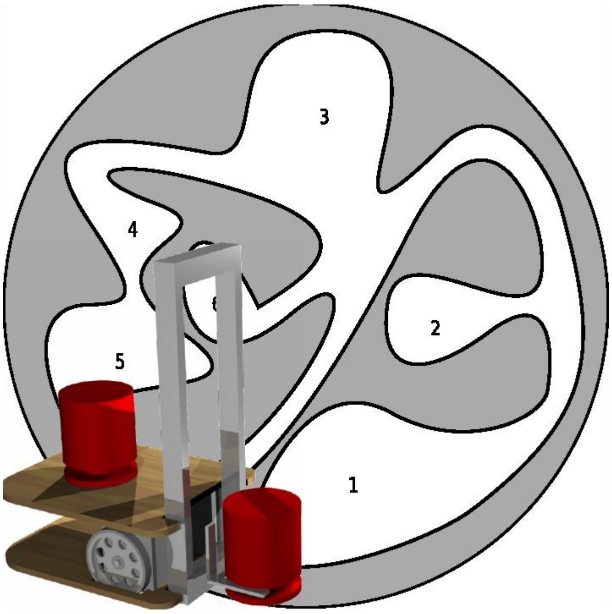
ROBOTICS RESEARCH LAB
DEPARTMENT OF COMPUTER SCIENCE
UNIVERSITY OF KAISERSLAUTERN

Seminar Report



Technical Overview of Robotic Mapping

Jonas Mitschang

July 14, 2007

Contents

1. Introduction

1.1 Motivation

Mobile robots navigate in their environment trying to reach their point of interest. The idea is to store knowledge for improving their performance and their flexibility. This knowledge is gathered by different sensors mounted on the robot. Goal of robotic mapping is to exploit the robot's sensorial capabilities to learn a model of its surrounding environment. Nowadays the problem of exploring and mapping an unknown environment is becoming increasingly important and is a highly active research area in robotics and artificial intelligence (AI). Building an accurate map of the real environment is essential for mobile robots that should interact truly autonomous. The problem could be solved really simple if the robot had idealized error-free perceptual capabilities. But as this is not the case (and maybe it will never be the case) one has to refine several mapping algorithms for getting the best out of the sensor measurements the robot acquires. The robot should be able to acquire a model of the environment in real-time by itself. Map building algorithms allow the acquisition of spatial models of physical environments using mobile robots.

Despite all the achievements in this area in the last two decades, it still presents great challenges. One problem is large-scale space maps that extend beyond sensory horizon of the robot. Large-scale space may include large nested loops and structural ambiguities between different places in environment. This paper gives a historical overview over robotic mapping and presents several mapping approaches with its advantages and disadvantages.

1.2 Historical Overview

Historically there are two methods for differentiating classes of maps. On the one hand mapping was divided into metric and topological approaches. This differentiation was state of the art when robotic mapping was in the fledging stages in the 1980s. Nowadays there is a smooth transition between metric and topological mapping. Both types have advantages and disadvantages that are discussed later.

Modern attempts try to implement hybrids between both types for combining their strengths and avoiding their weaknesses.

On the other hand robotic mapping algorithms were classified into world-centric and robot-centric. World-centric maps are stored as global spatial unit and the robot pose has global coordinates on this map. Objects in this map in generic do not carry information about sensor inputs. By contrast robot-centric maps store the information about sensor measurements relative to the robot pose. Robot-centric maps can be built easier because there are no transitions between different coordinate systems needed but they suffer from disadvantages: It is difficult to project the local sensor measurements to the measurements in another nearby spatial location. In robot-centric mapping it is more difficult to decide if the robot has already been at this place. And it is also hard to disambiguate between places that produce similar sensor measurements. These two problems are caused by lack of geometry in the sensor measurements. Caused by these disadvantages nowadays mapping approaches generate world-centric maps.

In the 1990s the robotic mapping developed towards probabilistic techniques. The mobile robot was supposed to be able to localize itself on the map and proceed mapping with the given data. This probabilistic approach is called simultaneous localization and mapping (SLAM, see 3.1.1) or sometimes concurrent mapping and localization (CML). Different algorithms are used to achieve the goal of simultaneous localization and mapping like Kalman filters (see 3.1.2 page 10) or expectation maximization algorithms (chapter 3.1.3).

2. Challenge in Robotic Mapping

Robotic mapping can be compared to the ”chicken and egg” problem. On the one hand there exist many very useful algorithms for determining the robot’s pose in a given (fully modeled) map just by analyzing the actual measurements. On the other hand it would also be really simple to construct a reliable map if one knows the actual robot’s pose in the environment.

