Factoring the Mapping Problem: Mobile Robot Map-building in the Hybrid Spatial Semantic Hierarchy

Patrick Beeson
Department of Computer Sciences
University of Texas at Austin
1 University Station C0500
Taylor Hall 2124
Austin, TX, USA 78712-0233
pbeeson@cs.utexas.edu

Joseph Modayil
Department of Computer Science,
University of Rochester,
PO Box 270226
734 Computer Studies Bldg.
Rochester, NY, USA, 14627-0226
modayil@cs.rochester.edu

Benjamin Kuipers
Department of Electrical Engineering and Computer Science,
University of Michigan,
2260 Hayward Street,
Ann Arbour, MI 48109-2121, USA
kuipers@cs.utexas.edu

Abstract

We propose a factored approach to mobile robot map-building that handles qualitatively different types of uncertainty by combining the strengths of topological and metrical approaches. Our framework is based on a computational model of the human cognitive map; thus it allows robust navigation and communication within several different spatial ontologies. This paper focuses exclusively on the issue of map-building using the framework.

Our approach factors the mapping problem into natural subgoals: building a metrical representation for local small-scale spaces; finding a topological map that represents the qualitative structure of large-scale space; and (when necessary) constructing a metrical representation for large-scale space using the skeleton provided by the topological map. We describe how to abstract a symbolic description of the robot's immediate surround from local metrical models, how to combine these local symbolic models in order to build global symbolic models, and how to create a globally consistent metrical map from a topological skeleton by connecting local frames of reference.

KEY WORDS—mapping, localization, autonomous agents, cognitive robotics

The International Journal of Robotics Research
Vol. 29, No. 4, April 2010, pp. 428–459
DOI: 10.1177/0278364909100586

Figures 1, 8–13, 15, 16 appear in color online: http://ijr.sagepub.com

1. Introduction

A map is a description of an environment allowing an agent – a human, or in our case a mobile robot – to plan and perform effective actions. From a single location, an agent's sensors can not observe the whole structure of a complex, large environment. For this reason, the agent must build a map from observations gathered over time and space. We distinguish between *large-scale space*, with spatial structure larger than the agent's sensory horizon, and *small-scale space*, with structure within the sensory horizon.

Most metrical approaches to mobile robot map-building define a single, global frame of reference in which to create the map. Range measurements are used to perform probabilistic inference about the location of features or about the occupancy of discretized cells in the map (Thrun et al. 2005). Existing simultaneous localization and mapping (SLAM) methods are highly effective for building local metrical models of

small-scale space and for providing reliable localization in the frame of reference of the local map; however, maintaining global consistency over large-scale environments is difficult, particularly when closing large loops in the environment. A popular approach is to use particle filters, where each particle represents a hypothesized exploration trajectory. The researcher must hope that with enough particles the distribution will include one that closes the loop correctly. Since the space of trajectories can be enormous, this hope is often optimistic.

The fundamental problem is representational: loop-closing hypotheses are alternative topological structures for the map, not alternative metrical structures. To be able to solve complex, multi-hypothesis loop-closing problems in a tractable manner, the robot must reason with symbolic topological maps. The space of metrical maps in a single frame of reference does not appropriately represent the states of incomplete knowledge that arise during exploration and map-building in complex, large-scale environments.

Our factored mapping framework is based on the Spatial Semantic Hierarchy (SSH) (Kuipers 2000, 2008), which uses multiple coordinated representations for knowledge of large-scale space. The *Hybrid SSH* (HSSH) (Kuipers et al. 2004; Beeson 2008) extends the basic SSH by including representations for small-scale space and defining the relationship between large-scale and small-scale spatial representations. Symbolic topological mapping methods such as the SSH provide a concise representation for the structural alternatives that arise in investigating loop closures. Topological maps provide the ability to store and access multiple local maps with separate frames of reference and topological connections annotated with weak metrical constraints. By separating small-scale from large-scale space, we postpone the problem of coordinating the local frames of reference until the global structure of the topological map has been identified. At that point, the global metrical map can be constructed, efficiently and accurately.

Therefore, our approach factors the mapping problem into four natural sub-goals: (1) building a metrical representation for local small-scale spaces; (2) detecting places and determining their symbolic descriptions; (3) finding a topological map representing the qualitative structure of large-scale space; and (4) constructing a metrical representation for large-scale space in a single global frame of reference, building on the skeleton provided by the topological map. While the global metrical map is useful for some purposes, it is worth noting that many autonomous planning and navigation goals can be achieved effectively using only the global topological map and/or the local metrical maps. Therefore, this approach to hybrid mapping is more robust than one that extracts topological relations from a global metrical map that must be built first (Thrun and Bücken 1996).

The multiple representations of the HSSH are described independently, while their semantic dependencies imply that they build on each other. However, this does *not* imply a simple serial processing pipeline. In fact, processing of sensory input to build representations of the different kinds is interleaved, providing various sorts of synergies. Two are particularly important. First, the local metrical map of small-scale space is a useful "observer" both for detecting and describing places and for low-level control with obstacle avoidance. Second, it may be useful to order candidate topological models by using the relative displacement of nearby places or even by using the global layout of places within a single frame of reference. Nonetheless, in order to clarify the distinct representational ontologies, we will describe them in this paper as though they operate independently.

The HSSH improves mobile robot capabilities in a variety of ways: efficient and robust map-building and navigation, "natural" human–robot interaction due to the multiple representations of space (Beeson et al. 2007), and hierarchical control. This paper cannot cover the full breadth of benefits obtained from using a hybrid topological/metrical framework; thus, this paper focuses solely on the issue of using the HSSH framework for map-building. Here we describe the HSSH theory and demonstrate key points of HSSH map-building using a particular implementation that focuses on perception using range sensors, although other sensory modalities can also be utilized in the HSSH framework – the hybrid, hierarchical framework is largely independent of the sensors used to create the local metrical model of small-scale space (cf. Murarka et al. (2006)). A more detailed description of the HSSH benefits to control, place detection/description, and human–robot interaction are discussed by Beeson (2008).

2. Background

2.1. Metrical Mapping

Powerful probabilistic methods have been developed for range-sensing mobile robots to perform SLAM within a single frame of reference (Thrun et al. 2005). These methods are accurate and reliable for online incremental localization within local neighborhoods. Sensing with sufficiently high frequency relative to local motion guarantees large overlap between successive sensory images. Current sensory information can be compared to the current map in order to improve localization. By analogy with radar signal interpretation, finding the correct match between observations and a model is called the *data association* problem. After improved localization occurs, the sensory information is used to update the map for the next SLAM iteration. In local regions, many data association problems, such as the closing of large loops, can be excluded. The absence of large loops means that the problem of large-scale *structural ambiguity* does not arise in the local metrical map.

While metrical SLAM methods work in small spaces, they do not extend well to larger environments. Global metrical maps become more expensive to update and access without

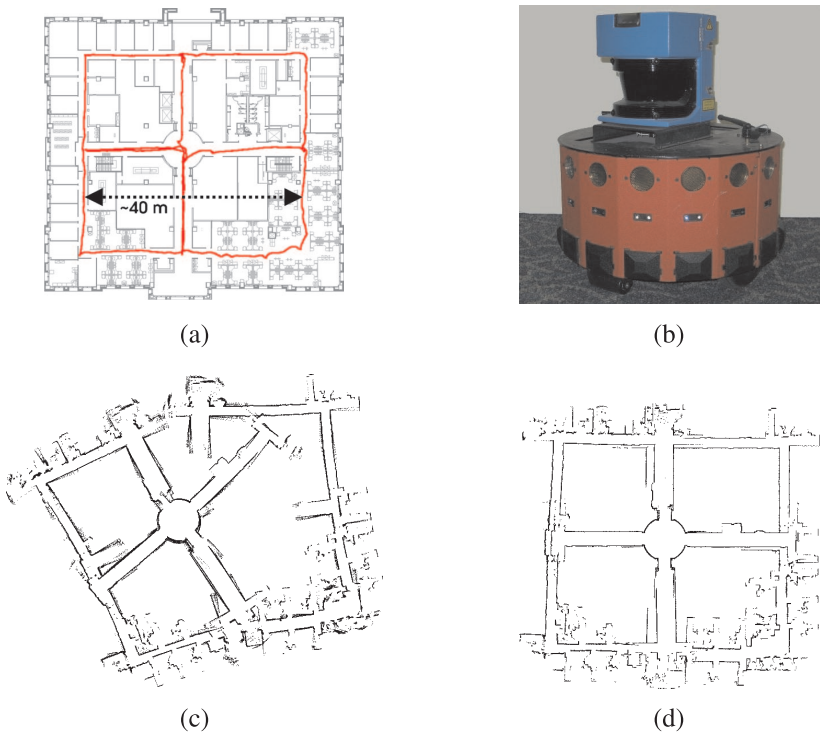


Fig. 1. Closing large loops reveals problems with cumulative errors when attempting to build metrical maps of large-scale environments in a single global frame of reference. (a) This environment and the robot's trajectory through it are used as an example throughout. (b) The data comes from a Magellan Pro research robot with differential-drive odometry and a SICK-brand lidar device for precise, planar range-sensing. (c) This robot-made map of the environment in image (a) shows the effect of accumulated raw odometry error. (d) This map shows the improvement in pose estimation over image (c) by using metrical SLAM methods, but it also shows that significant errors still accumulate with respect to the real environment.

clever storage schemes. More important is the difficulty that arises when closing large loops (Figure 1). Even with local SLAM methods that use perception to improve the accuracy of localization, odometry error accumulates in the relation between the map's global frame of reference and the ground-truth reference frame of the real-world environment. This global error becomes even more pronounced in environments with long paths that have few distinguishing features. Without proper data association along paths, localization often drifts from the ground-truth, both in the robot's distance along the path and in the robot's heading, causing straight paths to compress, stretch, or curve in the map.

There are *ad hoc* methods for hypothesizing loop closures when the global odometry error is small. When a large loop is closed, accumulated error will often result in the robot's current observations clashing with older portions of the map. Methods exist that search for a nearby pose in the older portions of the map where perceptions match the prediction (propagating detected global error backwards through the exploration trace) (Lu and Milios 1997; Hähnel et al. 2003a); however, these solutions can fail in sufficiently large or complex environments. For example, Cummins and Newman (2008) discuss closing loops over kilometers of travel, where small rotational errors lead to large positional errors, and the correct loop closure may never be considered by odometry-based solutions. Additionally, if the environment is subject to *perceptual aliasing* (different locations look the same), then the matching process may close the loop incorrectly, distorting the map as a whole. Depending on the amount of symmetry in the environment, a single incorrect match can lead the mapping agent down an arbitrarily long "garden path" before the error is discovered. It is still unclear how probabilistic methods applied to metrical maps can properly discover an incorrect map and how they might efficiently backtrack to hypothesize a different loop closure (Hähnel et al. 2003b).

Some research on map-building avoids loop-closing issues by explicitly assuming that the correct data association is known (Leonard and Newman 2003; Paskin 2003). In some cases, even without an explicit assumption about data association, impressive feats of large-scale map-making depend on locations in the environment being sufficiently distinguishable based on local cues (Montemerlo et al. 2002; Konolige 2004).

Others accept false negative matches in order to avoid false positives, sometimes improperly hypothesizing that a previously visited location is a new place (Bosse et al. 2003). This can eliminate the possibility of closing a loop correctly and finding the correct map, which leads to poor planning and navigation performance. In a rich environment with noise due to dynamic changes, it could be that every location is in principle distinguishable, but it is difficult or impossible to know which features identify the place, and which are noise. Methods created to distinguish between perceptually aliased states can get confused under scenarios of *perceptual variability* (the same place looks different on separate occasions) (Kuipers and Beeson 2002) causing a single physical location to be represented multiple times in the same map.

Early approaches to probabilistic localization and mapping used particles to represent a distribution over robot poses for localization, but a single shared map was updated from the maximum-likelihood pose hypothesis (Thrun et al. 2000b). This could produce an incoherent map due to an incorrect and premature commitment to a maximum-likelihood pose hypothesis that turned out to be incorrect. A more principled approach uses Rao-Blackwellized particle filters to explicitly represent the distribution of trajectories and maps by maintaining multiple metrical map hypotheses (Montemerlo et al. 2003; Eliazar and Parr 2003; Hähnel et al. 2003a). These methods are run offline (due to computational demands) after exploration is completed, forgoing useful *active exploration* techniques capable of eliminating some loop-closing hypotheses. Additionally, in large, symmetric environments, intractably large numbers of particles may be required to avoid particle depletion when closing large loops. Particle depletion is a failure to have a particle in the distribution that adequately models the correct map.

2.2. Topological Mapping

Topological mapping is the other major paradigm studied in mobile robotics. Cognitive map research supports the creation of topological maps of large, complex environments (Lynch 1960; Siegel and White 1975; Kuipers 1978; Chown et al. 1995; Kuipers 2000). A topological map, in its most basic form, represents an environment as a graph where nodes represent places and edges represent connections between places. Several groups of robotics researchers have presented distinct topological implementations that differ in their semantics for the graphs – how they define/describe places and actions between them (Kuipers and Byun 1991; Mataric 1992; Shatkay and Kaelbling 1997; Duckett and Nehmzow 1999; Choset and Nagatani 2001; Morris et al. 2005). Some implementations build topological maps autonomously, some are given topological maps *a priori* (Koenig and Simmons 1996), and some let the robot explore autonomously while the researcher provides place names to overcome perceptual aliasing issues (Kortenkamp and Weymouth 1994).

Topological maps are more compact representations than global metrical maps, allowing efficient large-scale planning. Additionally, since the environment is discretized into a graph, movement errors do not accumulate globally. Possibly the most important difference for future robotics research is that topological maps allow compact, efficient hierarchical models that support multi-level symbolic reasoning for robust navigation, planning, and communication.

The major hurdle for topological map-building has been the reliable abstraction of useful symbols from continuous, noisy perceptions of the environment, i.e. how to reliably detect and recognize places and paths. This is an instance of the more general *symbol grounding* problem (Harnad 1990) that has troubled the Artificial Intelligence (AI) community for many years. Probabilistic approaches are good at overcoming the kinds of local uncertainty and systematic noise that can hinder reliable symbol extraction. Incorporating probabilistic data association techniques into the topological map-building paradigm has sparked interest in hybrid map-building, including the HSSH approach presented in this paper.

2.3. Hybrid Approaches

Metrical and topological representations for space are very different in character, or, more precisely, in ontology. The topological map describes the structure of large-scale space. It abstracts away the specific nature of sensory input and the specific methods used for matching sensory images when the topological map is created. Metrical mapping techniques that rely on local overlap of successive sensations, on the other hand, precisely capture the structure within the local sensory horizon: small-scale space.

Recently, robotics researchers have begun to look at hybrid topological/metrical representations in order to try to leverage the benefits of both approaches. There are too many hybrid implementations to mention here, many with only very subtle differences, but publications about hybrid metric/topological representations fall into three basic categories, all of which are addressed by the work in this paper. We refer the reader to the survey of specific hybrid mapping implementations by Buschka (2005), as we refer to well-known or prototypical examples in this discussion.

In one category of hybrid map-building approaches, a robot uses local metrical models as local *observers* that help filter out sensor noise, aggregate observations over time, and create plans that avoid nearby obstacles. Much of this research is specifically interested in using metrical models to try to determine qualitatively distinct or interesting places as the robot explores a new environment (Yeap and Jefferies 1999; Lankenau et al. 2002; Tomatis et al. 2002; Ko et al. 2004). This is related to our work on grounding places and paths in local metrical models.

The second category of hybrid approaches focuses on using "places" in order to reduce the number of locations in

the world that must be considered when hypothesizing metrical loop closures. That is, the goal of most hybrid mapping techniques is still to achieve a global metrical map; however, they use some "topological" (i.e. graph) constraints to make the closing of loops more efficient. Many of these implementations record places arbitrarily (Duckett and Saffiotti 2000; Zimmer 2000; Blanco et al. 2008), e.g., every 5 m traveled, in order to reduce the number of locations in the world where loop closures can occur. Others use a feature buffer, so the robot creates a new place at every n corners or wall segments (Bosse et al. 2003). Some approaches simply have the researchers press a button to define places in the world (Thrun et al. 1998). Our research is related to these approaches as well, in that, given autonomous place detection at qualitatively distinct and metrically distant places, we can provide a compact graph representation of an environment that makes global metrical mapping extremely efficient.

Finally, a third category of hybrid mapping has only recently been investigated. There has been research looking at modeling the full Bayesian distribution over topological hypotheses (Ranganathan et al. 2006; Blanco et al. 2008). These methods are still strongly grounded in using odometry knowledge (which can be unreliable over large distances (Cummins and Newman 2007)) and/or aligning raw lidar measurements. As mentioned above, the "places" they utilize are determined by *ad hoc* means: by the researcher via button presses or by using distance thresholds or finite-sized feature buffers. We believe our topological representation is useful here as well. Section 10.2 discusses how this new area of hybrid research should mesh with the HSSH framework.

The *Hybrid Spatial Semantic Hierarchy* is, to our knowledge, the first framework and implementation to fully describe the process of going from metrical sensations to both metrical and symbolic models of both small-scale and large-scale space – moving from metrical models of small-scale space to symbolic representations of small-scale space, inferring large-scale structure via symbolic inference, before producing a consistent global metrical model from the symbolic structure. Our approach autonomously detects and describes qualitatively distinct places, creating far fewer places than other "hybrid" approaches. Additionally, these places are meaningful to humans as the SSH representations are inspired by human cognitive maps.

3. The Spatial Semantic Hierarchy

This section overviews the basic SSH (Kuipers 2000; Remolina and Kuipers 2004; Kuipers 2008).[1] The concepts and notation introduced here will be important for extension to the HSSH, which is presented starting in Section 3.5.

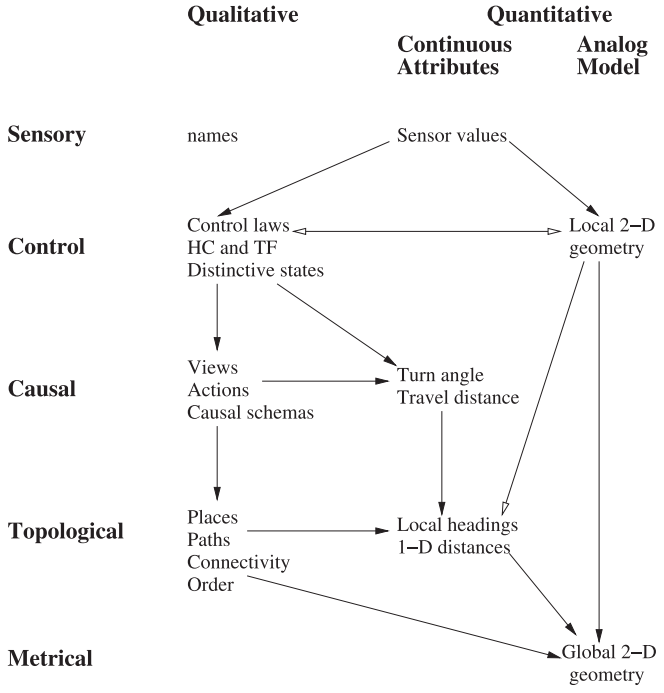


Fig. 2. The Spatial Semantic Hierarchy. Closed-headed arrows represent dependencies; open-headed arrows represent potential information flow without dependency. (From Kuipers (2000).)

