVG tracing procedure.

is tracing a saturated GVG edge. We denote the distance associated with the current local minima of the range sensor as $d(x)$, which is the distance to the true obstacle boundary. We additionally define the distance to the virtual minima to be $d_v(x) = 2r - d(x)$. All three points (the location of the true minima, the robot location, and the location of the virtual minima) are all assumed to be colinear. In Fig. 2, we show the local minima (represented by "+") and the virtual minima (represented by "×").

By having a virtual minima, we can represent the control problem in the same formulation as the conventional GVG tracing problem,

$$G_s(x) = [d(x) - d_v(x)]. \tag{4}$$

It follows that,

$$G_s(x) = 2[d(x) - r]. \tag{5}$$

In (5), we show that by creating the virtual minimum, we are controlling the robot to keep a distance r away from the nearest obstacle. This is the desired result for obstacle following for the tracing process with a saturated GVG edge.

Because we have formulated this control problem in the same way as the conventional GVG tracing problem (compare (4) with (2)), we can apply the same control law as in (1) to optimally follow a smooth path at a saturated distance. Although this is not the only way to control a robot to perform boundary following at a fixed distance from an obstacle, we feel that it is an important feature of our algorithm that a proven and previously implemented component of robot software can be incorporated to handle this new tracing control problem.

B. Line-Fitting

A robot with noisy sonar and/or a cheap laser sensor often has a delay between when a robot passes by an obstacle and when its boundary is sensed by the sensor. With the S-GVG, a delay in sensor data can cause significant problems when tracing the obstacle boundary at the saturated distance. As in Fig. 3-(a), for a robot whose sensed minima is trailing the robot due to a sensing delay, the distance between the robot and the obstacle will undesirably increase when driving forward. This will cause a control response to steer the robot towards the wall to reduce the error. Then, when additional sonar points are eventually acquired, the robot will immediately steer away from the wall to correct back to the saturated distance. This unstable behavior will then repeat due to the delay in the control system. The effect of the sensing delay and the control response can be seen in Fig. 4-(a).

To solve this problem, we introduce a line-fitting procedure to improve the performance of the S-GVG tracing algorithm. In most structured environments, the obstacles we encounter have straight edges (or at least can be approximated with several line segments locally). Thus, we are able to use the history of sensed minima to fit a line, which we extend forward past the robot, to simulate the as-of-yet unsensed portion of the obstacle (as the dashed line in Fig. 3-(b)). From this line, we can compute a more probable range minimum (as the + symbol in Fig. 3-(b)). We use a least square approach to fit the line and then compute the corrected local minima by projecting the robots position onto the fitted line. In addition, we use a clustering algorithm and an error metric to test whether the measured obstacle points conform to the computed line parameters. If not, we simply trust the uncorrected sensed minimum.

To show the effectiveness of this approach, we performed the S-GVG tracing process in the same environment with (and without) the line fitting procedure. Fig. 4-(a) shows the robot tracing the wall without line fitting and Fig. 4-(b) shows the result when applying line fitting. The path is straight and follows the wall at the saturated distance, which allows for stable mapping of the S-GVG.

C. Meet-Point Detection

With the saturated GVG, a saturated meet-point can occur in two ways as in Fig. 5. The first occurs when the robot is tracing a conventional GVG path and the equidistance to the two obstacles increases until it is equal to r. This indicates that if the robot were to continue tracing the conventional GVG path, it would lose sight of the obstacle (as in Fig. 5-(a)). The second occurs when the robot is tracing a saturated GVG path, and it encounters another obstacle. Since the robot just encountered this obstacle, the distance to this obstacle is equal to the sensor range r (as in Fig. 5-(b)).

For both of the two conditions, the saturated meet-point is a point which is equidistant to the two closest obstacles

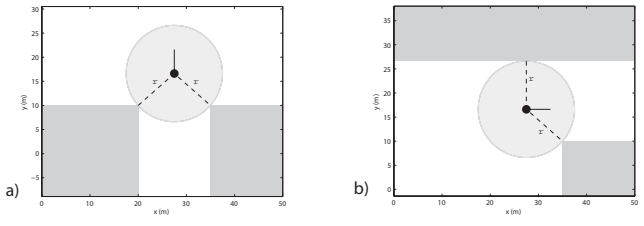

Fig. 5. The two types of saturated GVG meet-points

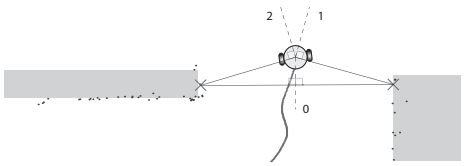

Fig. 6. Departure angle determination.

