International Journal of Computer Applications (0975 - 8887)

Volume 73 - No. 13, July 2013

Modified Boundary Fill for Complete Surface Coverage by Robotic Agents

Shanu Salunke

St. Francis Institute of Technology

Mt Poinsur, Borival, Mumbaii

ABSTRACT

Some robotic applications like vacuum cleaners require the robot to travel to each and every spot in the room while avoiding obstacles whose position may or may not be known. These obstacles may not always be in the same position. The robot itself may or may not have a map of the room. In the latter case, it has to rely on its sensors to navigate. It also may not know its absolute position in the room as the necessary hardware adds to both cost and complexity. This paper outlines an algorithm that can navigate to every spot in the room without requiring any knowledge of the robots absolute position nor maintaining a virtual map.

Keywords:

boundary fill, vacuum cleaner robots, household robots, surface navigationifx

1. INTRODUCTION

Robotics has wide and varied applications in education, military, industry as well as medicine. The one area where robotics has yet to carve a niche is the everyday household. Domestic robots are those used to perform everyday chores. Due to low performance, expense and more efficient unintelligent machines, robots have not been able to compete. That is slowly changing. Robotic Vacuum Cleaners are fast replacing their traditional counterparts in most households. These small, intelligent machines keep roaming the room, avoiding furniture and other obstacles, sucking up dirt as they go. Latest versions also have a docking facility where a robot that is running low on power returns to its charging station instead of dying in the middle of the room.

To be efficient, a robotic vacuum cleaner must cover the entire exposed area in the least amount of time. This paper reviews common navigation algorithms employed by commercial cleaners and proposes an alternative based on the famous Boundary Fill Algorithm commonly used in Computer Graphics.

2. BOUNDARY FILL ALGORITHM

The Boundary Fill Algorithm[1] as used in computer graphics, is a simple polygon fill algorithm. As shown in Fig. 1, one starts with a seed value i.e. a pixel inside the polygon and moves outwards toward the boundary. Given a seed value, the function checks all surrounding pixels. These neighbors will satisfy one of three conditions they form the boundary, they are already of the target color or they neither form the boundary nor are of the target color. When a pixel of the third type is encountered, it is colored. This method is widely used in interactive paint applications.

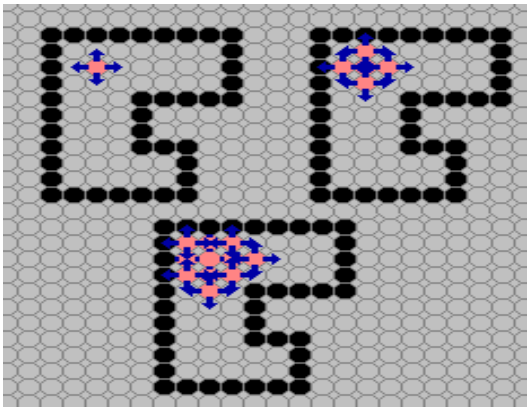


Fig. 1. Working of the simple Boundary Fill Algorithm

3. CURRENT PRACTICES

The first vacuum cleaner robot was released in 1996. However, it couldnt compete with traditional vacuum cleaners due to its limited efficiency and functionality. Today robotic vacuum cleaners like Neato and Roomba are steadily on the raise. This paper only covers the navigation algorithm used by various commercial vacuum cleaner agents. The operational mechanisms used by the top ten vacuum cleaner agents, as surveyed by toptenreviews.com [3] are listed in Table 1 As seen from Table I, the commonly used methods are Randomness, Spiral and Virtual Map.