The SSH represents knowledge of large-scale space with four distinct representations. Figure 2 illustrates the framework. At the SSH Control Level, control laws provide reliable motion among *distinctive states* (dstates) q_i. At the SSH Causal Level, state–action–state *schemas* $\langle q, a, q' \rangle$ explain how the distinctive states are linked by turn and travel *actions*, and relations $o(q) = v$ between a state and its observable *view* describe the potential experiences of the robot. Thus the Causal Level abstracts the continuous world to a deterministic finite automation (DFA) (Rivest and Schapire 1989; Dean et al. 1995), related to the way humans utilize route instructions in navigation. At the SSH Topological Level, a map consisting of discrete places, paths, and regions, describes the connectivity, order, containment, and boundary relations of large-scale environments. At the SSH Metrical Level, local metrical information about the location of obstacles, the magnitudes of actions, the lengths of path segments, and the directions of paths at place neighborhoods are incorporated into local and global metrical maps. One contribution of the HSSH is to clarify the relation between the metrical information and the symbolic abstractions of the basic SSH levels.

The SSH factors spatial uncertainty into distinct components, controlled in distinct ways. *Movement uncertainty* is

1. In order to combine the SSH theory with a probabilistic mapping framework, it is necessary to use a slightly different vocabulary and symbol set than in previous SSH publications.

controlled by the behavior of feedback-driven motion control laws. *Pose uncertainty* is controlled in the basic SSH by hill-climbing to dstates (and in the HSSH by incremental localization within a local metrical map). *Structural ambiguity* about the large-scale topology of the environment is controlled by search in a space of alternative topological maps. *Global metrical uncertainty* is controlled by relaxing metrical information from separate local frames of reference into a single global frame of reference, guided by the topological map.

3.1. The SSH Control Level

The SSH Control Level describes the system consisting of the agent and its environment as a piecewise continuous dynamical system. The agent's experience is represented as a fine-grained sequence of time-steps $0 \leq t \leq N$. At any time t, the agent–environment system is described by the state vector x_t (the agent's *pose* in a static world), the agent's sense vector z_t, and its motor vector u_t. We assume that both the environment and the agent's sensory system are very rich, so the sense vector z_t is very high-dimensional.

The dynamical system is described by the following equations, in which the functions F and G represent the physics of the agent's body in the environment and its sensorimotor system, respectively. These two functions are not explicitly known or available to the agent. The control law H_i, on the other hand, can be selected by the agent:

$$x_{t+1} = F(x_t, u_t),$$

$$z_t = G(x_t),$$

$$u_t = H_i(z_t).$$

The agent acts by selecting a control law H_i to determine its motor output signals as a function of its sensor input. In the basic SSH (Kuipers 2000), motion is controlled by alternating between two types of controllers. *Trajectory-following* control laws take the robot from one *distinctive state* (dstate) to the neighborhood of another. A *hill-climbing* control law guides the robot to the destination dstate $\bar{x}$ from anywhere in its surrounding neighborhood.

Hill-climbing localizes the agent by moving it reliably to a distinctive state within the local neighborhood, preventing the accumulation of position error, and paving the way for a discrete abstraction of the continuous space. Furthermore, hill-climbing control makes very weak assumptions about the properties of the sensors and the agent's knowledge of those properties. For example, a robot may hill-climb to a distinctive state, or follow a trajectory down a hallway, based on features extracted from sonar or laser range-finders, from monocular or stereo vision, or from sensors for luminance or electromagnetic fields. The robot's map is determined by the behaviors of its hill-climbing and trajectory-following control laws. It need never know how its sensory features correspond with environmental states.

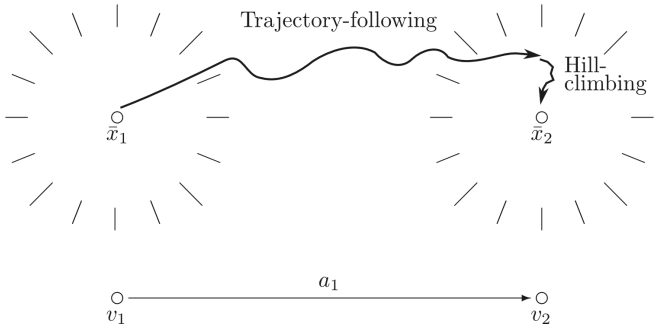


Fig. 3. SSH Control/Causal Level abstraction. In the SSH, dstates are defined by pairs of trajectory-following and hill-climbing control laws. These sequences are abstracted into actions, and the observations at dstates are abstracted into views. (Adapted from Kuipers (2000).)

Despite their simple conceptual definition, hill-climbing control laws can be difficult to define and may vary across domains. An agent often *does* have stronger knowledge of the properties of its sensorimotor system, and physical motion to distinctive states seems awkward and unnecessary in light of that knowledge. A key insight behind the HSSH is that accurate localization in the small-scale space model of a place neighborhood can substitute for the physical motion of hill-climbing to a particular distinctive state in that neighborhood. In Section 4, we discuss how the HSSH exploits metrical knowledge of small-scale space to build *local perceptual maps* (LPMs) of place neighborhoods, within which localization is reliable and effective.

3.2. The SSH Causal Level

Given pairs of trajectory-following and hill-climbing controls that represent motion between neighboring dstates at the Control Level, we begin to represent the robot's experiences as a set of symbolic abstractions (Figure 3). First, we define an *action* $a \in A$ to represent a pair of trajectory-following (TF) and hill-climbing (HC) controls that connect dstates. Since the sensory image at a dstate $\bar{z} = G(\bar{x})$ is a point in a very high-dimensional space, it will, in general, never be experienced twice. We will therefore assume that each distinctive state $\bar{x}$ has an associated *view*, $o(\bar{x}) = v \in V$, which is an abstracted description of the sensory image $\bar{z}$.

The actual content of a view will depend on the properties of the environment and of the robot's sensors. Views could include such things as: the direction and distance to nearby obstacles as detected by a range sensor; color, texture, and category of nearby objects as identified in a camera image; the

number and identity of accessible wireless routers or cell towers; or any of a number of other sensory features. Kuipers and Beeson (2002) describe a bootstrap-learning method for learning a view representation suitable for high-performance place recognition; however, for this paper, we will not require that the observation function o be discovered autonomously. As discussed in Section 5, the HSSH defines views by extracting a specific symbolic description of the local environmental structure.

The SSH Causal Level describes the agent's experience as a deterministic finite automaton (DFA) (Rivest and Schapire 1989; Dean et al. 1995). The Causal Level DFA,

$$M^C = \langle Q, A, V, R, o \rangle,$$

consists of sets of states Q, actions A, observable views V, a transition function $R : Q \times A \rightarrow Q$, and an observation function $o : Q \rightarrow V$. As the robot travels from one distinctive state $\bar{x}$ to the next, its experience is an alternating sequence of views and actions. Some actions are turns, while others are travels:

$$v_0 \quad a_1 \quad v_1 \quad a_2 \quad v_2 \quad \cdots \quad v_{n-1} \quad a_n \quad v_n.$$

At the SSH Control Level, a view v_i is experienced only when the agent is at a distinctive state $\bar{x}_i$, so the view v_i is an observable manifestation of the distinctive state, $v_i = o(\bar{x}_i)$:

$$\begin{array}{ccccccccc} \bar{x}_0 & a_1 & \bar{x}_1 & a_2 & \bar{x}_2 & \cdots & \bar{x}_{n-1} & a_n & \bar{x}_n \\ | & & | & & | & & | & & | \\ v_0 & & v_1 & & v_2 & \cdots & v_{n-1} & & v_n. \end{array}$$

At the Causal Level, each state $q \in Q$ represents an equivalence class of distinctive states $\bar{x}$ in the physical world.[2] Two distinctive states $\bar{x}_i$ and $\bar{x}_j$ are equivalent if they represent different experiences of the same distinctive state $q \in Q$. (We use the notation $[\bar{x}_i] = [\bar{x}_j] = q$ for this.) The set Q of distinctive states thus represents a specific hypothesis about which experiences $\bar{x}_i$ represent repeated encounters with the same state q in the environment; that is, Q specifies *data association* for loop closures.

All distinctive states in the same equivalent class q must have the same view:[3]

$$[\bar{x}] = [\bar{x}'] \rightarrow o(\bar{x}) = o(\bar{x}').$$

Thus, $o(q)$ is well-defined, and we can write

$$q = q' \rightarrow o(q) = o(q').$$

The full sensory input from high bandwidth sensors in a realistically complex environment is so rich that sensory images will never match exactly. Views must be defined in terms of some observation function that allows the same dstate to be reliably detected on separate occasions. Thus, an experience with repeated states such as

$$\begin{array}{ccccccccc} q_0 & a_1 & q_1 & a_2 & q_2 & \cdots & q_0 & a_n & q_1 \\ | & & | & & | & & | & & | \\ v_0 & & v_1 & & v_2 & \cdots & v_{n-1} & & v_n \end{array}$$

can only be consistent if $v_{n-1} = v_0$ and $v_n = v_1$. However, abstracted views are subject to perceptual aliasing (different places look the same), leading to ambiguities about the topological structure of the map: $o(q) = o(q') \not\rightarrow q = q'$.

The *transition function* $R : Q \times A \rightarrow Q$ is represented as a set of *schemas* $r = \langle q, a, q' \rangle$, where $context(r) = q$, $action(r) = a$, and $result(r) = q'$. As new observations are added to the robot's experience, new schemas $\langle [\bar{x}_n], a_{n+1}, [\bar{x}_{n+1}] \rangle$ are learned by the transition function R. The causal map is constructed by searching for an appropriate set Q of states (i.e. equivalence classes of distinctive state observations), such that M^C has a deterministic transition function R, predicted and observed views are consistent, and M^C is consistent with the axioms for topological maps (Remolina and Kuipers 2004).

For the purpose of building the SSH Causal Level from exploration experience, building and using a DFA is far more tractable than building and using a probabilistic state model, such as a hidden Markov model (HMM).[4] The key benefit of a DFA over HMMs (or stochastic finite automata in general) are that both the transition function and the observation function are deterministic. The deterministic transition function follows from the nature of the abstraction that results from moving reliably between dstates via trajectory-following and hill-climbing control laws. The deterministic observation function follows from the abstraction that defines the observation function o. One improvement of the HSSH over the SSH is that local small-scale space models make place detection and observational classification of states deterministic without the need for hill-climbing (Section 5).

2. The term *distinctive state*, abbreviated *dstate*, is thus overloaded. It refers both to the state $\bar{x}$ resulting from a hill-climbing control law at the SSH Control Level, and to the state $q = [\bar{x}]$ at the SSH Causal Level which is part of the discrete abstraction of the continuous environment. It is this abstraction from continuous to symbol that facilitates causal/topological mapping in the basic SSH.

3. The axioms provided here describe the nature of the spatial knowledge represented at each SSH level, but we omit auxiliary axioms required for logical completeness (e.g., unique names axioms, etc.). A complete set of axioms is provided by Remolina and Kuipers (2004). For clarity and conciseness, we use a typed logic in which variable names encode their types, and we assume that all free variables in axioms are universally quantified.

4. Finding the minimal DFA in the general case, like finding the minimal HMM, is NP-Complete (Angluin 1978; Gold 1978). However, given non-symmetries in the environment, it is possible to use active exploration routines to eliminate many DFA hypotheses. Active exploration routines that eliminate ambiguity among DFA hypotheses are closely related to *adaptive distinguishing sequences*, which can be computed in polynomial time (Yannakakis and Lee 1991), and should allow the robot to find a minimal DFA in polynomial time (Schapire 1991) in future work on active exploration.

The effect of the SSH hill-climbing (and HSSH place detection and localization) is that the Causal Level representation can assume that actions are deterministic. The determinism of the observation function rests on the abstraction from sensory images to views being sufficiently aggressive to eliminate perceptual *variability*. Although observations are deterministic, they are not necessarily unique since there may still be perceptual aliasing. This ambiguity is handled by creating multiple hypotheses of topological (thus causal) models, as explained in Section 3.3. In general, it is not possible for a robot to recover the complete spatial structure of any arbitrary environment (Dudek et al. 1991); therefore, keeping around the tree of possible maps allows the robot to continue navigation when the best hypothesis is refuted by an experienced counter example.[5]

3.3. The SSH Topological Level

In the SSH, a topological map is an instantiated model for two sets of axioms: one that describes topological maps in general and another that describes the exploration experience of the agent in a particular environment. We identify the global topological map by generating potential models of these axioms, discarding those that violate the axioms, and applying an ordering on the remaining ones so that a single best model can be selected. If there is no single best model, then a few closely competing models can be identified and can be used to make an exploration plan to help discriminate between models.

The SSH Topological map M^T describes the environment in terms of dstates, places, paths, regions, and the qualitative relations among them such as connectivity, order, and containment. A dstate $q \in Q$ represents a distinctive state or pose of the agent in the environment, a place $p \in P$ represents a zero-dimensional location, a path $\pi \in \Pi$ represents a one-dimensional structure, and a region $r \in R$ represents a two-dimensional subset of the environment. In this paper, we will not discuss regions or their relations, which are described by Remolina and Kuipers (2004).

We formalize a topological map as

$$M^T = M^C \cup Objects \cup Relations.$$

Here $Objects = \langle P, \Pi, R \rangle$, where P is a set of places, Π is a set of paths, and R is a set of regions. M^T thus includes (via M^C) the sets of states, actions, and views. $Relations$ encodes the relations over this set of objects. These relations, including at, $along$, on, $order$ (and the local topology "star" relations introduced in Section 5), allow for a richer description of the connectivity of places and paths, and are introduced below as needed.

At the SSH Causal Level, the experience is represented as an alternating sequence of states ($q_i \in Q$) and actions ($a_j \in A$).

$$q_0 \quad a_1 \quad q_1 \quad a_2 \quad q_2 \quad \cdots \quad q_{n-1} \quad a_n \quad q_n.$$

At the Topological Level, each distinctive state $q \in Q$ corresponds to being *at* a place, and facing *along* a path in some direction. Since a path is one-dimensional, it has two directions $d \in \{+, -\}$, for which $opp(+) = -$ and $opp(-) = +$. We define a directed path, π^d, to represent facing along a path in a particular direction:

$$\forall q \in Q \; \exists p \in P, \; \pi \in \Pi,$$

$$d \in \{+, -\} \; [at(q, p) \wedge along(q, \pi, d)], \tag{1}$$

$$along(q, \pi^d) \equiv along(q, \pi, d).$$

Additionally, there are two kinds of basic actions, turns and travels, and there is a TurnAround action:

$$A = Turns \cup Travels, \quad TurnAround \in Turns.$$

A place $p \in P$ corresponds to a set of states linked by turn actions:

$$\langle q, a, q' \rangle \in S \wedge a \in Turns \wedge at(q, p) \rightarrow at(q', p). \tag{2}$$

Similarly, a path $\pi \in \Pi$ corresponds to a set of states linked by travel actions, or by a TurnAround:

$$\langle q, a, q' \rangle \in S \wedge a \in Travels \wedge$$

$$along(q, \pi, d) \rightarrow along(q', \pi, d), \tag{3}$$

$$\langle q, a, q' \rangle \in S \wedge a = TurnAround \wedge$$

$$along(q, \pi, d) \rightarrow along(q', \pi, opp(d)). \tag{4}$$

The relation $on(\pi, p)$ means that the place $p \in P$ is on the path $\pi \in \Pi$:

$$at(q, p) \wedge along(q, \pi, d) \rightarrow on(\pi, p). \tag{5}$$

A path defines an order relation over the places on it:

$$\langle q, a, q' \rangle \in S \wedge a \in Travels \wedge at(q, p) \wedge$$

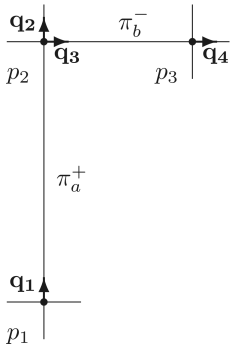
$$at(q', p') \wedge along(q, \pi, d) \rightarrow order(\pi, d, p, p'), \tag{6}$$

$$order(\pi, d, a, b) \rightarrow on(\pi, a) \wedge on(\pi, b), \tag{7}$$

$$\neg order(\pi, d, p, p), \tag{8}$$

$$order(\pi, d, a, b) \iff order(\pi, opp(d), b, a), \tag{9}$$

5. Long-term experience with the HSSH has yielded deterministic actions and views in research settings, but we can envision rare scenarios, e.g., an intersection crowded with people, that could lead to an undetected or misclassified place. Detecting and understanding these unusual events should allow us to still assume deterministic actions $(100 - \varepsilon)\%$ of the time.



Causal schema	Equation #	Topological relation
$\langle q_1, travel, q_2\rangle \in S$	(1) $\rightarrow$	$at(q_1, p_1),\ along(q_1, \pi_a^+)$
	(5) $\rightarrow$	$on(\pi_a, p_1)$
	(3) $\rightarrow$	$along(q_1, \pi_a^+),\ along(q_2, \pi_a^+)$
	(1) $\rightarrow$	$at(q_2, p_2)$
	(5) $\rightarrow$	$on(\pi_a, p_2)$
	(6) $\rightarrow$	$order(\pi_a^+, p_1, p_2)$
	(8) $\rightarrow$	$p_1 \neq p_2$
$\langle q_2, turn, q_3\rangle \in S$	(2) $\rightarrow$	$at(q_3, p_2)$
	(1) $\rightarrow$	$along(q_3, \pi_b^-)$
	(5) $\rightarrow$	$on(\pi_b, p_2)$
$\langle q_3, travel, q_4\rangle \in S$	(3) $\rightarrow$	$along(q_4, \pi_b^-)$
	(1) $\rightarrow$	$at(q_4, p_3)$
	(5) $\rightarrow$	$on(\pi_b, p_3)$
	(6) $\rightarrow$	$order(\pi_b^-, p_2, p_3)$
	(8) $\rightarrow$	$p_2 \neq p_3$

Fig. 4. Topological abduction example. Here we illustrate the abduction process, using the topological axioms to model exploration. Starting at dstate q_1, the agent reaches dstate q_2 at a place p_2 having traveled along directed path π_a^+. It then turns to dstate q_3, still at place p_2, and is ready to travel along another path, say π_b^-, from q_3 to dstate q_4 at some other place.

$$order(\pi, d, a, b) \wedge order(\pi, d, b, c) \rightarrow$$
$$order(\pi, d, a, c). \quad (10)$$

In order to create a Topological Level map from a Causal Level experience, such as $\langle q_1, travel, q_2\rangle$, $\langle q_2, turn, q_3\rangle$, $\langle q_3, travel, q_4\rangle$, the agent uses abduction to hypothesize the existence of several places and paths at which these distinctive states occur. Figure 4 shows an example of the abduction process.

