Topic 7: Navigation and Path Planning

7.1 What is navigation?
- 7.1.1 Odometry based navigation
- 7.1.2 Beacon based navigation
- 7.1.3 Probability based navigation
- 7.1.4 Map based navigation
- 7.1.5 SLAM based navigation

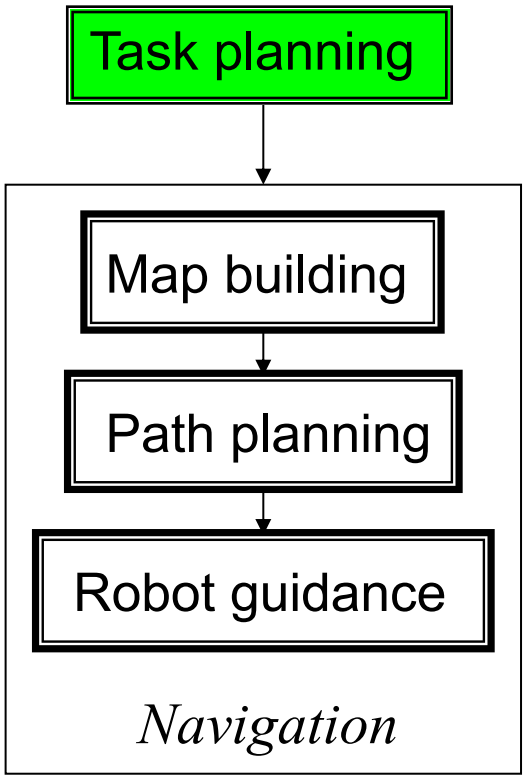
7.2 Path planning in known environments

7.3 Typical path planners for mobile robots
- 7.3.1 Teaching and Fixed Path Planner
- 7.3.2 Graph Search Path Planner
- 7.3.3 Composite-space Path Planner

7.4 Summary

7.1 What is navigation

Navigation is to direct the course of a mobile robot in order to reach a destination without crashing with anything.



Map building

- Prior maps of the terrain are stored in a map data structure.
- Using new sensor data to construct and modify maps.

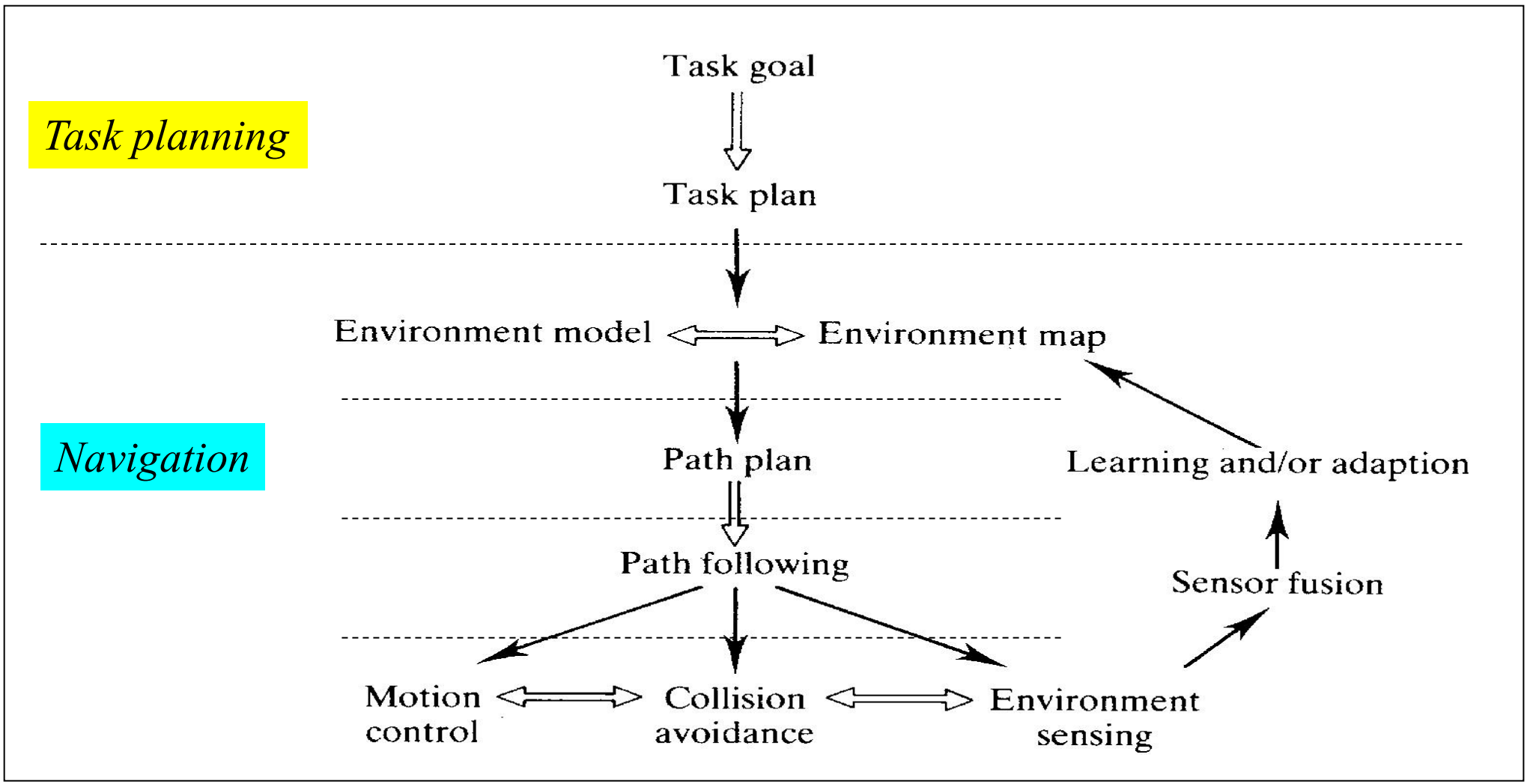
Path planning

- Searching the map for possible paths to the goal.
- Selecting suitable paths to match some criterions.

Robot guidance

- Controlling the robot to follow the planned path using its kinematics models.
- Monitoring potential collision and controlling the robot to take evasion actions.

7.1 What is Navigation?

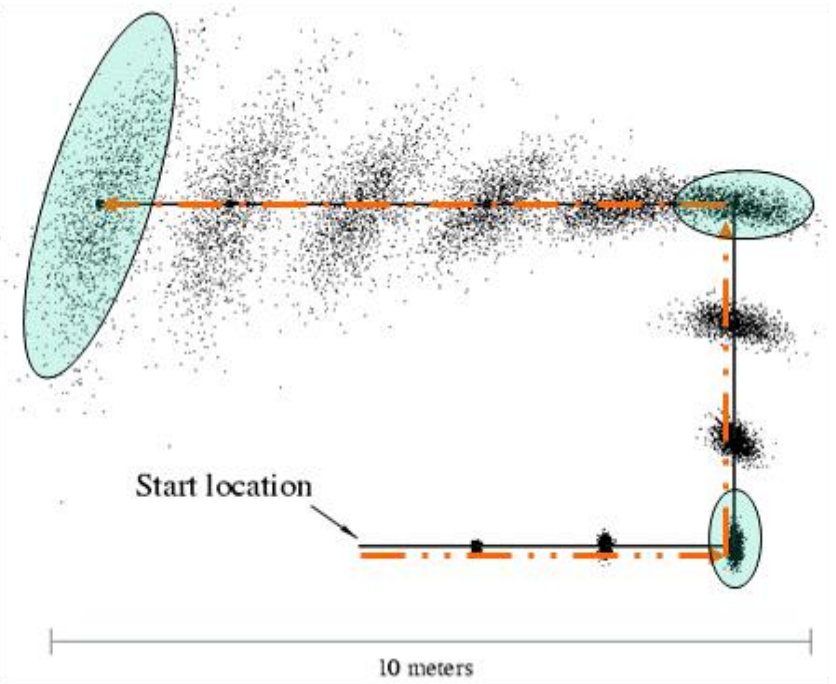


7.1 What is Navigation?

7.1.1 Odometry based navigation

Odometry is also called Dead Reckoning, and used for navigation (relative positioning) with accumulative errors.