The main problems in robotic mapping are the sensors that the robot uses for observing its environment. These sensors are not ideal but are victims of measurement noise (that cannot be avoided for analog measurements and data conversion) and range limitations (they can only sense the environment next to them). The sensor issue and the additional certainty of limited computational power on mobile robot systems lead to different problems:

Correspondence Problem

The correspondence problem or data association problem addresses the difficulty of determining which sensor measurements at different time belong to the same object and respectively the same location in the space-time continuum of the robot’s environment. The problem comes into fore when the mobile robot has to close a large cyclic loop. It has to detect which measurements belong to the same location in the built map for closing the loop. Caused by the inaccuracy of localization (odometry sensors) and the exponential growing number of hypotheses over time, the correspondence problem is the hardest problem in robotic mapping especially on environments with large cyclic loops (see figure 3.1).

Measurement Noise

As stated before, sensors are no ideal parts. The measurements they make suffer from different uncertainties (see figure 2.1). All electronic devices generate more or less noise which influences the measurements. The sensor inputs (environment) are time- and space-continuous but after signal processing and conversion to digital signals are both time- and value-discrete.

The noise problem can simply be reduced statistically by sampling the data multiple times. For mobile robots this implies slower movement and map exploration. But the measurement noise also affects control and results in pose errors and thus accumulates over time leading to misinterpretation of the subsequent sensor inputs.

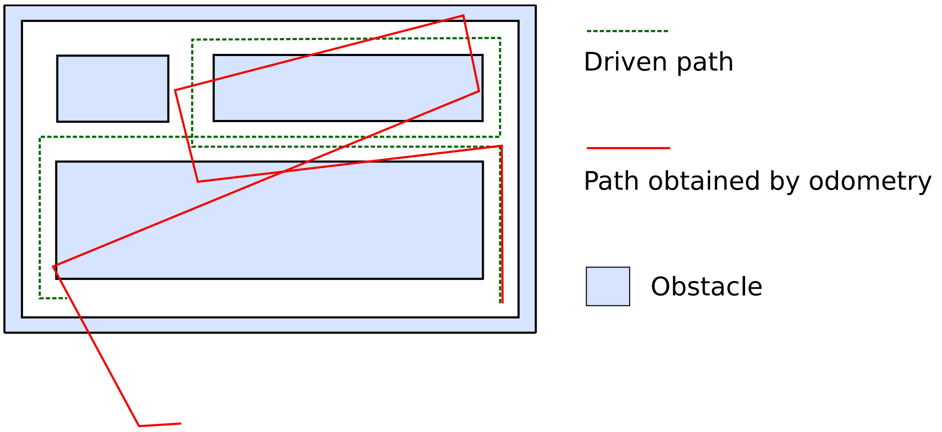


Figure 2.1: Example of odometry error

High Dimension

Each algorithm needs computational power for being completed. The problem of robotic mapping is the high dimension of the built map. It depends on the number of entities to store and the accuracy and type of the map to build. Topological maps (see 3.3) just need a minimum of data to represent the environment and thus does not consume much computational power and time. Dimension increases over metric two-dimensional grid maps to three-dimensional visual maps. Increasing dimension leads to serious runtime and storage problems. This prevents the robot from mapping larger environments and performing loop closing in real-time.

Dynamic of Environment

One critical fact in most environments is that they are dynamic which means they change over time. That means the mobile robot is not the only moving entity in the map. Whenever a map changes the robot has to make hypotheses on what has happened with the environment. It has to decide weather its surrounding has changed (moving people, doors opened and closed) or weather it is located at an other position on the map.

Many mapping algorithms assume that the robot is the only moving object in the map and all other dynamic variations of environment are just noise. This assumption is approximately true for short time windows because in short time the robot's surrounding is nearly static.

Exploration and Path Finding

Another challenge is the exploration of the robot's environment for building the full-fledged map. There are really good techniques for optimal robot motion on fully modeled maps but this is not the case for partial maps. The mapping algorithm has to construct the map in real-time in order to be able to explore the environment reasonable.

One always has to keep in mind that every movement of the mobile robot (every alternation of the robot's pose) produces a loss of information because the new pose is not hundred percent known and thus the successional measurement may be misinterpreted.

3. Types of Maps

Like stated in chapter 1.2 (page 3) maps are nowadays differentiated in metrical and in topological maps. Metrical and topological map representations have many differences [Kuipers 04]:

	Topological Maps	Metrical Maps
Scale	Large-scale space	Small-scale space
Sensor inputs	Abstracts sensor inputs	Stores sensor inputs
Computational power	Low	High
Memory consumption	Low	High
Sensitive to noise	Less	More
Real-time mapping	Yes	Depends on computational power

Topological maps describe large-scale space and abstract from sensor inputs. Local perceptual maps capture all sensor measurements within the sensor horizon and require plenty more resources than topological maps.

The choice of mapping algorithm is highly dependant on the field of application. Rescue robots [Lakaemper 05] for example build global overview maps where it is important to ensure that the whole target region has been searched. Under conditions of where rescue robots are used (e.g. collapsed buildings) there is just unreliable odometry information and landmarks are ambiguous.

