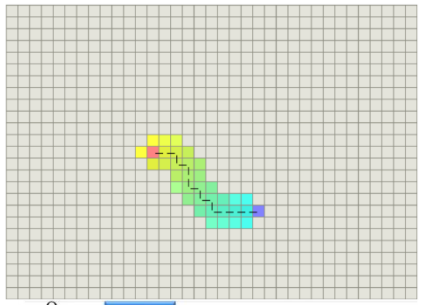
“Inaction breeds doubt and fear. Action breeds confidence and courage. If you want to conquer fear, do not sit home and think about it. Go out and get busy.” -- Dale Carnegie

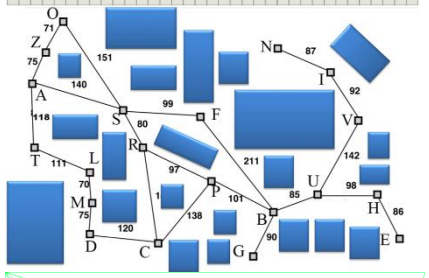
Announcements

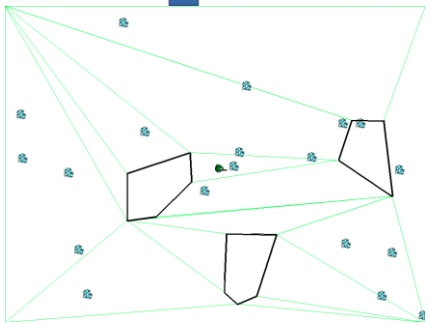
- AIIDE 2015 https://youtu.be/ZIAmoRsu3Z0?list=PLxGbBc3OuMgg7OuyLfvXQLR6HKcoglCfG&t=1713
- HW3. http://osi.gatech.edu/
- HW4 is posted, due this Sunday

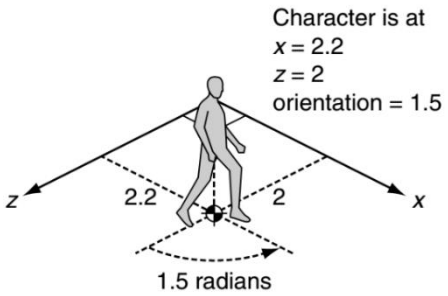
Review

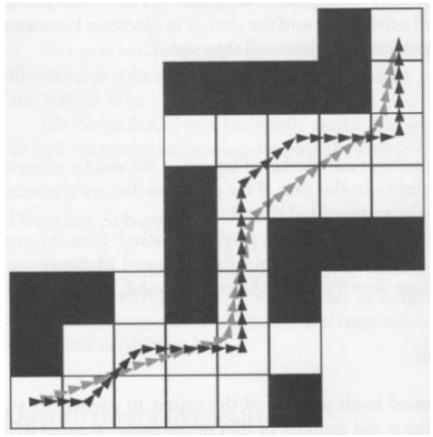
- The story thus far…

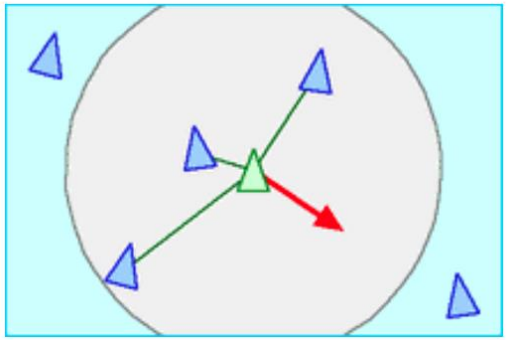












N-1: formations

1. What's the problem with independent shortest distance navigation with groups?
2. What's a simple solution, and its drawbacks? Other workarounds?
3. Is it always sensible to replan paths when problems occur?
4. What are 3 ways of implementing fixed formations?
5. Discuss some problems with entities of different sizes.