Remolina and Kuipers (2004) provide a non-monotonic axiomatization of the SSH topological map, including additional elements of the theory (regions, boundary relations, and metrical relations), along with more details and motivating examples. This theory provides a precise specification of the possible logical models (topological maps) that are consistent with the axioms and the sequence of actions and views observed while exploring. A prioritized circumscription policy (expressed as a nested abnormality theory (Lifschitz 1995)) specifies how distinct consistent logical models are ordered by simplicity. Furthermore, Savelli and Kuipers (2004) have developed the non-local *planarity constraint*, which enforces the requirement that a topological map is a graph embedded in the plane. Figure 5 presents an algorithm for constructing all possible topological maps by generating all possible sets Q.

3.4. The SSH Metrical Level

The SSH, often thought of as a framework for creating purely topological maps, has always allowed for local metrical knowledge to be utilized at the Control Level (Figure 2, right column). Additionally, the SSH Metrical Level has always supported a global metrical map to be created *after* the topological map – it is our belief that such a global metrical map is often unnecessary for navigation in and communication about large-scale environments. However, the SSH theory has lacked a formal description of exactly how metrical information influences the hierarchical abstractions of space. One contribution of this paper is to clarify the relationships between metrical and symbolic knowledge in a navigational agent.

In work leading to the development of the SSH, Kuipers and Byun (1991) created a "patchwork metrical map". Their mapping implementation annotated topological places and paths with metrical data gathered during exploration. Given a topological map hypothesis, the global place layout was relaxed to minimize errors with respect to the annotated metrical data before adding stored range information to create the obstacle map. This approach is similar to the probabilistic techniques we define formally in Section 7.3.

3.5. Extending the SSH

The SSH depends on the assumption that the environment naturally decomposes into place neighborhoods, connected by

0. Perform initial action a_0 that brings the robot to a place along a directed path. Initialize the tree of maps with the map hypothesis $\langle M_0, q_0 \rangle$, where M_0^C contains the single dstate q_0 with its observed view v_0, and M_0^T contains the single place p_0 and path π_0.

After performing a new action a and observing the resulting view v, for each consistent map $\langle M, q \rangle$ on the fringe of the tree:

1. If M^C includes $\langle q, a, q' \rangle$ in R and $v' = o(q')$,
   - if $match(v, v')$, then $\langle M, q' \rangle$ is the successor to $\langle M, q \rangle$, extending the tree;
   - if not, then mark $\langle M, q \rangle$ as inconsistent.
2. Otherwise, M^C does not include $\langle q, a, q' \rangle$ in R. Let M' be M extended with a new distinctive state symbol q' and the assertions $v = o(q')$ and $\langle q, a, q' \rangle$. Consider the $k \geq 0$ dstates q_j in M with $v_j = o(q_j)$, such that $match(v_j, v)$. Then $\langle M, q \rangle$ has $k+1$ successors:
   - $\langle M'_j, q' \rangle$ for $1 \leq j \leq k$, where M'_j is M' extended with the assertion $q' = q_j$;
   - $\langle M'_{k+1}, q' \rangle$, where M'_{k+1} is M' extended with the k assertions that $q' \neq q_j$, for $1 \leq j \leq k$.
3. Mark a new successor map inconsistent if it violates the axioms of topological maps.
4. Define a preference order on the consistent maps at the leaves of the tree.

In the Basic SSH:

$M = M^T$.
A view is a simple symbol.
$match(v, v')$ iff $v = v'$.
Both $a \in Turns$ and $a \in Travels$ can reach step 2 and cause a branch.
Preference order from prioritized circumscription policy (Remolina and Kuipers 2004).

In the Hybrid SSH (Section 6):

$M = \langle M^T, M^P \rangle$.
A view is a structure $\langle S_p, \tilde{\pi}^d \rangle$, where $p = place(q)$, consisting of a local topology and the directed local-path the robot arrived upon.
$match(v, v')$ iff there exists an isomorphism $\phi : S_p \rightarrow S'$ where $\phi(q) = q'$.
Only $a \in Travels$ can reach step 2 and cause a branch.
Future work: Preference order from map probabilities (Section 10.2).

Fig. 5. Building the tree of topological maps. This pseudo-code framework describes the algorithm for building a tree of all possible topological consistent with a sequence of actions and observations at discrete places. The different instantiations of this framework for the basic SSH and HSSH are also described.

path segments, which can then be abstracted to a topological map. That is, it uses the sparse structure of man-made environments (or man-made paths in natural environments) to define a small number of discrete places and connecting paths. Obviously, topological structure may be imposed even in unstructured environments. Defining places at visually distinctive locations along a single path (e.g., a water tower on the side of a highway) or even based on metrical path-integration in wide-open spaces (as the Puluwat navigators do when piloting dugout canoes between distant islands (Gladwin 1970)) are currently not handled by our SSH hill-climbing controllers or the HSSH place detection methods. We believe these type of places can be represented within the SSH framework, but we leave this problem for future work.

The basic SSH makes weak (i.e. very general) assumptions about the sensory capabilities of the navigational agent;

thus, abstraction from continuous sensations to discrete models of the environment depends on well-crafted control laws that move the robot reliably between distinctive states. The HSSH makes stronger (i.e. more specific) assumptions about the types of sensors available to the agent, e.g., range sensors. This allows the HSSH to extend the basic SSH by using existing metrical mapping techniques to create precise observational models of the local surround.

The HSSH has four major levels of representation that correspond to the four SSH levels (Figure 6). At the *Local Metrical Level*, the agent builds and localizes itself in the LPM, a metrically accurate map of the local space within its sensory horizon. The LPM is used for local motion planning and hazard avoidance. At the *Local Topological Level*, the agent identifies discrete places (e.g., corridor intersections, rooms, etc.) in the large-scale continuous environment, and symbolically describes the configuration of the paths through the place – its local decision structure. At the *Global Topological Level*, the agent resolves structural ambiguities and determines how the environment is best described as a graph of places, paths, and regions. The *Global Metrical Level* specifies the layout of places, paths, and obstacles within a single global frame of reference. It can be built on the skeleton provided by the topological map. Figure 6 diagrams the basic flow of data in the HSSH, from sensors, through the local metrical model of small-scale space and the local and global symbolic models of the large-scale environment, finally creating the global metrical model if desired.

Having small-scale space models of the local surround creates several advantages when implementing the HSSH versus the basic SSH. First, the robot represents the local environment using a local perceptual map (*LPM*) (Section 4). The robot can therefore use algorithms for local metrical motion planning and obstacle avoidance instead of relying on behavior-based controllers. Second, metrical localization can be done quickly after entering a place neighborhood, rather than requiring physical hill-climbing to a distinctive pose.

A symbolic description of the *local topology* is extracted from this precise small-scale-space model of the local surround via *gateways* (Section 5). Thus, the view of a distinctive state no longer need be some user-defined function of the perceptual inputs. Instead, the method relies on the local topology extracted from the LPM to describe places, thus describing all distinctive states at each place. Using local topology to *detect and describe* places allows the robot to model more complicated intersections of paths than with hill-climbing. Additionally, using local topology constrains the global topological model search (Section 6), as branching in the tree of possible maps occurs only when arriving at a place, not when visiting the various dstates of a place (Section 8).

Stored metrical information along topological connections between places can be used to efficiently obtain a global metrical layout of places (Section 7), which provides the "backbone" for a global map if desired. The HSSH also improves

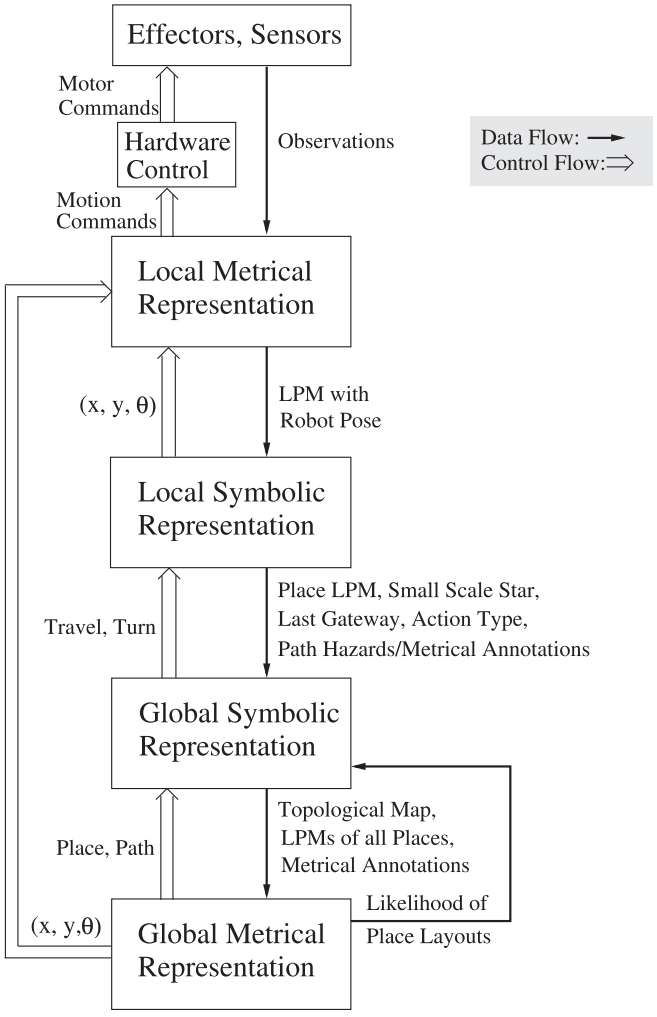


Fig. 6. HSSH description. The HSSH is an integrated framework of multiple, distinct representations of spatial knowledge. Each level of abstraction uses its own ontology with concepts motivated by human cognitive abilities and grounded to the environment via local metrical observations. The four major components here correspond to the four levels of the basic SSH shown in Figure 2.

navigational behaviors and facilitates multi-modal human–robot interaction (Beeson et al. 2007; MacMahon et al. 2006).

The rest of this paper is focused on discussion of the individual components of the HSSH. Section 4 describes local metrical modeling of small-scale space using existing SLAM methods, as well as local motion planning. Section 5 describes how the local topology of a place is abstracted from the local metrical model and how this abstraction leads to reliable place detections and descriptions. Section 6 describes how the global topological map is created and maintained as exploration provides a sequence of actions and local topologies of places encountered during travel. Section 7 describes how the global

metrical map is built on the qualitative skeleton provided by the global topological map. Section 8 discusses computational complexity issues for each level of the HSSH. Section 9 summarizes the paper, and Section 10 discusses future research on optimizing the HSSH.

4. HSSH Local Metrical Level

The critical difference between the basic SSH and the HSSH is the use of a local metrical model of small-scale space surrounding the robot. In our current work, we call this model a Local Perceptual Map. The LPM is currently built using sensor input from laser range sensors that see walls, but the LPM could be built from visual sidewalk (or road) detection or other sensor modalities. Similarly, the current LPM representation models occupied, free, and unknown regions of space. Work by Murarka et al. (2006) investigates incorporating semantic labels into the LPM to denote drop offs, pedestrians, and other types of hazards.

4.1. Local Perceptual Map

The LPM is a bounded-size metrical description of the small-scale space surrounding the agent. It functions as an *observer*, integrating sensor values over time to determine the locations of obstacles and other hazards, for localization, motion planning, and the derivation of local features for larger-scale mapping. The LPM represents the small-scale space within the robot's sensory horizon, not just what is currently in view. It is small enough to avoid the problem of closing large loops. The frame of reference of the LPM is local. Its relation with the world frame may be unknown, and will drift over time due to accumulating errors.

When the agent travels from one place to another, the LPM acts as a *scrolling map*, $\tilde{m}$, that describes the robot's immediate surround. Information that scrolls off the LPM is discarded, and new cells that scroll onto the map are initialized as unknown.[6] Because the LPM has a fixed, bounded size, the cost of updating it is constant in both time and space.

The full task of building metrical maps from exploration data can be described as finding the joint posterior over maps m and trajectories $x = (\mathsf{x}, \mathsf{y}, \theta)$ in $P(x, m|z, u)$ with the following symbol definitions.

t : The time-steps $0 \leq t \leq N$ of the agent's experience.

$x = x_{0:N}$: The sequence of agent poses x_t at each time-step t.

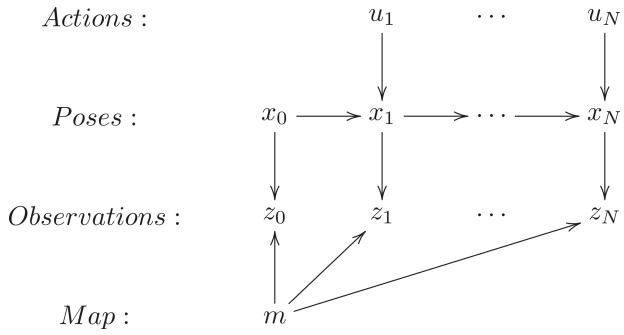


Fig. 7. Markov localization. The standard graphical dynamic Bayesian network (DBN) for Markov localization within a single frame of reference, which combines belief about actions $P(x_t|u_t, x_{t-1})$ and observation $P(z_t|x_t, m)$. SLAM algorithms combine localization, $P(x|z, u)$, with one of a number of mapping methods to estimate $P(x, m|z, u)$.

$z = z_{0:N}$: The sequence of observations z_t.

$u = u_{1:N}$: The sequence of actions u_t between time-steps.

m : The set of map elements, which may be landmarks or occupancy grid cells. $\tilde{m}$ (the scrolling LPM) is a particular example of a metrical map m.

The joint probability of the pose history x and the metrical map m can be decomposed as

$$P(x, m|z, u) = P(m|x, z, u) \cdot P(x|z, u)$$

by the chain rule for probabilities.

For simple, local regions, the maximum-likelihood map can be estimated incrementally given knowledge of x and z, so we really just need to solve for $P(x|z, u)$. Additionally, we are not concerned with the full distribution over pose trajectories, as we are updating the map from the maximum-likelihood pose at each time step. Thus, for our online metrical mapping we only need to determine the distribution over the current pose:

$$\begin{aligned} Bel(x_t) &= P(x_t|z_{0:t}, u_{1:t}) \\ &= \eta P(z_t|x_t, m) \int P(x_t|x_{t-1}, u_t) Bel(x_{t-1}) dx_{t-1}, \end{aligned}$$

where η is a normalization constant. Figure 7 illustrates the basic structure of Markov localization (Fox et al. 1999), which allows us to determine in an efficient and incremental algorithm the distribution of poses that best fit the current map.

Although multiple metrical mapping methods might be used for the LPM, we utilize the well-known occupancy grid representation (Moravec 1988; Elfes 1989), along with particle filter Markov localization (Fox et al. 1999) to overcome noisy odometry information. Stated more plainly, we model

6. Our rectangular LPM scrolls, horizontally or vertically, as needed to keep to keep the robot's pose in a central cell. Information in the occupancy grid is only shifted by integral numbers of cells to avoid blurring the model by rotations or partial-cell translations.

the world as a discretized grid, where each cell contains a probability of being occupied by an obstacle, as measured by a lidar sensor. Localization is performed by comparing hypothesis poses to the current map, and the map is updated accordingly. This is a well-known version of SLAM (Thrun et al. 2005). Discussions in this paper that refer to this implementation generalize to many SLAM implementations.

4.2. LPM Benefits

LPMs provide the HSSH with various information that allows both local and global abstractions of space. In Section 5, we discuss how the LPM supports the abstraction of a symbolic small-scale space description of the local-paths in the surround. Section 7 discusses how the local metrical information is used, along with the topological map, to find the global metrical place layout of an exploration trace and, if desired, the entire global metrical map of an explored environment. In addition to providing useful local metrical information for place detection, categorization, and layout, the LPM is a reliable observer for local control at the SSH Control Level.

Given a target pose in the LPM, the robot can compute a trajectory to reach the target without colliding into obstacles. This can be done using the Vector Field Histogram (Borenstein and Koren 1991), the Dynamic Window approach (Fox et al. 1997), gradient methods (Konolige 2000), or even a simple search (using A* or rapidly exploring random trees [RRTs] (Kuffner and LaValle 2000)) over the cells of the occupancy grid. The potential function (for gradient methods) or the cost function (for A*) reflects the distance of the agent from an obstacle or other hazard represented in the LPM. Object tracking may be implemented at this level, but our current robot implementation simply avoids obstacles by taking the first few steps along the planned trajectory before replanning. We discuss the selection of target poses as they apply to Causal Level *Travels* and $Turns$ in Section 5.3.

In the basic SSH (Kuipers 2000), an agent localizes itself in a place neighborhood by hill-climbing to a distinctive state. Localization by physically moving to maximize a "distinctiveness measure" requires very little knowledge about the nature of the environment or the sensors. In the HSSH, on the other hand, the agent uses an online SLAM method to localize itself unambiguously within the LPM. SLAM methods depend on stronger knowledge about the relation between sensor input and the agent's location in the local frame of reference – $P(z_t|x_t, \tilde{m})$. In return for these stronger assumptions, the agent does not need to move to a particular location to be adequately localized.

Finally, when the agent is in the neighborhood of a particular topological place p, a snapshot of the LPM $\tilde{m}$ serves as a small-scale space description of the place neighborhood that is stored as a place annotation m_p in the topological map. When a place p is first encountered, the local map m_p for its neighborhood is initialized with the information from the scrolling map $\tilde{m}$. The frame of reference defined for m_p may be different from that of $\tilde{m}$, appropriate to the characteristics of the place neighborhood. When the neighborhood of p is encountered on subsequent occasions, the agent may localize itself with respect to the stored map m_p and may update m_p with the more recent information in $\tilde{m}$.

5. HSSH Local Topological Level

As the robot and its scrolling LPM move continuously through the environment, the robot identifies a discrete set of isolated *places* and the *path segments* that connect them. In the small-scale space of the LPM, a place neighborhood is an extended region. In the large-scale space representation, a place is a node in the topological graph, and is connected by paths to other places. These are the local elements from which a global topological map is constructed. We abstract the structure of a place neighborhood to the *local topology* description of the place. Just as a path describes the linear order of places on it, a place describes the circular order of directed paths radiating from it. We call this the local topology S_p of a place p, and describe the circular order with a structure called a *star* (Kuipers et al. 2004). This section discusses how this symbolic representation of a place (in large-scale space) is grounded in the metrical description m_p of the place neighborhood (in small-scale space).

