Simulation and Execution of Learning Methods and Algorithms of an Automated Lawnmower
TJHSST Senior Research Computer Systems Lab
2009-2010

Jeff Hobson
Period 3

April 9, 2010

Abstract

The purpose this project sets out to achieve is to develop efficient technology in the field of automation, and to combine this technology with the day-to-day task of mowing one's lawn. Automated Lawnmowers, or ALMs, are already in circulation, but these function on the property that if the ALM moves in random directions consistently during most of the day, then the entire lawn will stay trim (Husqvarna's Automower uses this process). The project centers around the idea that the mowing of a lawn can be done more efficiently, such as when it is done manually. Identifying cut grass versus uncut grass, dividing the lawn into sections to be completed at separate times, and avoiding obstacles are just three of the many ways humans work more efficiently than their automated counterparts. This project uses the computer languages C# (in implementation) and Java used in conjunction with the Processing Development Environment (in simulation) in order to test and evaluate the performance of an ALM as it grows to learn its environment and work more efficiently. [1]

Contents

List of Figures

1 Introduction

Mundane tasks involving manual labor have been greatly replaced by automated processes in today's day and age. Though automated lawnmowers (ALMs) have been created before, none are efficient enough to completely replace manual labor and complete the job in the same (or less) amount of time. Most operate under a technique similar to that of the Roomba, an automated vacuum cleaner.

1.1 Problem Statement and Purpose

The purpose of this project is to create a more efficient ALM that can learn to adapt to any environment and become even more efficient and spend less time mowing with each run. The unique nature of each individual's lawn makes this a difficult task to accomplish. Different techniques will be tested in order to maximize efficiency and minimize time spent mowing.

2 Background

This project involves numerous algorithms and learning methods. SLAM—Simultaneous Localization and Mapping—can be used to help create a virtual environment identical to the real one the robot traverses. This topic is to be implemented by another student and used in conjunction with this project. [2] A thesis paper from a Virginia Polytechnic Institute graduate was released to the public titled *Navigation and Control of an Autonomous Vehicle*. This paper has provided invaluable information about different learning techniques to be considered when creating an ALM. [3]

3 Development

The simlation portion of the project was developed first in order to provide a basis for the physical robot. The simulation was developed in much of the same order that the ALM will implement efficient techniques as it cuts on a day-to-day basis. First, an environment was created in which to put the simulated robot. Then, the robot is programmed to move randomly about the environment, turning left whenever it bumps into an obstacle. The robot then learns the capability to detect cut grass from uncut grass and begin to ignore areas that have already been cut. The methods described below are also implemented in turn.

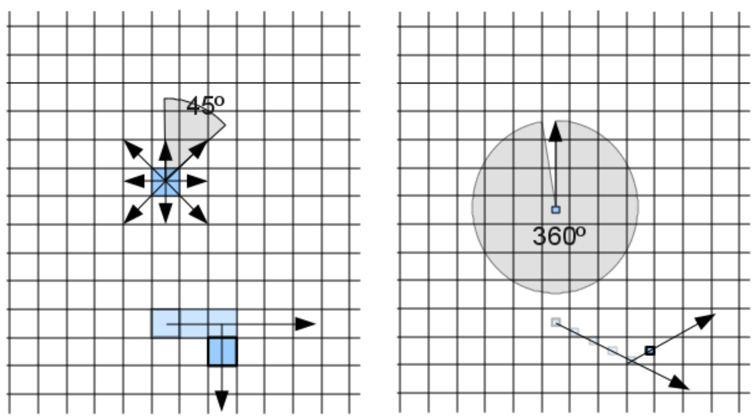
3.1 Description of Project Work

The ALM will learn to adapt to its environment based on a number of factors. The most prominent is SLAM. The robot will primarily use the virtual map it creates in order to determine where to move next. Initially, the robot wanders aimlessly around the yard, gathering data for its map. Once it has created a sufficient map, it proceeds to use the following methods to cut down runtime and maximize efficiency:

- Notice nearby obstacles and be aware of their proximity.
- Identify cut grass from uncut grass and avoid "backtracking," mowing over areas that have already been cut.
- Determine the width of the lawn at various points, so as to avoid moving into sections for which the entry point has already been mowed.
- Divide the lawn into sections as needed and complete entire sections rather than move about the entire lawn, possibly driving over cut grass unwillingly.
- Convert circular and elliptical obstacles into rectangular ones by cutting a border around them. This makes it easier for the ALM to move around them later.

