Vacuum Cleaner Agent

Wadood Alam, Matthew Pacey, Joe Nguyen

April 13, 2022

Abstract

This paper explores three different types of vacuum cleaner agents: a simple reflex agent, randomized reflex agent, and a deterministic model-based agent with 3 bits of memory. The agents have minimal sensors and can only do basic atomic operations. Their goal is two clean two different environments as efficiently as possible. The results show 1) the memory significantly improves the deterministic agent by allowing it to clean more cells, and 2) the random agent can match the performance of the deterministic agent with memory but at the cost of much larger number of actions taken.

1 Introduction

The vacuum agents each have the same precepts and simple list of actions they can perform. The percepts that each agent has are:

1. Whether a wall is directly in front of the agent
2. If there is dirt in the cell the agent is in
3. If the agent is in the home/starting cell

Each of the three agents will have five actions to choose from:

1. Go forward
2. Turn right 90 degrees
3. Turn left 90 degrees
4. Suck dirt
5. Turn off

The agents will consist of if-then rules than fires a single action. No combining of the actions is allowed in any of the agents (each run cycle of the vacuum must execute one and only one of the five actions). Our memory based agent uses its 3 bits of memory to maintain a state but still only executes a single action at each run cycle.

The vacuum agents will operate in two different environments. Both environments are a 10x10 grid of cells that all have dirt in them. The second environment has the addition of four walls that divide the rooms from each other, leaving one cell open for the agent to pass from one room to another.

2 Description of Agents

In our project, we will be working with 3 types of agent, each of which, has different mechanism to the other. The Agents' names and details are as follows:

2.1 Reflex Agent

The possible actions corresponds to the following sensors:

Dirt action Pick up the dirt if there is dirt in that cell.

Wall sensor

1. if wall sensor = 0, the reasonable choice is going forward to the next cell because it is clear there. [1]
2. if wall sensor = 1, since we can not go forward if there is a wall ahead of the robot, the only options are either turn left or right. Because the original place of the robot is left-most corner oriented upwards and [1], we must turn right when we first hit the wall.

In other words, the logic for the agent is as follows:

- IF Floor = Dirty THEN Suck-Dirt
- IF Floor = Clean THEN
  - IF Wall sensor = 1 THEN Turn-Right
  - IF Wall sensor = 0 THEN Go-Forward

2.2 Randomized Reflex Agent

Randomized Reflex agent makes use of biases and the relevant sensors to clean the environment. The logic for the agent is as follows:

- IF Floor = Dirty THEN Suck-Dirt
  If the floor is dirty then suck the dirt.
- IF Floor = Clean THEN Move-Forward 0.85 Turn-Right 0.10 Turn-Left 0.05
  If the floor is clean, then agent will chose to Move-Forward 85% of the time, chose to Turn-Right 90 Degrees 10% of the time, and chose to Turn-Left 90 Degrees 5% of the time.
- IF Wall Sensor = 1 THEN Turn-Left 0.15 Turn-Right 0.85
  If the wall is in front then agent will chose to Turn-Right 90 Degrees 85% of the time and chose to Turn-Left 90 Degrees 15% of the time.

2.3 Deterministic Model-Based Reflex Agent

The deterministic model-based agent with memory will use the 3 bits of available memory to cycle through 8 states (available using three bits). A diagram of the states is shown below. The state numbers represent how the 3 bits of memory are used at each 'runAction' iteration. It is not shown for readability sake, but before the state is checked the cell is first checked for dirt. If there is dirt in the cell, it is cleaned. Once the cell is clean the state diagram takes effect. Some states such as State 0 contain precept conditionals. In the case of State 0, if there is no wall in front of the agent it will go forward one cell; if there is a wall ahead, the agent will turn left and proceed to State 1. The goal of State 1 is for the agent to move up vertically one cell in the environment, so it will always go forward one cell and proceed to State 2. State 2 turns the agent left again, so it has now turned 180 degrees from it's starting orientation but is now one row above the starting row. This is assuming the row above is not blocked. If the agent is blocked, it will not move up but will still reverse from the original direction. Note that the descriptions below consist of a single action each loop. The states are transitioned by updating the 3 bits of memory before the vacuum agent performs the action (the action is more interesting though so it is listed first). The goal of each state is thus:

