TDDC17 Artificial Intelligence :

Lab 2 : Search Algorithms

Arnaud Pecoraro

September 2015

1.1 Aim

This first lab session explores the concept of intelligents agents. The agent is a Vacuum cleaner evolving in a simple grid. Each square of that grid is either clean or filled with dirt. The agent can only perform basic actions : move forward, turn right, turn left and suck. The purpose of this lab is to write an algorithm to automate the cleaning process. No supposition regarding the world dimension or the dirt position can be made. The vacuum cleaner has to be completly autonomous.

1.2 Strategy

Given that this first assignement does not imply obstacle avoidance, it is possible to implement an easy 3-step strategy :

— First, move the vacuum cleaner back to his home position, (1,1) on the grid, after he has performed its initial random moves.
— Then, start *snaking* : moving forward horizontaly and turning 180° each time it hits a wall.
— Finally when the vacuum cleaner reaches the last square, it goes back to the home position.

Note that the first step is arguable. Indeed in this lab, the number of random moves is set to 10. But, assuming a huge grid or a lot more random moves, as the score is based on the total amount of actions, it could be more efficient to implement a different strategy.

1.3 Decision Loop

The decision loop is a set of conditions based on either the vacuum cleaners sensors (*bump, clear, dirt*) or values defining the strategy (for example the boolean *isCleaned*). Each turn of the loop returns an Action object.

The first phase is performed with the *moveToHomePosition()* method. Indeed while phase 1 is not complete this method is executed. It initiates a sequence of movements to return to the home position.

Then, the real decision loop starts : this is *phase 2*. In this phase, the vacuum cleaner is snaking. Four different situations can occur : *clean*, *dirt*, *bump while going right* and *bump while going left*.

If *dirt* is *false*, the vacuum cleaner keep moving forward. If *dirt* is *true*, the vacuum cleaner simply suck the dirt.

Handling the bump is slightly more complicated as while snaking, the vacuum cleaner will perform a different sequence of actions depending on its current direction as it bump into the wall. The integer *turningDirection* is set to 0 if the agent was going in the east direction (agent_direction equal to 1) and 1 if it was going west. The boolean *isUTurning* is set to true.

Each turn *isUTurning* is tested to complete the 180° turning sequence. The variable *uMovement* is used as a conditional counter in this sequence.

This sequence is composed of three movements :
— Turning 90° in the given direction (*uMovement equal to 0*)
— Moving forward, down to the next grid line (*uMovement equal to 1*)
— Turning again in the same direction (*uMovement equal to 0 again*)

During phase 2, the agent should never bump while performing the second movement of this sequence. If it does, it means that it has reached the bottom of the grid and the cleaning is over. The boolean *isCleaned* is set to true and the program goes back to phase 1 but with this ending condition. Once the vacuum cleaner is home, the execution stops.

1.4 Possible improvement

Given that the java GUI give us the ability to add obstacles in the grid, an interesting improvement would be to implement search algorithms in order to find the best path while avoiding collision.

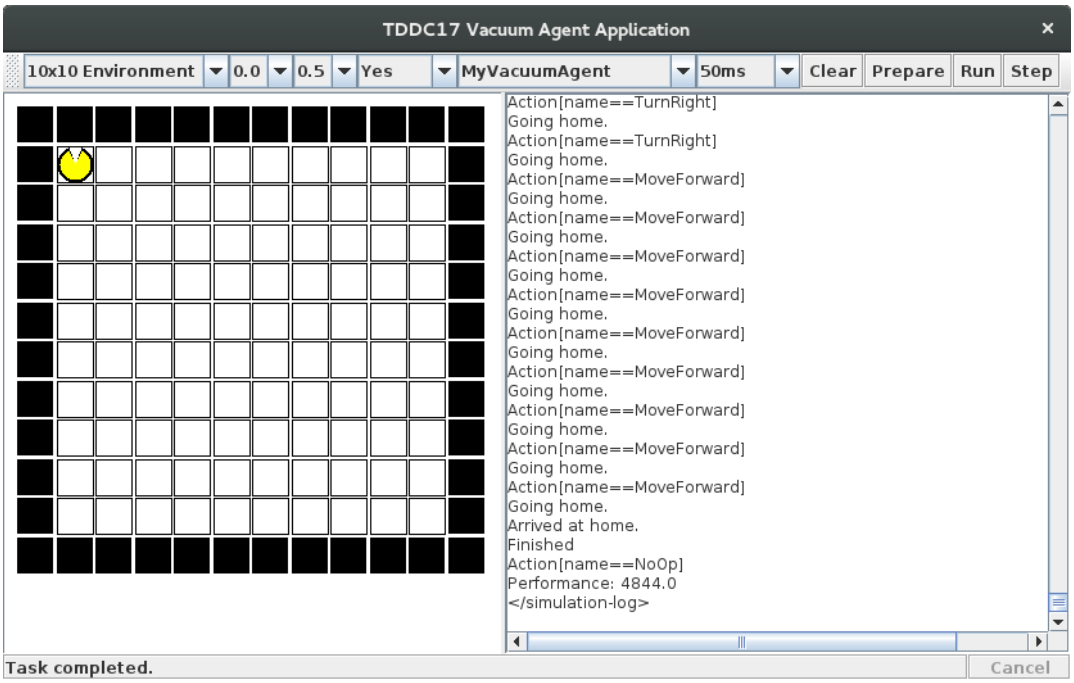

FIGURE 1 – Screenshot