RED lines show the odometry readings of the robot relative to a given map.



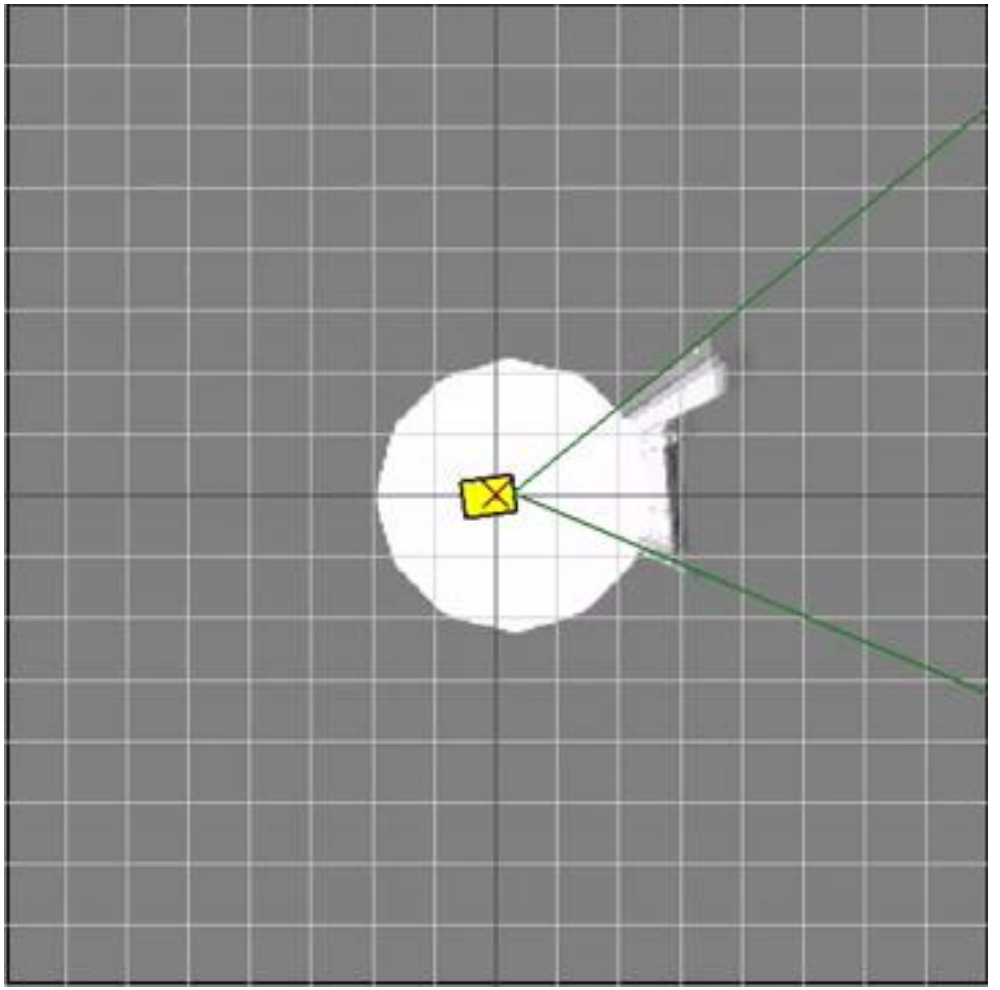
7.1.1 Odometry based navigation

The video shows the robot navigates in a place using its onboard sensors.

Yellow line is the reference trajectory.

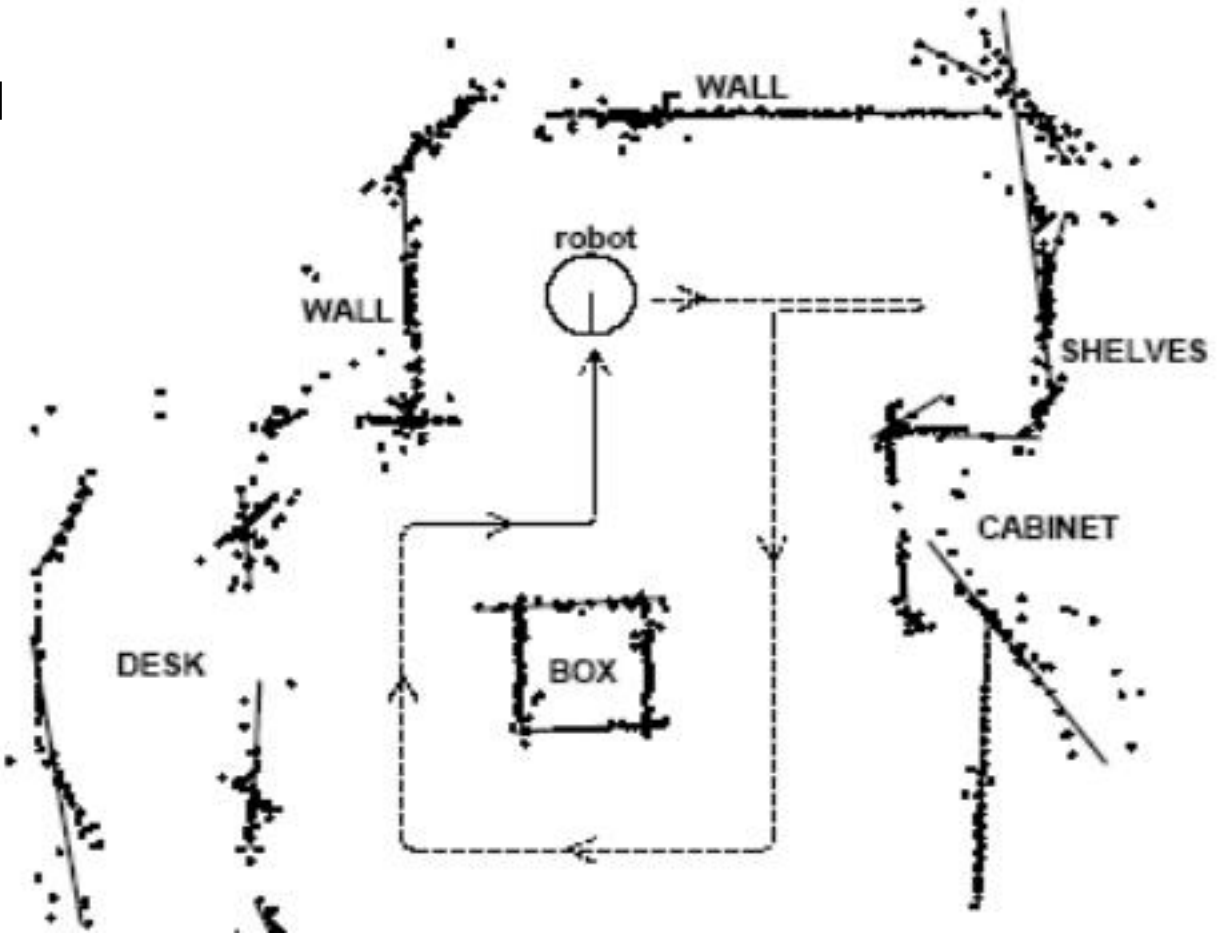
Blue line is the odometry readings.

