

CZ3005 Lab 2 - Wumpus World

Group0

SSP2

Group Members: Ma Xinyi, Zhang Chi, Zhang Meng'ao

1. Tasks

We designed 3 tests to test the correctness of the Agent.

Test 1: Preset action list; drive the Agent into the wall, to shoot the Wumpus, to pick up the coin, and into a Confundus Portal.

Test 2: Preset action list; drive the Agent into the wall, and into the Wumpus.

Test 3: No preset action list; drive the Agent to explore the world by repeatedly calling explore/1.

The map figure “Test X.Y” stands for the Yth action’s result relative map in Test X. “Test X_Y.Z” stands for the Zth action’s result relative map of the Yth action sequence in Test X. For example, “Test 1.0” means the initial relative map in Test 1. “Test 3_12.2” means the 2nd action’s result relative map of the action sequence generated by 12th explore/1 call in Test 3. (X = 1, 2, 3; Y = 0, 1, 2...; Z = 1, 2, 3...)

1.1 Correctness 1 - Agent’s localisation and mapping abilities

Approach:

1) Agent approach:

Use dynamic fact to implement current(X,Y,D), which is initialised with X = Y = 0, D = ‘rnorth’ and will be reset each time reposition/1 or reborn/0 is called. Each time the driver issues an action that changes location, the old current/3 will be retracted and new current/3 will be asserted accordingly.

2) Driver approach:

To test the localisation and mapping abilities of the Agent, the Agent was asked to perform a series of preset actions by the Driver. The preset actions involve ‘turnleft’, ‘turnright’, and ‘moveforward’. If the Agent preserves correct orientation and position after performing the preset actions, its localisation and mapping abilities are verified.

Output:

Take the partial printout of Test 1 as an example.

The following are the initial relative map and relative maps after the Agent correctly executed ‘moveforward’, ‘turnright’, and ‘turnleft’ right after initialization:

<pre>... .. ? s ?</pre>	<pre>... ### ... ? ### ? ... ### ...</pre>	<pre>... ### ... ? ### ? ... ### ...</pre>	<pre>... ### ... ? ### ? ... ### ...</pre>
<pre>... %.. ... s -^- s</pre>	<pre>... .. s -^- sB. ...</pre>	<pre>... .. s ->- s</pre>	<pre>... .. s -^- s</pre>
<pre>... .. ? s ?</pre>	<pre>... .. ? s ?</pre>	<pre>... .. ? s ?</pre>	<pre>... .. ? s ?</pre>
Test 1.0 Initial Relative Map	Test 1.1 “moveforward”	Test 1.2 “turnright”	Test 1.3 “turnleft”

As illustrated by the above printouts, the Agent performs correctly in terms of localisation and mapping.

1.2 Correctness 2 - Agent's sensory inference

Approach:

1) Agent approach:

Each time 'moveforward'/'pickup'/'shoot' action is issued, the Agent will perform a reasoning process to absorb the sensory data. For example, if it does not feel stench/tingle, it will retract all the fact wumpus(X,Y)/ confundus(X,Y), where (X,Y) is the adjacent cells of the Agent's current location. When it feels glitter, it will assert glitter(X,Y), where (X,Y) is the Agent's current location, and retract it after 'pickup'. After it smells stench, it will assert wumpus(X,Y) where (X,Y) satisfies the following conditions: 1. adjacent to the current location; 2. adjacent to all the visited stench cells; 3. all the visited adjacent cells have a stench. At the same time, all wumpus(X',Y') is retracted, where (X',Y') is not adjacent to the current location. Besides, whenever there is only one (X,Y) that makes wumpus(X,Y) true, we can assure that this cell will not contain a Confundus Portal, thus, confundus(X,Y) is retracted.

2) Driver approach:

To test the sensory inference of the Agent, the Agent was given lists of percepts while it was driven on the preset path.

Output:

Take the partial printout of Test 1 as an example.

Test 1.0 and Test 1.1 depict how the Agent reacted when precepting 'Bump', and Test 1.5 and Test 1.6 show the Agent perceived 'Glitter' and the 'Glitter' disappeared after the Agent performed 'pickup'.

<pre> ? s ? </pre>	<pre> ... ### ? ### ? ... ### </pre>	<pre> ... ### ... ? ? ### s ? ... ### ... </pre>	<pre> ... ### ... ? ? ### s ? ... ### ... </pre>
<pre> ... %.. s -A- s </pre>	<pre> s -A- sB. </pre>	<pre> ... s S ->- s ... *.. </pre>	<pre> ... s S ->- s </pre>
<pre> ? s ? </pre>	<pre> ? s ? </pre>	<pre> ... s s ? </pre>	<pre> ... s s ? </pre>
Test 1.0	Test 1.1	Test 1.5	Test 1.6

The Agent walked into a cell with both Wumpus and Confundus Portal around. Then, it shot an arrow and heard a 'Scream', which confirmed that the Agent was facing the Wumpus and the other side should be the Confundus Portal.

