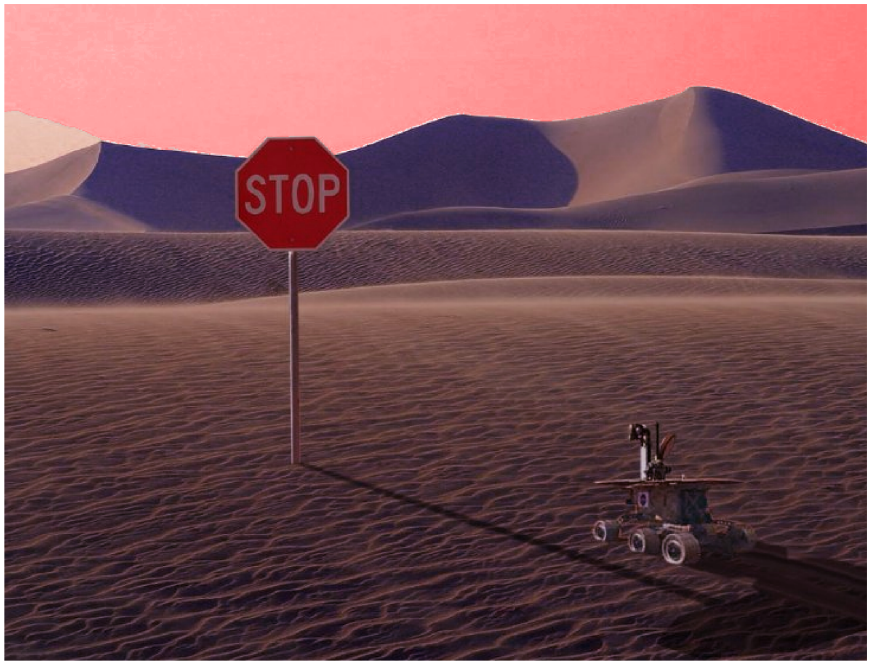
Search and Path Planing

CSCI-561 Course Project - Professor Laurent Itti

Developed by Nader Noori

January 23, 2012



introduction

This assignment is organized into two sections. In each section you will wrestle with a different path planning related problem. Each section contributes up to half of your grade for this assignment and includes a programming and a written part. For the programming part you will develop a program in c or c++. For the written part you will answer a few questions related to the problem and also the behavior of your c/c++ code. For more details about the formatting and grading guidelines please read the guidelines section of this document.

Path planning is the problem of finding the optimal rout from A to B which is usually defined as the shortest path between A and B. Search techniques can be adopted for solving this problem. You may consult with the textbook or visit `http://theory.stanford.edu/~amitp/GameProgramming/` to see how search techniques are applied to the problem of the shortest path.

In this project we will twist the problem of path planning a little bit just to give you the opportunity to deepen your understanding of search algorithms by modifying search techniques to fit the criteria of a realistic problem.

To give you a realistic context for expanding your ideas about search algorithms, we invite you to take part in a Mars exploration mission. The goal of this mission is sending a sophisticated mobile lab to the Mars to study the surface of the planet more closely. We are invited to solve two problems related to the navigation of the rover on surface of the planet. More specifically we are asked to help develop algorithms for finding landing sites and then finding the optimal path for the navigation of the rover based on a particular objective.

In the first section we will explain the problem of finding the optimal landing site and in the second section we will explain our path planning challenge.

The input of our program includes a topographical map of the mission site, plus some information about intended target locations and some other quantities that control the quality of the solution. The surface of the planet can be imagined as a surface in a 3 dimensional space. A popular way to represent a surface in a 3D space is using a mesh-grid with a Z value assigned to each cell that identifies the elevation of the planet at the location of the cell.

At each cell, the rover can move to each of 8 passable nearest cells: up, down, left, right, top-left, top-right, bottom-left and bottom-right. Actions are assumed to be deterministic (the rover will always end up within the intended passable cell). Note that moving to diagonal cells (top-left, top-right, bottom-left and bottom-right movements) will result to a movement along the grid plane $\sqrt{2}$ times larger than moving to other four closest neighboring cells.