- State 0: go forward (right in initial environment) until a wall is hit then turn left and transition to State 1
- State 1: Go forward then transition to State 2
- State 2: Turn left then transition to State 3
- State 3: Go forward (left in environment) until a wall is hit then turn right and transition to State 4
- State 4: Go forward (do nothing if there is a wall in front) and proceed to State 5
- State 5: Turn right and proceed to State 6
- State 6: If a wall is in front, turn left; if a wall is not in front, turn right and proceed to State 7
- State 7: Turn left and return to State 0

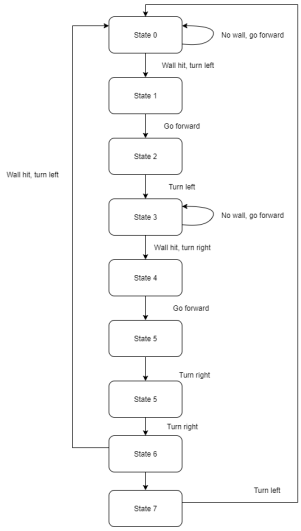


Figure 1: State Diagram of Memory Agent

3 Experimental Setup

- A 10x10 Grid. The boundaries of this grid are considered to be "walls". Total Number of Cells = 100

x x x x x x x x x x
x x x x x x x x x x
x x x x x x x x x x
x x x x x x x x x x
x x x x x x x x x x
x x x x x x x x x x
x x x x x x x x x x
x x x x x x x x x x
x x x x x x x x x x
^ x x x x x x x x x

- A 10x10 Grid divided into 4 rooms from the center with an entrance(size of 1 cell) to the adjacent room in the middle of first and last column respectively. The boundaries of the grid and the division of room in the center are considered to be "walls". Total Number of Cells = 85

x x x x x x x x x x
x x x x x W x x x x
x x x x x W x x x x
x x x x x W x x x x
x x x x x W x x x x
x W W W W W W W W x
x x x x x W x x x x
x x x x x W x x x x
x x x x x W x x x x
^ x x x x x x x x x

where W represents a wall, x is the dirty cell, and ^ is the initial position of the agent.

Each of the 3 agents will act in each of the 2 environments. The agent will always start at the bottom leftmost corner oriented upwards in both environments. The agents will follow the rules mentioned above in the paper. To conduct this experiment, we will be using Python programming language to code this entire phenomena and obtain the results. The agents are evaluated by the total number of clean cells as a function of the number of actions taken.

1. **Reflex Agent:** The agent will tried in the both the environments, once on each environment, and the data for each of the environments will be recorded.(Avg number of Actions and Avg number of Cells Cleaned)

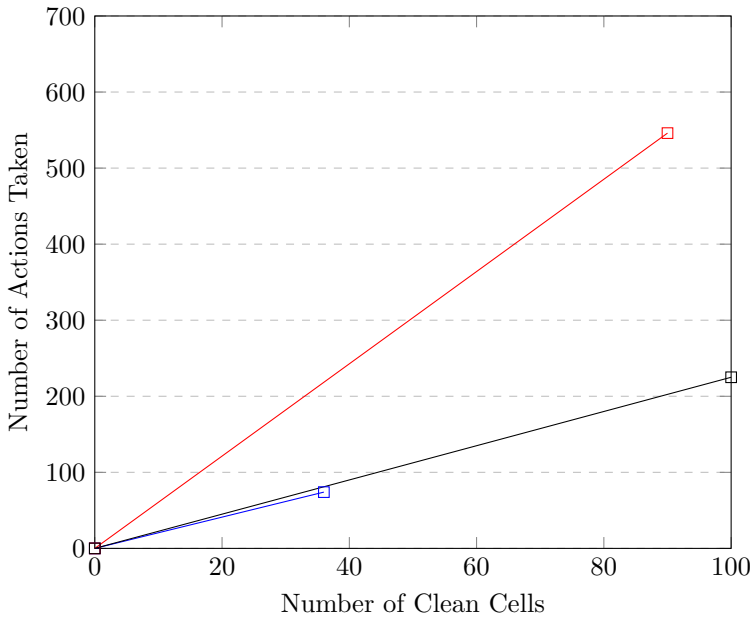
2. **Randomized Reflex Agent:** The agent will tried in the both the environments and, 50 times in each environment, and will record the number of actions to clean 90% of the room for each trial in each environment. Then the average will be taken of those 50 trails and recorded(2 averages, one for each environment).