3.1 Metrical Maps

The term metrical map refers to a map that consists of geometric objects (grids, shapes etc.) that represent a environment. Metrical maps provide some advantages over topological maps: They are finer grained (grid, polyhedra) than topological maps and thus solve more problems at high computational price.

Nowadays most metric maps are two-dimensional grids or shapes (specified by coordinates). Topological maps are less sensitive to noise but metrical maps are more common to humans perception.

3.1.1 Simultaneous Localization and Mapping

Simultaneous localization and mapping (SLAM) describes the problem of building a map within unknown environment while at the same time keeping track of the mobile robot's pose. Caused by several uncertainties described above in chapter 2 the task of SLAM is complex. SLAM uses mostly probabilistic techniques like Kalman filters or Monte Carlo methods and is only used for metrical maps. It can use very different types of sensors to acquire the required data for building the map.

3.1.2 Kalman Filters for Robotic Mapping

In the beginning of robotic mapping (1985) Kalman filters were used for simultaneous localization and mapping and they are still used nowadays [Thrun 02]. The Kalman filter was invented 1960 by Rudolf Kalman and is referred to as linear quadratic estimation (LQE). In general it is an efficient recursive filter that estimates the state of a dynamic system from a series of incomplete and noisy measurements. Kalman filters work incrementally and pertain to the SLAM approaches.

Kalman filters are Bayes filters and thus rely on basic assumptions:

- The sensor measurements must be linear with added Gaussian noise. Consequential the measurement noise ($\varepsilon_{measure}$) must not be correlated. Nonlinearities may be accommodated by linearization using e.g. Taylor series expansion.
- The initial uncertainty must also be independent Gaussian.

These assumptions are limitations to the Kalman filter approaches: Considering, for example, an environment with two indistinguishable landmarks. Measuring such a landmark will induce dependant noise which is not Gaussian. Thus Kalman filters are not able to cope the correspondence problem.

As a solution for the correspondence problem specially for laser range scanners an extension called Lu/Milios algorithm [Scott 00] has to be presented. It combines two basic estimation phases:

- In the first phase Kalman filters are used to calculate posteriors over the map.
- The second phase associates data of multiple scans with each other. The correspondence is achieved using maximum likelihood data association.

This improved version works well as long as errors in the initial pose estimate small (e.g., smaller than 2 meters). Larger pose errors can not be accommodated.

3.1.3 Expectation Maximization Algorithms

Expectation maximization algorithms (EM) are an alternative for Kalman filters also belonging to SLAM approaches. Nowadays they are the best solution to the correspondence problem described in chapter 2. A huge advantage over Kalman filters is achieved by repeatedly relocalizing the robot relative to the present map. EM algorithms perform hill climbing on all maps in order to find the best fitting map. That is why they cannot build the maps incrementally like the Kalman filters.

As stated before it is really simple to construct a reliable map if one knows the actual robot's pose in the environment and it is also no big deal to localize the robot's pose in a given map. The expectation maximization algorithm solves the SLAM problem by iterating between the following two steps to make the resulting map (m_i) more and more accurate with each step i:

- *Expectation step*: Generate all possible robot poses in the actual map m_i (the first map m_0 is empty).
- *Maximization step*: Calculates the most likely map for these given pose expectations.

The result of the expectation maximization algorithm are increasingly accurate maps m_0, m_1, ..., m_n.

3.1.4 Incremental Maximum Likelihood Method and Hybrid Approaches

Both Kalman filters and expectation maximization algorithms have several disadvantages. The errors in control accumulate over time which affect future sensor interpretations and disturbs the built map. Kalman filters are not able to solve the correspondence problem while expectation maximization algorithms are not working incrementally. A common approach is the incremental maximum likelihood method.

The basic idea is to incrementally build a single map as the sensor data arrives. Mathematically a series of maximum likelihood maps, m_1^*, m_2^*, ..., and a series of making likelihood poses s_1^*, s_2^*, ..., is maintained. Map m_t^* and pose s_t^* are built from the map m_{t-1}^* and pose s_{t-1}^* incrementally via maximization of the marginal likelihood. The incremental maximum likelihood method can be viewed as a expectation maximum algorithm without expectation-step, just consisting of a maximization-step. Although this approach is very simple it can build maps in real-time like Kalman filters and it maximizes likelihood like expectation maximization algorithms and thus is really popular. But it suffers from one big disadvantage: Like all algorithms that do not consider uncertainty when building maps it is not able to map cyclic environments. In cyclic environments the error in the pose s_t^* may grow without bounds (see figure 3.1).

To overcome this limitation hybrid approaches try to combine their strengths and avoid their weaknesses by maintaining uncertainty during mapping (see figure 3.2). Unlike the incremental maximum likelihood method, hybrid algorithms are able to correct maps backwards in time whenever an inconsistency is detected. But also hybrid approaches have disadvantages:

- The decision to change the map backwards in time is discrete which means that it may destroy the map if the decision is wrong.
- It can not handle complex ambiguities like multiple nested loops.
- It is not really real-time because the time needed for closing loops depends on the size of the loop. But if environment is limited to a specific size (e.g. office buildings) hybrid approaches work well.