Table 1. Commercial Vacuum Cleaner Robots and their Operating Mechanism [3]

Robotic Vacuum Cleaner	Operation Mechanism
Roomba 770	Back and Forth
Neato XV-21	Virtual Map
Cleanmate QQ3-T	S-Shaped, Wall and Spiral
Roomba 760	Random
Neato XV-12	Virtual Map
Cleanmate QQ2-T Plus	Repetitive
Roomba 560	Random
iTouchless Pro	N.A
DirtDevil RoomMate	Random, Spiral and Wall

3.1 Randomness

This is used extensively by Roomba. Whenever the robot bumps into an obstacle, it retracts, turns a random degree to the left or right and moves forward till it encounters another obstacle. The result of this type of motion is that the controller doesnt have to use processing power or memory for the purpose of navigation. This algorithm is also the easiest to implement. While the random method doesnt guarantee complete surface coverage, probabilistically, it covers most of the surface given an appreciable amount of time. [3]

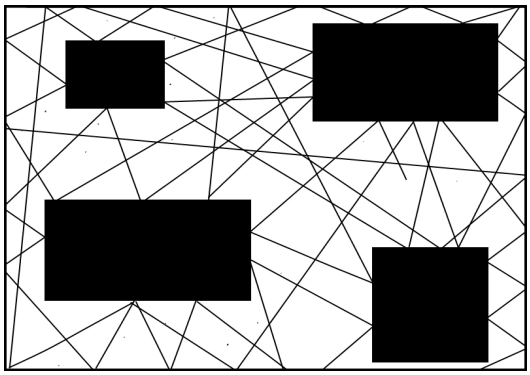


Fig. 2. Path taken by Roomba

3.2 Spiral

In this method [3], the robot first checks whether there is enough space for it to move. If yes, it moved spirally in the clockwise direction with increasing radius. This method is most effective when the robot is placed in the center of the room and when there are fewer obstacles placed far apart.

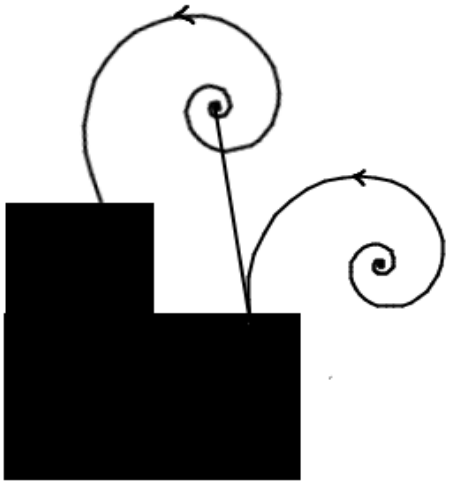


Fig. 3. Path taken by vacuum cleaner using spiral method

3.3 Virtual Map or SLAM (Simultaneous Localization and Mapping)

This method is used by the Neato vacuum cleaner robot. Neato initially scans the room to find out where it is located. It then moves to the nearest wall and starts cleaning the perimeter of the room. Its LIDAR (Light Detection and Ranging) keeps track of where the robot is and where it should be going. Although this method systematically covers the entire surface to be cleaned, it also makes the Neato one of the most expensive robotic vacuum cleaners in the market.

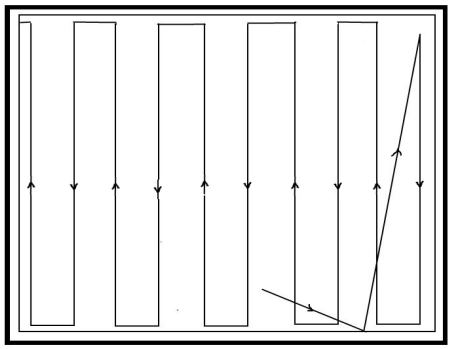


Fig. 4. Neato VX-12 navigating using SLAM

4. HARDWARE FEATURES OF THE ROBOT

A Robotic Vacuum Cleaner agent that uses the proposed algorithm must have some basic features. It should be able to sense an obstacle in four directions front, back, left and right. It should also be able to move in these four directions. Once the execution has begun, the robot must maintain its sense of direction i.e. if the robot starts while facing North, front always means moving North

