Mars Attacked!
2011 ICTSA Programming Challenge

Christian Colombo, Gordon J. Pace, Chris Porter and Sandro Spina

February 2011

Abstract

This document outlines the programming challenge task that has to be solved in the 2011 ICTSA Programming Challenge. The competition will open on Friday, xxxth February 2011 at 20:00 and closes on Sunday, xxxth February 2011 at 20:00 ZST[1]. This document also specifies the criteria that will be used to judge the entries. The rules of the Programming Challenge can be found on the website http://programming.ictsamalta.org/

Judging Panel

The judging panel is made up of the following four members:

- Christian Colombo, Department of Computer Science, University of Malta `<christian.colombo@um.edu.mt>`
- Gordon J. Pace, Department of Computer Science, University of Malta `<gordon.pace@um.edu.mt>`
- Chris Porter, Department of Information Systems, University of Malta `<chris.porter@um.edu.mt>`
- Sandro Spina, Department of Computer Science, University of Malta `<sandro.spina@um.edu.mt>`

Any queries to the panel have to be directed through the designated competition Google group which can be accessed by clicking the 'support' button in the navigation menu of the main competition website. No emails sent directly to the judges will be answered. The role of the panel is to judge all submissions made to this competition and declare the winners. The judging panel's decision is final. The formal rules of the competition can be found at the competition's website. Ensure you read through them before you start.

[1]Zeus Standard Time — the time as set on UNIX server zeus, based at the Department of Computer Science at the University of Malta. This time can be seen on the Programming Challenge website

A Planet with a View

NoodleMaps Street and Canals View is ready to map out the planet Mars. A robot with a 360 degree camera has been designed and built to travel around on the rough terrain of the red planet which has already been mapped but not photographed. Due to budget constraints, the solar-powered robot will not have an inbuilt AI, but will have a hardwired sequence of directions to follow once landed at a specific location.

Your task is to write a program, which given a map of the terrain of Mars and the landing spot of the robot, produces a sequence of instructions which maximises the photographed land on Mars within a short period after landing (for the service to go online before competing ones). There are a number of constraints which the robot has:

Energy: The robot consumes energy to move from one location to another, especially when climbing uphill. However, at any point, it may stop and open its solar power panels to absorb solar energy to replenish its batteries.

Drops and climbs: The robot has been very hardily built, and may withstand very deep falls. However, it cannot climb very steep hills.

Photography: Despite the disadvantage of climbing up (and consuming more energy), there is a strong advantage — the higher up you are, the less is in the way of the robot camera, and the more it can photograph.

Detailed Specification

The Map: The area where the robot will land has been mapped as a two-dimensional chessboard like area, with each square tagged with its altitude (normalized as a whole number between 0 and 999). In Table 1 on page 7, is a 30 by 20 map, with its three dimensional representation in Figure 1.

An x by y map is represented as a $y + 4$ line text file. The first two lines give x and y. The third and fourth give the initial x- and y-coordinates of the robot (the top left corner is $(0, 0)$, the bottom right being (x, y)). The remaining y lines give comma-delimited altitudes, with a newline separating each line, starting from the top row of the map, moving down with each line, and each line starting from the left proceeding to the right. You can safely assume that each line will have the same number of altitudes (i.e. the map is rectangular). The map given in the previous example would thus be represented as shown in Table 2 (see page 8).

Robot instructions: The robot understands just five instructions: `N`, `S`, `E`, `W`, `X`. The first four instruct the robot to move one square to the north, south, east or west respectively, while the fifth instructs the robot to stop and absorb solar energy.

A text file with robot instructions consists of one line of characters from the alphabet `N`, `S`, `E`, `W`, `X` (case sensitive), without any separating spaces or symbols.