As can be seen, the odometry data Is inaccurate and its error is increased without the boundary.



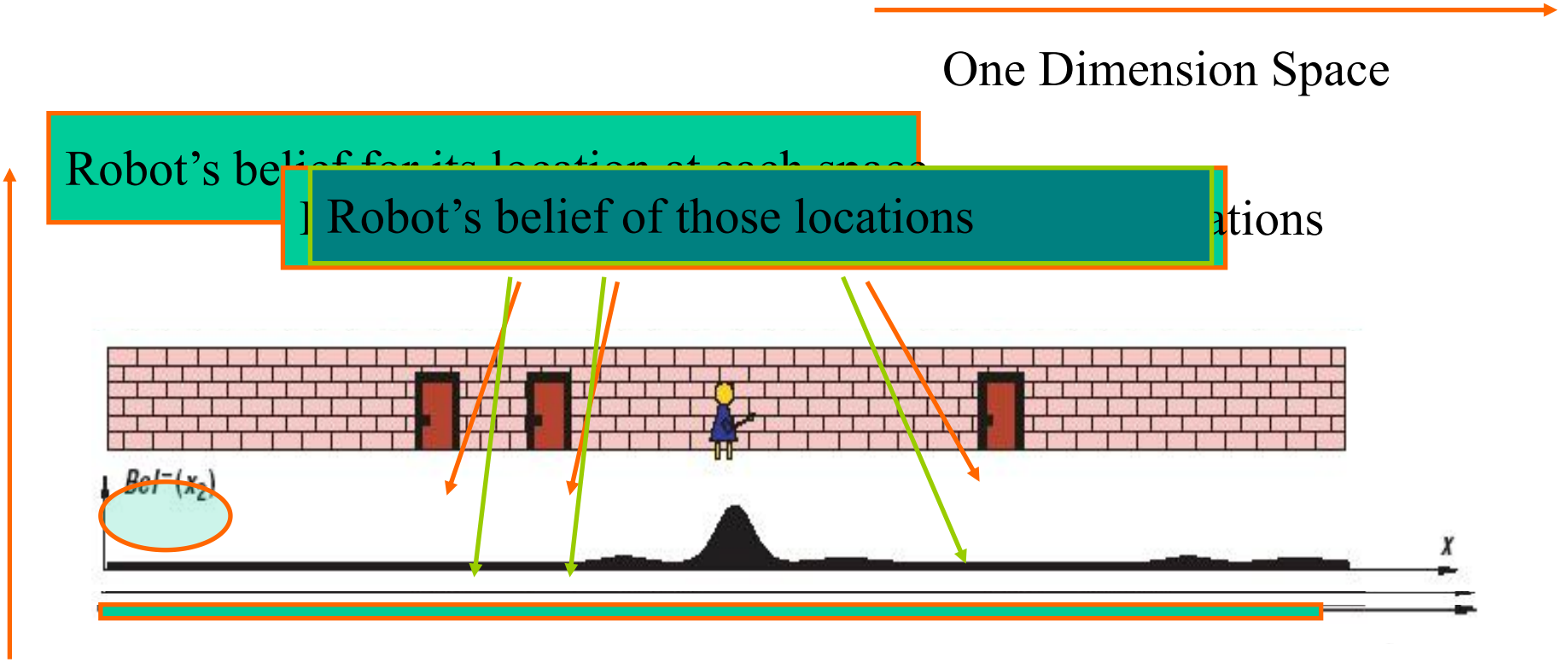
7.1.2 Beacon based Navigation

- Active beacons can emit energy, and are widely used in robot navigation, such as RF, Satellites, ...
- Artificial landmarks are specially designed passive objects to simplify navigation, such as barcode, name plate, …
- Natural landmarks are the features in the environment, which can be used for navigation, such as window, door, table, …



7.1.3 Probability based Navigation

It is based on Bayesian theory.



7.1.4 Map based Navigation

- The robots use their sensors to collect its local environment data and is then compared to a global map previously stored in memory.
- If a match is found, then the robot can compute its actual position and orientation in the environment.
- The pre-stored map can be a CAD model of the environment, or it can be constructed from prior sensor data.

Three steps in map building process:

-- Feature extraction from raw sensor data.

-- Fusion of data from various types of sensors.

-- Automatic generation of an environment map with different degrees of abstraction.

7.1.5 SLAM based navigation

SLAM means Simultaneous Localisation and Mapping.

7.1.5 Typic Navigation Pseudo code

Part 1

```
If there is a map then
   Search map for paths
   Select a path using an optimizing function
   If path is complex then decompose path into subgoals
   Sense environment
   While not at goal do
         Traverse path {move in direction indicated by path plar
         If at a subgoal then obtain next subgoal
         Sense environment
         If object on path then
            Halt robot
            If object is stationary then
               Update map
               Search for alternate path to sub-goal
               If an alternate path exists then
                  Follow alternate path
               Else {no alternate path}
                  Abort and replan task
```

Part 2

```
                  Abort and replan task
               Else {object is moving}
                  Halt till path clears
   End {of while loop}
   {robot at goal}
Else {no map – Initiate learning strategy}
   While not at goal do
         sense environment
         If clear in direction of goal then
            Move toward goal
         Else {object in the way}
            If clear in other directions then
               Select a direction using a heuristic
               Move in that direction
            Else {robot is trapped}
               Abort and replan task
   End {of while loop}
   {robot at goal}
End {of navigation task}
```

7.2 Path Planning in Known Environments

7.2.1 The requirements for a path planner

- To find a path through a mapped environment so that the robot can travel along it without colliding with anything.
- To handle uncertainty in the sensed world model and errors in path execution.
- To keep the robot away from those objects that may collide with the robot.
- To find the optimal path if there are multiple routes that exist.

Typical path planners for mobile robots:

- Teaching & fixed paths
- Graph search path planner
- Free-space planners
- Piano-mover planner
- Vertex graph planner
- Composite-space path planner

7.2.2 Map Data Structures & Planning

- Maps should be stored in a way that they can be retrieved and searched efficiently.
- Two main purposes: to record where a robot has been and
  to plan paths for a robot to follow.
- The data structure in map representation is depends on which path planning algorithm has been chosen, or vice versa.

Environment maps fall into four broad groups as follows:

– path maps
– free-space maps
– object-oriented maps
– composite-space maps

Our focus: path planning algorithms for a mapped environment.