All these methods are implemented in the simulation as well.

3.1.1 Vectors vs. Pixels

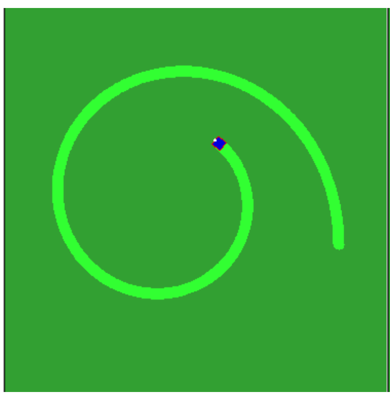


(a) "Pixel" Positioning (b) Vector Positioning

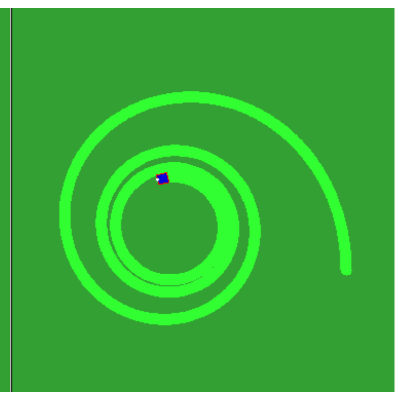
Figure 1: Two types of ALM positioning

When working with a simulation, the ALM is represented as a number of pixels on a computer monitor. This means simple processes such as how the ALM stores its location become much more complicated. The ALM's position is stored in a two-dimensional array, or a matrix, which acts very much like the pixels on a computer screen. An object must occupy the entire cell within a matrix, just as a pixel must be entirely turned on or off—there is no way to turn on *half* a pixel, for example. This can cause many issues when trying to "move" the ALM within the simulation. If coded based on the same principle as pixels, the ALM can only move in directions of 45 degree increments, as illustrated in Figure 1. This is where vectors are more useful. Vector positioning allows the ALM to move in any direction between 0 and 360 degrees. It also keeps track of the ALM's position *within* a cell of the matrix so that moving along a line with an irregular angle will stay straight and true. If the ALM were required to occupy the entire cell without knowing its position within it, then certain angles might cause the line to jump every few cells or follow a straight line along an incorrect angle. This is illustrated in the difference between the lower parts of Figure 1a and Figure 1b.

3.1.2 Problems



(a) Spiral – After 3 seconds



(b) Spiral – After 5 seconds

Figure 2: An incorrect spiral in which the distance between each loop becomes continually smaller, increasing at a quadratic rate

Just like any development process, many problems were encountered while creating the ALM. The first problem has already been described above: Vectors vs. Pixels. The next problem encountered along the way was finding methods of testing project data. There are numerous third-party programs that analyze and produce 3D graphs of scatter plot data, but very few that are available on the Linux operating system. A common one is gnuplot, which unfortunately refused to correctly display the data collected for grass heights over time. This problem still remains unsolved at the moment. An even larger (and still unsolved) problem encountered during the development process is depicted above in Figure 2: the spiral. The concept is simple: you have a length of rope. One end is tied to a self-propelling lawn mower, the other to a post in the middle of a lawn. The rope acts as a tether, constantly pulling the lawn mower towards the pole, in a large circle. As the rope winds around the pole, the radius of this circle gets smaller, and the lawn mower covers the entire lawn over time. Attempting to replicate this spiral involved much more advanced mathematics than originally anticipated, and has still not been solved.