(and the equidistance value is r). Since the meet-points in the GVG topological graph are important for successful mapping, the robot must precisely locate itself at the meet-points. We use a meet-point homing process that will control the robot to converge to the meet-point (as in [3]).

The control law for homing is similar to (1). Here, the function G is defined,

$$G(x) = \begin{bmatrix} d_1(x) - r \\ d_2(x) - r \end{bmatrix} = 0 \tag{6}$$

After homing, the robot is at the meet-point and must determine the departing angles for the edges that emanate from the current meet-point. With a saturated meet-point, there will be two local minima in the robot's omnidirectional range sensor. The bisector of the two lines drawn to the sensed minima will be one of the departing edges (in this case a conventional GVG edge). The angles of the two additional edges are perpendicular to each of the lines drawn to the sensed minima, away from the direction that leads towards the conventional GVG edge. Fig. 6 shows how the robot determines the departing angles of the three edges. In the figure, the two "×" symbols represent the sensed local minima, the three long dashed lines indicates the direction of each edge emanating from the meet-point.

After computing the departing angles of the new edges, the edge which the robot came from is then marked as *explored*. If this is a new meet-point, the robot will start to trace a new unexplored edge. If this is an old meet-point and all of the edges have been explored, the robot will use a branch search to find the nearest unexplored edge to trace for complete exploration of the environment.

IV. Topological SLAM using the S-GVG

Although we discussed incrementally building a topological map in the previous section and how to control the robot to navigate the S-GVG, we did not yet describe how the robot can perform vertex matching in this type of environment. In ambiguous maps, the robot must be able to detect loop-closures while mapping the S-GVG. This section introduces a topological SLAM approach that handles loop-closing with a multi-hypothesis filter. This type of SLAM approach is an extension of previous work [10], although we are presenting here, for the first time, a version of multi-hypothesis SLAM that is specific to the S-GVG.

A. Constructing a Hypothesis Tree

Each hypothesis h in our multi-hypothesis SLAM approach stores a possible topological graph, G_k^h, and a possible robot state, $X_k^h = (v_k^h, \alpha_k^h)$ (which stores the meet-point at which the robot is currently located, v_k^h, as well as the edge from which the robot arrived at that meet-point, α_k^h). The subscript k represents the time-step.

Ideally, at every time-step k, we would like to compute the possible map/state pairs, one of which will be correct and will incorporate the correct loop-closing and meet-point matching decisions. To do this, we maintain a hypothesis tree where each level of the tree corresponds to a different time-step. The tree structure we maintain is similar to that in [10], [14], [15].

At the start of an experiment, we assume the robot has no information except for the degree of the first meet-point it sees, δ_0, which equals the number of edges emanating from the meet-point. Therefore, we initialize the root node of our hypothesis tree as follows: $h = 0$, $k = 0$, $v_k^h = 0$, and $\alpha_k^h = 0$. All neighbors of the root node are labeled as *unexplored*.

During an experiment, the robot is continuously moving between meet-points. At each time-step k, the robot chooses a motion input u_k in order to transition to another meet-point. The motion input is a relative offset from the previous arrival edge, and produces the following departure edge β_k for a new hypothesis that is spawned from hypothesis h.

$$\beta_k = (\alpha_{k-1}^h + u_k) \bmod \delta_{k-1} \tag{7}$$

After departing along edge β_k, the robot drives to a new meet-point and then detects the number of edges emanating from that meet-point, which is stored as the degree δ_k.

After performing a new motion input u_k, all leaf nodes of the hypothesis tree must be expanded (the leaf nodes being the set of hypotheses at time-step $k-1$). Our algorithm expands all leaf nodes of the hypothesis tree in the following way. If the neighbor of v_{k-1}^h that is associated to the departing edge β_k is not *unexplored*, then we copy the hypothesis to a single child hypothesis but move the robot's state to the new meet-point. If the neighbor is *unexplored*, then the algorithm considers several possibilities: 1) that the robot traverses the unexplored edge and arrives at a new meet-point, or 2) that a loop is closed and the robot arrives at a previously visited meet-point via one of its unexplored edges. Different hypotheses are spawned for these cases. Fig. 7 demonstrates the expansion of the hypothesis tree.