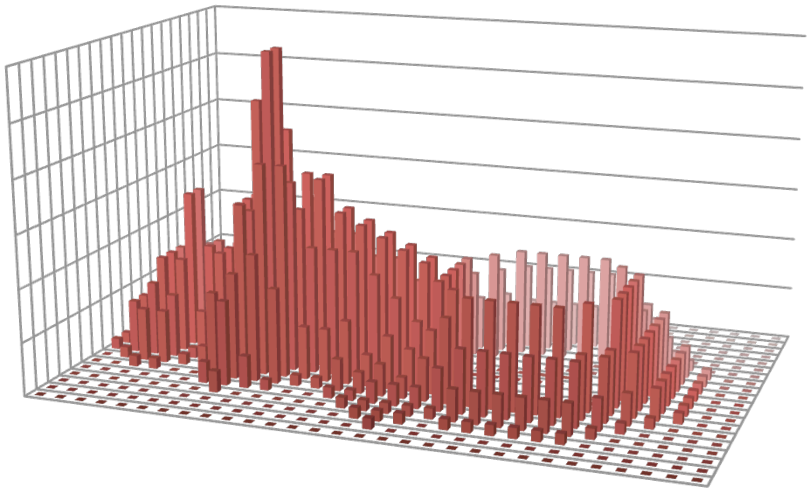


Figure 1: Three dimensional rendering of the area just south of the Greater Schiapparelli Canal

For example, the instructions to walk around in a 4 by 4 square resting twice in each of the corners, would be:

`NNNXXEEEXXSSSXXWWWXX`

(Yes, it's 4x4 draw it!)

Energy: Moving around Mars consumes energy according to the following formula:

$$e = \delta^2 + 10\delta + 10$$

where δ is the increase in height from the current location to the the next location. If there is a drop (the current location is higher than the next one), δ is taken to be 0.

Every time the robot executes the X instruction, it absorbs 500 energy units.

Constraints: The following are constraints on the robot movement:

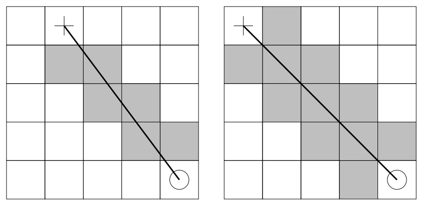
- The robot may never climb more than 50 units in a single step ($\delta \leq 50$).
- The robot may never move outside of the map boundaries.
- The robot may never perform a move which requires more energy than it currently has.

Other information: Other things to keep in mind:

- The robot may fall any distance without damaging itself.
- There is no maximum battery capacity.
- Initially, the robot starts off with 2000 energy units.

The view: At each position it visits, the robot photographs all the locations it can see up to a distance of 10 as long as there is no higher peak in between. The computation takes into account the straight horizontal line joining the centres of the current and observed location, and checks that no square which the line touches (even if it is just the corner) and which is higher than the current location of the robot.

The following pictures show which squares would be checked for higher altitude in two different positions (in both cases, the robot is in the bottom right hand corner of the grid marked by a circle, and the location which it trying photographed is in the one also marked by a cross in the top left region). Note that this analysis would be done for each target square whose centre is within a distance of 10 from the centre of the current location — since both squares shown in the analysis in the figure fall in this range (they are 5 and $\sqrt{32}$ units from the robot respectively) they would be amongst those checked.



More precisely, when at location (x, y), the robot photographs its current location and all other locations with coordinates (x', y') such that:

1. The horizontal distance[2] between (x, y) and (x', y') is not more than 10: $\sqrt{(x - x')^2 + (y - y')^2} \leq 10$.
2. There is no location (x_m, y_m) for which all the three following conditions hold:
   (a) The horizontal distance between (x, y) and (x_m, y_m) is smaller than the distance between (x, y) and (x', y').
   (b) There is at least one point in the square with top left corner $(x_m - 0.5, y_m + 0.5)$ and bottom right hand corner $(x_m + 0.5, y_m - 0.5)$ which lies on the line joining (x, y) and (x', y'). The outer edge and corners of the square also count towards occlusion.
   (c) The altitude of (x_m, y_m) is higher than the altitude of the current location.

Note that in this competition we use an approximation, since the real line-of-sight computation would have to take into account smaller peaks between the current location and the observed one.

[2]By horizontal distance, we mean 'the distance disregarding altitudes'.

The Task

The task is to write a program which given two command line parameters:

```
robot map.txt n
```

will read `map.txt` (a text file specifying the map — its size, the initial location of the robot and the map elevations), and output a text file `moves.txt` which specifies a valid route of up to `n` robot instructions.