Additionally, the rover is not designed to climb across steep hills and thus moving to a neighboring cell which requires the rover to climb up or down a surface which is steeper than a particular threshold value is not allowed. This maximum slope will be given as the input along the topographical map.

a. Search for the optimal landing site

A successful mission starts with landing in a good spot. So we are asked to help the exploration team find the optimal landing site. In this mission the quality of a landing site is defined by its accessibility to all sites that are identified by the field experts in advance. Target sites include those locations that look promising for gaining knowledge about the history of the planet, however, the schedule for

the exploration operation is subject to change and indeed depends on what is being discovered along the operation. So, from an operational perspective one might define a perfect landing spot as the location that has the best accessibility to all targets at the time of landing. This helps define a measure for the optimal solution: *the optimal landing spot is a location which has the shortest equal operational distance from all target locations on the surface of the planet* .

However note that the operational distance between two points is different from the shortest geometrical distance between them. This is because it may well be situations in which regions along the shortest path are not allowed for crossing and thus the rover has to take a longer but safer path to the destination. To avoid the risk of failing a several billion dollar mission, areas which are relatively flat are preferred for passing through and areas that are steeper than a threshold will be avoided. This threshold will be given to you in the input file that includes the map of the region.

Your task for this section is devising a search algorithm that finds the optimal landing site according to above measure. The location of the landing site will be used in the next section and should be reported in the output of your program. Meanwhile you need to prove the optimality of your solution in the written section of your homework.

Parallel BFS(s) for finding the optimal landing site

Regular path planning algorithms take the end points as the input and generate the optimal path between them. However in this problem we need to somehow find one of the endpoints along generating the optimal path(s) to that endpoint.

One approach is starting from all n target locations and search for a location that is reachable from all those target locations. So one can arrange n parallel search from n target locations till a particular location appears in all n explored spaces. However one should make sure that the first common explored state is equally distant from all n target locations by imposing a restriction on how these n parallel searches evolve.

b. Search for the optimal path

Now that we found the optimal landing site we can switch to the next section which is solving the problem of the optimal path form A to B. For an ideal rover that can cross every place, usually the shortest path is defined as the optimal path, however since in this project we have some operational concerns, our objective is first to avoid steep areas and then we want to minimize the path from A to B. This section begins with devising a suitable cost function to guarantee that we are not exposing our precious robot to danger and long and unnecessary trips. So the goal is roughly, finding the shortest path among the safest paths. *What defines the safety of a path is the maximum slope along that path.*

In this section of the project you need to devise a cost function which reflects the above concerns. *A complete uninformed search using your proposed cost function should result to the shortest path among all safest paths.*

This cost function then will be used in an informed search algorithm for finding the optimal path between the calculated landing site and the all target locations. Note that the cost function in this section does not necessarily

generate the shortest path which was used in the previous section. In fact for computing the optimal landing site we just considered the shortest operational path between two point.

The coding part

In your final submission, after compiling and liking your code you will have only one executable code which does both tasks: a. finding the optimal landing site b. finding the optimal path to all target locations.

The only input to your program is the path to an ASCII text file which contains the input for the problem. The information in this input file is organized as follow:

For your convenience the size of the world will be given in the first two lines of the input file. The first number (in the first line) represents the number of rows, the second number represents number of columns. The next line includes the threshold for a passable slope. This number will be a non-negative float.

Starting from the fourth line, the topographical map of the region will be given in consecutive rows. If the size of the world is $r \times c$ (r rows and c columns) then the map will be given in the form of r rows of c floats separated with the space character. Hence each line of text represents a single row of the grid, starting from row 1 and proceeding successively down the rows from there. So for each cell, the float value is the average elevation of the region.

Note that the given number is the average elevation of the region identified by the grid cell and it is not the slope of the region. Slopes and distances can be calculated using this information. Also note that elevations are given in the same unit as the sides of grid cells

The input text file will include the location of initial target sites at the end of file, in separate rows, each row including two integers indicating row and column address of the target. Here is an example of input file for the programing part:

```
6
8
0.9
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 6.0 6.4 5.7 4.5 6.6 6.8 0.0
0.0 6.3 6.4 5.7 4.5 6.7 6.8 0.0
0.0 5.8 6.4 5.7 4.5 6.8 6.7 0.0
0.0 6.0 6.1 6.7 6.5 6.6 6.9 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
2 2
4 7
```

Here the first line (6) shows the number of rows and the second line (8) shows the number of columns. The number of in the third line shows the threshold slop which defines whether moving from one cell to another neighboring cell is allowed. The next 6 lines define the elevation map of the region. The tenth line includes the grid address of one of targets (2,2) and the eleventh line shows the grid address of a second target location.