B. Computing the Posterior Probability

In order to solve the problem of topological SLAM for the S-GVG, we must determine which hypotheses among the leaf nodes of the hypothesis tree are likely to represent the true state and the true map. To do this, we compute the posterior probability of each hypothesis given a sequence

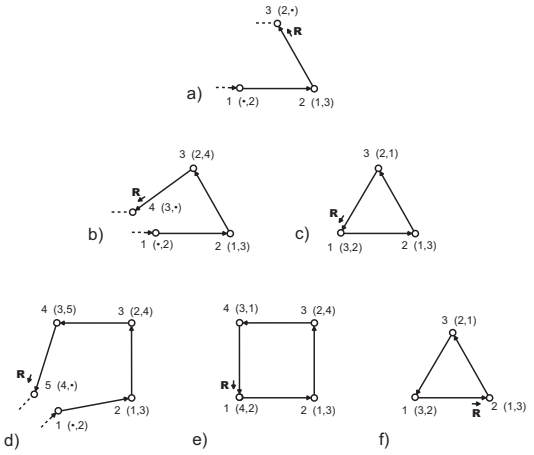

Fig. 7. This is an example of expanding the hypothesis tree due to robot motion. Hypothesis (a) spawns (b) and (c) after one edge traversal. After another edge traversal, hypothesis (b) spawns (d) and (e) while hypothesis (c) spawns only (f). The location of **R** in the figure marks the robot's state.

of sensor measurements. The hypothesis that better fits the sensor data will produce a higher probability measure and is therefore more likely to represent the true state and map.

During time-step k, the robot leaves the previous meet-point, traverses an edge in the graph, and arrives at a new meet-point (either saturated or conventional). A measurement z_k^e is obtained during the edge traversal (such as the distance travelled as measured by wheel odometry) and a measurement z_k^v is obtained when the robot arrives at the new meet-point (such as the range measurement for the equidistance to the nearby obstacles). The posterior probability of a hypothesis is $p(X_k^h, G_k^h | z_{0:k}, u_{1:k})$, where, as before, X_k^h and G_k^h represent the robot's state and graph respectively. Additionally, $z_{0:k} = (z_{0:k}^v, z_{1:k}^e)$ is the collection of all measurements during the experiment, which includes the edge measurement sequence, $z_{1:k}^e$, as well as the meet-point measurement sequence, $z_{0:k}^v$. The sequence $u_{1:k}$ represents the motion inputs through time-step k.

The posterior can be computed using Bayes law,

$$\begin{aligned} & p(X_k^h, G_k^h | z_{0:k}, u_{1:k}) \\ &= \eta \; p(z_{0:k} | X_k^h, G_k^h, u_{1:k}) \; p(X_k^h, G_k^h | u_{1:k}) \\ &= \eta \; p(z_{0:k} | X_k^h, G_k^h, u_{1:k}) \; p(G_k^h | u_{1:k}), \end{aligned} \tag{8}$$

where $p(z_{0:k} | X_k^h, G_k^h, u_{1:k})$ is the measurement likelihood and $p(X_k^h, G_k^h | u_{1:k})$ is a prior on the hypothesis. The prior reduces to $p(G_k^h | u_{1:k})$ in (8) because the robot correctly performs the motion input sequence. The scalar value η in Eq. 8 is to normalize over the space of possible hypotheses.

For a given time-step, we can compute the posterior probability of the new leaf nodes of the tree using Eq. 8. To reduce storage and computation, the likelihood term of a new hypothesis h' can be computed recursively given the likelihood of the parent hypothesis h, i.e.,

$$\begin{aligned} & p(z_{0:k} | X_k^{h'}, G_k^{h'}, u_{1:k}) \\ & \quad = p(z_k^e, z_k^v | z_{0:k-1}, X_k^{h'}, G_k^{h'}, u_{1:k}) \; p(z_{0:k-1} | X_{k-1}^h, G_{k-1}^h, u_{1:k-1}). \end{aligned}$$

It turns out that the S-GVG inherently stores additional information that is useful for SLAM. The robot is completely aware of the fact that it has homed into either a saturated meet-point or a conventional meet-point. The robot can distinguish between the two types because the chosen control laws are different and the number of equidistant obstacles will be different. Therefore, the detection of the meet-point type (saturation versus conventional) is a useful feature for proposing or rejecting loop-closure hypotheses.

