Why do we care about robot motion planning?

Regardless of the form of the robots or the task it must perform, robots must maneuver through the world.

**Motion planning** is the problem of finding a robot motion from a start state to a goal state in a cluttered environment to achieve various goals while avoiding collisions.

**In its simplest form, the motion planning problem is:**

how to move a robot from a “start” location to a “goal” location avoiding obstacles.

©Solmaz Kia, UCI
The problem of motion planning can be stated as follows:

Given:

- A start pose of the robot
- A desired goal pose
- A geometric description of the robot
- A geometric description of the world

Find a path that moves the robot gradually from start to goal while never touching any obstacle.

This problem is sometimes referred to as the “move from A to B” or the “piano movers problem” (how do you move a complex object like a piano in an environment with lots of obstacles, like a house).

https://youtu.be/HdfAzUXvmOQ

©Solmaz Kia, UCI
a robot described by a moving point (that is, the robot has zero size).

- A workspace $W \subset \mathbb{R}^2$ or $\mathbb{R}^3$, often just a rectangle;
- Some obstacles $O_1, O_2, \ldots, O_n$;
- A start point $p_{start}$ and a goal point $p_{goal}$;

**free workspace:** $W_{\text{free}} = W \setminus (O_1 \cup O_2 \cup \cdots \cup O_n)$: the set of points in $W$ that are outside all obstacles.
Motion Planning

Representing the Obstacles as polygon

© Solmaz Kia, UCI
A roadmap is a collection of locations in the configuration space along with paths connecting them.

- With each path, we associate a positive weight that represents a cost for traveling along that path, for example, the path length or the travel time.
- Think of a roadmap as a weighted graph $G = (V, E, w)$, where $w$ is a function that assigns the weight (e.g., path length) to each edge in $E$. 
Visibility roadmaps: the visibility graph $G = (V, E, w)$, is defined as

i. the nodes $V$ of the visibility graph are all the vertices of the polygons $O_1, \ldots, O_n$,

ii. the edges $E$ of the visibility graph are all pairs of vertices that are visibly connected. That is, given $u, v \in V$, we add the edge $\{u, v\}$ to the edge set $E$ if the straight-line segment between $u$ and $v$ is not in collision with any obstacle, and

iii. the weight of an edge $\{u, v\}$ is given by the length of the segment connecting $u$ and $v$. 

environments with polygonal obstacles.
According to Wikipedia: pseudocode, “pseudocode is compact and informal high-level description of a computer programming algorithm that uses the structural conventions of a programming language, but is intended for human reading rather than machine reading.” see also https://www.geeksforgeeks.org/how-to-write-a-pseudo-code/
How to program my problem in Python

How to write a program for the robot to choose the shortest path?

```python
### KCS LAB
## finding the shortest path in the array
import numpy as np
paths = np.array([5, 6, 4.5, 7])
# initializing the first item to be the minimum
min = paths[0]
# for loop to run through every elements in the list
for i in range(len(paths)):
    # comparing minimum values with the elements in the list
    if min >= paths[i]:
        # update the minimum value
        min = paths[i]
print('The shortest path is:', min)
```

Output: The shortest path is: 4.5

Assignment 1, Problem no 1

Write a program to sort the list of distances stored in an array in descending order?

Example: Input: distances = [9,6,7,1,4,5,8]  
Output: distances = [9,7,8,6,5,4,1]

You can also find the code here:
https://colab.research.google.com/drive/1BL5oWaeBqJTPb2w_dahmMewdmTucJ-7?usp=sharing
Motion Planning via Visibility Graph

- Create the visibility graph: clearly show the vertices and edges and write the distance between visible vertices as weights on the edges.

- Use Dijkstra algorithm to find the shortest path from the show start point to the shown goal point.
Sponsors

© Solmaz Kia, UCI