The output of your program should be a text file which include the row and column of the optimal landing site in its first row (row address first, space and

then column address). Then the path from the landing site to each of targets should be given by the following convention: each path is given in the form cell addresses along the path. Each cell address is given in the form of a pair of integers matching the row and column of the cell. The only allowed separator in each line is space. Here is an example of how an output for the above input file might look:

```
5 5
5 5 5 4 4 3 3 3 2 3 2 2
5 5 5 6 4 7
```

Here the fist line includes the address of the proposed optimal landing site (which may not be optimal in this example). The second line is a sequences of 12 integers which indeed show the path from the proposed landing site to the first target (this path includes 6 cells). The third line is a sequence of 6 integers which show the path from the proposed landing site to the second target.

So for a problem that has t number of target sites, the output will be given in $t+1$ lines. Generating a correct landing site (the firs line of your output file) contributes up to 30% of your grade. Calculating correct optimal paths (next t lines) also contributes up to 30% of your grade.

The written part

The written parts for each of two sections contribute up to 20% of your grade for this homework. Here are the questions to address in your written part of the homework marked with its maximum grade.

For section a.

1. Explain the detail of your algorithm (10%) :
Give your algorithm in pseudocode(5%) and then explain it in detail (5%)
2. The proof of the optimality of your solution (10%).
3. Proposing an informed search algorithm which is faster than our proposed algorithm will have up to 20% extra credit. You need to prove the optimality of your solution and compare the result of your search algorithm on a problem with that of our proposed algorithm. *(this part is optional)*

For section b.

1. Explain your choice of cost function (10%)
2. Explain your choice of the heuristic function (10%)

Answers on the written part will require justification and a simple yes/no answer will almost certainly garner you little or no points.

Guidelines

This assignment has a written and programming component. For the written part, please turn in typewritten answers. In general, it is safer to write longer answers than shorter ones, but stay focused as a long but off-topic answer will not work either. This way, we can discern your train of thoughts better and grant partial credit for a wrong answer, if you were on the right track.

The written portion should in some way be computer published. This means that graphs are produced with programs such as Adobe Illustrator or Microsoft Power Point and answers are typed. Thus, ideally you should produce a document in Latex, Word or even Quark XPress if that's your flavor. Really snappy looking graphs and homework hand-ins can sometimes get extra credit. What publishing software packages you use is totally up to you. This is a good opportunity to acquaint yourself with tools that will be extremely helpful in your future professional and academic career.

For the programming part, you will be provided sample inputs. Since each homework is checked via an automated Perl script, your output should match the example format exactly. Failure to do so will most certainly cost some points. Since the output format is simple. Additionally, if your code generates a large number of warnings during compilation, you will lose points, so try and eliminate compile-time warnings. Additionally, your code should be well documented. If something goes wrong during compile and grading, if the fix proves easy, the amount of points lost will be far less. As such, documentation makes fixing easier, so it is to your advantage to do so.

You will be provided with a stub Perl script called "stubby.pl", which works like the grading Perl script. You can run this script to make sure that your project will work properly. Thus, it is expected that your project will run through the grading Perl script without problems. Pedantically, we assert it will cost you points if your code does not run on the Perl script correctly. You will be able to tell if your output is correct if stubby shows each of the lines from your program output is exactly the same as from the comparison file which shows what output you should be getting.

To run stubby, copy it to your code directory along with the input, output and comparison files. Be sure to be in your code directory then type "perl stubby.pl". The script is loaded with all sorts of output and feedback, so you should be able to see what it is doing if for some reason it isn't working for you. If you don't understand the script and want to know more about Perl, go to `http://www.perl.org/` . Small amounts of extra credit (not more than 10% total) are given for all sorts of things. So long as you meet the base homework requirements anything creative, fun, interesting or outright cool will most likely earn you extra credit. What is cool and earns you extra credit is somewhat subjective and bound to the whim of the grader.

