Trying to KISS* with the Robot Vacuum Cleaner

(*Keep It Simple, Stupid)

James Gips and Donald Green

Computer Science Department
Boston College
Chestnut Hill, Mass. 02167

gips@bcvms.bc.edu

Abstract

We review some possible approaches to the robot vacuum cleaner problem and then propose our own approach, which is to adapt the classic depth-first maze-following algorithm to the problem. We have implemented a preliminary version of the algorithm on a mobile robot with sonar sensors, with encouraging results.

Possible Approaches

One of the many interesting aspects of the robot vacuum cleaner problem is the design space, which is rich and varied. No doubt there are many possible solutions to the problem. Good solutions will have pots of gold attached: fast-growing start-up companies, with exploding sales, that soon go public or lucrative licensing arrangements with multi-national vacuum cleaner companies excited about getting the jump on the next hot idea in vacuum cleaners.

How can we begin to explore and generate a large, complicated, unknown multi-dimensional design space ? One approach is to try to apply what we know, to try to apply existing solutions to similar problems. Here are some possible solution points scattered about the robot vacuum cleaner ("rvc") design space:

1. The pool cleaner approach. No learning, no mental map. The world is its best representation. The robot vacuum cleaner has sensitive bumpers and maybe a ring of sonars. The control program is simplicity itself: the rvc moves in a straight line. When it encounters an obstacle it goes off in a random direction. So the rvc continues, bouncing around the room vacuuming until it is turned off.

2. The simple learning robot. The user manually steers the rvc the first time. The rvc remembers the path it took, the distances and the turns. Ever after, the user starts it in the same location and orientation. The robot repeats the path it followed by dead reckoning. The robot might have bumpers and sonar so it can stop for pets and babies and for furniture out of position.

3. The survey robot. The rvc initially goes off and surveys the environment, building an internal map. Then the robot proceeds to use the internal map to determine the path it will follow and to guide it as it vacuums.

4. The user enters in a map of the environment using a mouse and screen. The rvc plans its path from the internal map.

5. The home equivalent of the office building mail-carrier robot. The user installs magnetic strips under the carpet for the robot to follow. Or the user sprays the carpet with a chemical that is invisible to the human eye but visible to a special sensing device on the robot. The chemical indicates the path over the carpet for the rvc to follow.

6. The user installs infrared reflectors on the walls and furniture so the robot can determine where the walls and furniture are. Or the user pastes barcode stickers on the walls and furniture for the robot to sense.

7. The telepresence rvc. The user puts on a special helmet with high-res stereo data displays and a dataglove and maybe even a body suit. The robot has video cameras mounted on it which communicate their images back to the high-res head-mounted display. Or maybe there are parallel supercomputers with a map of the environment that generates realistic stereo images of the view from the rvc in real-time. The user sits in the easy chair and controls the rvc remotely, lost in cyberspace.

8. The rvc is controlled by a simple Nintendo or Sega push-button controller. Through massive advertising we convince the ten year old son in the house and his millions of cohorts that trying to control the rvc so that it vacuums the entire room without bumping into furniture is the latest and greatest video game, a realistic 3-D Pac-Man for the 90's.

9. A voice activated rvc. The user sits in an easy chair and yells out the commands over the din of the motors: "Forward", "Left", "Right", "Stop", "Stop!", "STOP!!" ...

10. A million microbots spread out over the carpet to munch up the dirt in parallel a square millimeter at a time. Who cares how many of them fail ? The job still will get done. Of course, they are a bit crunchy under foot.

When the first automobile was developed it looked like ... a horseless carriage. The first robot vacuum cleaner will no doubt look like a vacuum cleaner with a computer inside or a mobile robot with a vacuum cleaner attachment. Gradually it will evolve down some unique path.

Maybe we are formulating the problem wrong. We don't want a robot vacuum cleaner. We want an automatic way of cleaning the carpets. An electrostatic system we install in the ceiling. A chemically self-cleaning carpet. Benign bacteria (that we have designed on our computer) that we sprinkle on the carpet to eat the dirt. A neural hypnosis helmet that convinces the wearer that the carpets are cleaned.

Instead of generating random solution points in the design space, we could think of an algorithm for systematically exploring the entire design space. The same idea works for the robot vacuum cleaner itself. We can think of the rvc as needing to explore the entire space of the room passing over each empty square foot of the room once, maybe even twice, not missing any areas and not going over any areas multiple times.

Our Approach

One solution is a classic depth-first "maze-following" algorithm, an algorithm for finding a path through a maze to a goal cell. We use the algorithm but eliminate the goal cell. Thus the algorithm starts at any cell and visits each empty cell in the space, eventually backing up and returning to the cell it started, where it quits.

In applying this algorithm to the robot vacuum cleaner problem we treat each square foot in the area as a grid cell. The robot uses a simple two-dimensional array as an internal map of where it has been and the path it followed. We start the robot in any location in a room. The array is blank. The initial coordinates for the robot are in the middle of the array, just to be safe. The robot proceeds in one direction. Before it moves it checks

the array to see if it has ever vacuumed the cell before. If not, it uses its sonar to check if there is an obstacle in the next cell. If not, it proceeds to the next cell and marks this off in the array. If there is an obstacle or if the rvc already has been in the cell, the rvc tries a different direction. It goes through the same procedure of checking its internal array and checking with the sonar before moving. When the robot is in a cell that is a "dead-end", a cell surrounded by cells that it has visited before and/or that contain obstacles, the robot backs up to the previous cell and resumes trying new directions from that cell. Eventually the robot vacuum cleaner backs up to the cell it began from and finds itself with no new cells to visit, whereupon it halts, its job completed.

Preliminary Implementations

We first tried a version of the algorithm in a simulation on a Macintosh. That worked -- whatever the configuration of obstacles and whatever the size and shape of the room, the square representing the robot visited every empty cell in the display, returned to the start cell, and halted when the job was done.

We then implemented the algorithm in a self-contained sonar-based mobile robot in our lab. The algorithm works fine, given the limitations of the robot hardware. We use cardboard boxes instead of furniture, as our current sonar has a difficult time seeing real tables. The robot visits each empty square foot, backing up through its path and trying new paths until it halts where it began.

Possible Improvements

We fully recognize that this is only the starting point for a real system:

- We would need to deal with children and pets darting in and out of the workspace. We handle this in the sense that if a cell is temporarily occupied the first time the robot is next to it the robot will avoid the cell the first time but still try the cell subsequent times it is in the vicinity.

- We should have touch-sensitive bumpers on our robot.

- Any real robot vacuum cleaner would have a power cord. We could modify the algorithm so that the length of the power cord bounds the distance from the starting point that the robot travels. Also, we could have the robot turn so as to minimize the cord wrapping around the robot.

- We could modify the algorithm so that it uses the sonar to "vacuum" right up close to an obstacle instead of avoiding any square foot that contains an obstacle.

- We would like to include a downward pointing sensor so the robot knows not to go plunging down the stairs. How to move our robot vacuum cleaner up or down the stairs is problematic as it would be fairly heavy. (A few years ago Nolan Bushnell, the founder of Atari, gave a talk in Boston on the possibility of home robots. A question from the audience pointed out that in California homes tend to be built all on one floor, whereas in New England we tend to build homes with several floors, often including a basement and attic. How is the homeowner in New England going to move his home robot from one floor to the next? Bushnell's answer: "It's tacky not to have at least one robot on each floor.")