A *local-path* $\tilde{\pi}$ at a place p is the fragment of a topological path that is visible within the stored LPM, m_p, of the neighborhood of p. A *directed local-path* is of the form $\tilde{\pi}^d$, where $d \in \{+, -\}$ represents the direction along $\tilde{\pi}$ moving away from p. Upon arriving at a new place, a local-path and its directions may not yet have been matched with a global topological path and its directions.

A *star* S is a set of directed local-paths such that $\tilde{\pi}^+ \in S \iff \tilde{\pi}^- \in S$. There are two functions that describe stars.

- $next : S \to S$ induces a clockwise circular order over $\tilde{\pi}^d \in S$. $next(\tilde{\pi}^d)$ is the next element from $\tilde{\pi}^d$ in the clockwise order.
- $\alpha : S \to \{0, 1\}$ associates an attribute value $\alpha(\tilde{\pi}^d)$ to $\tilde{\pi}^d \in S$, where $\alpha(\tilde{\pi}^d) = 1$ means that travel is possible along $\tilde{\pi}^d$ away from p, and $\alpha(\tilde{\pi}^d) = 0$ means that travel away from p along $\tilde{\pi}^d$ is not possible.

The star is naturally encoded as a sequence of pairs, where the sequence encodes the $next$ relation ($next$ of the last element being the first element), and the second element of each pair is the value of α applied to the first element. For example, consider the following local topology (star) descriptions of familiar intersection types[7] including local-paths $\tilde{\pi}_a$, $\tilde{\pi}_b$,

7. Note that we do not have a fixed set of equivalence classes for local topology abstraction. Although there is an upper bound on the number of paths that can fit into an LPM, this is determined by the path width and the LPM size. Thus, many types of intersections can exist that cannot be "named" using a letter.

and sometimes $\tilde{\pi}_c$. (For ease of visualization, the first directed local-path in the circular order is the one directed upward.)

$$+ \quad [\langle \tilde{\pi}_a^+, 1\rangle, \langle \tilde{\pi}_b^+, 1\rangle, \langle \tilde{\pi}_a^-, 1\rangle, \langle \tilde{\pi}_b^-, 1\rangle]$$

$$\mathsf{T} \quad [\langle \tilde{\pi}_a^-, 0\rangle, \langle \tilde{\pi}_b^+, 1\rangle, \langle \tilde{\pi}_a^+, 1\rangle, \langle \tilde{\pi}_b^-, 1\rangle]$$

$$\mathsf{L} \quad [\langle \tilde{\pi}_a^+, 1\rangle, \langle \tilde{\pi}_b^+, 1\rangle, \langle \tilde{\pi}_a^-, 0\rangle, \langle \tilde{\pi}_b^-, 0\rangle]$$

$$\mathsf{Y} \quad [\langle \tilde{\pi}_a^-, 0\rangle, \langle \tilde{\pi}_b^+, 1\rangle, \langle \tilde{\pi}_c^-, 0\rangle, \langle \tilde{\pi}_a^+, 1\rangle, \langle \tilde{\pi}_b^-, 0\rangle, \langle \tilde{\pi}_c^+, 1\rangle]$$

$$\mathsf{K} \quad [\langle \tilde{\pi}_a^+, 1\rangle, \langle \tilde{\pi}_b^+, 1\rangle, \langle \tilde{\pi}_c^+, 1\rangle, \langle \tilde{\pi}_a^-, 1\rangle, \langle \tilde{\pi}_b^-, 0\rangle, \langle \tilde{\pi}_c^-, 0\rangle]$$

$$\psi \quad [\langle \tilde{\pi}_a^+, 1\rangle, \langle \tilde{\pi}_b^+, 1\rangle, \langle \tilde{\pi}_c^-, 0\rangle, \langle \tilde{\pi}_a^-, 1\rangle, \langle \tilde{\pi}_b^-, 0\rangle, \langle \tilde{\pi}_c^+, 1\rangle]$$

An *isomorphism* $\phi : S \to S'$ between two stars S and S' is a bijective function such that

$$next(\phi(\tilde{\pi}^d)) = \phi(next(\tilde{\pi}^d)),$$

$$\alpha(\phi(\tilde{\pi}^d)) = \alpha(\tilde{\pi}^d),$$

$$path(\phi(\tilde{\pi}^d)) = path(\phi(\tilde{\pi}^{opp(d)})),$$

where $path(\tilde{\pi}^d) = \tilde{\pi}$. An isomorphism means that the two stars have the same local topology under a suitable rotation of the circular order. Note that two stars may have multiple distinct isomorphisms. For example, there are four distinct isomorphisms between two + intersections.

The local topology description provides a purely qualitative account of "left" and "right", avoiding the need to define them in terms of thresholds on some angular variable. A particular directed local-path at a place p, $\tilde{\pi}^+$, and its opposite, $\tilde{\pi}^-$, partition the other directed local-paths in the star into two groups. Those that are between $\tilde{\pi}^+$ and $\tilde{\pi}^-$ in the clockwise direction can be described as being "to the right" of $\tilde{\pi}^+$. Those between $\tilde{\pi}^+$ and $\tilde{\pi}^-$ in the counter-clockwise direction can be described as "to the left" of $\tilde{\pi}^+$. This also defines the appropriate destination for a route instruction such as "turn right" when the agent is at a place p, facing along a directed path $\tilde{\pi}_a^+$. The pragmatics of natural language requires that "turn right" must uniquely specify a directed local-path $\tilde{\pi}_b^d$ that is "to the right" of $\tilde{\pi}_a^+$, such that $\alpha(\tilde{\pi}_b^d) = 1$ (i.e. $\tilde{\pi}_b^d$ is navigable from p).

5.1. Grounding Local Topology in the Local Perceptual Map

We have illustrated how to describe a place symbolically as a circular order of directed local-paths. Here we discuss how to use *gateways* to ground local-paths in the LPM. Gateways allow the robot to ground large-scale actions in the small-scale metrical models, abstract a symbolic local topology description from the small-scale model, and detect and compare places in the environment.

5.1.1. Gateways

The term "gateway" is adapted from Chown et al. (1995), who 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. (Chown et al. 1995, p. 32)

We define a *gateway* as a boundary in the LPM that separates the local place neighborhood from the larger environment. That is, a gateway is the boundary where control shifts between localization within the local place neighborhood and travel from one place neighborhood to another. A gateway has two directions, *inward* (looking into the place) and *outward* (looking away from the place), according to the direction of that shift. The location, extent, and orientation of gateways at a place are saved as annotations of the local place neighborhood map m_p.

In much of human experience with large-scale environments (both natural and man-made) local place neighborhoods are separated from each other (either by boundaries or by distance), and they are connected by travel actions along paths. Navigation in large-scale space is thus typically an alternation between motion along travel paths and motion within place neighborhoods. The existence of gateways, as interfaces between the two types of travel, is therefore a requirement for the abstraction from small-scale to large-scale space. Certainly extreme situations occur, such as place neighborhoods that overlap or are immediately adjacent, or environments with (apparently) no distinctive states at all. These will be discussed in appropriate sections below.

Schröter et al. (2004) and Yeap (1988) discuss finding gateways by looking for occlusions from laser data or local models.[8] Schröter (2006) also details a visual door recognition system for determining gateways. Below, we describe an alternative algorithm for identifying gateways within the small-scale space of the LPM. This algorithm relies on a Voronoi skeleton computed from the free space in the LPM. Our implementation first prunes the Voronoi skeleton, using the fact that we have a bounded LPM to determine the true skeleton of free space in the local surround. It then defines the "core" of the local region by grouping nearby Voronoi junctions, if they exist. Walking along the graph, from the core, to the frontiers of the local map, the algorithm looks for *constrictions* as locations for gateways. Constrictions can be defined several ways, e.g., local minima in the distances to the closest obstacles (what

8. Gateways are called exits by Yeap (1988); Yeap and Jefferies (1999).

Thrun and Bücken (1996) call "critical points"). Gateways are then defined as line segments that separate distinct regions of free space in the LPM.

Pruning the Voronoi graph. A Voronoi graph is the set of points equidistant from the two (or more) closest obstacles. It lies on the boundaries of Voronoi regions (Fortune 1992). Using a Voronoi graph to describe the free space of a metrical model can be useful; however, given noisy measurements, a Voronoi graph can contain many branches and spurs that do not contribute to the "base" skeleton that describes the "backbone" of the modeled environment. As a result there has been work on pruning of Voronoi graphs (Choset and Nagatani 2001; Wallgrün 2005) and on using thinning-based approximations of Voronoi graphs for mobile robot navigation (Choi et al. 2002). *Thinned skeletons* often have far fewer spurs into concave corners; thus, they represent an approximation of a partially pruned Voronoi skeleton.

In order to determine the "critical" skeleton of a noisy Voronoi graph, we assume that the robot is computing a Voronoi graph in the LPM. Currently, we also remove "island" obstacles by removing occupied or unknown cells in the occupancy grid that are completely surrounded by free cells. This reduces drastic changes in the skeleton due to pedestrians. The Voronoi graph is computed by treating occupied cells in the occupancy grid as obstacles.

Because we use a small, bounded LPM, there is always some region of free space that touches the edge of the occupancy grid or some region of "unknown occupancy" (gray cells in the figures) that may provide an option to leave the current region. We define a terminal point of the Voronoi graph that reaches the edge of the LPM or reaches unexplored cells to be an *exit*. We can then define the branches of the Voronoi graph that contain exits to be "critical" branches. Instead of actively pruning away branches, a better approach is to include only the union of all shortest paths that connect each exit to all other exits in the LPM.[9] Figure 8 shows how this spanning tree eliminates all spurious junctions and branches in these small-scale models.

Determining Gateways. Gateways can be grounded in an LPM by using the Voronoi graph. Although the full, continuous generalized Voronoi graph can be computed using Fortune's algorithm (Fortune 1992), it is usually more efficient to approximate the Voronoi graph of an occupancy grid, using pixel-based "brush-fire" algorithms: imagine a brush fire along all defined obstacle pixels, burning inward at a constant speed, and the skeleton is marked by all points where two or more fires meet. Similarly, there exists a thinning algorithm (Zhang and Suen 1984) that gives a pixel-based skeleton, but with many spurious terminal branches pre-pruned. As both the brush-fire approaches and the thinning approach are linear in the grid size, we utilize the thinned skeleton, as it speeds up the pruning process due to having fewer branches that need to be examined.

Figure 9 shows a few steps of the gateway algorithm on a thinning-based skeleton. The algorithm below has been tested on both thinned skeletons and true Voronoi graphs, and works well on both kinds of skeletons. Given a pruned skeleton, the method for finding gateways is as follows:[10]

- l is the location of the physical robot with respect to the LPM.
- c is the point on the Voronoi graph closest to l (Figure 9(a)).
- J is the set of Voronoi junctions j (Figure 9(b)).
- r_p is the "Voronoi radius": the distance from the 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 $dist(j, j') \leq max(r_j, r_{j'})$.
- Define the "core" of the place neighborhood as F: the equivalence class of neighboring junctions that includes K as the starting set of junctions. See Figure 9(b).
- If $F = \emptyset$ then define $F = \{c\}$.
- Q is the set of all Voronoi points q, where $\exists f \in F$ such that $dist(f, q) = r_f \wedge \forall f' \in F\ dist(f', q) \geq r_{f'}$. 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 (Figure 9(c)). Look for a point p that corresponds to a *constriction*.
- At each of these constrictions m, define a line segment g of length $2 \cdot r_m$, centered on m, oriented normal to the branch at point m. See Figure 9(d).

A recursive version of this algorithm was implemented that runs quickly enough to recompute gateways in real-time (two to three times a second for a 300×300 cell occupancy grid with 10 cm cells using an older 450 MHz research robot

9. Dead-ends are a special case, where only one exit exists. Here we just keep the branch that connects the exit to the Voronoi junction at the dead-end.

10. Symbols used in the discussion of the gateway implementation are local in scope and have no relation to symbols used elsewhere in this paper, even if they are spelled the same.

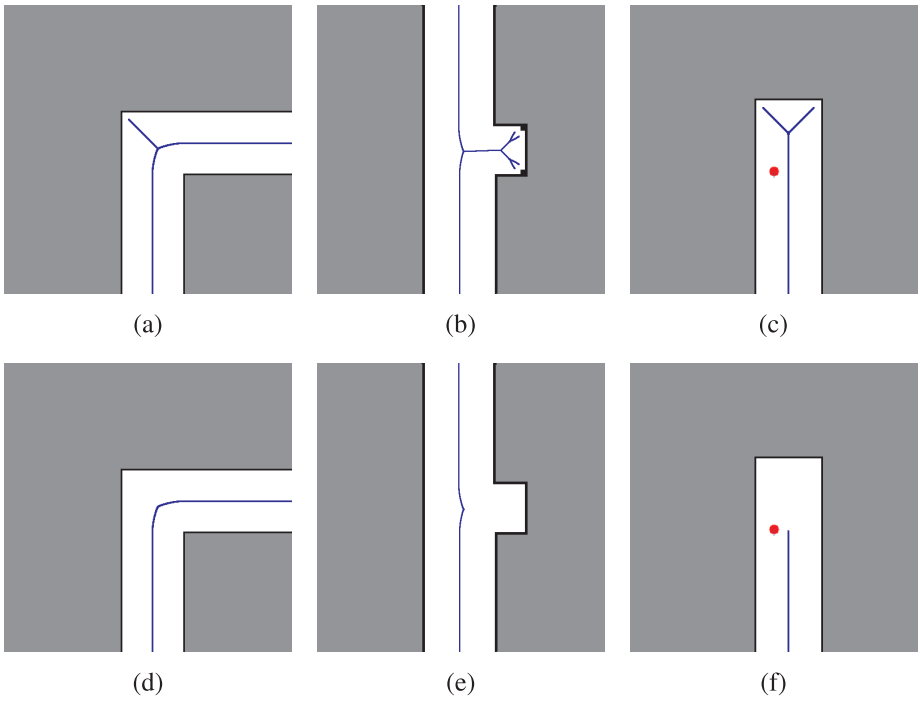


Fig. 8. **Pruning a Voronoi graph using the LPM boundary.** (a),(b),(c) Examples of Voronoi graphs at common places. (d) Pruning the skeleton at an L leaves no junction, which means methods that rely on junctions in pruned graphs to define places (Choset and Nagatani 2001) ignore these types of intersections. (e) Small bits of noise around objects can cause spurs and hierarchical branches in the Voronoi skeleton. Methods that use fixed-depth or distance-based pruning (Choset and Nagatani 2001) can leave junctions in the graph, while the LPM pruning eliminates all non-critical branches. (f) Dead-ends are a special case where we keep the minimal branch that connects the single exit to the junction closest to the robot.

computer).[11] This implementation was used to produce Figures 9(d), 10(a), 11(a)–(c), and 12(b),(c), and it was the implementation used to detect the places in Figure 15(a). A constriction is currently a local minimum over $r_p \in R$. Defining gateways where the *change in distance* to nearby obstacles is minimal (or plateaus) provides useful gateways at the beginning of hallways and doorways in corridor environments.

5.1.2. Local Topology

Given an implementation for detecting gateways in a stored map of a place, m_p, we can ground the local topology concepts of local-paths in our small-scale model of the surrounding environment.

- For each outward-facing oriented gateway $\langle g, out \rangle$, define a *directed local-path* $\tilde{\pi}_g^+$ that leads away from the current place.
- Initialize a circularly ordered *star* S_p with a list (clockwise from an arbitrary starting point) of associations between directed local-paths and oriented gateways, $(\langle \tilde{\pi}_g^+, 1 \rangle \leftrightarrow \langle g, out \rangle)$. Since these are traversable paths, each $\alpha(\tilde{\pi}_g^+) = 1$.[12]
- Test each pair of gateways, g and g', via a *path continuity* test, to determine whether their directed local-paths belong to a single continuous path. If so, give both directed local-paths the same path name (e.g., $\tilde{\pi}_a$ below and in Figures 10(b),(c)), and include the inward oriented gateways in the association. For example, change

11. Note that the gateway algorithm is useable on LPMs with quite low resolution (Beeson 2008). Because higher resolution LPMs do not improve the reliability of gateways and because the thinning algorithm is linear in the number of cells in the LPM, we chose LPM parameters that facilitate reliable SLAM and local motion planning, while still fitting nicely within the cache of our robot's onboard computer.

12. In future implementations, $\alpha : S \rightarrow \{$MIDLINE, LEFTWALL, RIGHTWALL, DEADEND, NONE$\}$ should associate directed local-paths with attributes representing the control laws for traversing the path in that direction. (LEFTWALL and RIGHTWALL imply coastal navigation scenarios. For terminating local-paths, DEADEND means that further travel is blocked, while NONE means that no control law is applicable.)

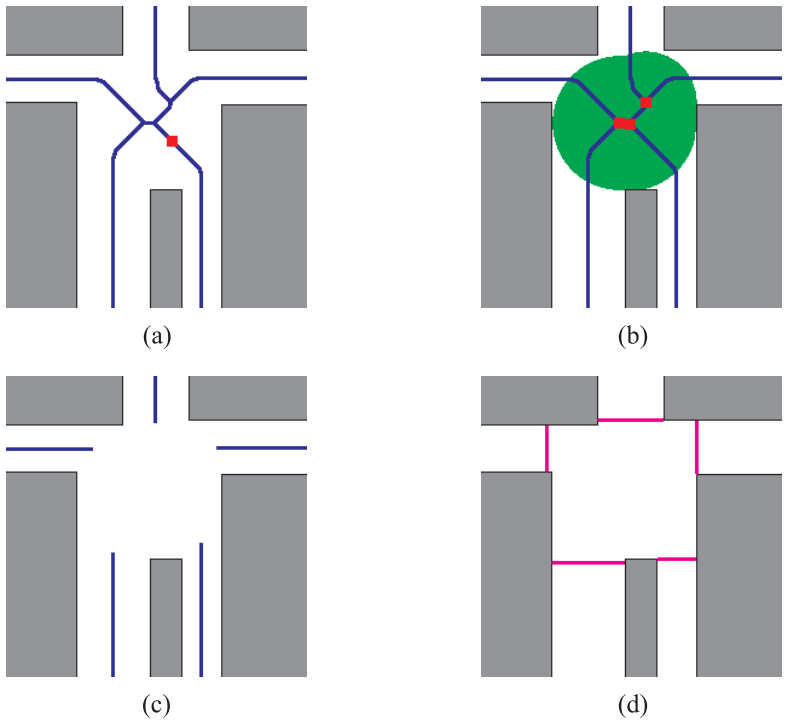


Fig. 9. Example of gateway detection. Here we show certain steps of the gateway algorithm. We begin with a model of the local surround. (a) We then calculate a pruned Voronoi skeleton. Here we use a thinning-based approximation of the Voronoi skeleton (Zhang and Suen 1984), which we have found to be much faster to calculate on slower processors. We then locate the closest point on the skeleton to the robot. (b) The robot then determines the "core" of the local region. (c) The algorithm ignores all portions of the skeleton inside of the core, only looking for gateways along portions of the skeleton outside of the core. (d) The algorithm looks for a local minimum in the rate of change of the Voronoi radius. These constrictions define the locations of gateways, while the skeleton itself defines the orientation of the gateway.

$$\begin{array}{ll} (\langle \tilde{\pi}_g^+, 1\rangle \leftrightarrow \langle g, out\rangle) & (\langle \tilde{\pi}_a^+, 1\rangle \leftrightarrow \langle g', in\rangle, \langle g, out\rangle) \\ \text{to} & \\ (\langle \tilde{\pi}_{g'}^+, 1\rangle \leftrightarrow \langle g', out\rangle) & (\langle \tilde{\pi}_a^-, 1\rangle \leftrightarrow \langle g, in\rangle, \langle g', out\rangle). \end{array}$$

- For each $\tilde{\pi}_g^+ \in S_p$ such that $\tilde{\pi}_g^- \notin S_p$, insert the association $(\langle \tilde{\pi}_g^-, 0\rangle \leftrightarrow \langle g, in\rangle)$ into the circular order of S_p, in a position determined by its failure of the path continuity test.