Thus, to customize the implementation of multi-hypothesis SLAM for the S-GVG, we purposefully fabricate an extra measurement at each meet-point probability update that represents the detected type of meet-point: saturated or conventional. The meet-point measurement is therefore $z_k^v = [\varepsilon_k^v, s_k^v]$, where ε_k^v is the measured equidistance and s_k^v is the detected meet-point type (saturated or conventional). The measurement model for the detected meet-point type is,

$$\begin{aligned} p(s_k^v &= saturated | X, G) \begin{cases} \alpha \approx 1.0 & T(X,G) = saturated \\ 1 - \alpha & T(X,G) \neq saturated \end{cases} \\ p(s_k^v &\neq saturated | X, G) = 1 - p(s_k^v = saturated | X, G), \end{aligned}$$

where $T(X,G)$ is the meet-point type assigned to the meet-point in question according to the hypothesis, X and G. The idea is that if the meet-point is labeled as saturated, the likelihood of the robot detecting the meet-point as a saturated GVG vertex is nearly one.

This measurement model is then incorporated into the likelihood update for the posterior probability computation. By adding this measurement to the likelihood term, we can greatly penalize a hypothesis that expects the robot to be in a conventional region of the graph when the robot is currently homing into a meet-point with two-way equidistance at the range limit of its sensors (in a saturated region).

Neglected thus far in our discussion is the prior $p(G_k^h | u_{1:k})$ in Eq. 8. This term represents, without any sensor information, the probability that the robot happens to be placed in an environment with a topology G_k^h. What should this distribution be? There is no way to know the right answer. But we can do better than a uniform distribution. We use the following distribution for experiments,

$$p(G_k^h | u_{1:k}) \propto \exp\left(-N_k^h \log k\right),$$

which favors smaller maps that still explain the measurement data properly. This makes sense, because we would like to prevent over-fitting when testing loop-closing hypotheses. It turns out that this formulation is equivalent to using the Bayesian information criterion [16] for model selection.

By combining in Eq. 8 the prior developed here with the recursive likelihood function, we are effectively trying to capture the perfect balance between concise maps that are typical for structured environments and large intricate maps that better fit the data. Lastly, we incorporate a conservative pruning step, based on thresholding the posterior probability of a topological map hypothesis, that improves the efficiency of our multi-hypothesis SLAM algorithm.

The result is a SLAM algorithm that uses the meet-point type as information towards testing loop-closing hypotheses.

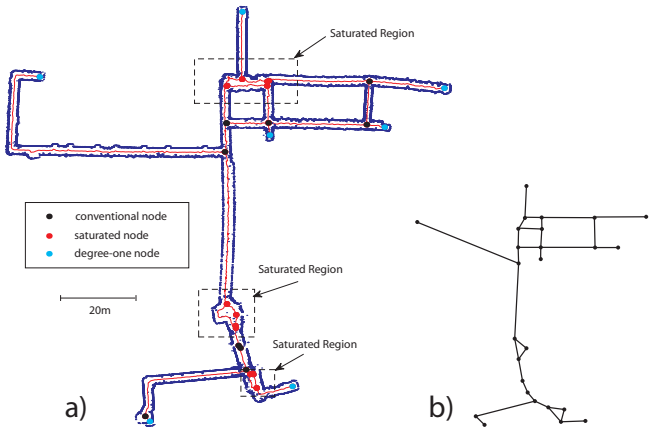

Fig. 8. This is the experimental result for our S-GVG SLAM algorithm.

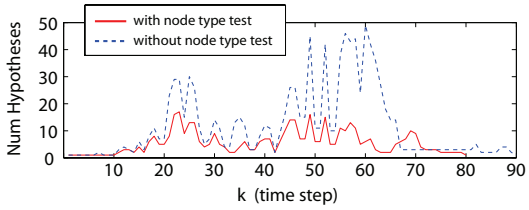

Fig. 9. This plot shows the number of hypotheses in our SLAM filter for an S-GVG experiment.

The algorithm also incorporates traditional measurements to help resolve the posterior probability over topological maps so that the SLAM process will find the most likely solution.

V. Evaluation

We have performed extensive testing of our S-GVG incremental construction algorithm. We are using a Nomad Scout robot with an omnidirectional sonar array for sensing range in an office-like environment (Wean Hall and Newell Simon Hall at Carnegie Mellon University). The robot primarily navigates in the hallways, but intermittently encounters large regions that extend beyond the sensing range.

In Fig. 8-(a), we show the environment in which the robot performed a SLAM experiment. In this experiment, the robot navigated three different saturated regions in which the layout of the floor plan was too large for the sensor range while also traversing many conventional GVG edges. If the robot only used conventional GVG edge tracing approach, it would fail when it moved into the saturated regions. We note that this experiment was created over several days for convenience and the data was post-processed at a later date.