If the trace is not of the right length, or results in either (i) the robot trying to climb a hill that is too steep or (ii) the robot running out of energy (it requires more energy than it has to perform one of the steps); or (iii) the robot walks beyond the limits of the known, mapped world – then the trace is considered invalid and will be disqualified.

Judging Criteria

The judging panel will run all submissions on a number of pre-defined Mars terrains and number of permissible steps. Each program will be given up to 1 minute to generate a route for each problem instance. Programs not terminating on a particular problem within 1 minute, or producing wrong output (too long a route, trying to climb up edges which are too steep, trying to perform a step which requires more energy than the current charge or invalid syntax) will get no points for that problem

For each problem instance, the programs will be ranked according to how much of the terrain has been explored and photographed.

The score at the end of a trip is a triple: (i) camera coverage — the number of distinct squares in the grid which the robot has managed to view; (ii) robot coverage — the number of distinct locations the robot has walked through (because the robot is also collecting soil samples as it goes along); (iii) energy efficiency — the energy remaining at the end of the trip.

For each problem instance, the teams are ordered according to the camera coverage criteron (i) (the larger the better). A score is then given to each team according to their ranking: best 50 points, second 30 points, third 20 points, fourth 10 points and fifth 5 points. The rest will not be awarded any points.

In the case of ties on an individual problem, all solutions with the same camera coverage, the team with the highest robot coverage, criterion (ii), is given priority. If they still give the same results, their energy efficiency (iii) is used (the higher the better). In case the tie is still not resolved, they will be given the same ranking for that problem.

The winning team will be the one which obtains the largest total number of points. In the unlikely case of unresolved ties, the ranking will be done according to the execution time of the program over all the problem set (fastest program wins) after which the judging panel reserves the right to resolve any further tie in as fair a way as possible using a tie breaker.

Submission Rules

All submissions have to be either (i) a single Microsoft Windows executable file – a command line program called `robot.exe`; or (ii) a zip file with one jar file and a batch file called `robot.bat`. In both cases, the executable or batch file must take exactly two parameters: the name of the map file it should read and the maximum length of the route.

No new windows or graphical user interfaces are to be opened by the program. The program will be executed on an Intel-based PC and having 2GB of RAM. The operating system will be Microsoft Windows XP and it will also have the latest version of the .NET Framework installed and the latest Java version installed. Any submissions that fail to execute on the mentioned environment or any submission that take longer than 1 minute to output a result will not be considered by the judging panel.

Each registered team is allowed to submit as many solutions as it wishes. The last submission on Sunday, xxxth February 2011 at 20:00 ZST will be considered for the prize. After these times no corrections or resubmissions will be allowed. Submissions can only be done through the programming challenge website — note that the program must finish uploading before the aforementioned time.

The Judging Suite

To ensure fairness, the problems that will be used for judging the submissions will also be made available when the challenge opens. These will be encrypted and at the end of the competition, the key will be given to ensure that the files were unchanged.

Clarifications

During the period between Friday, xxxth February at 20:00 and Sunday, xxxth February 2011 at 20:00, all the teams can send queries for clarification through the use of the designated competition Google group which can be accessed by clicking the 'support' button in the navigation menu of the website. No emails sent directly to the judges will be answered. If the judging panel deems the query to be a valid one, and not answered in this document, the answer will be posted back to the group. The judging panel will do its best to answer these queries in as short a time as possible.