5. MODIFIED BOUNDARY FILL FOR SURFACE NAVIGATION

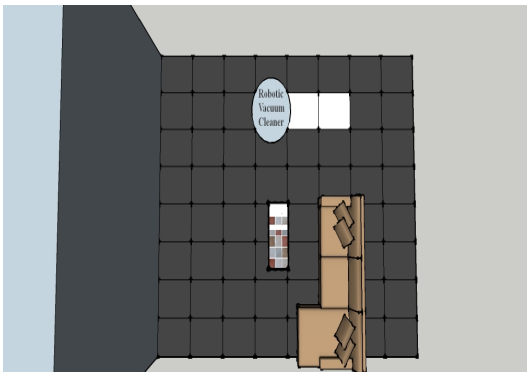
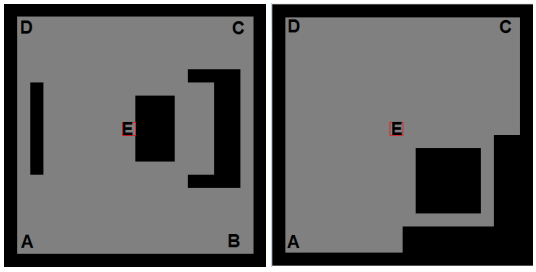


Fig. 5. The floor as a bitmap

A variation of the standard boundary fill algorithm can be used to determine that navigation path of the robot agent. The floor of a room can be thought of as a 2D bitmap as shown in Fig. 5. It would be composed of a fixed number of quadrilaterals which the robot can move to. Then the initial position of the robot can be considered its seed value and modifying the boundary fill algorithm appropriately to move from one square position to another can give a deterministic way to navigate the entire surface.
The robot needs to maintain three integer pairs its previous position (xp, yp), its current position (x, y) and its next position (xn, yn). It also needs to maintain a stack of some predefined length. This stack will be used to push and pop (x, y) integer pairs of the neighbors of the current position. These positions are with respect to its initial position and are not absolute coordinates.
The initial position of the robot in the room is assumed to be its

origin and represented by the integer pair (0, 0). The robot then checks whether any of its four immediate neighboring positions is a boundary. All positions that are free except the one that led to the current position i.e. (xp, yp) are pushed into a stack. The next position is obtained by popping the topmost element of the stack. Whenever the stack is full, the robot stops checking its neighbors and travels to all the positions in the stack by popping the stack elements till it is empty.

The boundary fill algorithm can potentially get stuck in an in-



(a) Map 1 (b) Map 2

Fig. 6. Samle maps used for the simulation

finite loop where the same positions are travelled in the same order and newer positions are left at the bottom of the stack, untraveled. The fixed length stack prevents infinite loops from being formed as whenever the stack is full, all the positions are travelled and invariably, newer neighbors are discovered. It also keeps a cap of the amount of memory being used and prevents Stack Overflow errors

6. SIMULATOR DESIGN

A simple simulator was designed using the Java programming language. Two algorithms were tested using this simulator the randomness algorithm and the modified boundary fill algorithm. Two cases were considered as shown in Fig. 6. They will be referred to as Map 1 and Map 2 respectively. In order to understand how the original seed value affects the working of the robot, five generic seed positions were used to simulate the algorithm. These seed positions are named A, B, C, D and E as shown in the figures.

7. RESULT

The simulator was run for seven hundred iterations in an effort to allow the random algorithm enough time to converge. The random algorithm was run thrice for each position in each map. The modified boundary fill (BFA) algorithm was run thrice with three different stack sizes 150, 300 and 450. Efficiency was calculated as the percentage effective area covered per run.

Fig. 7 shows the results as a bar graph. The random algorithm is highly inconsistent in terms of area coverage. The BFA, although consistent, performs well when placed in an uncluttered area. Fig. 8 show the simulated room after running the algorithms for seven hundred iterations. White represents the cleaned area while gray represents the unclean floor and black represents the furniture/obstacles.

8. CONCLUSION

The aim of this paper is to modify the Boundary Fill Algorithm so as to create a deterministic algorithm that could potentially replace the randomness that guides current commercial robotic vacuum cleaners. This algorithm ensures that the robot travels to each and every position in the room without maintaining a virtual map of the surroundings.