For this experiment, the robot traced the GVG quite well, as seen in the result (Fig. 8-(a)). The robot also mapped the correct topology for the environment, as seen in Fig. 8-(b), which precisely compares to the true environment. We attribute this to the success of the multi-hypothesis SLAM algorithm at detecting loop-closure despite ambiguities.

In Fig. 9, we depict the number of hypotheses that were maintained by our SLAM filter during the experiment. The robot observed ambiguous information, due to structural similarity between different regions in the environment, and had to consider different hypotheses about loop-closure after several edge traversals in the environment. We note that, as time progresses, the number of hypotheses converges to 1, with the only remaining hypothesis equaling the correct true map/state pair. The experiment was run twice (once while using the detected meet-point type and once without).

VI. Conclusion

The Generalized Voronoi Graph (GVG) is a popular topological representation for roadmap based path planning and sensor-based navigation. When the robot navigates in a large open area, though, the conventional GVG approach fails due to sensor range limitations. This paper investigates the saturated generalized Voronoi graph (S-GVG) and provides a rigorous method for the incremental construction of the S-GVG with a sensor-limited mobile robot. Our contributions include a method for incrementally constructing the S-GVG, which includes a control law for following the obstacle boundary and a line fitting technique for smoothing the sensor-based control of the robot. We also introduce a novel loop-closing method that considers multiple hypotheses when performing SLAM with the S-GVG.

References

[1] H. Choset and J. Burdick, "Sensor based planning, part I: The generalized voronoi graph," in *Proceedings of the 1995 IEEE International Conference on Robotics and Automation*, 1995.

[2] ——, "Sensor based planning, part II: Incremental construction of the generalized voronoi graph," in *Proceedings of the 1995 IEEE International Conference on Robotics and Automation*, 1995.

[3] H. Choset, I. Konukseven, and A. Rizzi, "Sensor based planning: A control law for generating the generalized voronoi graph," in *Proc. of the 1997 IEEE Intl. Conf. on Advanced Robotics*, 1997.

[4] B. Lisien, D. Morales, D. Silver, G. Kantor, I. Rekleitis, and H. Choset, "The hierarchical atlas," *Robotics, IEEE Transactions on*, vol. 21, no. 3, pp. 473–481, June 2005.

[5] W. Huang and K. Beevers, "Complete topological mapping with sparse sensing," *Rensselaer Polytechnic Institute, Technical Report 05-06*, March 2005.

[6] P. Beeson, N. Jong, and B. Kuipers, "Towards autonomous topological place detection using the extended voronoi graph," in *Proc. of the 2005 IEEE Intl. Conf. on Robotics and Automation*, 2005.

[7] C. O'Dúnlaing and C. Yap, "A retraction method for planning the motion of a disc," *The Journal of Algorithms*, vol. 6, no. 1, pp. 104–111, 1985.

[8] K. Nagatani and H. Choset, "Toward robust sensor based exploration by constructing reduced generalized voronoi graph," in *Proc. of the 1999 IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems*, 1999.

[9] S. Tully, H. Moon, D. Morales, G. Kantor, and H. Choset, "Hybrid localization using the hierarchical atlas," in *Proc. of the 2007 IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems*, October 2007.

[10] S. Tully, G. Kantor, H. Choset, and F. Werner, "A multi-hypothesis topological slam approach for loop closing on edge-ordered graphs," in *Proc. 2009 IEEE/RSJ Intl. Conf. on Int. Robots and Systems*, 2009.

[11] K. Nagatani, H. Choset, and S. Thrun, "Towards exact localization without explicit localization with the generalized voronoi graph," in *Proc. of the 1998 IEEE Intl. Conf. on Robotics and Automation*, 1998.

[12] A. Ranganathan and F. Dellaert, "Online probabilistic topological mapping," *The International Journal of Robotics Research*, 2010.

[13] H. Choset, K. Lynch, S. Hutchinson, G. Kantor, W. Burgard, L. Kavraki, and S. Thrun, *Principles of Robot Motion: Theory, Algorithms, and Implementations*. MIT Press, Cambridge, MA, 2005.

[14] G. Dudek, P. Freedman, and S. Hadjres, "Using local information in a non-local way for mapping graph-like worlds," *Proc. 3rd International Conference on Artificial Intelligence*, pp. 1639–1645, 1993.

[15] E. Remolina and B. Kuipers, "Towards a general theory of topological maps," *Artificial Intelligence*, vol. 152, no. 1, pp. 47–104, 2004.

[16] G. Schwarz, "Estimating the dimension of a model," *The Annals of Statistics*, vol. 6, pp. 461–464, 1978.