What algorithm should I implement to program a room cleaning robot?

For this question assume that the following things are unknown:

- The size and shape of the room
- The location of the robot
- The presence of any obstacles

Also assume that the following things are constant:

- The size and shape of the room
- The number, shape and location of all (if any) obstacles

And assume that the robot has the following properties:

- It can only move forward in increments of absolute units and turn in degrees. Also the operation that moves will return true if it succeeded or false if it failed to move due to an obstruction
- A reasonably unlimited source of power (let's say it is a solar powered robot placed on a space station that faces the sun at all times with no ceiling)
- Every movement and rotation is carried out with absolute precision every time (don't worry about unreliable data)

Finally please consider the following properties of the robot's environment:

- Being on a ceiling-less space station the room is a safe but frustratingly close distance to passing comets, so the dust (and ice) are constantly littering the environment.

I was asked a much simpler version of this question (room is a rectangle and there are no obstacles, how would you move over it guaranteeing you could over every part at least once) and after I started wondering how you would approach this if you couldn't guarantee the shape or the presence of obstacles. I've started looking at this with Dijkstra's algorithm, but I'm fascinated to hear how others approach this (or if there is a well accepted answer to this? (How does Roomba do it?)

mobile-robot artificial-intelligence algorithm coverage theory

edited Dec 5 '12 at 19:39

Josh Vander Hook
3,830 9 31

asked Dec 5 '12 at 18:40

Jason Sperske
191 1 1 6

tags like +algorithm and +theory would help a question like this but I don't have the reputation to add them yet – Jason Sperske Dec 5 '12 at 18:42

definitely something better than Roomba – Octopus Apr 26 '13 at 3:34

Interesting. I have bobsweep and it is programmed just perfectly momblogsociety.com/meet-newest-addition-family-bobsweep I suggest it to everyone. Greetings! – user7400 Nov 3 '14 at 15:57

1 Is this an advertisement? If not, you may want to post information, rather than just the link, explaining how the robot behaves and why it is just perfect. – Shahbaz Nov 3 '14 at 17:09

5 Answers

As far as I know, this problem hasn't been "solved."

Formally, this is an online coverage problem. Coverage, because we must cover each point on the floor, and online because we do not have offline access to the map. If you are interested in the most recent results, I suggest you lookup "**robotic online coverage algorithms**," perhaps in google scholar (there are *lots and lots* of great results). In addition to @embedded.kyle's very colorful (re)post, I'll add some details (I'll also try to quickly find a few simple results):

- Dijkstra's will get you a path, but not necessarily a coverage. For example, how do you specify to Dijkstra's that you must visit *each* point in the graph, instead of visiting *one* point as quickly as possible? You can run all-pairs shortest paths, but what are the points? You don't have a map.
- Online algorithms like this are often called "bug" algorithms because they tend to look like a bug wandering across an area, bumping into something, then wandering around it a bit.
- With no obstacles, and a rectangular room, and assuming you start on the *boundary* a boustrophedon (way of the ox) path is optimal.

Funny that farmers have been doing this forever, right? http://en.wikipedia.org/wiki/Boustrophedon. This can be extended to rooms with obstacles by finding roughly-rectangular area which are free of obstacles. Howie Choset worked on this a bit.

- To cover an area with unknown perimeter, and assuming you *don't* start on the perimeter, what is the optimal strategy? Well, imagine you drop into the world, and can't see the perimeter. You can walk in a straight line until you reach the perimeter, then do a boustrophedon, right? Except now you "wasted" the time spent finding the perimeter, because you will cover that part twice. This is why robots tend to spiral: By the time you reach a boundary, you have covered a lot of the area ($\pi * d^2$ where d is the distance to the *nearest* boundary, right?). This is helpful: now you are guaranteed to find the nearest boundary, and can trace it out.
- This is a huge, fascinating area. I'm sorry I can't provide a better summary!

The biggest problem is you don't have a map. Without a map, you are limited to simple actions like perimeter following, and moving along a path (like the spiral mentioned). So, there exist some robots that actually build the map while cleaning, decompose the mapped-out area into shapes, then cover each shape to ensure coverage. See: http://mintcleaner.com/

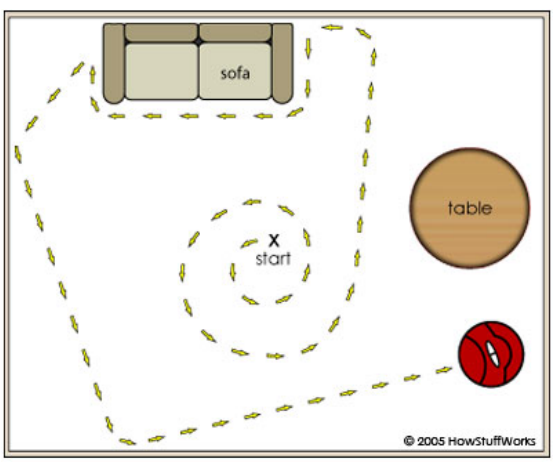
edited Dec 5 '12 at 20:06

answered Dec 5 '12 at 19:39

Josh Vander Hook
3,830 9 31

Roomba starts in a spiral until it hits something, then does a perimeter sweep. Then it just bounces around. Roomba being the de facto standard in household robotic vaccum cleaners, I guess you could call it the "accepted solution". But from personal experience (I own two), there is definitely room for improvement.