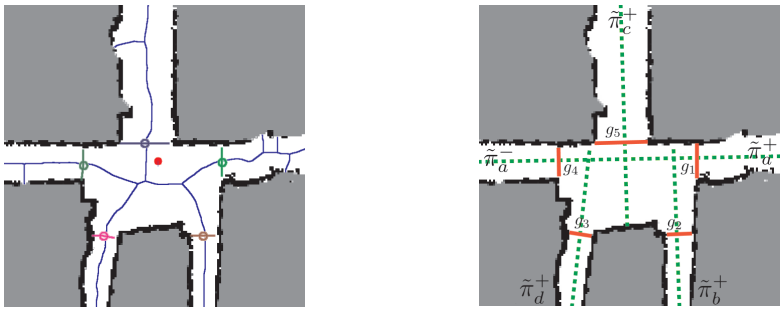
In our current implementation, the gateways g and g' belong to a single continuous path if: (1) a ray normal to the orientation of gateway g and centered at the midpoint of gateway g intersects the line segment that defines gateway g'; and (2) *vice versa* for a ray from g' towards gateway g. (Note that the failure of this test should determine a pair of gateways where the non-traversable path continuation falls between.)

At this point, the star S_p is a complete representation of the local topology of the neighborhood described by the LPM. Since this representation is expressed completely in terms of small-scale space (the gateways g and the directed local-paths $\tilde{\pi}_g^d$), we refer to this as the *small-scale star* (Figure 10(c)). Binding the directed local-paths to directed paths in the topological map of large-scale space implies the appropriate *large-scale star* (Figure 10(d)). This binding is part of the HSSH topological mapping process, and is discussed in Section 6.1.

5.2. Detecting Places

To explain local topology extraction, we provided examples where the robot was already at a place. Perhaps surprisingly, the method for constructing the local topology of a place neighborhood does not actually depend on being at a place neighborhood. Gateways can also be defined along paths, as they separate the robot from the "frontier" of the LPM. Therefore, if we recalculate gateways and local topology at each time step, we can very easily *detect* places.

We define the robot to be *on-path* when the local topology of the LPM contains exactly two gateways and exactly one path (e.g., Figure 11(a)). When the agent is on-path, it is selecting and executing control laws (and hence primitive motions) to perform a travel action. The LPM scrolls as the agent



(a) (b)

Small-scale star description

$$
\begin{array}{lcl}
((\langle \tilde{\pi}_a^+, 1\rangle & \leftrightarrow & \langle g_4, in\rangle, \langle g_1, out\rangle) \\
(\langle \tilde{\pi}_b^+, 1\rangle & \leftrightarrow & \langle g_2, out\rangle) \\
(\langle \tilde{\pi}_c^-, 0\rangle & \leftrightarrow & \langle g_5, in\rangle) \\
(\langle \tilde{\pi}_d^+, 1\rangle & \leftrightarrow & \langle g_3, out\rangle) \\
(\langle \tilde{\pi}_a^-, 1\rangle & \leftrightarrow & \langle g_1, in\rangle, \langle g_4, out\rangle) \\
(\langle \tilde{\pi}_d^-, 0\rangle & \leftrightarrow & \langle g_3, in\rangle) \\
(\langle \tilde{\pi}_c^+, 1\rangle & \leftrightarrow & \langle g_5, out\rangle) \\
(\langle \tilde{\pi}_b^-, 0\rangle & \leftrightarrow & \langle g_2, in\rangle))
\end{array}
$$

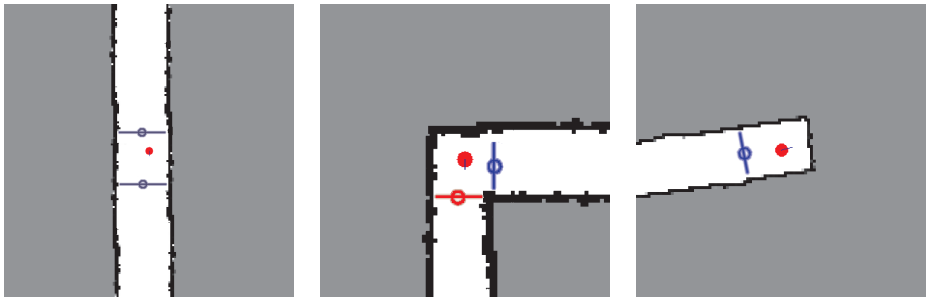
(c)

An example large-scale star abstraction

$$
\begin{array}{lcl}
((\langle \pi_1^+, 1\rangle & \leftrightarrow & q_1) \\
(\langle \pi_2^+, 1\rangle & \leftrightarrow & q_2) \\
(\langle \pi_3^-, 0\rangle & \leftrightarrow & q_3) \\
(\langle \pi_4^+, 1\rangle & \leftrightarrow & q_4) \\
(\langle \pi_1^-, 1\rangle & \leftrightarrow & q_5) \\
(\langle \pi_4^-, 0\rangle & \leftrightarrow & q_6) \\
(\langle \pi_3^+, 1\rangle & \leftrightarrow & q_7) \\
(\langle \pi_2^-, 0\rangle & \leftrightarrow & q_8))
\end{array}
$$

(d)

Fig. 10. Identifying gateways and local topology in the LPM. The LPM is implemented as a bounded occupancy grid. The robot is shown as a circle in the center of the LPM. (a) The gateways separating the core from the exits are defined. In our implementation this is done using a pruned Voronoi skeleton. (b) Gateway locations and directions are used to identify the directed local-paths and to determine which pairs satisfy the path continuity requirements. (c) The small-scale star enumerates directed local-paths in clockwise order, describing their traversability and association with gateways. Note: the robot entered the place via g_5; thus, it arrived on directed local-path $\tilde{\pi}_c^-$. (d) The large-scale star (Section 6.1) replaces local-paths with topological paths from the global topological map, and defines a distinctive state for each directed path at this place. This environment has five gateways, four paths, and eight distinctive states.



(a) (b) (c)

Fig. 11. More real-world gateways. Our current gateway algorithm uses a Voronoi skeleton to find the gateways surrounding a location. (a),(b) Even at locations with no Voronoi junction points, the gateway algorithm works. Example (a) shows the robot on a path, where two gateways on either side of the robot give a stable topology, and (b) shows the robot at a place. (c) At dead ends, there is only a single gateway.

moves, keeping the agent near its center cell, and serving as an observer for the local small-scale space.

When the agent is not on-path, it is in a place neighborhood. In this situation, the agent establishes a fixed correspon-

dence between the LPM and the structure of the place neighborhood. Here, the LPM serves as a local metrical map m_p of the place neighborhood (and does not scroll with the agent's motion within the place neighborhood). Thus, the number and location of places in an environment depends in part on the predetermined size of the LPM[13], although places are not sensitive to small changes in LPM size.

When the robot is not on-path, it either has more than one local-path, which occurs at intersections or open doorways, or one local-path with only one gateway, which occurs at dead ends. These are all places (Figure 11). There is a degenerate case where no gateways exist. Due to our implementation of gateways, this situation means there is no way out of the current location, so the robot's entire world is simply modeled by a single place and LPM.[14]

When traveling along a path, the robot may see multiple unaligned gateways and suspect it is at a place. Sometimes, false gateways appear in the LPM due to the boundary between observed free space (i.e. white cells in the occupancy grid) and unknown space (i.e. gray cells in the grid). This is often the case when the robot's sensors do not provide a 360° field of view, as with SICK-brand lidars. Before the robot commits itself to being at a place, it must perform some local exploration in the fixed map of the potential place to eliminate any false gateways. We have found that, for a robot with a 180° field of view, simply rotating in place eliminates most false gateways.

Using the local topology defined by gateways allows the robot to detect places more reliably than when using methods that simply look for Voronoi graph junctions. First, Voronoi graphs can have many spurious junctions. This is especially true given noisy sensors or environments, but even occurs in the face of no noise at small alcoves and other common architectural features. Similarly, complex intersections can have multiple junctions. The gateways and local topology can see one place, whereas a junction-based approach (including Delaunay triangle approaches (Silver et al. 2004)) must define multiple strangely connected places. Figures 9(b) and 10(a) both show multiple junctions at a single place. Additionally, there exist important places detected via the gateway approach, like L intersections, that contain no junctions at all after pruning the Voronoi graph (Figures 8(d)–(f)).

5.3. Selecting Local Motion Targets

Instead of relying on the dynamical system approach to motion used in the basic SSH, we introduce gateways as an alternative approach. Gateways provide a geometric method for control of motion – where midline or coastal navigation along paths is applicable. The motion of the robot in large-scale space can be adequately captured by noting which oriented gateways the robot passes through. Reconsider the example of abductive inference for a topological map in Figure 4 that modeled the world as points connected by lines. Compare this to the HSSH approach illustrated in Figure 12 where turns and travels correspond to moving towards gateways.

As discussed in Section 4.2, local motion planning consists of selecting a target pose in the LPM, computing a safe trajectory to it, executing the first step of that trajectory, sensing the environment, updating the LPM, and repeating the cycle. The selection of target poses for local motion control corresponds to the action or goal being pursued. There are three distinct cases.

- If the agent is not in a place neighborhood, it is *on-path*, in which case it is moving along the local-path in the LPM toward one of the two gateways. Just beyond the forward gateway, in the outward orientation, is an appropriate target for local motion planning; however, a more robust approach with respect to obstacle avoidance is to aim at a point well beyond the gateway, like the edge of the LPM. As the LPM scrolls, the gateway location is constantly refreshed. The robot never reaches the gateway until its location becomes stable (which only happens when the agent arrives at a place).[15]

- If the agent is in a place neighborhood, the LPM is fixed to the local environment, so motion planning is confined to the small-scale space of the place neighborhood. The agent may have a pragmatic destination within the place neighborhood, for example an intelligent wheelchair may have the goal of bringing its driver to her desk after entering her office, in which case the local motion target is a pose associated with that destination. Such motion targets can also be generated when exploring the fixed LPM of a potential place.

- The agent may be executing a turn action as part of a route through large-scale space. In this situation, the LPM is fixed in the local frame of reference, and a large-scale turn action corresponds to moving from an inward-facing oriented gateway to a location just beyond an outward-facing oriented gateway. After passing through the outbound gateway, the robot is in position to begin following another path. Note, that the TurnAround action simply corresponds to traveling past the same gateway the robot entered the place through, facing the outward instead of the inward orientation. Continuing along a path that passes through a place (no turn) also falls into this case.

13. One interesting avenue of future research is to try adapting the LPM size by environment characteristics.

14. There is another degenerate case when the robot is in the middle of a featureless environment. As mentioned in Section 3.5, the HSSH currently does not handle these types of environments.

15. Lee (1996) calls control algorithms that continuously re-plan for a moving point ahead as "red wagon" controllers.

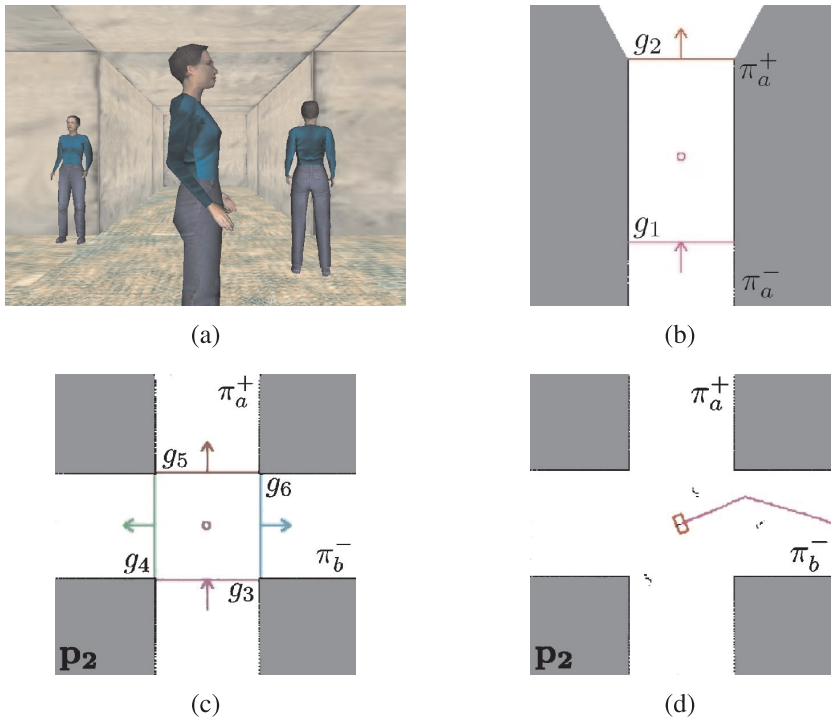


Fig. 12. Grounding control using gateways. (a) The example from Figure 4 is further examined in a simulated 3D office environment with obstacles. The gateways are found and drawn on the LPM in real-time, with arrows representing the outward orientations that leave the current area. The gateway associated with the robot's past motion is depicted using an arrow pointing in the inward orientation. (b) Traveling along directed path π_a^+ corresponds to aiming for an oriented gateway, e.g., $\langle g_2, out \rangle$, in the appropriate direction. The gateway is continually recomputed, which keeps moving the local motion target along the path, until it becomes stable at the entrance to a place. (c) Arriving at dstate q_2 at place p_2 corresponds to arriving at a gateway $\langle g_3, in \rangle$ associated with a directed local-path $\tilde{\pi}_a^+$ in the LPM for place p_2. The turn action from dstate q_2 to q_3 corresponds to local motion within the LPM through outward-facing oriented gateway $\langle g_6, out \rangle$ on directed local-path $\tilde{\pi}_b^-$. (d) In calculating local topology, "island" obstacles that are surrounded by free space are removed to ensure reliable gateway detection. Planning to move through a gateway requires consideration of these obstacles. Once the robot moves past gateway $\langle g_6, out \rangle$, two new aligned gateways appear that will flank the robot throughout the next travel action, as in image (b).

In certain scenarios, such as two large rooms connected by a doorway, it may be possible for an agent to move directly from one place neighborhood to another, moving between two distinct local topologies, without ever being significantly *on-path*. The SSH can accommodate this transition with a dummy travel action whose effect is simply to transition between the reference frames of two adjacent, or even slightly overlapping, places.[16]

16. Taking this idea to an extreme, the Atlas system (Bosse et al. 2003) creates new frames of reference based on feature counts, building a "patchwork" map of overlapping frames of references. However, if the entire environment is described in terms of overlapping place neighborhoods, the benefit of the topological map as a concise description of large-scale space is decreased. Likewise, the local and global distinctiveness of places is sacrificed.

6. HSSH Global Topological Level

The next two sections will address the problems of building a global topological map to describe the qualitative structure of large-scale space and building a global metrical map to describe its geometric structure within a single global frame of reference. We describe these two map-building problems separately, but their solutions benefit from each other and should be interleaved in future research (Section 10.2).

The first problem is to identify the best global topological map consistent with exploration experience. The process of generating possible topological maps from experience and testing them for consistency can provide formal guarantees that the correct map is generated and never discarded (Dudek et al. 1993). A logic-based theory of topological maps (Remolina and Kuipers 2004) makes explicit the assumptions upon which those guarantees depend.

If the robot knows it is in an environment with no loops, creating a topological map is quite easy. This is especially true given deterministic actions, as the robot simply moves deterministically between known places when it revisits parts of the environment. Even with non-deterministic actions, creating the topology of such environments is still possible (Tomatis et al. 2002). The difficulty in map-building arises from closing loops: determining when a newly-encountered place is the same as a previously-experienced place, and creating a hypothesized new loop in the topological map. When large loops in the environment result in structural ambiguity, a topological representation can concisely represent the loop-closing hypotheses by generating a single topological map for each qualitatively distinct alternative.

6.1. From Small-scale to Large-scale Star

In small-scale space, the LPM is used for the detection of gateways, local-paths, and places, and to create the local map m_p that is stored at places. The small-scale star describes both a circular order on the set of directed local-paths in a place neighborhood and also the correspondence between directed local-paths and oriented gateways. A place p in large-scale space is associated with the local map m_p, a model of the place neighborhood in small-scale space. At a place, each directed local-path $\tilde{\pi}^d$ in small-scale space corresponds to a directed path π^d in large-scale space. This allows us to determine the large-scale star that describes the circular ordering of topological directed paths at the place.

Assimilating the local topology of a place into the global topological map requires a one-to-one mapping between the directed local-paths in the small-scale star and a set of directed paths from the global topological map. In Figures 10(c),(d), we illustrate such a mapping between the local-paths, $\tilde{\pi}_a$, $\tilde{\pi}_b$, $\tilde{\pi}_c$, and $\tilde{\pi}_d$, and the corresponding global topological paths π_1, π_2, π_3, and π_4, respectively. To keep this example simple, we specified $+$ and $-$ on the directed paths to correspond consistently, but of course this need not be true in general.

In large-scale space, a distinctive state q corresponds uniquely to a place, a path, and a direction on that path (Equation (1)). Thus, the dstate q is at a particular place p, and there is a bijective association between a dstate and a directed local-path: $\psi_p(q) = \tilde{\pi}^d$ where $\tilde{\pi}^d \in S_p$. This implies that in the case where the directed local-path passes through the place, the distinctive state q will correspond with two different oriented gateways, one $\langle g, in \rangle$ entering the place neighborhood, and the other $\langle g', out \rangle$ departing from it.

