Autonomous Pathfinding

Stephen J. Tobias, A. Antonio Arroyo
Machine Intelligence Laboratory
Dept. of Electrical Engineering
University of Florida, USA
Tel. (352) 392-6605
Email: SteveT@mil.ufl.edu, Arroyo@mil.ufl.edu
URL: http://www.mil.ufl.edu

2000 Florida Conference on Recent Advances in Robotics
May 4-5, 2000, Florida Atlantic University

Abstract

One of the goals of autonomous robotics is for a root to learn about his environment and to be able to move about its environment in an intelligent manner. Using sensors that are nearly blind when compared to a human's, it is possible to learn this information and adapt to the environment at a fraction of the computing power.

Introduction

A number of methods have been used before to attempt to solve the problems of autonomous mapping and navigation. One method a robot could use to learn about its environment is to move around in it, and as it encounters object, map the object and place it in memory. By this method a robot would then learn where obstacles are as it encounters them and can avoid them in the future. Using this information the robot can decide how to move from point A to point B in an efficient manner without running into the previously mapped obstacles. This is a very straight forward method, but requires some perfect knowledge of its environment, like its exact position in space and its bearing when it encounters an object. Humans don't use this type of method to determine the best path across a room, would it be possible to create a procedure that a robot could use to navigate without this perfect information?

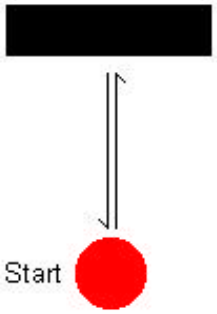
Reasons for Pathfinding

The net effect of mapping with a robot is that the robot moves from point A to point B without running into objects and it takes at least a fairly short path. Mapping out the objects requires fairly sophisticated sensor apparatus on the mobile robot, but after its environment is mapped, it is not nearly as necessary. Perhaps instead of finding the objets in its environment, find possible paths. The end result is the same, and the mapping process may be easier to the robot. Additionally, the robot may not find all the obstacles in its surroundings, causing problems later should it run into one. By locating the clear paths the robot is not concerned with how many objects it could run into in its environment nor would it need this information.

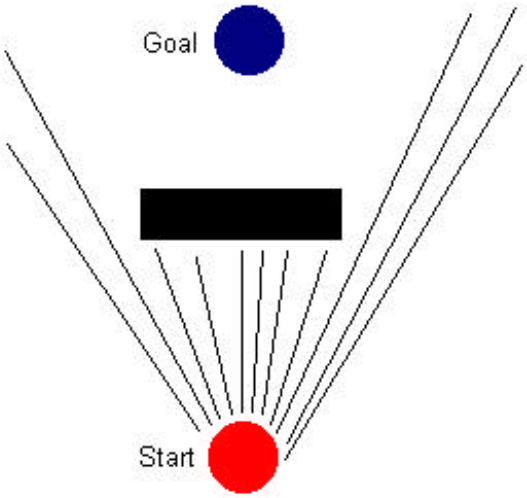
Procedure

How could a robot learn the potential paths to a destination? The robot must start from a know location that it knows to be correct. This would be the robots start or home points, or node. In this area, it has to have fairly accurate measure of its position, and be able to

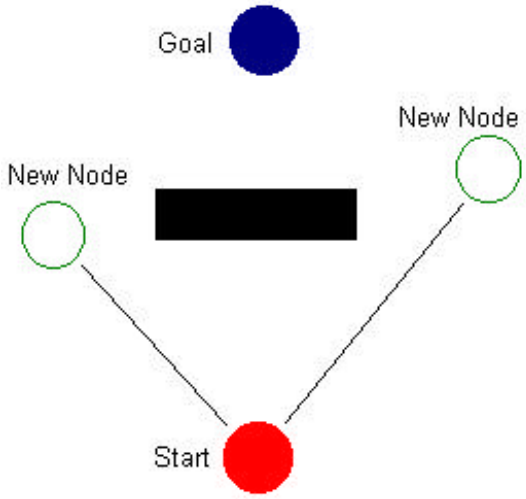
determine what direction it is facing. Defining the start region could be very simple, a small colored circle on the ground for example, would tell the robot it is in its start node. Direction is slightly more difficult, but could be solved with a compass or a beacon the robot could use to initialize its bearings to every time it came by its home point. From the start node, it could venture off in a direction until it encounters an object, then record the direction and approximate distance based on time of flight, or other means. An exact measurement is not necessary. Then move back to its start point.



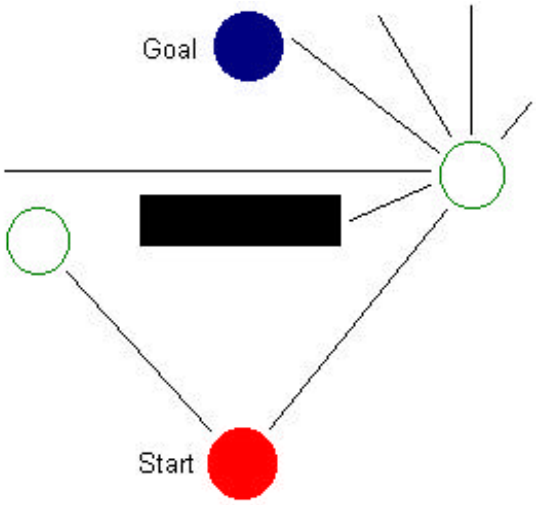
The robot would repeat this many times, each time from its start point in a slightly new direction.