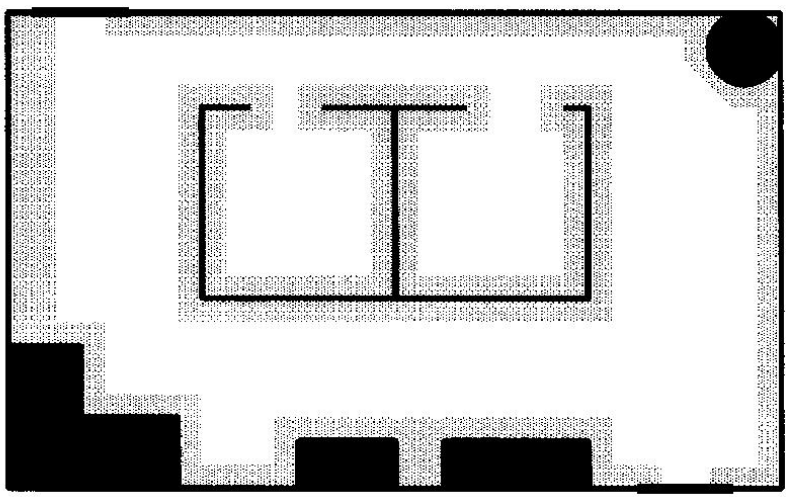
Data structure usage

Data storage	*Data manipulation*
Maps	Record & display
Features	Extract features
Object models	Identify objects
Paths	Search for paths
Sensor data	Update & learn maps

7.2.2 Map Data Structures & Planning

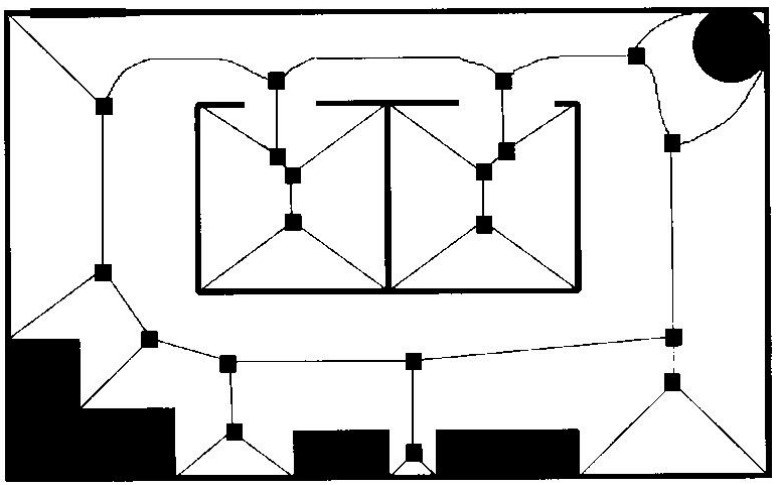
a) **Configuration space representation**

C-space representation enlarges the dimension of all obstacles and walls so that the robot can be shrunken into a point. The path planning can be easily carried out for a robot with a point size.



b) Voronoi graph representation

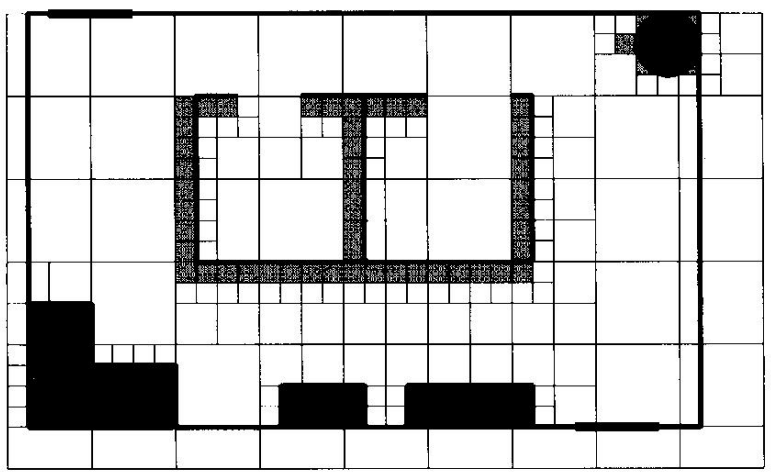
Voronoi graph is to generate a Voronoi edge, equidistant from all points. It makes it much easier for a robot to follow a path staying equidistant from all obstacles.



7.2.2 Map Data Structures & Planning

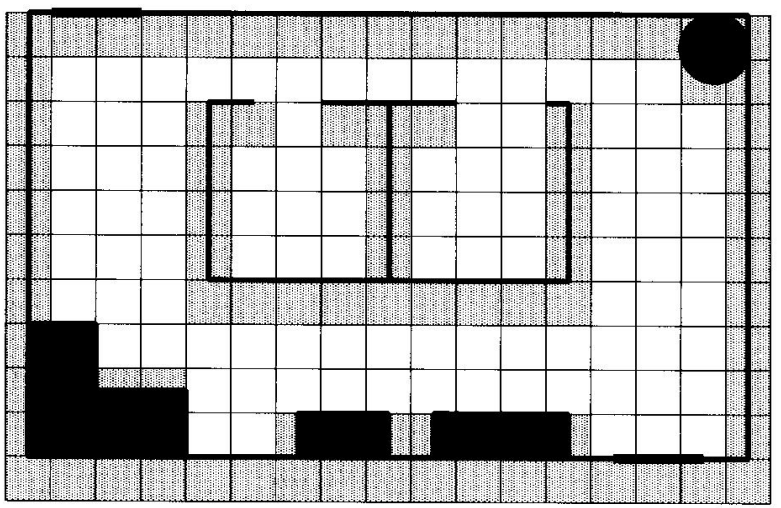
c) **Quad tree representation**

A quadtree representation has a variable size of grids to reduce the number of grids needed to represent a space. In other words, quadtree is a recursive grid representation.



d) **Regular grid representation**

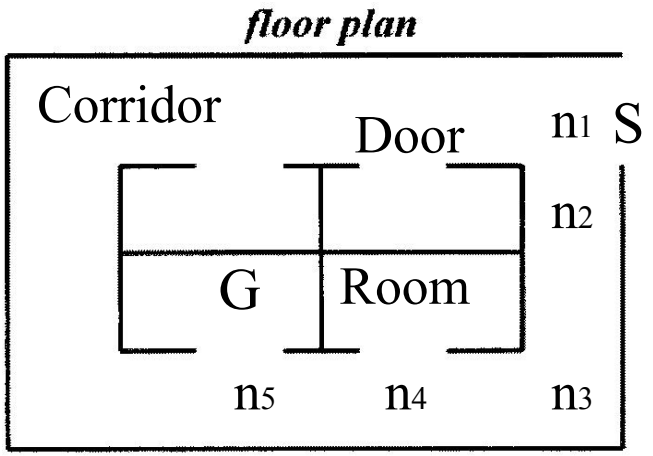
A regular grid map is a simple way to partition the world space, in which the grid is marked as empty if there is no object inside. Otherwise it is marked occupied.



7.3 Typical Path Planners for Mobile Robots

7.3.1 Teaching and fixed routes

- When guide wires or landmarks that are clearly defined in a map, the operator is able to teach a mobile robot to follow a specific path.
- The paths planned by an operator are stored in a control computer which controls the mobile robots from one place to another.
- Operational research techniques can be used to optimise the path.