An isomorphism $\phi : S \rightarrow S'$ between two stars implies a bijective mapping between the associated dstates as well. We will extend ϕ to write these implied mappings as $\phi(q) = q'$. For a topological map M^T, and the set P of places in M^T, we can now define the set of local place maps,

$$M^P = \{\langle p, m_p, S_p, \psi_p \rangle : p \in P\},$$

associating each place $p \in P$ with its local metrical map m_p, its local topology S_p, and ψ_p, the association between dstates and directed local-paths in the local topology.

Assuming that the LPM is sufficiently well explored, the set of directed local-paths and gateways in the small-scale star is complete, so the description of the distinctive states and directed paths in the circular order of the large-scale star is also complete. A *turn* action in large-scale space corresponds to motion in small-scale space within a place neighborhood from the inward-facing oriented gateway the robot arrived upon to an outward-facing oriented gateway (Figure 12(c)). Thus, for every pair of dstates q_i and q_j at the place, a causal schema for the turn action $\langle q_i, turn, q_j \rangle$ is implicitly defined. Exploration experience can now be described as an alternating sequence of travel actions and place neighborhoods, which simplifies construction of the global topological map (Figure 5).

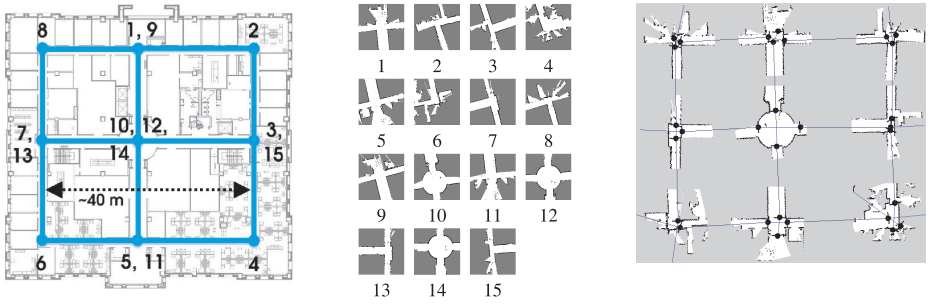
6.2. The Tree of Possible Topological Maps

The topological map-builder maintains a tree whose nodes are pairs $\langle M, q \rangle$, where M is a topological map (augmented below for the HSSH), and q is a distinctive state within M representing the robot's current position. The leaves of the tree represent all possible topological maps consistent with current experience (Dudek et al. 1993). The algorithm for growing the tree of possible topological maps was presented in Figure 5. This figure also describes the differences between map-building in the basic SSH and HSSH.

After each action a and resulting view v, we extend each map hypothesis at a leaf of the tree. If the current action moves within known territory, the map $\langle M, q \rangle$ will predict the resulting dstate q' and the view to be observed, so the hypothesis can be updated or refuted according to whether the prediction was correct or not. If the current action explores new territory, then either the resulting dstate is also new, or the action closes a loop and connects with a previously known dstate. Since there may be multiple possibilities that all match view v, the tree of topological map hypotheses will branch. For purposes of generating and testing candidate topological maps in the HSSH, we will extend the basic SSH topological map M^T with $M^P = \{(p, m_p, S_p, \psi_p) \ : \ p \in P\}$, the set of local metrical maps and local topologies of individual place neighborhoods,

$$M = \langle M^T, M^P \rangle.$$

In the SSH, a view v is an abstracted description of the agent's perception of the local environment from a distinctive state q. We select the level of description to ensure that the view is a deterministic function of the dstate ($v = o(q)$), although we allow perceptual aliasing (different states with the same view) (Kuipers and Beeson 2002). In the basic SSH, a view is a symbol, abstracting away the nature of the perceptual system, and views are matched only for equality. In the HSSH, we define a view to be the local topology S_p of the



(a) (b) (c)

Fig. 13. An environment with multiple nested loops. In the CAD drawing (a), we show the path traveled between places in the environment. We enumerate the sequence of places as experienced by the robot. (This exploration trace was also used for Figure 1.) In (b), we show the LPMs created at the places during the travel. We constrained the gateway algorithm in order to ignore open office doors and cubicle openings, which ensures places only at hallway intersections. The stars generated from these LPMs are used to search through the space of consistent topological maps. In (c), we show the unique topological map generated after matching local stars and LPMs. The map is overlaid with the LPMs generated at the places, with the gateways, and with the connections between gateways which lie on the same path.

current place p and the current directed local-path the robot is on; thus, the new view description is derived from the local topology, which is grounded in LPM m_p:

$$v = \langle S_p, \tilde{\pi}^d \rangle \text{ where } d \in \{+, -\}$$

$$= \langle S_p, \psi_p(q) \rangle.$$

Given two views v and v', we say that $match(v,v')$ holds iff there is an isomorphism $\phi : S \rightarrow S'$ such that $\phi(q) = q'$. That is, from the perspectives of the specified dstates, the local topologies match.

As exploration progresses, the map M is extended with new information. For example, after an exploration step that closes a loop in the map, the resulting map M' is M extended with a new dstate q' and assertions $\langle q, a, q' \rangle$, $v = o(q')$, and $q' = q_j$. A new version of $M^C = \langle Q, A, V, S, o \rangle$ is created, and the implications of the loop-closing assertion $q' = q_j$ propagate through new versions of M^T and M^P to unify place and path labels as necessary. Because we are matching complete local topologies in the HSSH, the tree of maps only branches on travel actions. Turn actions are already fully described by the large-scale star.

6.3. Topological Mapping Example

We applied an implementation of the HSSH map-building to a pre-specified route through an office environment including multiple nested loops. This office had a large number of cubicles and office doorways. To respect student and faculty privacy, we pruned the Voronoi skeleton so that Voronoi branches, thus gateways, were defined only for large hallway intersections, not at doorways or cubicle openings. The environment, as defined by the robot, contained six paths and nine places with four distinct local topologies. Figure 13 shows the exploration route as a sequence of place visits, the sequence of LPMs observed at successive place neighborhoods, and the unique simplest topological map that resulted from the mapping algorithm, with LPMs overlaid at corresponding places in the correct topological map.

After a sequence of 14 travel actions, the topological mapper finds 83 possible configurations of the environment that are consistent with the observed local topologies and the topological axioms – that is there exist 83 leaves in the tree of maps. The prioritized circumscription (Remolina and Kuipers 2004) on this set of maps produces four minimal models. All but one of these can be eliminated with further exploration or by simply matching LPMs using the alignments specified by the four minimal maps. This final map model is the correct topological representation of the environment.

If we assume planarity of the environment, we can use a more sophisticated version of the topological map-building algorithm (Savelli and Kuipers 2004) that rules out many more models as inconsistent. Here, there are only 46 consistent configurations of the exploration experience, and the circumscription policy produces a single minimal model, which is the correct topological map of the environment (Figure 13(c)). Currently, our implementation can build the complete tree of maps for this exploration trace and determine the unique minimal map of this environment in ∼200 ms on the robot's Pentium III 450 MHz processor. Notably, the results presented on this office environment would be unchanged if the path segments were longer or even very convoluted, as the number of

places and paths would not change. Additionally, the tree of maps ensures the correct map is never discarded.

6.4. Levels of Spatial and Temporal Granularity

At this point, we summarize the three different levels of granularity, with different ontologies, that we are using to describe space and time.

The agent's experience is a trajectory through the environment. At the SSH Control Level and in the LPM, the trajectory is represented using a fine-grained representation for time t, pose x, motor signal u, and sensory image z. These are used both for control laws, and for simultaneous localization and mapping to build the LPM. Expanding Figure 7, the agent's exploration experience is described by

$$\begin{array}{ccccccccccccc} & & \cdots & & u_{t-1} & & u_t & & u_{t+1} & & \cdots & & u_N \\ & & & & \downarrow & & \downarrow & & \downarrow & & & & \downarrow \\ x_0 & \rightarrow & \cdots & \rightarrow & x_{t-1} & \rightarrow & x_t & \rightarrow & x_{t+1} & \rightarrow & \cdots & \rightarrow & x_N \\ \downarrow & & & & \downarrow & & \downarrow & & \downarrow & & & & \downarrow \\ z_0 & & \cdots & & z_{t-1} & & z_t & & z_{t+1} & & \cdots & & z_N. \end{array}$$

At the SSH Causal Level (which is part of the topological map), exploration experience is described by an alternating sequence of actions and distinctive states, with each distinctive state associated with a view:

$$\begin{array}{ccccccccccc} q_0 & a_1 & q_1 & a_2 & q_2 & \cdots & q_{n-1} & a_n & q_n \\ | & & | & & | & & | & & | \\ v_0 & & v_1 & & v_2 & \cdots & v_{n-1} & & v_n. \end{array}$$

In both the basic and hybrid versions of the SSH, distinctive states q correspond to being at a place, facing along a directed path. In the basic SSH, the distinctive states q are grounded by isolated distinctive states $\bar{x}$ where hill-climbing control laws terminate. In the HSSH, dstates are grounded by a directed local-path extracted from the LPM of a place neighborhood.

At the SSH Topological Level, a particular place p_j can correspond to several distinctive states, say q_{i-1} and q_i and the turn action a_i between them. A travel action a_{i+1} from q_i at p_j to q_{i+1} at a different place p_{j+1} can be used to infer the displacement λ_{j+1}, which is the pose of place p_{j+1} in the frame of reference of place p_j. This lets us abstract the sequence of distinctive states and actions to an alternating sequence of place p_j and displacements λ_j as follows:

$$p_0 \quad \lambda_1 \quad p_1 \quad \lambda_2 \quad p_2 \quad \cdots \quad p_{m-1} \quad \lambda_m \quad p_m.$$

As described in Section 7 and illustrated in Figure 14, in order to define the λ_i, each place neighborhood must have its own frame of reference and we must select a set of distinguished time-points $0 \leq t_0 < t_1 < \cdots \leq t_N = N$ such that

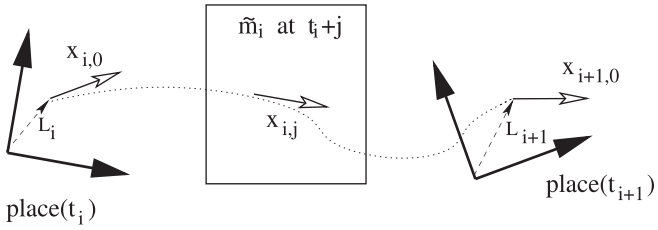


Fig. 14. Defining local frames of reference. The agent creates the local scrolling map $\tilde{m}_i$ when traveling between places. The agent's poses at the distinguished time-points t_i and t_{i+1} are $L_i = [x_{i,0}]_{place(t_i)}$ and $L_{i+1} = [x_{i+1,0}]_{place(t_{i+1})}$. The displacement between the two place frames of reference is $\lambda_{i+1} = L_i \oplus [x_{i+1,0}]_{\tilde{m}_i} \oplus (\ominus L_{i+1})$.

adjacent time-points belong to different place neighborhoods, and the pose x_{t_i} at each time-point t_i can be unambiguously localized in its place neighborhood. To fit this into the SSH causal framework, we select distinguished time-points at the termination of each travel action: in the basic SSH, this is after hill-climbing terminates, and in the HSSH, this is after a place is detected. In the HSSH, the dividing poses are near the incoming gateways in place neighborhoods. The net effect of the turn and travel actions between these dividing points is used to estimate the displacements λ_i between the frames of reference of adjacent place neighborhoods connected by path segments.

7. HSSH Global Metrical Level

The topological map identifies a discrete set of places, each with its own local metrical map within its own frame of reference. The topological map also encodes decisions about how loops are closed and which aliased local neighborhoods represent the same places. The global metrical map is built on the structural skeleton provided by the topological map (Modayil et al. 2004). The steps in building the global metrical map are: (1) describe the *displacements* $\lambda = \{\lambda_i\}$, each describing the change in pose from one place neighborhood to the next in the frame of reference of the first; (2) describe the *layout* $\chi = \{\chi_p\}$, specifying the poses of places in a global frame of reference; (3) describe the *trajectory* $x = \{x_t\}$ of robot poses within the global frame; and (4) create the *global map* m^* from sensor readings given the trajectory.

7.1. Terminology

The global topology τ, used below, consists of the set $M^P = \{\langle p, m_p, S_p, \psi_p \rangle : p \in P\}$ of places with their local information, the set of distinguished time-points $0 \leq t_0 < t_1 < \cdots <$

$t_n \leq N$ that divide the fine-grained sequence of exploration experience into segments corresponding to travel between adjacent place neighborhoods, and the relation $place(t_i) = p_j$ between them. It is convenient to relabel the variables x, z, and u, defining $x_{i,j} \equiv x_{t_i+j}$. At each distinguished time-point t_i, where $place(t_i) = p_j \in P$ and $place(t_{i+1}) \neq place(t_i)$, the agent is localized in the local metrical map m_{p_j}.

Much of our metrical inference consists of defining an appropriate set of reference frames, and estimating the values of local and non-local metrical quantities. Many of these concepts can be simply understood by examining Figure 14.

$[x]_p$: The coordinates of the pose x in the frame of reference of place p.

O_p : The pose x such that $[x]_p = (0, 0, 0)$.

$L_i \equiv [x_{t_i}]_{place(t_i)}$: The coordinates of the pose x_{t_i} in the reference frame of $place(t_i)$.

$\tilde{m}_i$: The scrolling map that models the agent's surroundings between distinctive time-points t_i and t_{i+1}. The map's origin is defined as the agent's pose at time t_i. That is, $O_{\tilde{m}_i} = x_{t_i}$.

$\lambda_i \equiv [O_{place(t_i)}]_{place(t_{i-1})}$: The location of $O_{place(t_i)}$ in the reference frame of $place(t_{i-1})$, estimated using the experience from t_{i-1} to t_i.

$\chi_p \equiv [O_p]_{m^*}$: The pose of O_p in the global reference frame of m^*.

m^* : The global metrical map.

7.2. The Theory of the Global Metrical Map

To build a global metrical map m^*, we want to find the maximum-likelihood path the robot traveled, using the topological skeleton in addition to odometry. As discussed in Section 4.1, the joint probability of the pose history x and the global map m^* can be decomposed as

$$P(x, m^*|z, u) = P(m^*|x, z, u) \cdot P(x|z, u)$$

by the chain rule for probabilities. This decomposition is valuable since $P(m^*|x, z, u)$ (map-building given accurate localization) can be computed analytically and incrementally for popular map types, so we can focus our attention on $P(x|z, u)$ (pose estimation).

To include the effect of possible global topologies τ on pose estimation, we marginalize over the space of topologies. If we assume that the correct global topology $\bar{\tau}$ has been identified, only one topological hypothesis $\tau = \bar{\tau}$ has non-zero probability:

$$\begin{aligned} P(x|z, u) &= \sum_{\tau} P(x|z, u, \tau) \cdot P(\tau|z, u) \\ &= P(x|z, u, \bar{\tau}). \end{aligned}$$

On the other hand, suppose there are multiple topologies τ with significantly non-zero values of $P(\tau|z, u)$. While the weighted sum provides a mathematically correct characterization of the probability distribution $P(x|z, u)$, it can easily lead to a nonsensical metrical map due to the dramatic qualitative impact of topological structure on the metrical map. Thus the summation should be regarded as describing a disjunction over topological maps, with $P(\tau|z, u)$ being the likelihood of each map. This is exactly the tree of possible topological maps we have already constructed. Therefore, even in the case where there are multiple plausible topological maps, we will construct global metrical maps for each one individually.

Given a particular topology $\bar{\tau}$, we can marginalize over the global poses of all topological places $\chi = \chi_i$ and their estimated relative displacements λ:

$$P(x|z, u, \bar{\tau}) = \int\int P(x|\chi, \lambda, z, u, \bar{\tau}) \cdot P(\chi|\lambda, z, u, \bar{\tau}) \cdot P(\lambda|z, u, \bar{\tau})\, d\lambda\, d\chi.$$

Because x is conditionally independent of λ given χ, and χ is conditionally independent of z, u given λ, we can simplify this equation:

$$P(x|z, u, \bar{\tau}) = \int P(x|\chi, z, u, \bar{\tau}) \cdot \int P(\chi|\lambda, \bar{\tau}) \cdot P(\lambda|z, u, \bar{\tau})\, d\lambda d\chi.$$

We divide this equation into simpler components, defining the following functions representing probability distributions over their arguments:[17]

$$\begin{aligned} F(\lambda) &= P(\lambda|z, u, \bar{\tau}), \\ G(\chi) &= \int P(\chi|\lambda, \bar{\tau})\, F(\lambda)\, d\lambda, \\ H(x) &= \int P(x|\chi, z, u, \bar{\tau})\, G(\chi)\, d\chi. \end{aligned}$$

Thus, we use the topological map $\bar{\tau}$ to factor the localization term $P(x|z, u) = H(x)$ into three separate probability distributions: place-to-place displacements $F(\lambda)$ derived from local metrical maps; the metrical layout $G(\chi)$ of places in the global topological map; and the global metrical layout $H(x)$ of the robot's pose trajectory. Finally, we can combine the pose trajectory with $P(m^*|x, z, u)$ to define the joint distribution $P(x, m^*|z, u)$.

17. We assume that there is no opportunity for confusion between these probability functions F, G, and H, and the dynamical system functions F, G, and H_i used in Section 3.1.

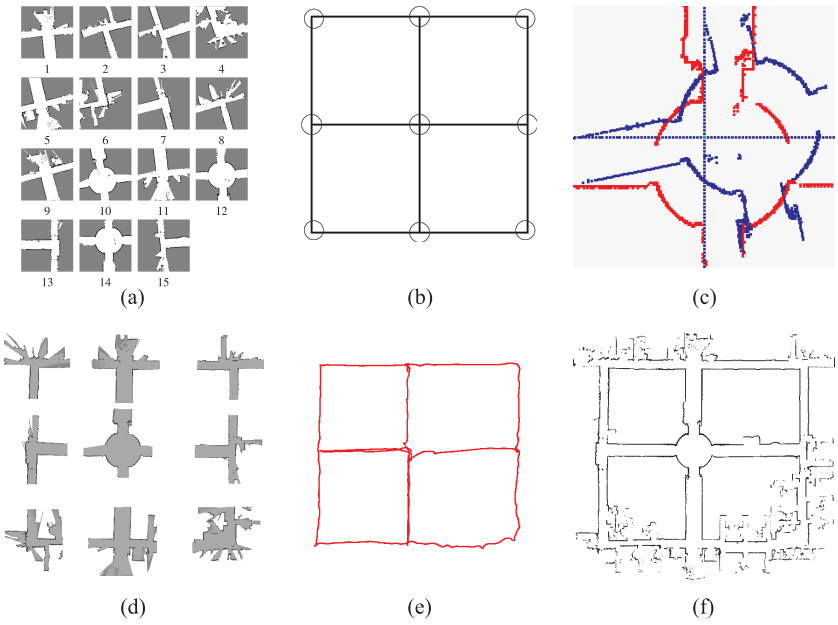


Fig. 15. Global map-building process. (a) The sequence of local place maps m_p experienced. (b) The unique topological map consistent with topological and planarity constraints. (c) We determine λ_i for loop closures by finding the offset between the current pose and the place origin (defined on the initial place visit). (d) The layout χ derived from the topological map and the place-to-place displacements λ. (e) The pose trajectory $x(t)$ anchored at points where the robot is localized in place neighborhoods in the layout χ. (f) Given the localized pose trajectory $x(t)$ in the global frame of reference, the global metrical map m^* is created accurately and efficiently. Compare with Figure 1.