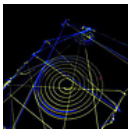
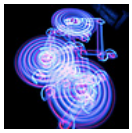
From How Stuff Works:



From an interview with Nancy Dussault Smith, Vice President of Marketing Communications at iRobot:

> When it starts you'll notice a spiral pattern, it'll spiral out over a larger and larger area until it hits an object. When it finds an object, it will follow along the edge of that object for a period of time, and then it will start cris-crossing, trying to figure out the largest distance it can go without hitting another object, and that's helping it figure out how large the space is, but if it goes for too long a period of time without hitting a wall, it's going to start spiraling again, because it figures it's in a wide open space, and it's constantly calculating and figuring that out.
>
> It's similar with the dirt sensors underneath, when one of those sensors gets tripped it changes its behaviors to cover that area. It will then go off in search of another dirty area in a straight path. The way that these different patterns pile on to each other as they go, we know that that is the most effective way to cover a room.
>
> The patterns that we chose and how the algorithm was originally developed was based off of behavior-based algorithms born out of MIT studying animals and how they go about searching areas for food. When you look at how ants and bees go out and they search areas, these kinds of coverage and figuring all of that out comes from that research. It's not exact, obviously, I'm not saying we're honeybees, but it's that understanding of how to search out an area in nature that is the basis behind how our adaptive technology is developed.

Some long exposure pics of Roombas with LEDs on them illustrate how it works in practice:


edited Jan 10 '13 at 14:12
Ian
9,455 1 12 43

answered Dec 5 '12 at 19:01
embedded.kyle
1,434 7 15

Is this a copy-paste content from the link? – Josh Vander Hook Dec 5 '12 at 19:06

@Josh The relevant material to answer the question has been copied from the linked sites and placed in a blockquote in order to prevent link rot. – embedded.kyle Dec 5 '12 at 20:19

The first thing you need to establish is the goal of the robot -- not quite clear from your question. There are two main tasks that your robot has to accomplish: discovering the shape of the clean-able area, and then cleaning it.

But is the amount of dirt constant? Is dirt added constantly? Is it your goal to minimize the average time that dirt remains on the floor, or the median time? Is the goal to keep the floor equally clean? Or is it just to clean once, as quickly as possible? Are there patterns in the accumulation of dirt that you can measure and use to your advantage in accomplishing your goal?

The answer to these questions will help guide what algorithm you choose. In the Roomba's robot's case, there might be no point learning the exact layout of the room because furniture (like chairs around a table), people, and other obstacles move very often. However, in your case it might be better to explore the space to build a complete map (some combination of a lawnmower pattern with finding edges), then use that map to compute the shortest path through the space (the term is "coverage", and there are several ways to do it e.g. spanning tree algorithm).

One more thing you'll need to worry about is how to discretize the space you're in. Because you can move in any direction -- even with integer degree amounts and integer units of distance -- your X and Y positions can have fractional values. You'll need to decide how to represent obstacles on that map without having it grow to an infinite number of data points.

answered Dec 5 '12 at 19:18
Ian
9,455 1 12 43

Well since I am sadistically making an interview question harder on myself I suppose I could come up with some more information. I like the points you are raising :) – Jason Sperske Dec 5 '12 at 19:23

Neato uses an organized approach. Using SLAM and bumpers, it maps the 'current' room, perimeter first, then applies some algorithm for cleaning as efficiently as possible. I've never

owned a Roomba, but given what I have read about it's algorithm, I would never switch from a neato.

The Laser Range Finder in the neato is often cannabilized for robotics, as it is a cost effective sensor for SLAM algorithms.

If I was given your task, First I would find a suitable SLAM implementation, based on the hardware I had.

Then I would use a CNC ISLAND Motion planning algorithm. My experience is they tend to be very efficient at covering an arbitrary area with the least amount of movement.

answered Dec 5 '12 at 19:21

Spiked3
1,853 6 14

SLAM isn't quite appropriate for this problem because it's a simulation in which there is no uncertainty in the positioning. For a real robot, you're absolutely right though. – Ian Dec 5 '12 at 19:25

I missed that (if its there). Actually it says the following are unknown; the location of the robot. – Spiked3 Dec 5 '12 at 19:27

I was thinking that the location began as unknown but as a topology of the room was created it might become known. – Jason Sperske Dec 5 '12 at 19:29

This is one of those odd academic problems where simplifications produce strange implications. Since you have no map, no starting location, perfect positioning, and no external entities to coordinate with, absolute position is irrelevant. You arbitrarily decide that (0,0) is your starting place, then build your map relative to that point. This isn't really practical in the real world, but does give you some practice with coverage algorithms. – Ian Dec 5 '12 at 19:39

Your answer is good from a systems perspective. However, I believe this question is a better theory/algorithms question. – Josh Vander Hook Dec 5 '12 at 20:07

Looking at the simpler problem that you were asked - a rectangular room, with no obstacles and clean every part **at least** once.

The solution is to find a corner of the room, and finding a corner won't be a big problem. Once that has been achieved, then just follow a spiral path to the center of the room.

edited Mar 24 '16 at 9:24

Greenonline
1,013 2 4 22

answered Mar 23 '16 at 20:00

user13259
1