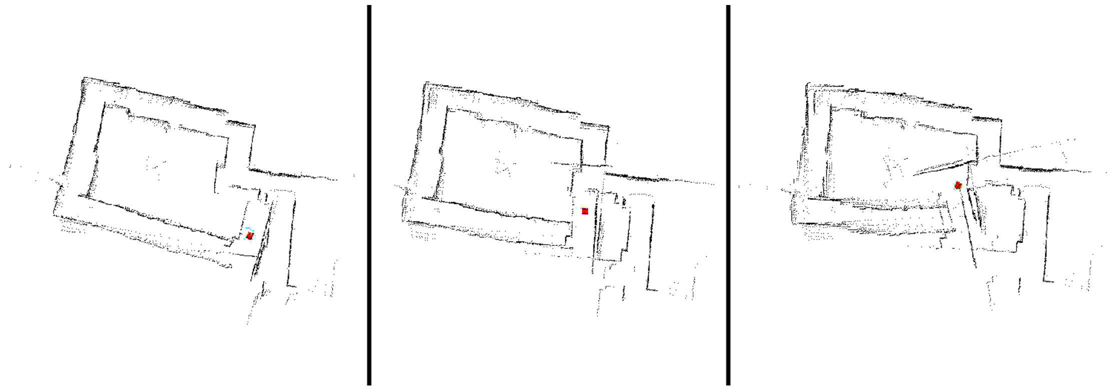


Figure 3.1: Incremental maximum likelihood method mapping is not able to handle cycles.

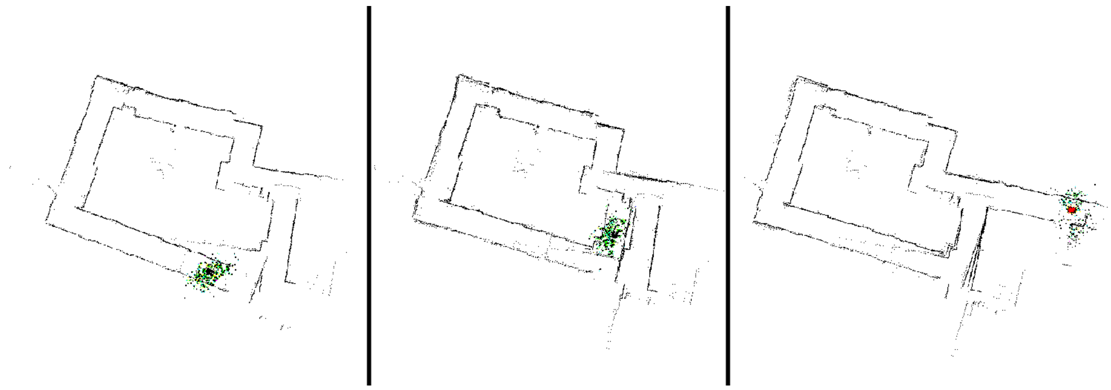


Figure 3.2: Hybrid approaches use probabilities and are able to close loops which are not too complex.

3.1.5 Discrete Segment Evolution

The goal of Discrete Segment Evolution is to build a global map from range sensors (two-dimensional) without using the odometry information. Discrete segment evolution solves the problem of statistically dependant measurement data without any assumptions about the error characteristics. In [Lakaemper 05] the application are rescue robots (no odometry, ambiguous landmarks) that should localize victims in catastrophe scenarios.

Following steps are performed for converting the measured data into a *scan* S_i (see figure 3.3):

- Scan points are approximated with line segments. These line segments don't need to be connected because this would imply additional noise in the approximation.

- *Segment grouping step*: An ordered list of segments if constructed by minimizing the sum of the distances of their endpoints.

- Finally the ordered list is split into multiple lists at positions where the end-point distance exceeds a defined maximum.

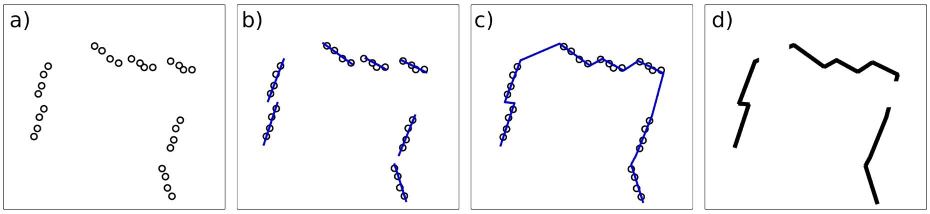


Figure 3.3: The figure shows the different steps of the discrete segment evolution algorithms. a) Scan points; b) Approximated line segments; c) Segment grouping step; d) Splitting into sublists

The first map G_0 is equal to the first scan: $G_0 = S_0$ and the subsequent maps are created by the following steps:

- *Correspondence*: The new scan S_i is positioned over previous map G_{i-1} at the position and angle of the old pose. Correspondence is established by mapping each segment endpoint of S_i to the closes point of G_{i-1}. This assumes small pose changes from scan to scan (given for high scan rates).