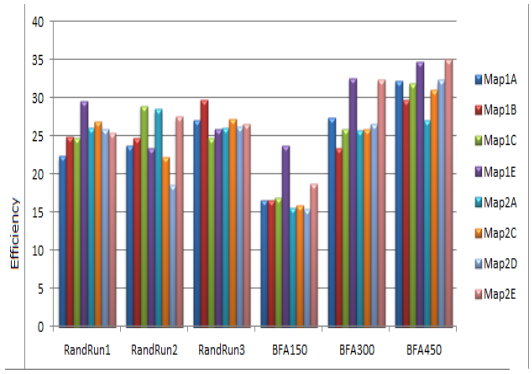
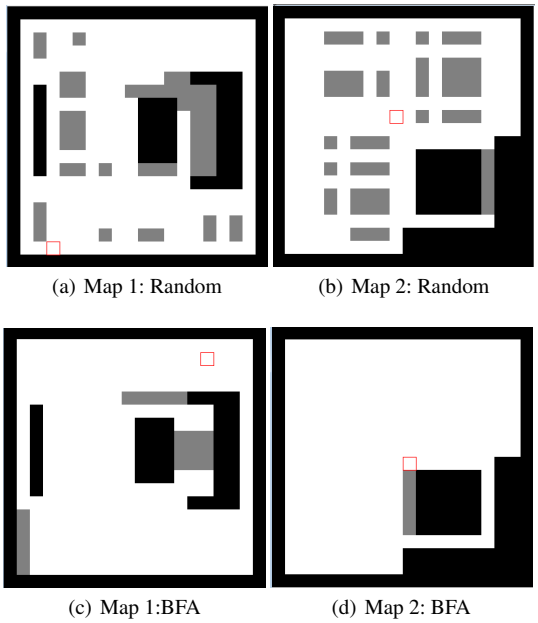


Fig. 7. Graphical Representation of Simulation Results. Map1E represents the Map1 and position E.



(a) Map 1: Random (b) Map 2: Random

(c) Map 1:BFA (d) Map 2: BFA

Fig. 8. State of Maps after Algorithm Execution

Although the random algorithm is easy to implement, the robot consumes a lot of power since its inherent inconsistency prevents it from covering the entire area effectively. The BFA could potentially replace this algorithm without adding to the cost of the machine.

References

[1] Donald Hern, M. Pauline Baker, Computer Graphics C Version, 2nd ed., Pearson Education pp.147.

[2] MIT Computer Science and Artificial Intelligence Laboratory, Boundary Fills,www.groups.csail.mit.edu/graphics/classes/ accessed on 10th September, 2012.

[3] Top Ten Reviews, 2012 Best Robot Vacuum Reviews and Comparison http://robot-vacuum-review.toptenreviews.com/ accessed on 10th September, 2012

[4] Krzysztof Skrzypczyk, Agnieszka Pieronczyk, Surface covering algorithms for semiautonomous vacuum cleaner , Proceedings Of The 12th WSEAS International Conference On Automatic Control,Modelling & Simulation, 2010.

[5] Roomba Art, http://geekartgallery.blogspot.in/2011/07/gallery-roomba-art.html, accessed on 15th September, 2012.

[6] Neato Robotics,Neato XV-11 All Floor Vacuum System White Paper, http://techrevu.com/storyfiles/

DE2010/0623NYC/Neato%20Robotics/Neato%20XV-11%20Robotic%20Vacuum %20Cleaner%20White%20Paper.pdf , as accessed on 13th September, 2012.

[7] Zelinsky, Alexander, et al. "Planning paths of complete coverage of an unstructured environment by a mobile robot." Proceedings of international conference on advanced robotics. Vol. 13. 1993.

[8] Ulrich, Iwan, Francesco Mondada, and J-D. Nicoud. "Autonomous vacuum cleaner." Robotics and autonomous systems 19.3 (1997): 233-245.

[9] Doty, Keith L., and Reid Harrison. "Sweep strategies for a sensory-driven, behavior-based vacuum cleaning agent." AAAI 1993 Fall Symposium Series Instantiating Real-World Agents Research Triangle Park, Raleigh, NC. 1993.