<pre> ? ? ? ? ? ? ? ? </pre>	<pre> ? ? ? ? ? ? ? ? </pre>
<pre> ? ? ? ? ? ? ? ? </pre>	<pre> ? ? ? ? ? ? ? ? </pre>
<pre> ... ### ? ? ? ? ### s -0- ? ? ... ### </pre>	<pre> ... ### ? ? ? ? ### s -0- ? ? ... ### </pre>
<pre> ... T ? ? ? s s S -0- ? </pre>	<pre> ... T ? ? ? s s S -0- ? </pre>
<pre> ... s s S s ? ? ? ? ? s s S s ? </pre>	<pre> ... s s S s ? ? ? ? ? s s S s ? </pre>
<pre> ... =T ? ? ? ? ? s S ->- -U- </pre>	<pre> ... T ? ? ? ? ? s S ->- s ... @ </pre>
<pre> ... -U- ? ? ? ? ? ? s -U- ? </pre>	<pre> ... -0- ? ? ? ? ? ? s -0- ? </pre>
Test 1.12	Test 1.13

The above printouts demonstrate that the Agent can receive sensory data and infer based on them correctly.

1.3 Correctness 3 - Agent's memory management in response to stepping through a Confundus Portal

Approach:

1) Agent approach:

Once the Agent feels 'Confounded', it will call reposition/1. In reposition/1, all dynamic facts are retracted except the fact of hasarrow/0. Besides, current/3 is reset. Then the Agent will do an initial reasoning according to the sensory data.

2) Driver approach:

Drive the Agent into a Confundus Portal and check its memory.

Output:

Take the partial printout of Test 1 as an example.

Before (Test 1.14) and after (Test 1.15) the Agent stepped into a Confundus Portal:

<pre> ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ### s -0- ? ? ### ? ? ? s S S S -0- ? ? ? ? ? s s S s ? ? ? ? ? ? s S -v- s ? ? ? ? ? ? s -0- ? </pre>	<pre> ? -0- ? % . T ... -0- -^ -0- ? -0- ? </pre>
---	--

Test 1.14

Test 1.15

As shown, the Agent correctly cleared its memory according to the requirements after stepping into a Confundus Portal.

1.4 Correctness 4 - Agent's exploration capabilities

Approach:

1) Agent approach:

First, the Agent will check if there is any safe accessible unvisited cell according to its knowledge base. If yes, it will conduct a Depth First Search to find a path. We used a list as a stack to record the current explored path, and maintained a flag set to record which cells have been visited. Each time it will expand an adjacent cell and check whether the following conditions are satisfied: 1. safe; 2. not a wall; 3. not visited. This process is done recursively until the end condition, namely safe, accessible, unvisited, is met. Then the path is generated according to the nodes in the stack. Every time explore/1 is called, it will pick up the coin if there is one in the initial position, this ensures no coin is missed. After safely accessible unvisited cells are exhausted, it will generate a path to the relative origin.

Repeatedly call the function, `explore/1`, of the Agent and drive the Agent according to the return of `explore/1`. Terminate the test if the Agent has explored all the safe unvisited cells and returns to the origin. Set a maximum number of looping as 42 in case that the Agent is stuck in at some point of time. 42 is chosen because there are $7 * 6 = 42$ cells in total, which allows the Agent to explore all the cells and perform some actions while protecting the system from infinite looping.

The relative maps after each action of the above action list:

1.5 Correctness 5 - Agent’s end-game reset in a manner similar to that of Confundus Portal reset

Before (Test 2 1.5) and after (Test 2 2.0) the Agent meet the Wumpus:

```

... ..
? -W- -0- s ? ? ?
... ..

### .=. ..T ... ..
### -^- S S s ? ?
### ... ..

... ..
? -W- -0- s ? ? ?
... ..

```

Test 2_1.5

```

... ..
? s ?
... ..

... %.. ...
s -^- s
... ..




... ..
? s ?
... ..

```

Test 2_2.0

The Agent correctly cleared its memory after meeting the Wumpus, which is also the end of a game.

2. Contribution

Name	Individual Contribution to Lab 2	Contribution (%)	Signature
Ma Xinyi	Analysis of the problem and discuss the solutions. Contribute in writing the source code and final report.	33.3%	
Zhang Chi	Analysis of the problem and discuss the solutions. Contribute in writing the source code and final report.	33.3%	
Zhang Mengao	Analysis of the problem and discuss the solutions. Contribute in writing the source code and final report.	33.3%	

Friend agent: "Elseif" - SSP6

Friend driver: "TeamName" - SSP2

3. Conclusion

From this assignment, we have obtained a broader view of the programming methodology by learning the Knowledge Representation Language and Knowledge Based System. This assignment provides us with another viewpoint on solving logic-related complex problems. We also improved our Python skills and we are familiar with the library, PySwip, and the use of it. We practised Object Oriented Programming and implemented some design principles, like Encapsulation, that we learned from other modules making the code more reusable, more extensible, and more maintainable.

4. References

PySwip GitHub: <https://github.com/yuce/pyswip>

SWI-Prolog syntax: <https://www.swi-prolog.org/>

Prolog tutorial: <https://www.amzi.com/AdventureInProlog/index.php>