- *Alignment*: Minimize the distance between the points found in correspondence step and the actual scan S_i by translating and rotating S_i. This step calculates the new robot's pose.

- *Merging*: Now the actual scan S_i and the previous map G_{i-1} are aligned thus they can be merged into a new map G_i (noise is reduced on the map).

3.1.6 Occupancy Grid Maps

Unlike SLAM, occupancy grid maps work with known robot poses. They build a metric map from noisy or incomplete sensor data and rely on the estimated pose. Occupancy grid maps mostly use range sensors (sonar sensors or laser range finders) for detecting objects in the covered cone in space (see figure 3.4). In general the grids are two-dimensional (there also exist three-dimensional versions) grids of variables representing the probability that the corresponding field is occupied. Regions of high values represent obstacles, low values represent free space and special values (e.g. intermediate values) represent lack of knowledge. The advantage of occupancy grid maps is that they are very robust and easy to implement but they rely on pose certainty.

As occupancy grid maps use Bayes filters they assume independent Gaussian noise (see 3.1.2). With correlated noise the maps will be erroneous. This is usually the case when sonar measurements are integrated while the robot pose does not change. Usually this problem is avoided by discarding all sensor measurements while the robot is not moving.

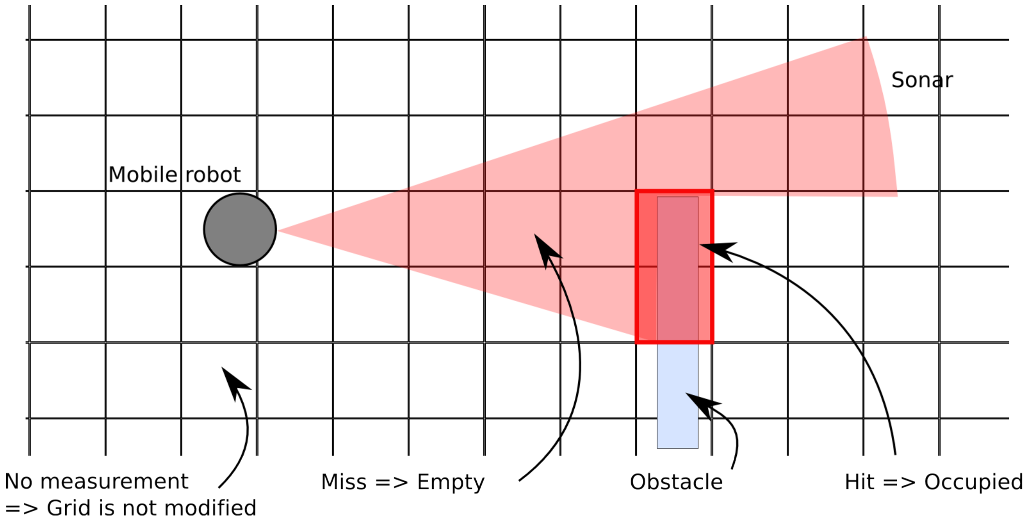


Figure 3.4: Example of occupancy grid mapping using sonar sensor

ApriAlpha™

An example for occupancy grid maps is the mapping algorithm that the home robot ApriAlpha™[Monteiro 05] uses. It implements a variation of the occupancy grid mapping which does not use Bayes filters but an alternative simple counting method. The mobile home robot uses a sonar sensor for observing its environment. Its occupancy grid consists of cells $C(i, j)$ that may be calculated by the following equation:

$$C(i, j) = \frac{hits(i, j)}{hits(i, j) + misses(i, j)}$$

$hits(i, j)$ is the number of cases in which the cell in (i, j) was occupied. $misses(i, j)$ on the contrary is the number of cases in which the cell was empty. This simplification for the occupancy grid results in less computational effort for the robot (no Bayes filters needed) but provides similar results. ApriAlpha™uses a grid of 80x80 cells with a resolution of 20cm x 20cm. It has five sonar sensors with a maximum range of approximately 2.1m. Figure 3.5 shows the environment exploration algorithm of the robot: It starts with simple exploration procedure (e.g. driving straight forward) and while driving it measures the sonar data for filling the grid map. When an obstacle is detected, it begins to follow its boundary. Caused by the slow movement speed of $10\frac{cm}{s}$, features like corners and edges can be detected and matched with the grid map in real-time.

3.1.7 Object Maps

The idea of object maps is to construct the map out of geometric shapes rather than grids. These geometric shapes in the simplest case may just be lines. Object mapping has several advantages over grid maps:

- The objects can change their position over time but remain the same object (e.g. same instance / ID) with other properties unchanged. So object maps are better for dynamic environment.