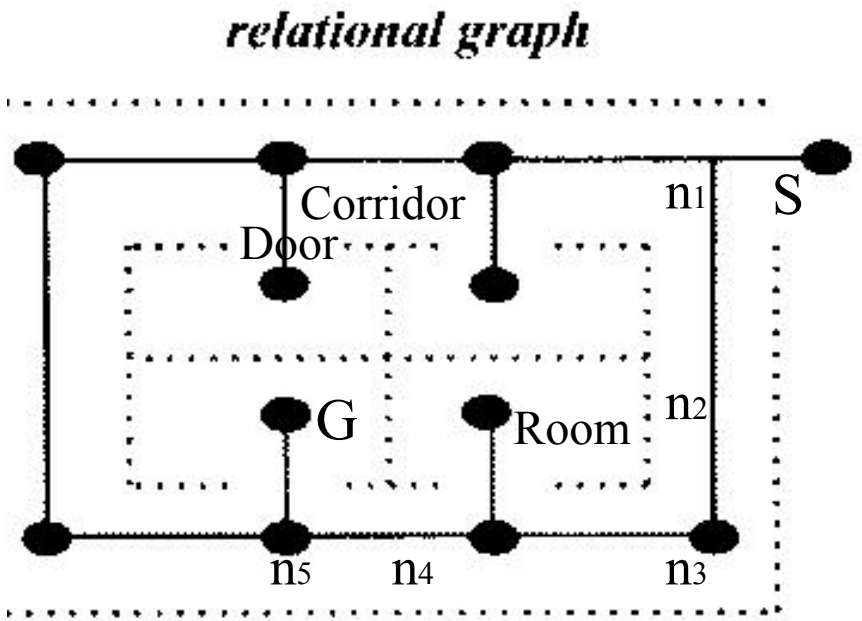


Task: Plan a path:
S → n1, …, n5 → G

7.3.1 Teaching and Fixed Routes

Build a relational graph:

- Nodes represent the gateways, sub-goals, landmarks, or any distinctive places.
- Edges represent the direction and distance between nodes.

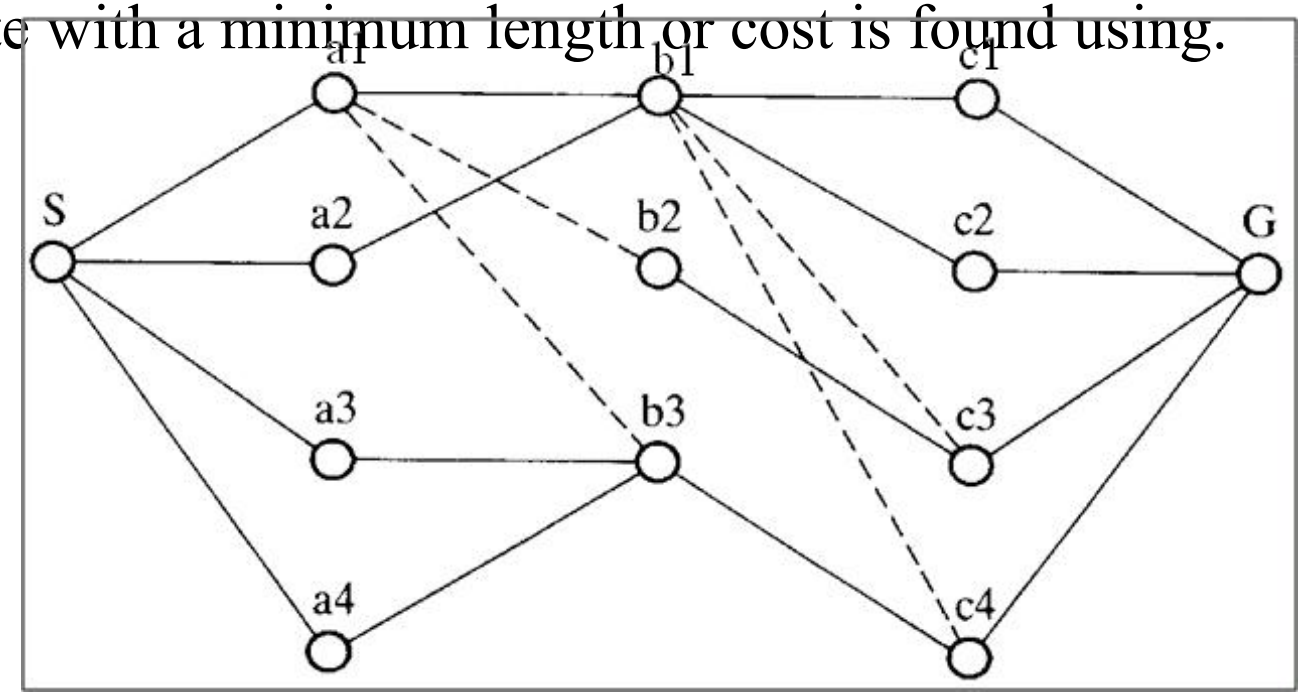


The path being planned:

1. Move from S to n_1, going WEST.
   In navigating corridor behaviour
   Wall following …
2. Move from n_1 to n_2, going SOUTH
   In navigating corridor behaviour
   Wall following …
3. Move from n_1 to n_2, going SOUTH
   In navigating corridor behaviour
   Wall following …
4. ….

7.3.2 Graph Search Path Planner

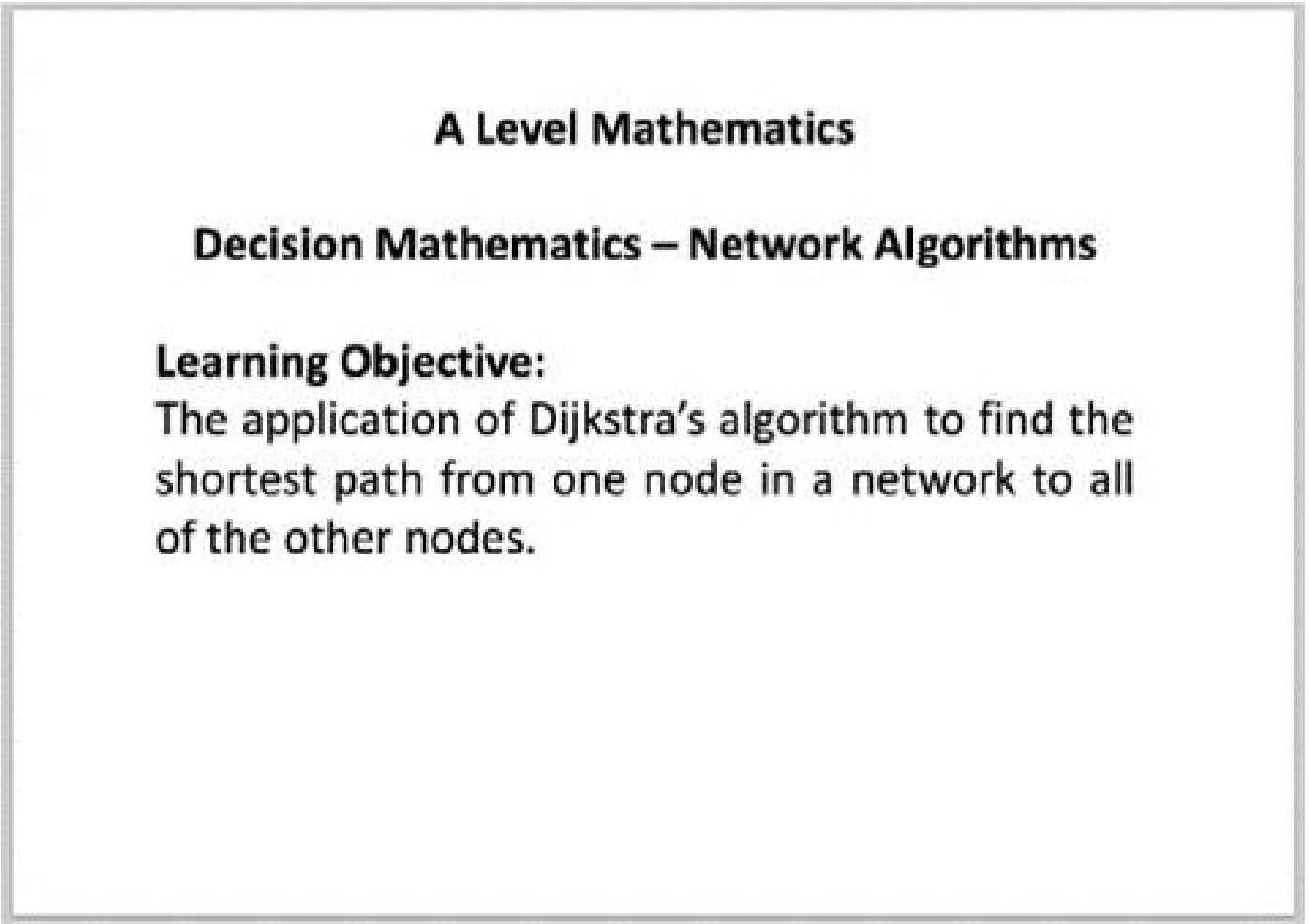
- The graph of nodes is treated as a tree with the root at the starting node.
- A path planner searches the tree to find a route to the goal using AI.
- If multiple route exist, dynamic programming technique is used to find the route with a minimum length or cost is found using.