4 Testing and Results

The scope of research for this project is narrow. The topics of SLAM and the different ways to mow a lawn are crucially fundamental in this project, but most methods an ALM is capable of, a human is also capable of, and are therefore easily researched through observation and logic.

4.1 Testing

The program is tested based upon three factors of efficiency:

- Time taken to mow entire lawn
- Percent coverage
- Amount of backtracking allowed

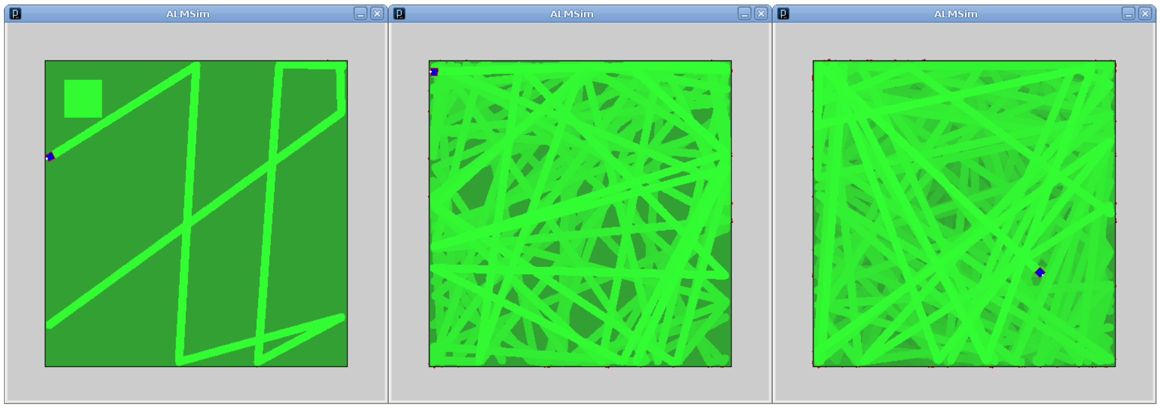
The robot will be timed as it mows the lawn on several simulated days to be sure the robot learns to better itself. Once the robot is able to determine the difference between cut and uncut grass, both backtracking (how much uncut grass was driven over more than once) and percent coverage (based upon how much uncut grass was missed) will be recorded by the robot and easily accessible to be sure the robot is making forward progress.

4.1.1 Example

Figure 2 demonstrate the progress of an ALM at its first run—a random-angle turning vehicle. Analyzing the ALM at different time steps will allow for a better model of its efficiency. One algorithm or method may be more efficient or quicker at the beginning of its run, but slow down or waste resources as it runs longer.

4.2 Results

The robot starts with poor efficiency during its first few runs on a new lawn as it moves randomly. Since the purpose of these runs is only to generate a map for the future, their efficiency is relatively unimportant in the large scheme of the project. The ALM should provide data that suggests a unique mixture of all or most of the methods will maximize the efficiency while minimizing the time spent on the lawn. Currently, the ALM will run making random-angle turns whenever it hits a wall or boundary. Although Figure 2c may make the ALM seem efficient, it has a hard time covering the small areas that are left after covering the majority of the lawn.



(a) After about ten seconds. (b) After just under one minute. (c) After just over two minutes.

Figure 3: An ALM on its first run making random-angle cuts in straight lines

5 Conclusion

I expect my results to conclude a unique combination of the proposed methods of cutting as the ”best” form. This will especially include the technique of dividing the lawn into sections. I hope to see little to no difference between the methods the ALM uses to cut and the methods *I* use to mow my lawn by utilizing effective divisions of the lawn and knowing where the lawn has and hasn't been cut.

References

[1] Husqvarna, “Husqvarna Automower”, http://www.automower.us, 2009.

[2] Sren Riisgaard and Morten Rufus Blas, “SLAM for Dummies”, pp. 1-44, 2003.

[3] Ian Schworer, “Navigation and Control of an Autonomous Vehicle”, pp. 1-84, 2005.

[4] Dustin Bates and Evan Dill, “The Ohio University Autonomous Lawnmower”, pp. 1-21, 2009.