

6.1 Introduction to mapping

In the previous chapter we studied different methods for localizing a robot equipped with a sensor able to take measurements to a number of landmarks with known positions, that is, the map m of the environment was given before hand. Now we will review a number of techniques addressing the opposite problem: to build a map m of the environment given a set of known robot poses x and sensor measurements z , that is:


$$p(m|z, x)$$

Notice that:

- x is not a random variable here, but a know vector (3×1 in 2D).
- since the robot pose x is given, it is not necessary to consider the motion command u !
- z is a sample from a distribution with mean $h(x, m)$.

There exist different types of maps, being the most used ones:

Landmarks based maps

 No description has been provided for this image
Fig.1 - Landmarks based map of an office environment. In this case Landmarks are QR codes placed next to doors.

Occupancy grid maps

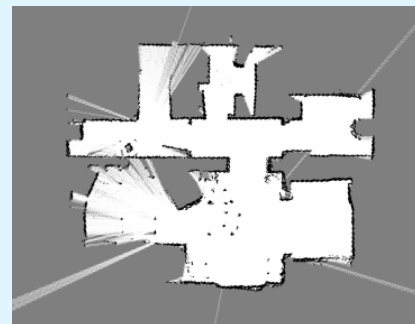


Fig.2 - Example of an occupancy gridmap of a house. In the image, white pixels represent traversable space, black pixels stand for obstacles, and gray ones are unknown space.

Topological maps

Semantic maps

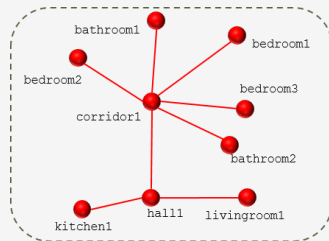


Fig.3 - Example of a topological map of a house where nodes represent rooms and edges link connected rooms.

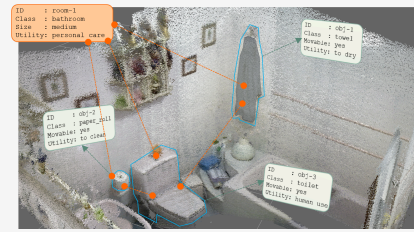


Fig.4 - Example of a semantic map of a bathroom where the objects within it are linked with meta-information about their category, size, utility, mobility, etc.

Notice that both Landmarks based maps and occupancy grid maps are **geometric maps**. In the next section of this book we are going to take a look at how to build landmarks based maps relying on the Extended Kalman Filter (EKF).

OPTIONAL

Surf the internet looking for more general information about robot mapping. You can include additional definition, examples, images, videos,... anything you find interesting!

Types of Maps

1. Geometric Maps

- **Grid Maps:** These divide the environment into fixed-size cells, each categorized as free, occupied, or unknown. They are often used in indoor scenarios, such as robotic vacuum cleaners navigating homes.
- **Landmark-Based Maps:** These rely on distinctive features like QR codes, doors, or natural markers to define the environment. Such maps are well-suited for dynamic or feature-rich environments.

2. Topological Maps

Represent environments as a graph with nodes (e.g., rooms) and edges (pathways). These maps abstract spatial layouts, making them computationally efficient for high-level planning and navigation.

3. Semantic Maps

Incorporate detailed information about objects, such as their type, size, and utility. For example, a semantic map of a bathroom might label sinks, toilets, and cabinets, enabling robots to perform context-sensitive tasks.

4. Hybrid Maps

Combine elements of different map types. For instance, **topometric maps** integrate metric precision with topological structure, enabling flexible and detailed navigation.

The Mapping Process

1. Data Collection

Robots use sensors such as LiDAR, cameras, or ultrasonic devices to gather spatial data.

2. Map Representation

The type of map chosen depends on the task. For obstacle avoidance, detailed geometric maps are ideal, while for abstract planning, topological maps suffice.

3. Map Updating

As robots explore new areas, maps must adapt to include updated or previously unknown information.

Advanced Techniques: SLAM

Simultaneous Localization and Mapping (SLAM) is a widely used approach where robots build a map and localize themselves simultaneously. SLAM uses probabilistic methods, such as Bayesian filtering, to handle uncertainty and ensure real-time map updates.

Applications of Robot Mapping

Robot mapping is integral to tasks such as:

- Navigation and path planning.
- Obstacle avoidance in dynamic settings.
- Contextual decision-making in environments requiring semantic understanding.

By combining innovative algorithms and adaptive techniques, robots can achieve robust mapping capabilities, paving the way for greater autonomy in real-world applications.

END OF OPTIONAL PART