7.3.2 Graph Search Path Planner

Dijkstra's Algorithm

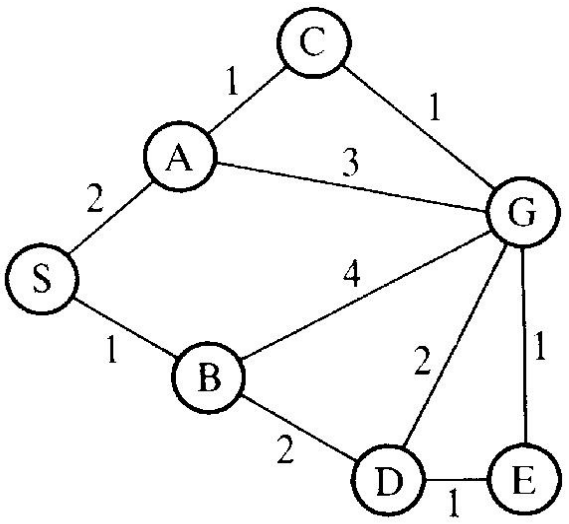
It is a popular method to find a shortest path in a graph.



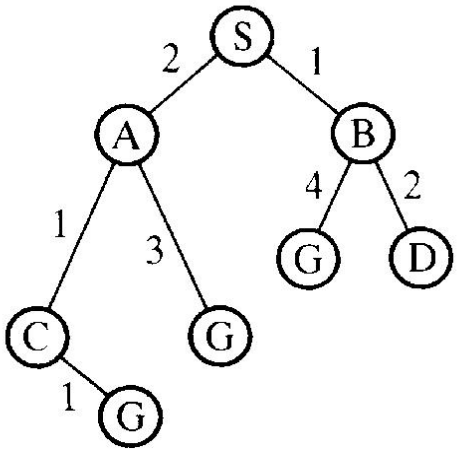
7.3.2 Graph Search Path Planner

A* search algorithm

- It is a refinement of the branch & bounds search.
- It is based on dynamic programming principle to delete multiple paths and leave the lowest cost path.
- It adds an estimated cost ***h*(i)*** for remaining path to the actual cost ***g* (i)*** of the path so far to produce an optimal solution, i.e. ***f* (i) = g* (i) + h* (i)*** (***i*** is a node in a graph)



(a) Graph of paths



(b) Branch and bounds

f(i) -- measures how good the move to node i is.*
g(i)-- measures the cost of getting to node I from the initial node.*
h(i)-- the cheapest cost of getting from node i to goal.*

7.4.2 Graph Search Path Planner

Step 1: The choices are node A and node B.

$$f^*(A) = g^*(A) + h^*(A) = 2 + 2 = 4$$
$$f^*(B) = g^*(B) + h^*(B) = 1 + 4 = 5$$

A path going from node A has the potential to be shorter than a path going from node B to the goal G.

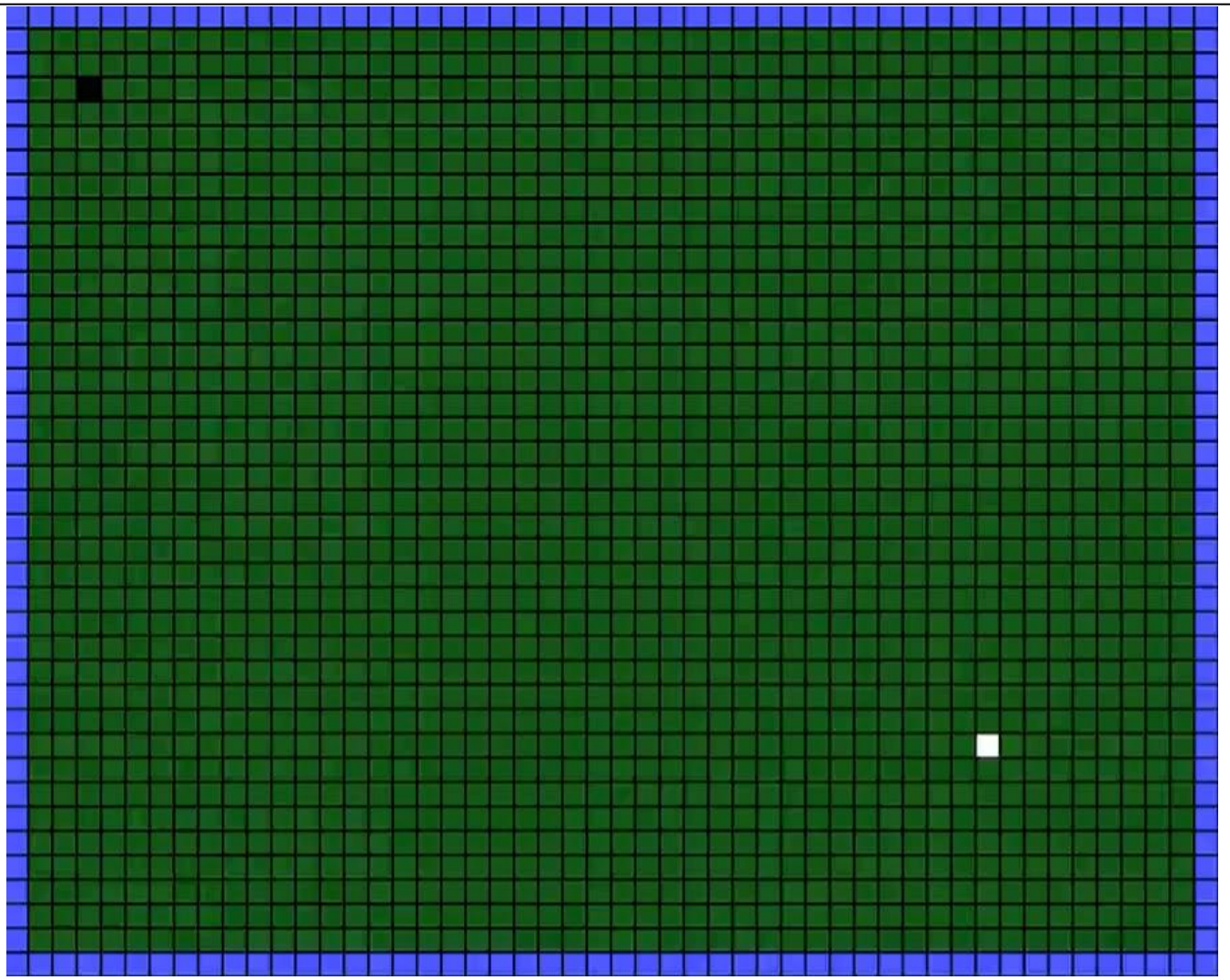
Step 2: The choices are node C and goal G.

$$f^*(C) = g^*(C) + h^*(C) = 3 + 1 = 4$$
$$f^*(G) = g^*(G) + h^*(G) = 5 + 0 = 5$$

Node C is chosen since a path going from it has the potential to be shorter than a path directly from node A to the goal.

