EXPERIMENT 39

Prolog program for GENERATING computational agent

|  |
| --- |
| % Define the agent's actions |
|  | action(move\_up). |
|  | action(move\_down). |
|  | action(move\_left). |
|  | action(move\_right). |
|  |  |
|  | % Define the agent's rules for choosing actions |
|  | choose\_action(Agent, Action) :- |
|  | current\_position(Agent, X, Y), |
|  | (goal\_reached(X, Y) -> Action = stop; |
|  | (obstacle\_ahead(Agent) -> choose\_alternate\_action(Agent, Action); |
|  | Action = move\_towards\_goal(Agent))). |
|  |  |
|  | % Define the alternate actions for when there is an obstacle |
|  | choose\_alternate\_action(Agent, Action) :- |
|  | (clear\_path(Agent, move\_left) -> Action = move\_left; |
|  | (clear\_path(Agent, move\_right) -> Action = move\_right; |
|  | (clear\_path(Agent, move\_up) -> Action = move\_up; |
|  | (clear\_path(Agent, move\_down) -> Action = move\_down; |
|  | Action = stop)))). |
|  |  |
|  | % Define the action to move towards the goal |
|  | move\_towards\_goal(Agent) :- |
|  | current\_position(Agent, X, Y), |
|  | goal(GX, GY), |
|  | (GX > X -> move\_right; |
|  | (GX < X -> move\_left; |
|  | (GY > Y -> move\_up; |
|  | move\_down)))). |