The robot could then place new nodes along its longer paths for future investigation.

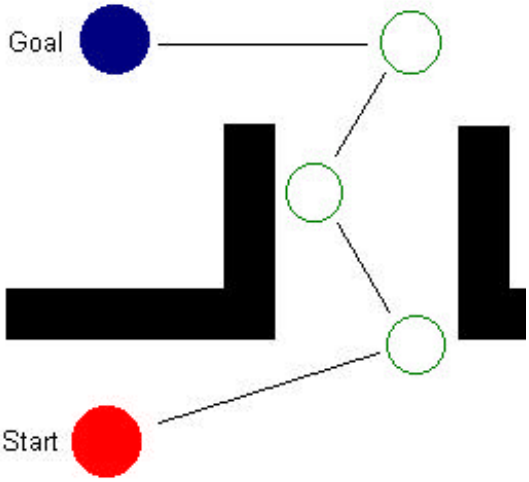


Then begin systematically exploring new paths from the new nodes.



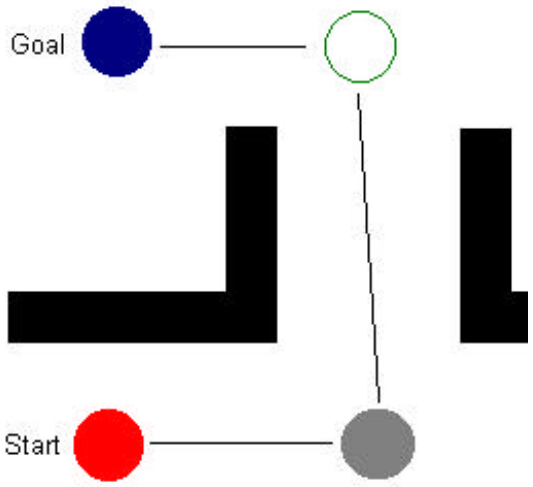
Once the goal has been reached, the robot can either discard all the other nodes, or continue to explore other possible paths. Rules used to explore trees or networks in pure data structures can be applied to the physical movement of the robot. The distances from a node to another node would correspond to the weight of that edge. However, this process may begin to break down when

it is used in tight spaces, like perhaps a hallway. The robot may not pick a good spot for a node and end up bouncing off walls unnecessarily.



Clearly, this is an over simplified example, but illustrates a point. The robot may not find the optimal path.

How do humans navigate? Typically people use landmarks to determine where they are, and where they should go. When someone leaves on a long trip, they take a direction until they find a landmark, then chose a new direction an move to the next. The robot in the procedure described above does something similar, except its landmarks are internal. It's a common direction and distance it uses to arrive at a node on the network it is building. One solution may be to supply the robot with some artificial landmarks to help it learn some possible paths.



This is paramount to giving it a hint to that it might be a good idea to explore a node based there instead of allowing it to make its own determination. Once the robot has learned the path however, it would not need the artificial landmark.

Disadvantages

Despite the fact that this method uses little computational power, and can rely on few sensors, there are a number of disadvantages to using it. The foremost would be that this process takes time. Each path, even if it is fairly short, would take at the least a few seconds. The number of paths radiating out from each node while the robot is in an exploration mode would be considerable. Additionally, the robot would probably have to calibrate its direction and distance according to the start node for each path, even if it is trying to expand another node. This is because, the robots sensor and actuation are not accurate, and can not be trusted over long periods of time. The characteristics of each individual robot, even if they are constructed from exactly the same components, are often slightly different. Therefor when an identical robot is given the prior information a previous robot found, it may not accurate for that specific robot. The maps are not portable. Directions may be ported to other robots, if they both use a global direction sense, like a compass.

Additionally, often a single robot can not be expected to behave in a perfect manner. A robot told to move in a straight line may, despite the designer's best efforts, turn slightly as it moves forward due to the slight differences in its wheels or motors driving those wheels. Again, the process I describe is ideal for an individual robot, but the

internal representation may not be portable to another. The process takes no assumptions about a robot, about how it might move, how fast or even what type of locomotion it has, so the algorithm may be portable.

Multiple of robots also complicate the process. Unless steps are taken to identify another robot on a path, when a robot is exploring, it may interpret another robot as an immovable object. Also, it will destroy results if the robot has to stop while it is exploring a path. Once a network has been established, multiple robots on any given edge may cause additional problems. The robots only know how to cross an edge in a straight line. This could be solved by assigning directions to edges and not allow robots to move either direction on a given edge. Another solution would be to use a controller to allow or deny robots down edges.

Conclusion

This process seems to be an excellent solution for those who can not place more expensive more accurate sensors on their mobile robot. It requires no global positioning information, except for that when it is at its start or end nodes. It is my hopes to prove this as a potential solution to other methods of mapping and navigation.