Step 3: No choice at this stage. A path S - A - C - G is an optimal solution.

7.4.2 Graph Search Path Planner



7.4.3 Composite-space Path Planner

Simple Distance transform method

- Based on an area grid data structure, the path planner starts at the goal cell and propagates distances through free space.
- A distance transform value is calculated for each grid square, and stored in its data record.
- The simple distance transform can be calculated by adding 1 for each grid square crossed, as shown in the 8-connected distance transform in the next figure.
- If a grid square contains an object, even partially, it should be given a high value so that the paths will naturally flow away from it.
- An occupancy probability factor can be used to separate the grid squares without objects from the grid squares with objects, say 1 for a square known to be free space and 3 for a square known to be an object.
- Once the distance transforms are generated, the path is planned to follow the valley from start to goal.

7.4.3 Composite-space Path Planner

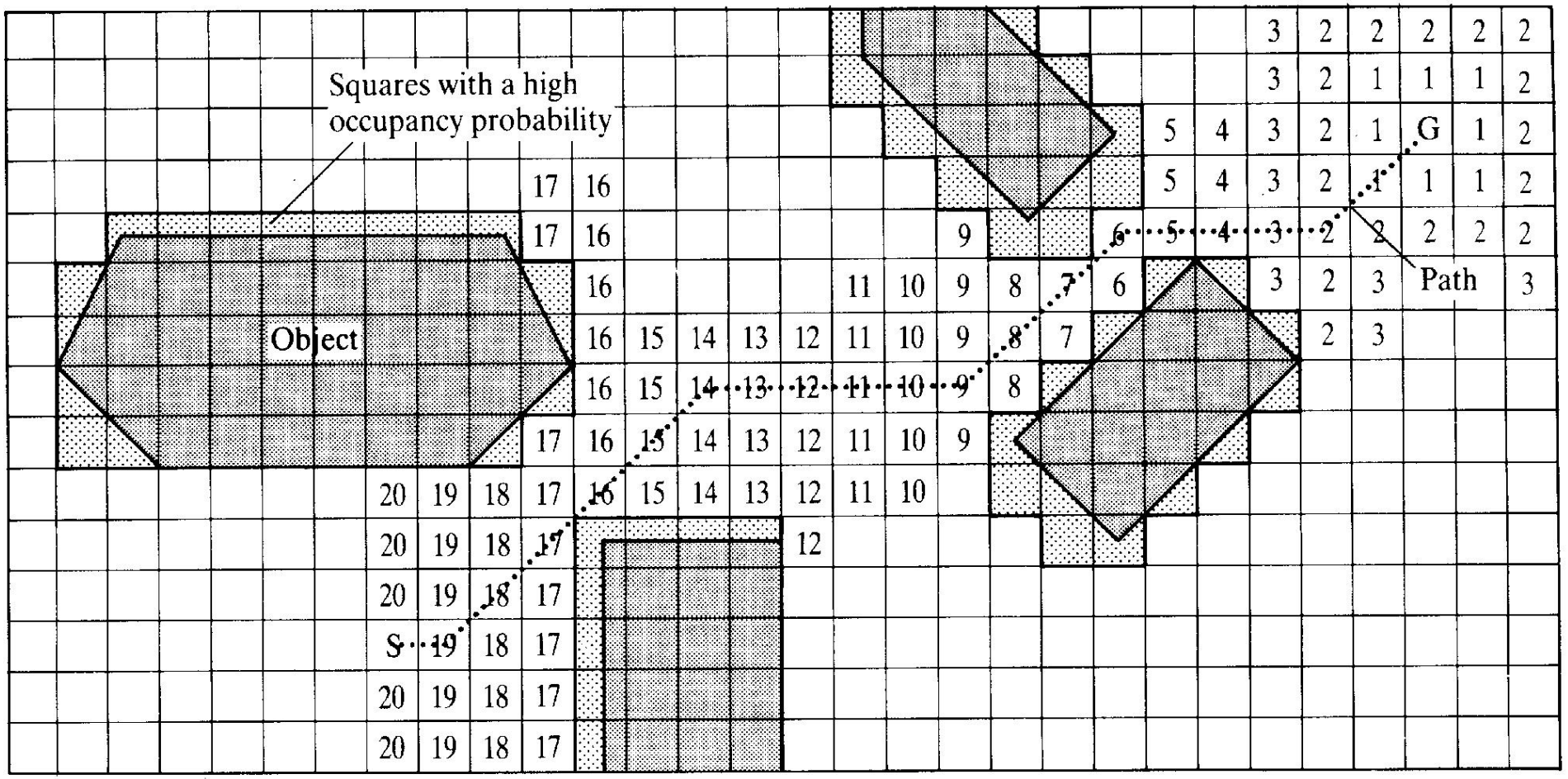
Algorithm 8.4 Distance transform path planning

```
Move to start cell
Scan all neighbours to see if a cell has a lower value
If no lower value cell is found then terminate {because there is no path}
Else {there is a path}
    Add start cell identifier to list
    Repeat
        Scan neighbouring cells to find cell with lowest value
          {4- or 8-connected}
        Move to this cell
        Add cell identifier to list
    Until at goal cell
End
Return path found flag and cell identifier list
```

However, the algorithm for propagating distance transforms in computationally expensive on a large grid space.