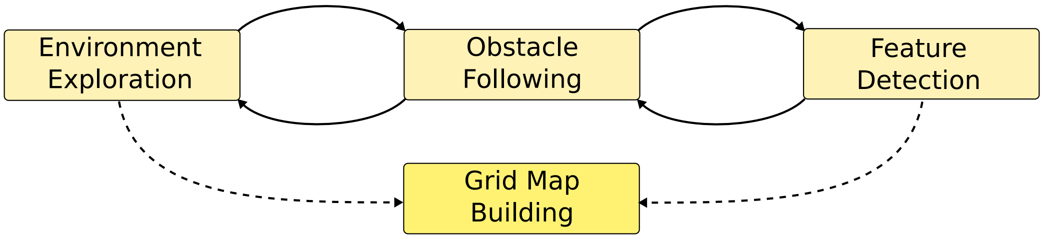


Figure 3.5: ApriAlpha$^{\text{TM}}$map building scheme

- They are more compact especially for structured environment.
- For all objects the coordinates in space are stored thus object maps are more accurate.
- Interaction of the mobile robot to humans can be improved because object maps are closer to the people's perception of the environment.

But object maps suffer from one big disadvantage: Often real environments are too complex to represent them with simple shapes. Fine grained grid maps would represent these environments (or objects) better than object maps. However there are two solutions how to solve this problem: On the one hand the mapping can be implemented as hybrid between object- and grid map. On the other hand the object map might ad-hoc learn new objects.

When building three-dimensional maps occupancy grids are often (depends on the sensors) not sufficient because occupancy grid maps require that each feature of the environment can be measured multiple times for being integrated using Bayes filters. But for some scanners (e.g. horizontal laser scanners) each feature is measured only once. Object maps remedy this problem by assuming that the environment consists of flat surfaces (surface-objects) and the number, position and size of these surfaces have to be discovered. This problem can be solved by expectation maximization algorithms (see 3.1.3).

3.2 Topological Maps

Topological maps are concise descriptions of large-scale structure of environment [Kuipers 04]. Environment is described as a collection of places that are linked by paths (see figure 3.6). These paths represent relations between the places and may include additional information e.g. how to navigate from one place to another or distance-information. An important advantage of topological maps is that it is possible to provide formal guarantees that the correct map is generated and never discarded [Dudek 93] [Remolina 04]. Topological maps are also built from geometric data but processing topological maps (generation, path finding) needs less computations than for metrical maps.

The TOUR Model

The TOUR model [Kuipers 77] is a topological mapping approach that divides spatial knowledge into five categories. It is originated of the year 1977 and is referenced

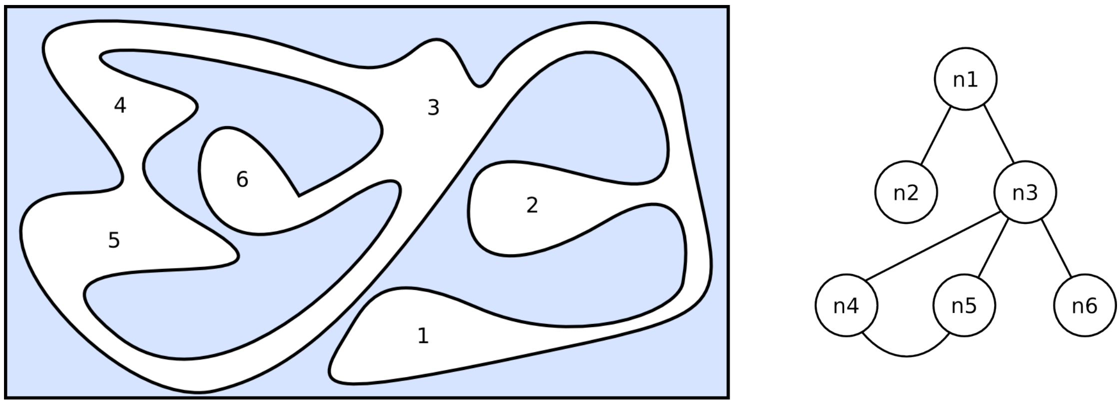


Figure 3.6: Example of a topological map (right) constructed out of a real environment (left).

in chapter 3.3.2. All representations provide a special range of states of partial knowledge and are able to solve a certain range of problems.

- Routes represent sequences of actions taking the traveler from one place to another. Three sources are needed for representing a route description: Observations of the Environment, recalled old routes, and intermediate states of the route-planning process.
- Street networks are represented topologically by descriptions of streets and places and of the local geometry of the intersection of two streets.
- Relative positions of two places are stored as two-dimensional vector.
- Dividing boundaries define regions on either side. The kind of knowledge dividing boundaries offer is particularly useful in route-finding.
- Regions related by their containment provide useful levels of abstraction for stating relations among their elements. Regions allow places to be grouped and referred to collectively.