7.3. Global Mapping Example

Here we detail each step of creating the global map and discuss our current implementation, which runs offline. Figure 15 demonstrates the stages of creating an accurate global metrical map of a large, complex environment using these methods.

7.3.1. Estimating $F(\lambda)$

Given the topology τ, we can compute $F(\lambda)$. Each λ_i corresponds to a single experience of a path segment. Since closing large loops is not a problem when considering a single path segment, traditional SLAM methods may be employed to estimate $F(\lambda)$ by decoupling it into a set of independent probabilities:

$$D_i = z_{i,0}, \ldots, z_{i,n_i}, u_{i,1}, \ldots, u_{i+1,0},$$

$$F_i(\lambda_i) = P(\lambda_i | D_{i-1}, L_{i-1}, L_i),$$

$$F(\lambda) = \prod_{i=1}^{n} F_i(\lambda_i).$$

See Figure 14 to understand L. Our current implementation is an incremental maximum-likelihood method (Fox et al. 1999), modeling each $F_i(\lambda_i)$ as a Gaussian.

Using the notation of the compounding operator (Smith et al. 1990), we compute the distribution of λ_i by composing three uncertain vectors: the vector L_{i-1} from $O_{place(t_{i-1})}$ to $x_{i-1,0}$; the vector $[x_{i,0}]_{\tilde{m}_{i-1}}$ from $x_{i-1,0}$ to $x_{i,0}$; and finally the vector $-L_i$ from $x_{i,0}$ to $O_{place(t_i)}$:[18]

$$F_i(\lambda_i) = P(\lambda_i = (L_{i-1} \oplus [x_{i,0}]_{\tilde{m}_{i-1}} \oplus (\ominus L_i))).$$

The essential connection is that the pose $x_{i,0}$ at the end of a path segment is described in the frame of reference of place p_{i-1} by the expression $L_{i-1} \oplus [x_{i,0}]_{\tilde{m}_{i-1}}$, and simultaneously in the frame of reference of place p_i by L_i.

The problem of estimating $[x_{i,0}]_{\tilde{m}_{i-1}}$ is relatively simple along individual path segments, since loops cannot be involved. The more difficult problem arises from determining $\ominus L_i$ after a loop closure. Here we need to align the map m_{p_i} with a previously stored map m_{p_h} in order to determine $[O_{p_h}]_{p_i}$, which allows us to solve $\ominus L_i$. Matching maps can be expensive and can lead to false positives due to local minima (e.g., two LPMs of a + intersection can be matched four ways). To eliminate this problem, we first align the LPMs based on the locations of corresponding gateways, consistent

18. Given two poses a and b, we write $[b]_a$ for the coordinates of b in the frame where a lies at the origin and faces along the positive x-axis (Smith et al. 1990). Then, $[c]_a = [b]_a \oplus [c]_b$. The inverse operator is $[b]_a = \ominus[a]_b$.

with m_h, S_h, ψ_h and m_i, S_i, ψ_i, before refining the alignment using the obstacles and free space of the LPMs. (Figure 15(c) omits this gateway alignment step in order to better illustrate the process of LPM alignment.)

7.3.2. Estimating $G(\chi)$

The layout $\chi = \{\chi_p\}$ represents the poses of the places in the topological map, with respect to the frame of reference of the global metrical map m^*. $G(\chi)$ is a probability density function over possible layouts χ. Among other things, it reflects the distortion in the place layout due to a loop-closing hypothesis, compared with the observed displacements λ.

Given the topological map, which specifies the data association between observations and places, we can evaluate $G(\chi)$ for an arbitrary distribution of $F(\lambda)$. For a particular value of χ, $P(\chi|\lambda, \bar{\tau})$ will only be non-zero for a single value of λ, namely when each $\lambda_i = (\ominus \chi_{place(t_{i-1})}) \oplus \chi_{place(t_i)}$. Hence, $P(\chi|\lambda, \bar{\tau})$ is a Dirac delta function, which gives us a simple expression for $G(\chi)$:

$$G(\chi) = \int P(\chi|\lambda, \bar{\tau})\ F(\lambda)\ d\lambda,$$

$$= \prod_{i=1}^{n} F_i((\ominus \chi_{place(t_{i-1})}) \oplus \chi_{place(t_i)}).$$

When $F(\lambda)$ is represented as a Gaussian, an Extended Kalman Filter (EKF) is a simple way to approximate $G(\chi)$. The idea is to consider place p to be a landmark with pose χ_p. These landmarks are observed one at a time, linked by actions λ_i. This is essentially the classic approach of Smith et al. (1990). Given Gaussian uncertainty along each action u_i connecting the n robot poses, along with constraints that give Gaussian uncertainty between poses taken from multiple visits to the same place (to associate poses after loop closures), we can solve for $H(x)$ in time $O(n \log n)$ using the sparse matrix methods of Konolige (2004). However, often we may only want $G(\chi)$, which can be computed in $O(m \log m)$ time for m places, where $m \ll n$.

In our current implementation, we utilize a hill-climbing search to quickly converge to a local maximum of $G(\chi)$ (Figure 15(d)). The Levenberg–Marquardt algorithm for non-linear optimization (Press et al. 1992) treats the λ_i as "springs" between the poses of the places p_k in χ, and relaxes their configuration to reach a local minimum-energy configuration. Efficient estimations of this non-linear optimization also exist (Olson et al. 2006). A good initial layout χ for this hill-climbing search can be derived from the displacements λ_i, which represent SLAM-corrected odometry from the scrolling map. We use the term $\hat{\chi}$ to denote the computed estimate of $G(\chi)$.

7.3.3. Estimating $H(x)$

An extended Kalman filter can be used to estimate $H(x)$ using $G(\chi)$ and individual pose covariances from the experienced trajectory. Alternatively, if accurate pose covariances are not available, a simple method can estimate the maximum-likelihood trajectory through the environment. We calculate the independent trajectory $H_i(x)$ along each path segment, as each place location is fully determined by a global layout χ. In most cases, there will be some discrepancy between the measured distance λ_i along the path segment and the fixed distance between the places in χ. We transform the experienced motion along the path segment to fit the global path segment distance using a simple affine transformation. This process is similar to methods of distributing odometry error after closing a loop in a global metrical map (Thrun et al. 2000a).

For each trajectory between adjacent places p_i and p_{i+1}, we transform the relative, incremental displacements $\Delta(\mathsf{x}, \mathsf{y}, \theta)$ from the pose estimates in the scrolling LPM $\tilde{m}_i$ into relative displacements $\xi_{i,j}$ in the global frame of reference. This uses a simple affine transformation T_i to anchor the beginning of the LPM experience to the global frame of reference:

$$[x_{i,0}]_{m^*} = \chi_{place_{t_i}} \oplus L_i,$$

$$[x_{i,j}]_{m^*} \equiv T_i([x_{i,j}]_{\tilde{m}_i}),$$

$$\xi_{i,j} \equiv [x_{i,j}]_{m^*} - [x_{i,j-1}]_{m^*}.$$

We compute a final trajectory for x from the set of incremental displacements $\hat{\xi}$ by satisfying the constraint that the travel experience between the places must fit the globally defined distance between the places. This uses another affine transformation T_i' to map the final pose along the path segment (x_{i,n_i}) to the global frame of reference:

$$[x_{i+1,0}]_{m^*} = \chi_{place_{t_{i+1}}} \oplus L_{i+1},$$

$$[x_{i+1,0}]_{m^*} - [x_{i,0}]_{m^*} \equiv T_i'([x_{i,n_i}]_{m^*} - [x_{i,0}]_{m^*}),$$

$$\equiv T_i'(\sum_{j=1}^{n_i} \xi_{i,j}),$$

$$\hat{\xi}_{i,j} \equiv T_i'(\xi_{i,j}).$$

7.3.4. Creating a Map m^*

The trajectory above can be used as a starting trajectory for gradient descent methods to align the pose positions with map estimates to converge upon a locally optimal map (Lu and Milios 1997; Thrun et al. 2000a). A more principled approach is to run a Rao-Blackwellized particle-filtering algorithm, using the maximum-likelihood trajectory as the mean of a proposal distribution: $P(x, m|z, u) = P(m|x, z, u) \cdot H(x)$. However, we

have found that in practice the x values defined by the above scaling method adequately approximate the mode of the posterior (Modayil et al. 2004); thus the global map can be built by projecting the recorded range measurements from poses in the new global coordinates. The final map produced from the topological skeleton is shown in Figure 15(f). Compare this to Figure 1(d) to see the improved map.

8. Complexity in the HSSH

The different components of the HSSH fulfill different complexity requirements. The algorithms that pertain to small-scale space operate in real-time on the robot, and are designed for constant run-time complexity at each time step. The algorithms that pertain to large-scale space depend on the number of places and paths in the environment, and thus require more computation with increased topological ambiguities. Nonetheless, the topological complexity terms are functions of variables with far smaller values than the number of poses during exploration, due to the coarse granularity of the topological map.

The theoretical run-time complexity for all small-scale space operations is bounded by a constant due to the fixed size of the LPM. Updating the map is a function of the observation size and the resolution of the grid, but these are constant with respect to the number of actions and observations gathered during exploration. Localization with the fixed-sized LPM takes a constant amount of time since incremental localization within a fixed-size LPM requires a bounded number of particles. Computing the local topology for an LPM (generating a thinned skeleton, pruning the skeleton, and finding gateways) is linear in the number of grid cells, so for a fixed-size LPM this algorithm also has a constant run-time complexity.

In practice, the robot is able to update the LPM, use the LPM for local motion planning, and compute the local topology from the LPM in real-time for a grid size of 300×300 with 10 cm cells. Although the LPM must have enough resolution to support control, localization, and topological identification, it must also be small enough to allow these algorithms to run in real-time. Highly detailed models of the small-scale space for visualization or other purposes can be generated along with the LPM, but they are not required for the mapping process.

Topological maps for metrically large spaces can be computed efficiently in practice since the complexity grows with the size of the topological exploration instead of the distance traveled; however, in general, the complexity of learning the topological map can be exponential. Let n be the number of poses in the exploration trajectory; let m ($m \ll n$) be the number of topological places; let k ($k < m$) be the maximum number of places matching an observed local topology; and let l be the maximum number of directed local-paths in the local topologies (often $l \leq 4$). For example, for the environment in Figure 13, $n \approx 7,300$, $m = 9$, $k = 4$, and $l = 4$. In the HSSH, the maximum branching factor in the tree of maps is $k + 1$. Branches only occur when the robot travels between two connected places for the first time, which can only happen at most $ml/2$ times. This means the size of the tree of maps is $O(k^m)$; thus, computing the tree of maps is exponential in m (not in n). In the HSSH, the exponent decreases by at least a factor of three compared with the basic SSH version due to branching only on travels, not on turns, and matching local topologies of places.[19]

Savelli and Kuipers (2004) show that the planarity constraint gives an additional improvement in the branching factor k by rejecting many loop-closing hypotheses. They also point out that, for each map m_i in the tree to be expanded, the reduction of the branching factor k_i due to the planarity constraint is proportional to the number of closed loops already present in m_i. In other words, "the more loops [that] have been closed, the more topologically compact the map must be, and therefore the fewer ways there are to close new loops while preserving planarity," which reduces the branching factor further.

Once the topological map is known, the computation of a global metrical map will be linear in the number of poses n in the exploration trace. The work in generating the LPMs has a constant run-time, so is linear in n for the complete trace. Computing the distribution of the relative place displacements F is linear in n. Computing the layout of the places G uses an iterative non-linear optimizer whose computation is $O(m \log m)$, which, for $m \ll n$, is bounded by $O(n)$. Computing the pose layout H, and inserting the scans along the poses in the final map is again linear in n.

To summarize, the computation of the LPM and the local topology is done in constant time per pose. Additionally, for a specific global topological map, exploration and the construction of the global metrical map is linear in the length of the exploration. When constructing the global topological map, the worst-case number of loop-closing hypotheses is the hyper-exponential Bell's number (Ranganathan and Dellaert 2005), but this is a function of the number of topological places, not the number of poses. In practice, this number can be made much smaller by exploration strategies that close smaller loops earlier. A detailed examination of how exploration strategy affects the number of topological hypotheses is proposed for future work (Section 10.2).

9. Summary

We have presented a hybrid metrical/topological framework that processes information at both small-scale and large-scale abstractions. Our *Hybrid Spatial Semantic Hierarchy* is inspired by human cognitive maps; thus, it represents the environment using human-like concepts, such as places and paths,

19. There are at most $ml/2$ unique travel actions. There are at most l turns at each of the m places. Thus, in the worst-case environment, we have ml turns and $ml/2$ travels, resulting in $3ml/2$ actions in the *basic* SSH.

which support hierarchical navigation, human–robot interaction, and logical reasoning. Specifically, we have focused on the problem of map-building – discussing how the HSSH builds metrical representations for local small-scale spaces, finds a topological map representing the qualitative structure of large-scale space, and constructs a metrical representation for large-scale space in a single global frame of reference by building on the skeleton provided by the topological map.

Unlike many robotic implementations that attempt to build a monolithic, Cartesian global metrical map, we propose an alternative approach that handles closing large loops by hypothesizing symbolic place matches. This ensures that all possible loop closures are considered, not just ones where the robot, with accumulated odometry error, happens to be near some older portion of the map. The minimal topological map that results from large-scale exploration is sufficient for navigation and necessary for efficient planning, especially to rule out alternative topological structures during exploration.

The thrust of this paper has been to formally describe how concepts of large-scale space can be grounded in the robot's low-level observations. This problem has hindered topological map-building research, as it is an example of the hard AI problem of *symbol grounding* (Harnad 1990). Our innovation has been to utilize metrical approaches to model the immediate, local surround of the robot in order to ground *gateways* in small-scale space. Gateways provide the robot with local motion targets that facilitate control along paths. They also provide a *local topology* description of the local surround, useful for detecting and describing places and the paths that emanate from places.

We have demonstrated an implementation of the HSSH within an environment with fairly large, nested loop closures. The results support our claims of efficient, online map-building in the presence of multiple loop closures. We have demonstrated that a global layout of places is easily achieved given a topological map hypothesis, and a full global metrical map can be accurately achieved by filling in exploration experience along the path segments that connect places in the environment.

10. Future Work

There are obvious avenues of future work at all levels of the HSSH: creating semantically labeled LPMs using vision, demonstrating a HSSH interface that improves human–robot navigation tasks, and exploring very large environments to demonstrate claims about scalability. Below we discuss several specific issues that relate directly to the issues in this paper.

10.1. Gateways for Coastal Navigation

In Section 5.1.1, we describe an initial "constriction-based" algorithm for gateway detection that works in well-structured LPMs with boundaries on both sides of the underlying paths. We are currently utilizing a new "anchor-based" gateway algorithm that is essentially functionally equivalent to our constriction-based algorithm in corridor environments; however, it also handles *coastal navigation* scenarios, where constrictions do not exist, as well as improving certain boundary cases that can occur at places with no Voronoi junctions. We refer the reader to the dissertation work by Beeson (2008, Chapter 6) for a detailed discussion of the "anchor-based" gateway algorithm, including an empirical evaluation that shows the robustness of this new gateway algorithm under noisy conditions and using low-resolution LPMs.

Given gateways that define paths along the perimeter of walls, our robot can explore and map the areas around the outside of buildings or rooms that are larger than the LPM size. When using the new gateway algorithm, upon entering a large room, there will be paths to follow, at least around the edge of the room (see Figure 16(a)). Even without a global metrical map, the robot could find places at the corners of the room, and paths between them (Figures 16(b)–(d)). By using the LPM and the symbolic local topology of the detected places, the robot has enough evidence from local information along the paths to know that it was circumnavigating a large space.

For many navigation tasks such a model may be sufficient; however, by, using the relative displacements λ to calculate a global layout $\hat{\chi}$, a metrical map of the obstacles near the walls of large rooms can be created. Starting with a global metrical map near the walls of a large room, it should be possible to define control laws that set off into unknown space, using SLAM and/or dead reckoning to stay localized in the global frame of reference. Such a strategy could estimate where the robot should intersect the far side of the room, and compare that with its observation when it actually arrived, in order to create a new kind of "path" across the open space. This strategy should make it possible to find "islands" of interest in the middle of unexplored space in the middle of large open rooms. The perimeter of these islands may also be explored using coastal navigation.

10.2. Efficient Expansion of the Tree of Maps

Currently, the tree of maps contains every topological map consistent with exploration experience and the topological axioms. This guarantees soundness, which is useful in the case where observations refute the current best map and the next best map must be identified. However, there remain two related problems that need to be addressed in future work. First is the need for a reliable method to identify the best candidate among a set of possible topological maps, given odometry and perceptual information (Ranganathan et al. 2006). Second is the need to reduce the tree of maps from a "breadth-first" search to a more focused search that tracks a small number of maps at a time.

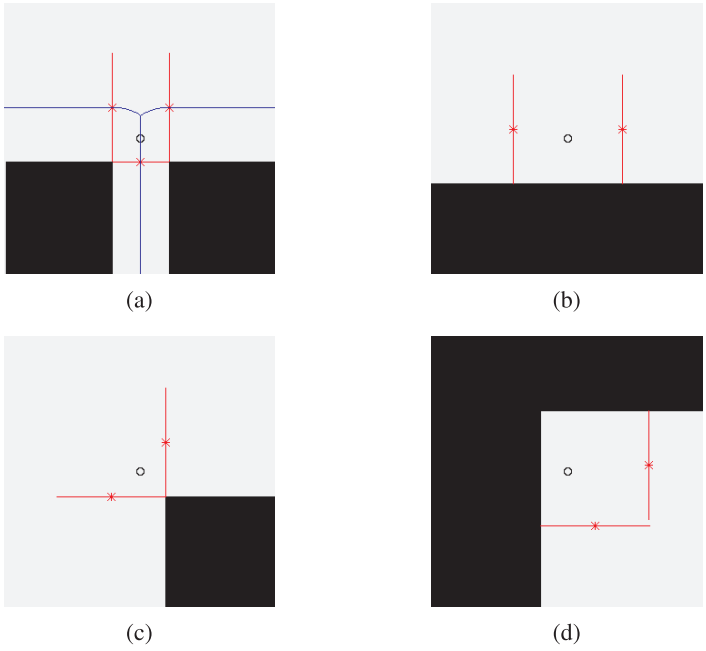


Fig. 16. Coastal navigation gateways. In regions of the environment where corridors cease to exist, the traditional Voronoi graph will lead the robot away from all nearby obstacles. Instead we utilize the extended Voronoi graph (EVG) (Beeson et al. 2005), which is equivalent to the Voronoi graph in corridors but switches to perimeter following at a maximum distance threshold. (a) The "constriction-based" gateway algorithm described in Section 5.1.1 is not applicable in coastal navigation scenarios, because branches of the EVG may have no nearby constrictions. Our new "anchor-based" gateway algorithm (Beeson 2008), handles these situations in addition to any corridor situations a robot will encounter. (b)–(d) The anchor-based algorithm works when the path is defined by a single wall, and in the convex and concave corners encountered in large rooms or when navigating the exterior of buildings.