3. **Deterministic Model-Based Reflex Agent:** The agent will tried in the both the environments, once on each environment, and the data for each of the environments will be recorded.(Avg number of Actions and Avg number of Cells Cleaned).

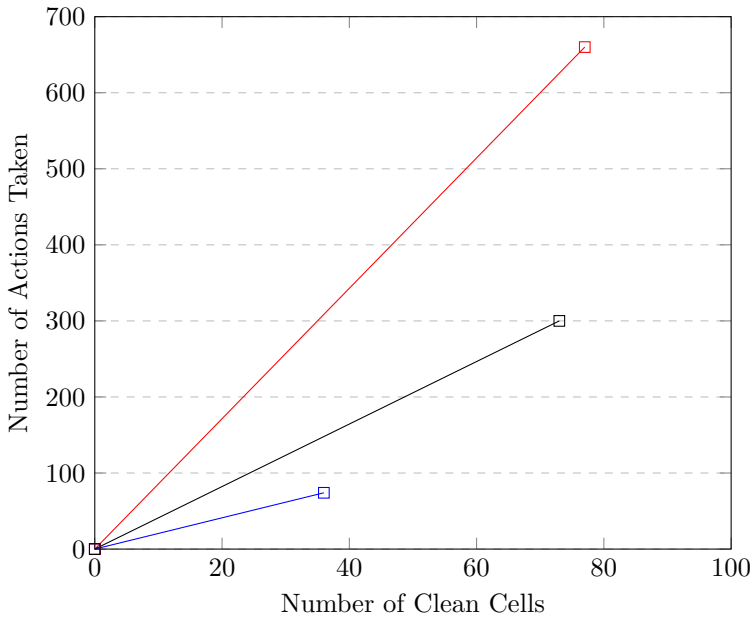
After the data is recorded, we will plot 3 lines(one for each agent) respectively on 2 graphs for each of the environments respectively.

4 Results

The results for three agents is presented in Table 2 for 10x10 grid case and in Table 3 for 10x10 Grid with 4 rooms. The following are graphs the for each of the 2 environments:



(a) Graph for 10x10 Grid



(b) Graph for 10x10 Grid Divided into 4 Rooms

Figure 2: Blue = Reflex Agent, Red = Randomized Reflex Agent, Black = Deterministic Model-Based Reflex Agent

Trial Number	Number of Actions for 10x10 Grid	Number of Actions for 4 rooms
1	471	772
2	368	643
3	534	586
4	542	447
5	638	801
6	749	799
7	721	505
8	544	531
9	398	454
10	414	416
11	505	450
12	430	518
13	467	685
14	660	894
15	394	1060
16	709	473
17	501	695
18	449	695
19	389	925
20	464	837
21	488	425
22	552	898
23	651	758
24	539	1018
25	405	765
26	985	553
27	344	568
28	692	638
29	624	679
30	562	1114
31	575	510
32	548	503
33	599	465
34	625	788
35	524	616
36	943	680
37	614	525
38	413	1195
39	482	543
40	520	479
41	480	867
42	603	553
43	581	442
44	618	684
45	478	692
46	556	447
47	508	532
48	573	469
49	519	835
50	527	499

Table 1: Randomized Reflex Agent's Performance to clean 90% of the environment(s)

Agents	Trials	Average Number of Actions	Average Number of Cells Cleaned
Reflex	1	74	36
Randomized Reflex	50	546	90
Deterministic Model-Based Reflex	1	225	100

Table 2: Agents' Performance for 10x10 Grid

Agents	Trials	Average Number of Actions	Average Number of Cells Cleaned
Reflex	1	74	36
Randomized Reflex	50	660	77
Deterministic Model-Based Reflex	1	225	73

Table 3: Agents' Performance for 10x10 Grid Divided into 4 Rooms

5 Discussion

5.1 Reflex agent

The best possible performance achievable by the simple reflex agent in the two environments is the agent only covers the outer edges of the grid. Lacking of memory prevents the agent from achieving the goal of cleaning the room perfectly in each case.

Specifically, because 1) we only have two perception states: either hitting the wall or not (we do not have memory to represent other states) and 2) we can only one action at one time, the reasonable action for each perception state is as follows:

- IF Wall sensor = 1, we can only do either turn left or turn right all the times. This basically turns the robot to do the circle loop around the outer edges of the grid in the normal grid and in the four-room grid as well (we define doors are in the outer edges of the whole grid).
- IF Wall sensor = 0, the reasonable thing to do is going forward.