3.2.1 Spatial Semantic Hierarchy

The spatial semantic hierarchy (SSH, basic SSH) is a computational model of knowledge of large-scale space [Kuipers 00]. Large-scale space is defined as space whose structure is a much larger than the sensory horizon of the robot. So the mobile robot has to travel through the space, gathering local observations and finding their spacial relationship. It consists of multiple different interactive but related representations for space. The multiple levels of the SSH expresses states of partial knowledge (see figure 3.7) [Kuipers 90]:

- The *sensory level* deals with continuous sensing of the continuous world.

- The *control level* describes the world in terms of continuous control laws. It is used to navigate among distinctive states of environment and thus supports higher level behaviors. These distinctive states eliminate cumulative position errors while moving (but not while exploring). Local geometrical maps improve the speed of finding a distinctive state because hill-climbing is no longer needed.
- The *causal level* abstracts the environment to discrete states. As the causal level is sequence of control laws (trajectory-following and then hill-climbing) reliably takes the agent from one distinctive state to another. The sequence of control laws is called *action* A. *View* V and V' are the sensor inputs before and after the action A. Their association is represented by $\langle V, A, V' \rangle$.
- The *topological level* is a topological model with places, paths and regions and their connectivity of the robot's environment. The minimal set of places, paths, and regions required to explain the set of observed views is created.
  - A *place* is zero-dimensional point that may lie on paths.
  - A *path* describes a one-dimensional subspace of environment. Travel actions may between places take place on paths.
  - A *region* represents a two-dimensional subset of environment.
- The *metrical level* represents a global geometric map of the environment in a single frame. This level is useful sometimes but it is rarely essential. It requires relatively high computational effort but can simply be omitted if not needed.

SSH can be improved by using local metrical and global topological hybrid approaches as described below (page 18).

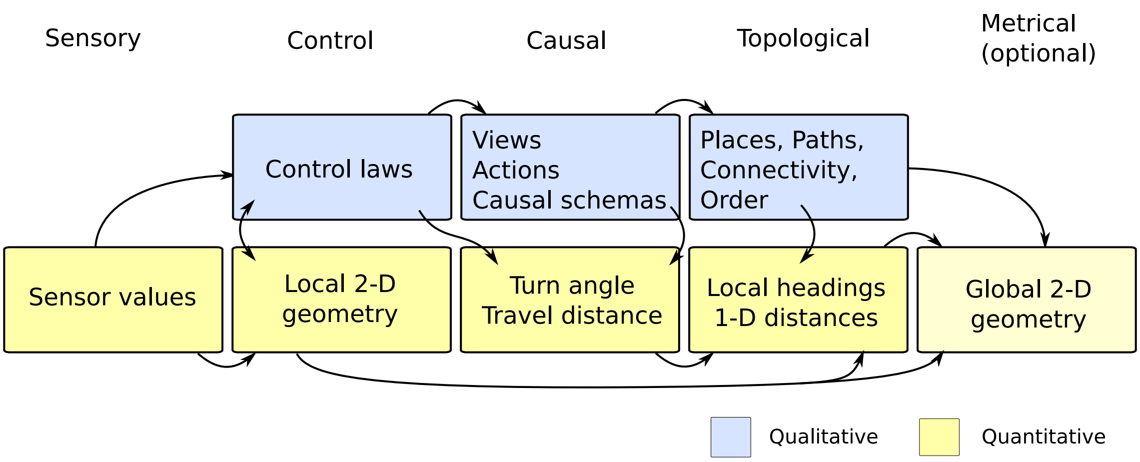


Figure 3.7: Levels in spatial semantic hierarchy

3.3 Metrical-Topological Hybrids

As stated on page 9 metrical and topological mapping algorithms have a very different character. Hybrid approaches try to combine both types for aggregating their advantages and avoiding some disadvantages. The most common hybrid mapping type is the hybrid spatial semantic hierarchy.

3.3.1 Hybrid Spatial Semantic Hierarchy

The hybrid spatial semantic hierarchy extends the basic SSH by a *local perceptual map* (LPM) which is a metrical map resulting from applying an online SLAM method to a simple local region. This local metrical map can be used for local motion planing and obstacle avoidance and can utilize any reliable metrical mapping, preferably occupancy grids. Hybrid SSH suffers from several advantages that aid online map-building:

- Local motion and collision avoidance can take place within the LPM.
- Metrical localization can be done quickly when searching for a distinctive point. The SSH hill-climbing is not needed any longer. If the mobile robot is out of resources for processing the local perceptual map it can still fall back to basic hill-climbing.
- Ambiguities caused by closing large loops or false positive place matches can be represented by a set of alternative topological maps which may be discarded by future observations.