In Section 6.3, we identified the "best" map as the simplest one based on a prioritized circumscription policy over the models generated by the non-monotonic theory of topological maps (Remolina and Kuipers 2004). This is sufficient for the environment in Figure 15, but Savelli and Kuipers (2004) describe larger environments where extreme symmetry and aliasing cannot so easily be resolved by purely qualitative methods, as the tree of maps grows too large to maintain in real-time.[20] These are not entirely unrealistic examples, since large grid-structured neighborhoods in real cities provide opportunities for vast topological ambiguity (Lynch 1960).

In the example of Section 6.3, the exploration sequence was provided to the robot. One obvious improvement that will limit the number of map hypotheses in future work is to perform *active exploration* that occasionally exploits knowledge of asymmetries in the environment to eliminate entire branches from the tree of maps. Such strategies are similar to the localization procedures advocated by proponents of DFA-style maps (Kuipers and Byun 1991; Dean et al. 1995; Rekleitis et al. 1999). Dudek et al. (1991) propose an exploration algorithm that finds the correct topological structure in polynomial number of travel actions, but this requires the robot to drop markers and backtrack to determine which loop-closing hypothesis was correct.

Along with utilizing intelligent exploration strategies, we would like reduce the tree of maps by drawing on perceptual information currently unused in the topological map-building process. We should be able to use observational data, such as the likelihood of the global metrical layout $P(\hat{\chi}|\lambda, \bar{\tau})$, probabilistic local topology matching, or the likelihood of visual observations at places (Cummins and Newman 2008), to define weights on the tree of maps. These weights should allow us to have a quantitative ordering on the map hypotheses, and should allow best-first expansion of the tree that focuses on a limited number of highly ranked candidates at a time. This should allow the robot to map larger environments including those with large amounts of symmetry and perceptual aliasing.

20. Ranganathan and Dellaert (2005) claim that, because (in the worst case) the number of aliased places grows with the amount of exploration experience, the number of possible topological maps is given by Bell's number, which grows hyper-exponentially with the number of perceptually aliased places.

Acknowledgments

This work has taken place in the Intelligent Robotics Lab at the Artificial Intelligence Laboratory, The University of Texas at

Austin. Research of the Intelligent Robotics Lab is supported in part by grants from the Texas Advanced Research Program (3658-0170-2007), from the National Science Foundation (IIS-0413257, IIS-0713150, and IIS-0750011), and from the National Institutes of Health (EY016089).

References

Angluin, D. (1978). On the complexity of minimum inference of regular sets. *Information and Control*, **39**: 337–350.

Beeson, P. (2008). Creating and utilizing symbolic representations of spatial knowledge using mobile robots. *Ph.D. Thesis*, The University of Texas at Austin.

Beeson, P., Jong, N. K. and Kuipers, B. (2005). Towards autonomous topological place detection using the extended Voronoi graph. *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, Barcelona, Spain, April, pp. 4373–4379.

Beeson, P. et al. (2007). Integrating multiple representations of spatial knowledge for mapping, navigation, and communication. *Proceedings of the Symposium on Interaction Challenges for Intelligent Assistants*, AAAI Spring Symposium Series, Stanford, CA, AAAI Technical Report SS-07-04, March, pp. 1–9.

Blanco, J.-L., Fernández-Madrigal, J.-A. and González, J. (2008). Toward a unified bayesian approach to hybrid metric-topological SLAM. *IEEE Transactions on Robotics*, **24**(2): 259–270.

Borenstein, J. and Koren, Y. (1991). The Vector Field Histogram—fast obstacle avoidance for mobile robots. *IEEE Transactions on Robotics and Automation*, **7**(3): 278–288.

Bosse, M. et al. (2003). Atlas framework for scalable mapping. *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, Taipei, Taiwan, September, pp. 1899–1906.

Buschka, P. (2005). An investigation of hybrid maps for mobile robots. *Ph.D. Thesis*, Örebro University.

Choi, C.-H. et al. (2002). Topological map building based on thinning and its application to localization. *Proceedings of the IEEE/RSJ Conference on Intelligent Robots and Systems (IROS)*, Lausanne, Switzerland, October, pp. 552–557.

Choset, H. and Nagatani, K. (2001). Topological simultaneous localization and mapping (SLAM): toward exact localization without explicit localization. *IEEE Transactions on Robotics and Automation*, **17**(2): 125–137.

Chown, E., Kaplan, S. and Kortenkamp, D. (1995). Prototypes, location, and associative networks (PLAN): towards a unified theory of cognitive mapping. *Cognitive Science*, **19**(1): 1–51.

Cummins, M. and Newman, P. (2007). Probabilistic appearance based navigation and loop closing. *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, Rome, Italy, November, pp. 2042–2048.

Cummins, M. and Newman, P. (2008). FAB-MAP: Probabilistic localization and mapping in the space of appearance. *International Journal of Robotics Research*, **27**(6): 647–665.

Dean, T. et al. (1995). Inferring finite automata with stochastic output functions and an application to map learning. *Machine Learning*, **18**(1): 81–108.

Duckett, T. and Nehmzow, U. (1999). Exploration of unknown environments using a compass, topological map and neural network. *Proceedings of the IEEE International Symposium on Computational Intelligence in Robotics and Automation (CIRA)*, Monterey, CA, September, pp. 312–317.

Duckett, T. and Saffiotti, A. (2000). Building globally consistent gridmaps from topologies. *Proceedings of the International IFAC Symposium on Robot Control (SYROCO)*, Vienna, Austria, pp. 357–361.

Dudek, G., Freedman, P. and Hadjres, S. (1993). Using local information in a non-local way for mapping graph-like worlds. *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, Chambéry, France, September, pp. 1639–1647.

Dudek, G. et al. (1991). Robotic exploration as graph construction. *IEEE Transactions on Robotics and Automation*, **7**(6): 859–865.

Elfes, A. (1989). Occupancy grids: a probabilistic framework for robot perception and navigation. *Ph.D. Thesis*, Carnegie Mellon University.

Eliazar, A. and Parr, R. (2003). DP-SLAM: fast, robust simultaneous localization and mapping without predetermined landmarks. *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, Acapulco, Mexico, August, pp. 1135–1142.

Fortune, S. (1992). Voronoi diagrams and Delaunay triangulations. *Computing in Euclidean Geometry* vol. 1 of *Lecture Notes Series on Computing*, Du, D.-Z. and Hwang, F. (eds). World Scientific, River Edge, New Jersey, pp. 193–234.

Fox, D., Burgard, W. and Thrun, S. (1997). The dynamic window approach to collision avoidance. *IEEE Robotics and Automation Magazine*, **4**(1): 22–33.

Fox, D., Burgard, W. and Thrun, S. (1999). Markov localization for mobile robots in dynamic environments. *Journal of Artificial Intelligence Research*, **11**: 391–427.

Gladwin, T. (1970). *East is a Big Bird: Navigation and Logic on Puluwat Atoll*. Cambridge, MA, Harvard University Press.

Gold, E. M. (1978). Complexity of automaton identification from given sets. *Information and Control*, **37**: 302–320.

Hähnel, D. et al. (2003a). An efficient FastSLAM algorithm for generating maps of large-scale cyclic environments from raw laser range measurements. *Proceedings of the IEEE/RSJ Conference on Intelligent Robots and Systems (IROS)*, Las Vegas, NV, October, pp. 206–211.

Hähnel, D. et al. (2003b). Towards lazy data association in SLAM. *Proceedings of the International Symposium on Robotics Research (ISRR)*, Sienna, Italy, October, pp. 83–105.

Harnad, S. (1990). The symbol grounding problem. *Physica D*, **42** 335–346.

Ko, B.-Y., Song, J.-B. and Lee, S. (2004). Real-time building of a thinning-based topological map with metric features. *Proceedings of the IEEE/RSJ Conference on Intelligent Robots and Systems (IROS)*, Sendai, Japan, September, pp. 797–802.

Koenig, S. and Simmons, R. G. (1996). Unsupervised learning of probabilistic models for robot navigation. *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, Minneapolis, MN, April, pp. 2301–2308.

Konolige, K. (2000). A gradient method for realtime robot control. *Proceedings of the IEEE/RSJ Conference on Intelligent Robots and Systems (IROS)*, Takamatsu, Japan, November, pp. 639–646.

Konolige, K. (2004). Large-scale map-making. *Proceedings of the National Conference on Artificial Intelligence (AAAI)*, San Jose, CA, July, pp. 457–463.

Kortenkamp, D. and Weymouth, T. (1994). Topological mapping for mobile robots using a combination of sonar and vision sensing. *Proceedings of the National Conference on Artificial Intelligence (AAAI)*, Seattle, WA, August, pp. 979–984.

Kuffner, J. J. and LaValle, S. M. (2000). RRT-Connect: an efficient approach to single-query path planning. *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, San Francisco, CA, April, pp. 995–1,001.

Kuipers, B. (2000). The Spatial Semantic Hierarchy. *Artificial Intelligence*, **119**: 191–233.

Kuipers, B. (2008). An intellectual history of the Spatial Semantic Hierarchy. *Robotics and Cognitive Approaches to Spatial Mapping*, vol. 38 of *Springer Tracts in Advanced Robotics*, Jefferies, M. E. and Yeap, W.-K. (eds). Berlin, Springer, pp. 243–264.

Kuipers, B. and Beeson, P. (2002). Bootstrap learning for place recognition. *Proceedings of the National Conference on Artificial Intelligence (AAAI)*, Edmonton, Canada, July, pp. 174–180.

Kuipers, B. et al. (2004). Local metrical and global topological maps in the Hybrid Spatial Semantic Hierarchy. *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, New Orleans, LA, April, pp. 4845–4851.

Kuipers, B. J. (1978). Modeling spatial knowledge. *Cognitive Science*, **2** 129–153.

Kuipers, B. J. and Byun, Y.-T. (1991). A robot exploration and mapping strategy based on a semantic hierarchy of spatial representations. *Journal of Robotics and Autonomous Systems*, **8** 47–63.

Lankenau, A., Röfer, T. and Krieg-Brückner, B. (2002). Self-localization in large-scale environments for the bremen autonomous wheelchair. *Spatial Cognition III*, vol. 2,685 of *Lecture Notes in Artificial Intelligence*. Berlin, Springer, pp. 34–61.

Lee, W.-Y. (1996). Spatial semantic hierarchy for a physical mobile robot. *Ph.D. Thesis*, The University of Texas at Austin.

Leonard, J. and Newman, P. (2003). Consistent, convergent, and constant-time SLAM. *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, Acapulco, Mexico, pages 1,143–1,150.

Lifschitz, V. (1995). Nested abnormality theories. *Artificial Intelligence*, **74**: 351–365.

Lu, F. and Milios, E. (1997). Globally consistent range scan alignment for environment mapping. *Autonomous Robots*, **4**: 333–349.

Lynch, K. (1960). *The Image of the City*. Cambridge, MA, MIT Press.

MacMahon, M., Stankiewicz, B. and Kuipers, B. J. (2006). Walk the talk: connecting language, knowledge, action in route instructions. *Proceedings of the National Conference on Artificial Intelligence (AAAI)*, Boston, MA, July, pp. 1,475–1,482.

Mataric, M. J. (1992). Integration of representation into goal-driven behavior-based robots. *IEEE Transactions on Robotics and Automation*, **8**(3): 304–312.

Modayil, J., Beeson, P. and Kuipers, B. (2004). Using the topological skeleton for scalable, global, metrical map-building. *Proceedings of the IEEE/RSJ Conference on Intelligent Robots and Systems (IROS)*, Sendai, Japan, September, pp. 1,530–1,536.

Montemerlo, M. et al. (2002). FastSLAM: a factored solution to the simultaneous localization and mapping problem. *Proceedings of the National Conference on Artificial Intelligence (AAAI)*, Edmonton, Canada, July, pp. 593–598.

Montemerlo, M. et al. (2003). FastSLAM 2.0: an improved particle filtering algorithm for simultaneous localization and mapping that provably converges. *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, Acapulco, Mexico, August, pp. 1,151–1,156.

Moravec, H. P. (1988). Sensor fusion in certainty grids for mobile robots. *AI Magazine*, **Summer**: 61–74.

Morris, A. C. et al. (2005). Towards topological exploration of abandoned mines. *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, Barcelona, Spain, April, pp. 2,117–2,123.

Murarka, A., Modayil, J. and Kuipers, B. (2006). Building local safety maps for a wheelchair robot using vision and lasers. *Proceedings of the Canadian Conference on Computer and Robot Vision (CRV)*, Quebec City, Canada, June, p. 25.

Olson, E., Leonard, J. and Teller, S. (2006). Fast iterative optimization of pose graphs with poor initial estimates. *Pro-*

ceedings of the IEEE International Conference on Robotics and Automation (ICRA), Orlando, Florida, May, pp. 2,262–2,269.

Paskin, M. A. (2003). Thin junction tree filters for simultaneous localization and mapping. *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, Acapulco, Mexico, August, pp. 1,157–1,166.

Press, W. H. et al. (1992). *Numerical Recipes in C: The Art of Scientific Computing*, second edition. London, Cambridge University Press.

Ranganathan, A. and Dellaert, F. (2005). Data driven MCMC for appearance-based topological mapping. *Proceedings of Robotics: Science and Systems (RSS)*, Cambridge, MA, June, pp. 209–216.

Ranganathan, A., Menegatti, E. and Dellaert, F. (2006). Bayesian inference in the space of topological maps. *IEEE Transactions on Robotics*, **22**(1): 92–107.

Rekleitis, I. M., Dujmovic, V. and Dudek, G. (1999). Efficient topological exploration. *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, Detroit, MI, May, pp. 676–681.

Remolina, E. and Kuipers, B. (2004). Towards a general theory of topological maps. *Artificial Intelligence*, **152**: 47–104.

Rivest, R. L. and Schapire, R. E. (1989). Inference of finite automata using homing sequences. *Proceedings of the ACM Symposium on Theory of Computing (STOC)*, Seattle, WA, May, pp. 411–420.

Savelli, F. and Kuipers, B. (2004). Loop-closing and planarity in topological map-building. *Proceedings of the IEEE/RSJ Conference on Intelligent Robots and Systems (IROS)*, Sendai, Japan, September, pp. 1,511–1,517.

Schapire, R. E. (1991). The design and analysis of efficient learning algorithms. *Technical Report MIT/LCS/TR-493*, MIT Laboratory for Computer Science.

Schröter, D. (2006). Region & gateway mapping: acquiring structured and object-oriented representations of indoor environments. *Ph.D. Thesis*, Technical University of Munich.

Schröter, D. et al. (2004). Detection and classification of gateways for the acquisition of structured robot maps. *Proceedings of the Symposium of the German Association for Pattern Recognition (DAGM)*, Tübingen, Germany, August, pp. 553–561.

Shatkay, H. and Kaelbling, L. P. (1997). Learning topological maps with weak local odometric information. *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, Nagoya, Japan, August, pp. 920–929.

Siegel, A. W. and White, S. H. (1975). The development of spatial representations of large-scale environments. *Advances in Child Development and Behavior*, vol. 10, Reese, H. W. (ed). New York, Academic Press, pp. 9–55.

Silver, D. et al. (2004). Feature extraction for topological mine maps. *Proceedings of the IEEE/RSJ Conference on Intelligent Robots and Systems (IROS)*, Sendai, Japan, September, pp. 773–779.

Smith, R., Self, M. and Cheeseman, P. (1990). Estimating uncertain spatial relationships in robotics. *Autonomous Robot Vehicles*, Cox, I. J. and Wilfong, G. T. (eds). New York, Springer, pp. 167–193.

Thrun, S. and Bücken, A. (1996). Integrating grid-based and topological maps for mobile robot navigation. *Proceedings of the National Conference on Artificial Intelligence (AAAI)*, Portland, OR, August, pp. 944–950.

Thrun, S., Burgard, W. and Fox, D. (2000a). A real-time algorithm for mobile robot mapping with applications to multi-robot and 3d mapping. *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, San Francisico, CA, April, pp. 321–328.

Thrun, S., Burgard, W. and Fox, D. (2005). *Probabilistic Robotics*. Cambridge, MA, MIT Press.

Thrun, S., Fox, D. and Burgard, W. (2000b). Monte Carlo localization with mixture proposal distribution. *Proceedings of the National Conference on Artificial Intelligence (AAAI)*, Austin, TX, August, pp. 859–865.

Thrun, S. et al. (1998). Integrating topological and metric maps for mobile robot navigation: A statistical approach. *Proceedings of the National Conference on Artificial Intelligence (AAAI)*, Madison, July, WI, pp. 989–995.

Tomatis, N., Nourbakhsh, I. and Siegwart, R. (2002). Hybrid simultaneous localization and map building: Closing the loop with multi-hypotheses tracking. *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, Washington, DC, May, pp. 2,749–2,754.

Wallgrün, J. O. (2005). Autonomous construction of hierarchical voronoi-based route graph representations. *Spatial Cognition IV. Reasoning, Action, Interaction*, vol. 3,343 of *Lecture Notes in Artificial Intelligence*, Berlin, Springer, pp. 413–433.

Yannakakis, M. and Lee, D. (1991). Testing finite state machines. *Proceedings of the ACM Symposium on Theory of Computing (STOC)*, New Orleans, LA, May, pp. 476–485.

Yeap, W.-K. (1988). Towards a computational theory of cognitive maps. *Artificial Intelligence*, **34** 297–360.

Yeap, W.-K. and Jefferies, M. E. (1999). Computing a representation of the local environment. *Artificial Intelligence*, **107**(2): 265–301.

Zhang, T. Y. and Suen, C. Y. (1984). A fast parallel algorithm for thinning digital patterns. *Communcations of the ACM*, **27**(3): 236–239.

Zimmer, U. R. (2000). Embedding local metrical map patches in a globally consistent topological map. *Proceedings of the International Symposium on Underwater Technology (UT)*, Tokyo, Japan, May, pp. 301–305.