Specifically, we have following two final results:

- A 10x10 grid:

```
o o o o o o o o o o
o x x x x x x x x o
o x x x x x x x x o
o x x x x x x x x o
o x x x x x x x x v
o x x x x x x x x o
o x x x x x x x x o
o x x x x x x x x o
o x x x x x x x x o
o o o o o o o o o o
```

- A 10x10 grid with walls:

```
o o o o o o o o o o
o x x x x W x x x o
o x x x x W x x x o
o x x x x W x x x o
o x x x x W x x x v
o W W W W W W W W o
o x x x x W x x x o
o x x x x W x x x o
o x x x x W x x x o
o o o o o o o o o o
```

where W represents a wall, X is the dirty cell, and o is the cleaned cell.

5.2 Random reflex agent

The random agent on average takes more actions to clean a single cell:

- For 10x10 Grid: 6.06 actions per clean cell
- For 10x10 Grid Divided into 4: 8.67 actions per clean cell

The random agent is successful in cleaning 90% of both the environments. The average number of actions of the best 45 trials are 524 and 632 for the 10x10 grid and 4-room respectively. The cost-benefit of this agent can be best described as assurance of cleaning most of the environment(90%) but more actions are required.

5.3 Deterministic Model-based Reflex Agent

The memory significant improves the deterministic agent in term of the number of cleaned cells.

The memory-based deterministic agent was able to fully clean the single 10x10 room. The agent was not able to perfectly clean the four room environment, it left 12 dirty cells after 225 moves. Even if allowed to run for 1,000 or more actions it could not clean the remaining cells. If it was allowed to have some randomness, it likely could clean the remaining cells. If the memory agent had enough memory to map the entire room (more than 3 bits needed), it could clean 100% by tracking exactly where it had been, where the walls are, and which cells have not be visited yet.

The final result is presented as follows:

o o o o o o o o o o
o o o o o o o o o o
o o o o o o o o o o
o o o o o o o o o o
o o o o o o o o o o
o o o o o o o o o o
o o o o o o o o o o
o o o o o o o o o o
o o o o o o o o o o
o o o o o o o o o o

- 10x10 Grid

o o o o o o o o o o
o o o o o W o o o o
o o o o o W o o o o
o o o o o W o o o o
o o o o o W o o o o
o W W W W W W W W o
o x x x x W o o o o
o x x x x W o o o o
o x x x x W o o o o
o o o o o o o o o o

- 10x10 Grid Divided into 4 rooms

6 Conclusion

We address the following questions:

What are the trade-offs between the random and deterministic agents?

1. Due to the stochastic nature of the random agent, the random agent takes more actions to clean the same number of cells, but achieving a better number of cleaned cells than the deterministic one with no memory.
2. The random agent does not produce a consistent result as it is the random process. Meanwhile, the deterministic result is always guaranteed.
3. The random agent and the deterministic agent with memory produce the roughly same result (the number of cleaned cells) but the random agent requires more steps to complete the same goal.

How would you design better agents for more complex environments, say with polygonal obstacles? We can do the following improvements:

1. Get more memory to the agent: in that way, it can remember more history (action history and perception history), have a better understanding of the current perception state.

2. Increase the action space: the agent will be more flexible in moving and therefore resulting in less perception space needed to handle. For example, we can add "turn around 180 degree" from the current angle of orientation. Another example is allowing more degrees of turning (not limited to 90 degrees). This enables the agent to move in a more complex map with polygonal obstacles.

3. Allow the agent to take multiple actions at the same time. For example, if the agent hits the wall, we turn left and go forward. Like the previous solution, this requires less perception space and less history to remember.

What did you learn from this experiment? Were you surprised by anything?

1. One key learning from this was that it is hard to program an effective agent with limited memory. If the agent had the memory to maintain an internal map of where it had explored and cleaned, it would be easier for it to direct itself to new cells to get the clean coverage to 100%.

2. The random agent can cover more cleaned cells than the deterministic one with no memory. In the case of no memory, randomness allows you to have more freedom and more flexibility in the action space. Even this process is stochastic, we still have more actions for the agent to move around, rather than just either "moving left" or "moving right" when the agent hits the wall. Therefore, it can cover more cells, which ends up clearing more cells.