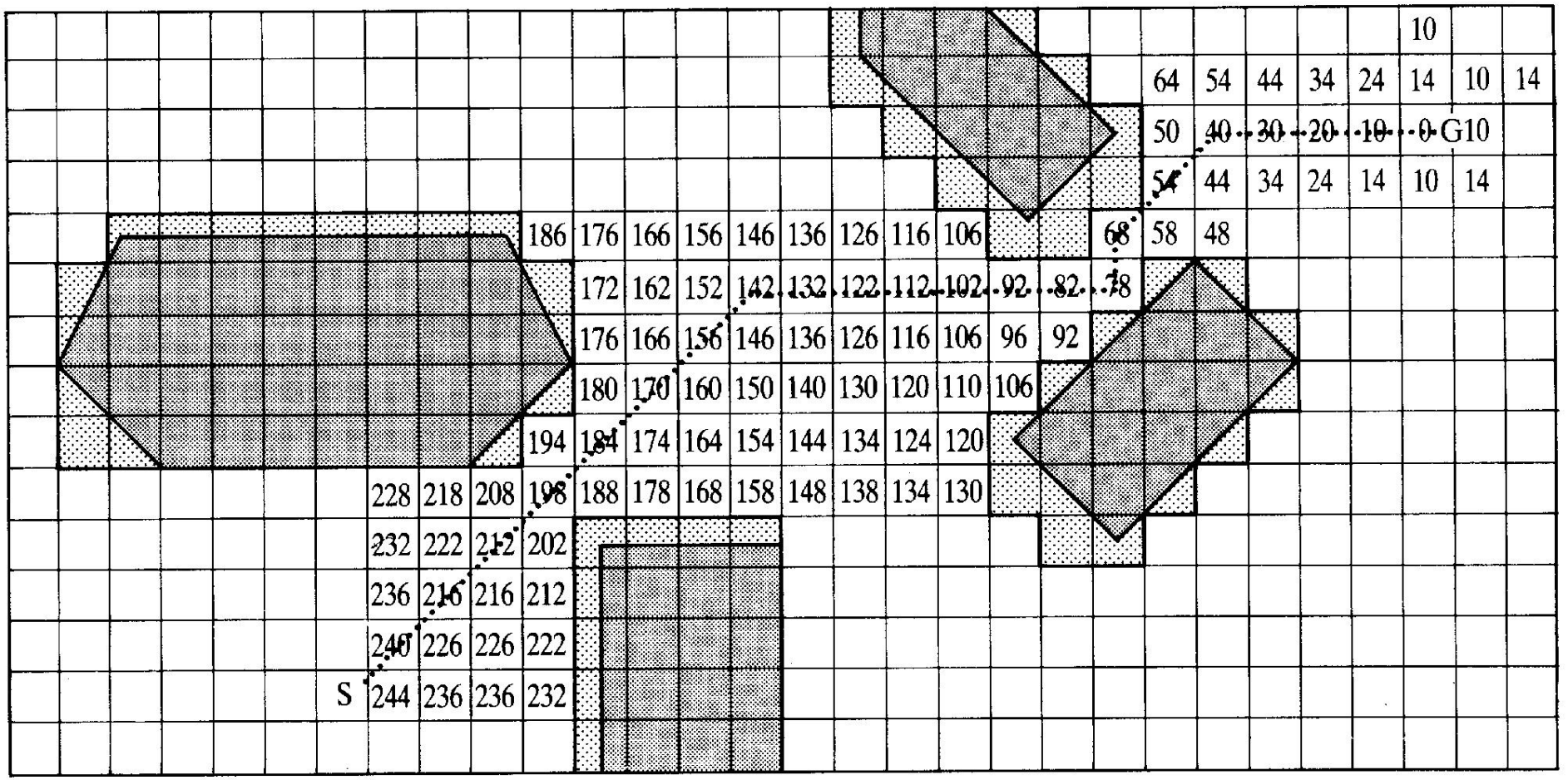
7.4.3 Composite-space Path Planner

8-connected distance transforms for the path planner

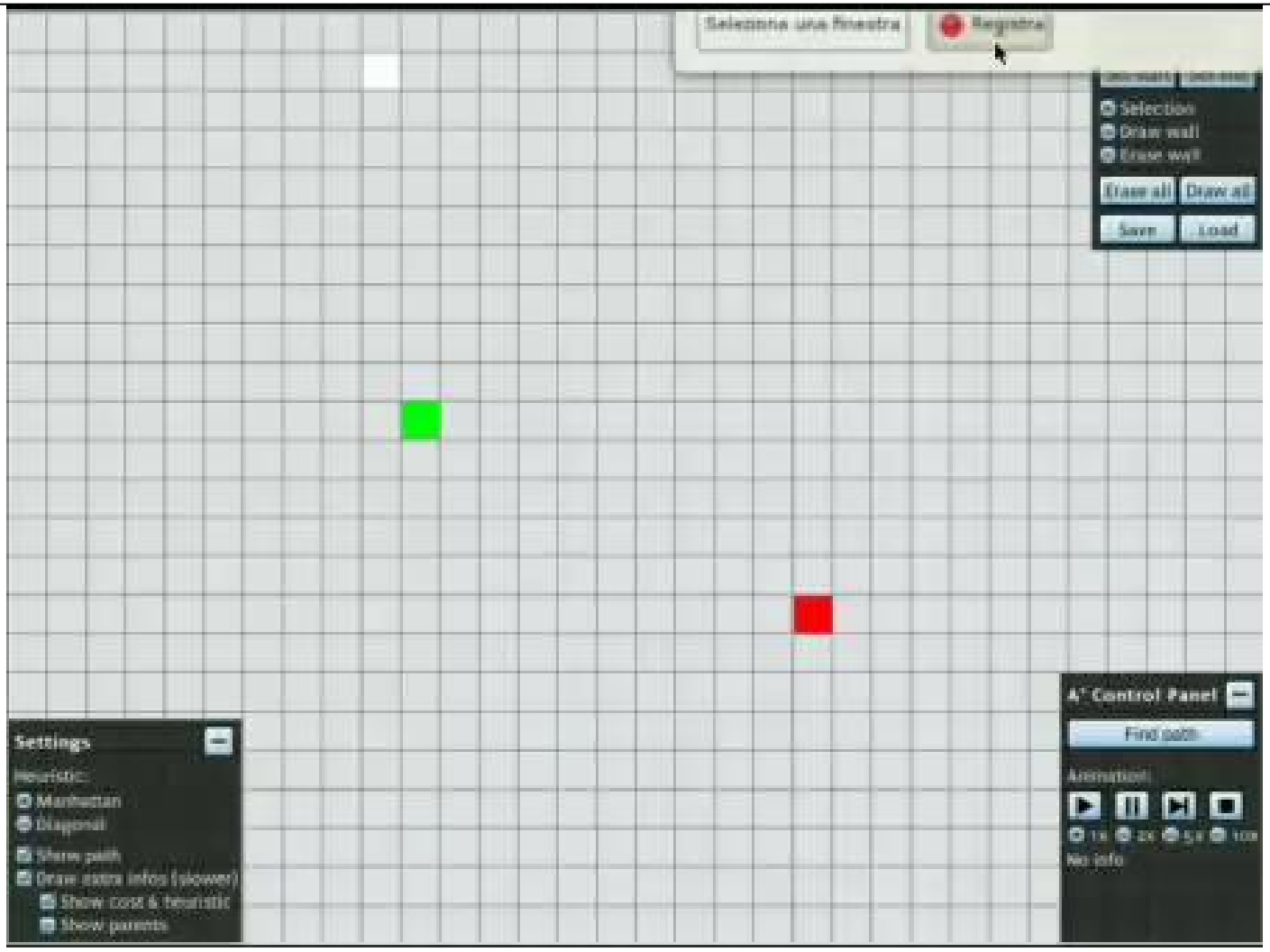


7.4.3 Composite-space Path Planner

8-connected transforms (***actual distances)*** for the path planner:



7.4.3 Composite-space Path Planner



7.5 Summary

- Navigation is to direct the course of a mobile robot in an environment.
- Navigation involves three tasks:

  environment mapping, path planning, guidance.
- The choice of a map data structure is usually a trade-off between
  - efficient representation of the environment, and
  - efficient representation of paths.
- Maps for known environments fall into four broad groups:

  path maps, free-space maps, object maps, composite/grid maps
- The path planning algorithms and the methods of sensor integration are determined by data structures.
- SLAM is a key technology for navigation in unknown environments.