ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Institute for Dynamic Systems and Control
IDSC
Institut für Dynamische Systeme und Regelungstechnik

151-0563-01 **Dynamic Programming and Optimal Control** (Fall 2012)

Programming Exercise #1 Topic: Deterministic Systems and the Shortest Path Problem

Issued: Oct 24, 2012 Due: Nov 07, 2012

Nico Huebel (nhuebel@ethz.ch) , 24. Oktober 2012

Find a shortest path using the Dynamic Programming Algorithm, the Label Correcting Method, and the A^* Algorithm

The goal of this programming exercise is to find an optimal path for a robot through a known rough and hilly terrain. The terrain is discretized into a grid and the altitude of each grid cell is given by the elements of the terrain matrix T, $T \in \mathbb{R}^{m \times n}$. All elements of T are nonnegative. Obstacles, i.e. regions that cannot be traversed by the robot, will have the value ∞ in the terrain matrix T. The state $x_k \in \mathbb{N}^2$ without zero corresponds to the grid position of the robot at time k, i.e. the state $x_k = (x_{k,1}, x_{k,2}) = (1,1)$ corresponds to the first row and first column of the terrain matrix T and the state $x_k = (1,2)$ corresponds to the first row and second column of the terrain matrix T. The objective is to find an optimal path from the starting cell s to the terminal cell t. At each step the robot can choose to go to one of its four neighboring grid cells. Diagonal 'jumps' are not allowed. To prevent the robot from leaving the known terrain, the border of the terrain is treated as an obstacle, i.e. the entries of the first and last row and column of T are ∞. While it is difficult for the robot to climb hills, it has no problems to drive downhill. Therefore, the cost function to be minimized is

$$g_k(x_k) = \begin{cases} 0 & \text{if } x_k = t, \\ \max(\ 0\ ,\ T(x_{k+1}) - T(x_k)) & \text{otherwise.} \end{cases}$$

Part I The optimal path problem is to be solved by

(a1) creating a cost matrix $P \in \mathbb{R}^{(m-2)(n-2)\times(m-2)(n-2)}$ from the terrain matrix T using the cost function $g_k(x_k)$. For creating P you need to label all elements of the terrain matrix T with a unique node number $i = 1, 2, \ldots, (m-2)(n-2)$. Ignore the first and last row and column of T to reduce the dimension. For example, state $x_k = (2,2)$ becomes node $i = 1$, $x_k = (2,3)$ becomes $i = 2$, and $x_k = (3,2)$ becomes $i = n-1$.

(b1) Using the cost matrix P you can convert the shortest path problem to a deterministic finite-state problem and solve it with the Dynamic Programming algorithm (see class textbook, p. 67/68), **and**

(c1) apply the Label Correcting algorithm (see class textbook, Sec. 2.3.1) with depth-first method for selecting a node from the candidate list at each step (see textbook p. 85).

Part II The cost function is changed and is now given by

$$g_k(x_k) = \begin{cases} 0 & \text{if } x_k = t, \\ \max(\ 1\ ,\ T(x_{k+1}) - T(x_k)) & \text{otherwise.} \end{cases}$$

The shortest path problem is to be solved by

(a2) creating a cost matrix P from the terrain matrix T using the new cost function $g_k(x_k)$, **and**

(b2) converting the shortest path problem to a deterministic finite-state problem and solving it with the Dynamic Programming algorithm (see class textbook, p. 67/68), **and**

(c2) applying the Label Correcting algorithm (see class textbook, Sec. 2.3.1) with depth-first method for selecting a node from the candidate list at each step (see textbook p. 85), **and**

(d) applying the A^* algorithm with the lower bound on the cost-to-go from x_k to node t given by the so-called city block distance or Manhattan distance $\sum_{i=1}^{2} |x_{k,i} - t_i|$. Modify your implementation from (c2) to obtain the A^* implementation.

Provided Matlab Files

A set of Matlab files is provided on the class website. Please use them for solving the above problem.

`script.m`	Matlab script that can be used to load terrain matrices, execute the shortest path algorithms and display the results.
`generateCostMat.m`	Matlab function template to be used for creating a cost matrix P from the terrain matrix T.
`sp_dpa.m`	Matlab function template to be used for your implementation of the Dynamic Programming algorithm for the shortest path problem.
`sp_lca.m`	Matlab function template to be used for your implementation of the Label Correcting algorithm for the shortest path problem.
`sp_astar.m`	Matlab function template to be used for your implementation of the A^* algorithm for the shortest path problem.
`terrainTX.mat`	Terrain matrices T specifying the terrain the robot should move in.

Tasks

Implement your solution for (a1+2) in the file `generateCostMat.m`. Implement your solutions for problem (b1+2), (c1+2), and (d) in the files `sp_dpa.m`, `sp_lca.m`, and `sp_astar.m`, respectively. Be aware that you can use the same files for ***Part I*** and ***Part II***, but you have to supply a different cost matrix. Your code must be able to run with the Matlab script `script.m`.

For evaluating your solution, we can test it on the given terrain matrices as well as on other examples of the same structure.

Deliverables

Please hand in by e-mail

- your implementation of the function `generateCostMat.m`;
- your implementation of the DP algorithm `sp_dpa.m`;
- your implementation of the Label Correcting algorithm `sp_lca.m`;
- your implementation of the A^* algorithm `sp_astar.m`;
- in a `pdf`-file, answers to the following questions

  1. What is the shortest path (cost and path for ***Part I*** and ***Part II***) for the problem given in
     `terrainT1.mat` for starting node $s = 1$ and terminal node $t = 7$?

2. Explain the difference in the behavior of the robot caused by changing the cost function from ***Part I*** to ***Part II***. What does that change in the cost function mean physically?
3. Why does the given heuristic for the A^* algorithm not work in ***Part I***?
4. Compare the three algorithms. Which algorithm is the fastest to compute the shortest path and which one is the slowest? Why is this the case?
5. What does the following cost function

$$g_k(x_k) = \begin{cases} 0 & \text{if } x_k = t, \\ T(x_{k+1}) - T(x_k) & \text{otherwise} \end{cases}$$

mean physically? How would the optimal path that minimizes the following cost function look like?

Please include all four files into one `zip`-file, which you name `DPOCEx1_Names.zip`, where *Names* is a list of the surnames plus initial letter of the first name of all students who have worked on the solution[1]. Example: `DPOCEx1_HuebelN_GajamohanM.zip`

Send your file to Nico (`nhuebel@ethz.ch`) until the due date indicated above. We will send a confirmation e-mail upon receiving your e-mail. You are ultimately responsible that we receive your solution in time.

Plagiarism

When handing in any piece of work, the student (or, in case of a group work, each individual student) listed as author confirms that the work is original, has been done by the author(s) independently and that s/he has read and understood the *ETH Citation etiquette* (http://www.ethz.ch/students/exams/plagiarism_s_en.pdf). Each work submitted will be tested for plagiarism.

[1] Up to three students are allowed to work together on the problem. They will all receive the same grade.