Table 1: Just south of the Greater Schiaparelli Canal

0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	10	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	10	10	10	0	10	10	10	10	10	10	0	0	0	0	0	0
0	0	0	0	0	0	0	20	0	0	0	0	0	10	10	10	10	10	30	30	30	30	30	30	10	0	0	0	0	0
0	0	0	0	0	0	20	80	30	10	0	0	10	10	20	20	20	40	60	60	60	60	60	60	30	10	0	0	0	0
0	0	10	10	0	0	80	100	120	90	10	10	30	20	30	20	40	80	100	100	100	100	100	60	60	30	10	0	0	0
0	10	50	50	10	20	10	160	200	200	50	50	60	30	50	40	60	100	0	0	0	0	0	100	60	60	30	10	0	0
10	50	70	60	10	50	20	150	300	180	120	120	120	100	80	60	100	0	0	0	0	0	0	0	100	60	30	10	0	0
10	50	90	90	40	100	150	250	300	150	180	150	140	130	120	110	100	0	0	0	0	0	0	0	100	60	30	10	0	0
10	50	90	150	100	100	150	200	220	180	180	150	140	130	120	110	100	0	0	0	0	0	0	0	100	60	30	10	0	0
10	50	90	150	100	100	150	150	100	150	120	120	100	100	80	80	100	0	0	0	0	0	0	0	100	60	30	10	0	0
10	50	90	90	40	100	100	100	80	120	80	90	70	70	50	70	100	0	0	0	0	0	0	0	100	60	30	10	0	0
10	50	70	60	10	50	70	100	50	80	50	70	50	50	30	50	80	100	0	0	0	0	0	100	60	60	30	10	0	0
0	10	50	50	10	20	30	70	10	50	30	30	10	20	20	30	50	80	100	100	100	100	100	60	60	30	10	0	0	0
0	0	10	10	0	10	10	30	0	30	20	10	0	10	10	20	30	50	80	80	80	80	80	40	40	10	0	0	0	0
0	0	0	0	0	0	0	10	0	20	10	0	0	0	0	10	20	30	40	40	40	40	40	30	20	0	0	0	0	0
0	0	0	0	0	0	0	0	0	10	0	0	0	0	0	0	10	20	30	30	30	30	30	20	10	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	10	20	20	20	20	20	10	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	10	10	10	10	10	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 2: The representation of the map shown in Table 1

```
30
20
10
10
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,10,0,0,0,0,0,0,0,0,0,0,0,0,0,0
0,0,0,0,0,0,0,0,0,0,0,0,0,0,10,10,10,0,10,10,10,10,10,10,0,0,0,0,0,0
0,0,0,0,0,0,0,20,0,0,0,0,0,10,10,10,10,10,30,30,30,30,30,30,10,0,0,0,0,0
0,0,0,0,0,0,20,80,30,10,0,0,10,10,20,20,20,40,60,60,60,60,60,60,30,10,0,0,0,0
0,0,10,10,0,0,80,100,120,90,10,10,30,20,30,20,40,80,100,100,100,100,100,60,60,30,10,0,0,0
0,10,50,50,10,20,10,160,200,200,50,50,60,30,50,40,60,100,0,0,0,0,0,100,60,60,30,10,0,0
10,50,70,60,10,50,20,150,300,180,120,120,120,100,80,60,100,0,0,0,0,0,0,0,100,60,30,10,0,0
10,50,90,90,40,100,150,250,300,150,180,150,140,130,120,110,100,0,0,0,0,0,0,0,100,60,30,10,0,0
10,50,90,150,100,100,150,200,220,180,180,150,140,130,120,110,100,0,0,0,0,0,0,0,100,60,30,10,0,0
10,50,90,150,100,100,150,150,100,150,120,120,100,100,80,80,100,0,0,0,0,0,0,0,100,60,30,10,0,0
10,50,90,90,40,100,100,100,80,120,80,90,70,70,50,70,100,0,0,0,0,0,0,0,100,60,30,10,0,0
10,50,70,60,10,50,70,100,50,80,50,70,50,50,30,50,80,100,0,0,0,0,0,100,60,60,30,10,0,0
0,10,50,50,10,20,30,70,10,50,30,30,10,20,20,30,50,80,100,100,100,100,100,60,60,30,10,0,0,0
0,0,10,10,0,10,10,30,0,30,20,10,0,10,10,20,30,50,80,80,80,80,80,40,40,10,0,0,0,0
0,0,0,0,0,0,0,10,0,20,10,0,0,0,0,10,20,30,40,40,40,40,40,30,20,0,0,0,0,0
0,0,0,0,0,0,0,0,0,10,0,0,0,0,0,0,10,20,30,30,30,30,30,20,10,0,0,0,0,0
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,10,20,20,20,20,20,10,0,0,0,0,0,0
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,10,10,10,10,10,0,0,0,0,0,0,0
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
```