Handing in the Assignment:

The homework is due before class on the due date shown. There are two parts of the homework, which you must hand in. These are:

1. Your code tar/gzipped in one file. Do not send the binaries. - This should just include your uncompiled code and a readme file called readme.txt. When I run tar/ gzip to uncompress your file, it should uncompress into its own directory. To keep things uniform, do not use **bzip2**. I should then be able to run my stubby Perl script and grade your code. See below on how to tar/gzip your code. Also include my.environment.txt as generated by stubby.

2. Your "written" part on paper. You must submit a paper copy either in class or in a drop-box in front of my office in HNB (NOTE: the HNB building closes at 5:00pm). Late submissions will be noted. Be sure your homework is stapled together and is generally tidy. Be sure that your name is clearly visible

on all material you hand in. To hand in your code you can compress it with tar/gzip with a command like:

`tar -czvf my.name.platform.hw1.code.tgz my.name.platform.hw1.code`

You might also try:

`tar -cvf my.name.platform.hw1.code.tar my.name.platform.hw1.code`

and then type `gzip my.name.platform.hw1.code.tar` This will compress the contents of the directory named "my.name.platform.hw1.code" into a single file my.name.platform.hw1.code.tgz. If you need more info on this, try "`man tar`". Be sure to replace "`my.name`" with your own name and not literally hand in a file with the example name (I'm serious, some people actually do this).

Replace the word platform with the platform or machine (If it's a common machine) you compiled your project on. For instance linux or aludra are valid entries. Be sure to use all lower case letters in your file name like in the example. When we download your homework, we will extract each one. So if you have followed the directions given, all your material will be in a directory named platform.my.name.hw1.code on our side. This way your homework stays separate from other students and is easy to find and importantly grade when we need to.

Electronic Submission of code:

Submit your code via the Blackboard Turn-in facility. As mentioned, everything should be in one file. When you submit your file, be sure to fill in the information fields provided. This helps us resolve issues with misplaced homework and other issues. In general you should stamp your name all over the things you submit (better safe than sorry).

1. In the field marked "Name" put your name and USC email address. Put in your given name first and your family name last. So as an example, if your name is John Smith you would want to put in: John Smith johnsmit@usc.edu

2. In the comment field put in information about your submission. This should include which homework you are submitting, which revision it is if you turn in multiple copies and any additional messages you would like to include for the grader on the third line. As an example:
Homework 1
Submission 1
Comment Blah Blah Blah

3. Be sure and send your code when you feel you are totally ready to send it.

4. Due to limits on space, we cannot keep your homework forever on the Blackboard site. So keep in mind your submission record will be deleted after a few weeks.

5. The date and time of your submission are automatically recorded.

What exactly is Stubby?

This is very important. We use a Perl script to automate the grading process. Stubby is a Perl script we give to you so that you can check and make sure that your project conforms to specifications. That is, you can use Stubby to make sure that your assignment when handed in will run with our grading script. Thus, we have our own grading script like stubby, which we use to grade

your project. By checking to make sure your project works with stubby, you make sure that your project will work with our automated grading script. It is important to note that we will not use your version of stubby. We have our own script. As such, if you edit stubby to work with your code and not the other way around, this will not be very helpful. Additionally, since there are so many projects to grade, it is imperative that your program works with the grading Perl script. If your program does not work with the grading Perl script you will lose points.

Academic Honesty and Integrity

All homework material is checked vigorously for dishonesty using several methods. Over the past semesters this has netted around 10% of the students in each class at least once for a serious infraction. We are increasing the volume of our warning this semester in the hopes of bringing that number down. All detected violations of academic honesty are dealt with seriously. To be safe you are urged to err on the side of caution. Do not copy work from another student or off the web. Keep in mind that sanctions for dishonesty are reflected in your permanent record and can negatively impact your future success. As a general guide:
Do not copy code or written material from another student. Even single lines of code are dangerous to copy.
Do not copy code off the web. This is easier to detect than you may think.
Do not copy code from past students. We keep copies of past work to check for this.
Do ask the professor or TA if you are unsure about whether certain actions constitute dishonesty. It is better to be safe than sorry. Please, please, please stay honest. Enduring an inquiry for dishonesty is an experience much like having a root canal done and getting dumped on your senior prom all on the same day. We don't particularly enjoy the experience either. Why Follow Instructions Exactly as Written? In some instances you will lose points for not following directions. In general, this turns out to be easy points for most students. Additionally, you are more likely to receive your graded