## **Rational Agents**

Thinking humanly

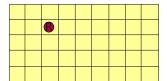
Acting humanly

Thinking rationally

Acting rationally

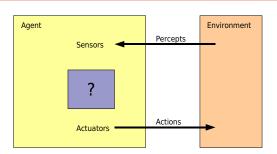
## **Acting Rationally**

- Rational behavior: doing the right thing
- The right thing: that which is expected to maximize goal achievement, given the available information

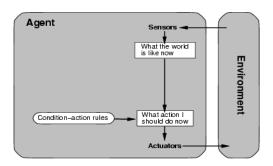


If the goal is to find a wall and follow it, and the agent can only perceive the neighboring cells, what kinds of behavior would be rational/irrational?

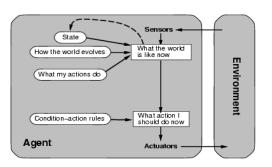
## What is an agent?



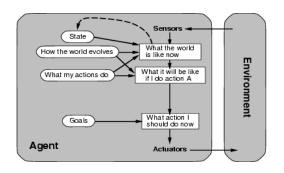
## Reflex Agent or Stimulus-Response Agent



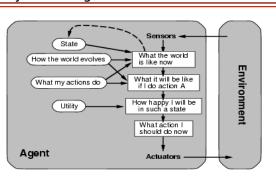
## Model-based Reflex Agent



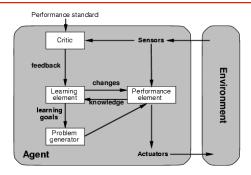
## Goal-based Agent



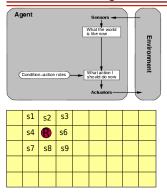
## **Utility-based Agent**



## Learning Agent



## A Wall-following Reflex-Agent



## The Agent Program

if s2 and not s6:
 move(east)
elif s6 and not s8:
 move(south)
elif s8 and not s4:
 move(west)
elif s4 and not s2:
 move(north)
else:
 move(north)

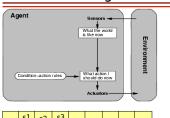
### Wall-following with Obstacles (Agent Program)

northwall = s2 or s3 eastwall = s6 or s9 southwall = s8 or s7 westwall = s4 or s1 if northwall and not eastwall:
 move(east)
elif eastwall and not southwall:
 move(south)
elif southwall and not westwall:
 move(west)
elif westwall and not northwall:
 move(north)
else:
 move(north)

## Our Requirements for Intelligence

- make up new answers
- · learning / adapting
- reasoning / making decisions / analyzing
- perception (body language, intonation ...)
- planning
- emotions

## A Wall-following Reflex-Agent



#### percepts:

[s1,s2,s3,s4,s5,s6,s7,s8,s9]

si == 'w': there is a wall at position si

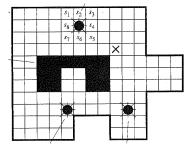
si == ' ': position si is free

s1	s2	s3			
s4	8	s6			
s7	s8	s9			

#### actions:

move(x) with x being north, east, south, west

# Wall-following with Obstacles



## In Python: the grid world

### In Python: the grid world

### In Python: sensing – first try

```
# x_pos: variable holding the agent's horizontal
position
# y_pos: variable holding the agent's vertical position

def sense():
    return [
        world[y-1][x-1], world[y-1][x], world[y-1][x+1],
        world[y][x-1], world[y][x], world[y][x+1],
        world[y+1][x-1], world[y+1][x], world[y+1][x+1]
        ]
```

## In Python: moving

## In Python: the agent function

```
def agent_function():
    percepts = sense()
    northwall = percepts[1]=='w' or percepts[2]=='w'
    eastwall = percepts[5]=='w' or percepts[8]=='w'
    [...]
    if northwall and not eastwall:
        move('e')
    elif eastwall and not southwall:
        move('s')
    [...]
    else:
        move('n')
```

## In Python: sensing

## In Python: sensing – second try

```
def get_cell(x,y):
    if x < 0 or y < 0 or x >= XSIZE or y >= YSIZE:
        return 'w'
    else:
        return world[y][x]

def sense():
    return [
        get_cell(x-1, y-1),get_cell(x, y-1),
        get_cell(x+1, y-1),
        get_cell(x+1, y),
        get_cell(x+1, y),
        get_cell(x+1, y),
        get_cell(x+1, y),
        get_cell(x+1, y+1),get_cell(x, y+1),
        get_cell(x+1, y+1)
```

### In Python: moving

### **Exercises**

- Modify the layout of the map (e.g. change the size of the grid, change the position, size, form and number of the obstacles) and see how the agent behaves.
- Once the agent finds a wall, it moves along it in a specific direction. For example, if it hits the outer wall, it follows that wall clockwise. Change the direction in which the agent goes when following walls.