3.3.2 Cognitive Maps

Cognitive maps are related to the human perception if its environment [Kuipers 01]. They combine metrical and topological maps for avoiding their weaknesses. Cognitive maps describe the knowledge of large-scale environment that is acquired by integrating observations gathered over time, and are used to determine relative positions of places for finding routes. Thus the mobile robot integrates local observations in the large-scale environment rather than being perceived from a single vantage point.

As usual in robotic mapping the robot must be simultaneously building the map, planning its activities (path planning), solving problems, and dealing with interrupts (hazards / collision avoidance during travel). Thus the cognitive mapping algorithm must operate under limitations of resources: The map is initially created analog to the environment into a two dimensional map that is called "Map in the Head" [Kuipers 83b]. While traveling the robot will add more places to its store of knowledge, drawing a progressively more complete map of its environment. This two dimensional analog map is not sufficiently powerful to serve as cognitive map because is has too few states of partial knowledge that causes several problems:

- Errors of odometry propagate over time and become very serious so the map must offer probabilistic methods.
- If an error is detected (which is only a question of time) no corrections can be worked out.
- Large computational effort for correcting the analog map when erroneous orientation is detected.
- During correction process the map could be left in inconsistent state.

So it is necessary to store spatial relationships between certain places while tolerating ignorance of relationships between others. This leads to the topological mapping approach. The idea is to build multiple two dimensional maps of the environment in different places ("Map in the Head") and connect them via topological information ("Atlas in the Head"). These topological information can be described like stated in TOUR model [Kuipers 77] (see section 3.2). While exploring the environment the robot creates links among places to connect the several maps in the atlas to one unified structure.

Cognitive maps have several advantages:

- If further resources become available or if new observations are made the topological connections between the metrical maps can be strengthened.
- When the topological connection between metrical maps is strong enough (exactly known) the maps may be joined together.
- Topological connections that are learned incorrectly can be easily modified without needing to analyze and process metrical data.

4. Conclusion

Several mapping approaches have been presented in this paper and their relative strengths and weaknesses were pointed out. All approaches are different versions of Bayes filters and thus they are probabilistic. Of course they have to be probabilistic to deal with the real environment. Most algorithms assume a static world but the real world is really dynamic. They approximate the static world by evaluating small time differences which is far away from reality. We need technologies for understanding environment dynamics rather than just adapting changes.

Although the situation encouraged over the last two decades of research there is still much to do. Unstructured outdoor environment like vegetation or water (underwater) environments are problematic. As computational power of embedded devices is becoming bigger and bigger and the resources available for mapping algorithms also increases in future other mapping approaches might be thought of.

In future there will be new application domains for other mapping approaches like e.g. multi robot mapping, thus the emerging correspondence problem will be much more complex than nowadays. Robot control has to be improved so that robots just have the function to "do the right thing" [Thrun 02]. So in future mapping will continue to be a highly active research area in robotics and artificial intelligence.

Bibliography

[Dudek 93] Gregory Dudek, Paul Freedman, Souad Hadjres. Using Local Information in a Non-Local Way for Mapping Graph-Like Worlds. 1993.

[Kuipers 00] B. J. Kuipers. The Spatial Semantic Hierarchy. *Artificial Intelligence*, 119, 2000. http://www.cs.utexas.edu/users/qr/papers/Kuipers-aij-00.html.

[Kuipers 01] Benjamin Kuipers. The Skeleton in the Cognitive Map: A Computational Hypothesis. 2001.

[Kuipers 04] B. Kuipers, J. Modayil, P. Beeson, M. MacMahon, F. Savelli. Local metrical and global topological maps in the hybrid spatial semantic hierarchy, 2004.

[Kuipers 77] Benjamin Kuipers. Modeling Spatial Knowledge. In *IJCAI*, 1977.

[Kuipers 83b] B. J. Kuipers. The cognitive map: Could it have been any other way? 1983b.

[Kuipers 90] Benjamin Kuipers, Yung-Tai Byun. A Robot Exploration and Mapping Strategy Based on a Semantic Hierarchy of Spatial Representations. Technischer Bericht AI90-120, 1, 1990.

[Lakaemper 05] Rolf Lakaemper, Longin Jan Latecki, Xinyu Sun, Diedrich Wolter. Geometric Robot Mapping, 2005.

[Monteiro 05] Sidomar T. Monteiro, Hideichi Nakamoto, Hideki Ogawa, Nobuto Matsuhira. Robust mobile robot map building using sonar and vision. 2005.

[Remolina 04] E. Remolina, B. Kuipers. Towards a general theory of topological maps, 2004.

[Scott 00] A. Scott. Quantitative and qualitative comparison of three laser-range mapping algorithms using two types of laser scanner data, 2000.

[Thrun 02] S. Thrun. Robotic Mapping: A Survey. In G. Lakemeyer, B. Nebel (Hrsg.), *Exploring Artificial Intelligence in the New Millenium*. Morgan Kaufmann, 2002. to appear.