Decision Making – FSMs

2018-02-13

B Ch2, M 5.1,5.3

Decision Making

- Historically a “mathematical model of computation” – Wikipedia
- Classic AI:
  - making the optimal choice of action (given what is known or is knowable at the time) that maximizes the chance of achieving a goal or receiving a reward (or minimizes penalty/cost)
- Game AI:
  - choosing the right goal/behavior/animation to support the experience
- Decision-making must connect directly to animation so player can see the results of decision-making directly (explainable AI)
  - What animation do I play now?
  - Where should I move?

Most game agents do more than move

- Shoot
- Patrol
- Farm
- Sleep
- Hide
- Hunt
- …
- We will call these “states”

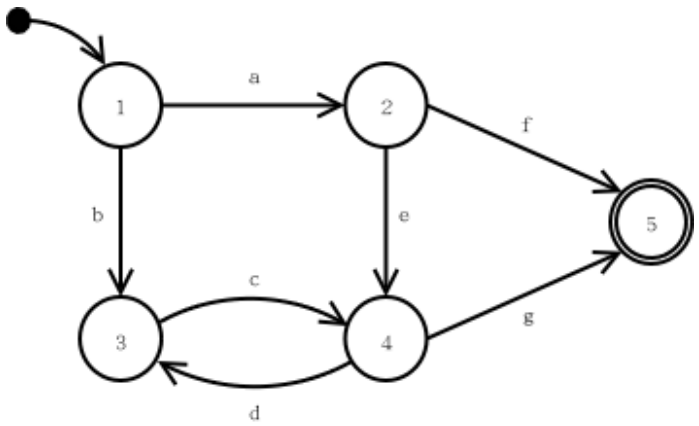
Most game agents go through more than one state

Game agents will have different sets of states, and different ways of moving through those states based on their purpose in the game:

- Soldier: Patrol -> Shoot -> Hunt
- Farmer/Villager: Farm -> Patrol -> Sleep

FSM theory

- A (model of a) device which has
  - a finite number of states (S)
  - an input vocabulary (I)
  - a transition function T(s,i) → s'
  - a start state ∈ I
  - zero or more final states ⊂ S



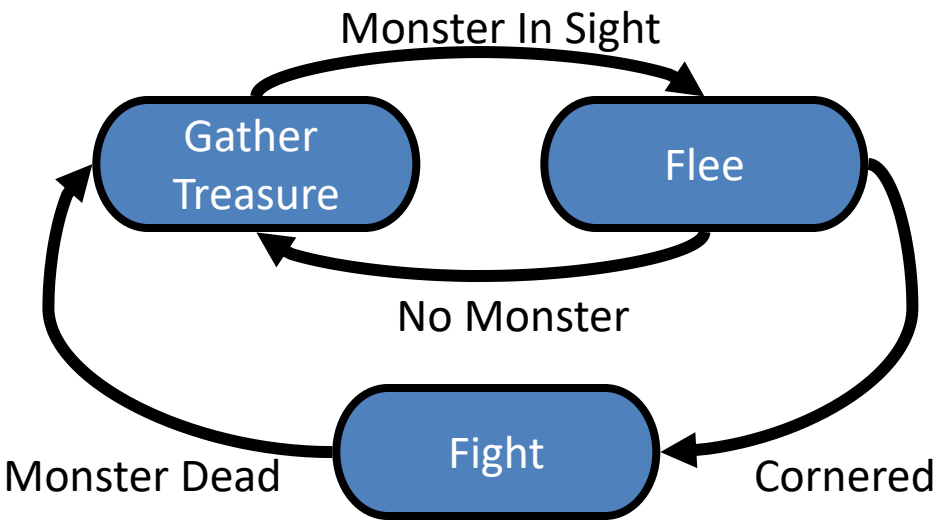
- Behavior
  - Can only be in one state at a given moment in time
  - Can make transitions from one state to another or to cause an output (action) to take place.

FSMs in Practice

- Each state represents some desired behavior
- Transition function often resides across states
  - Each state determines subsequent states
- Can poll the world, or respond to events (more on this later)
- Support actions that depend on state & triggering event (Mealy) as well as entry & exit actions associated with states (Moore)

FSM as GAI

- Character AI modeled as sequence of mental states
- World events (can) force a change in state
- Mental model easy to grasp (for all)



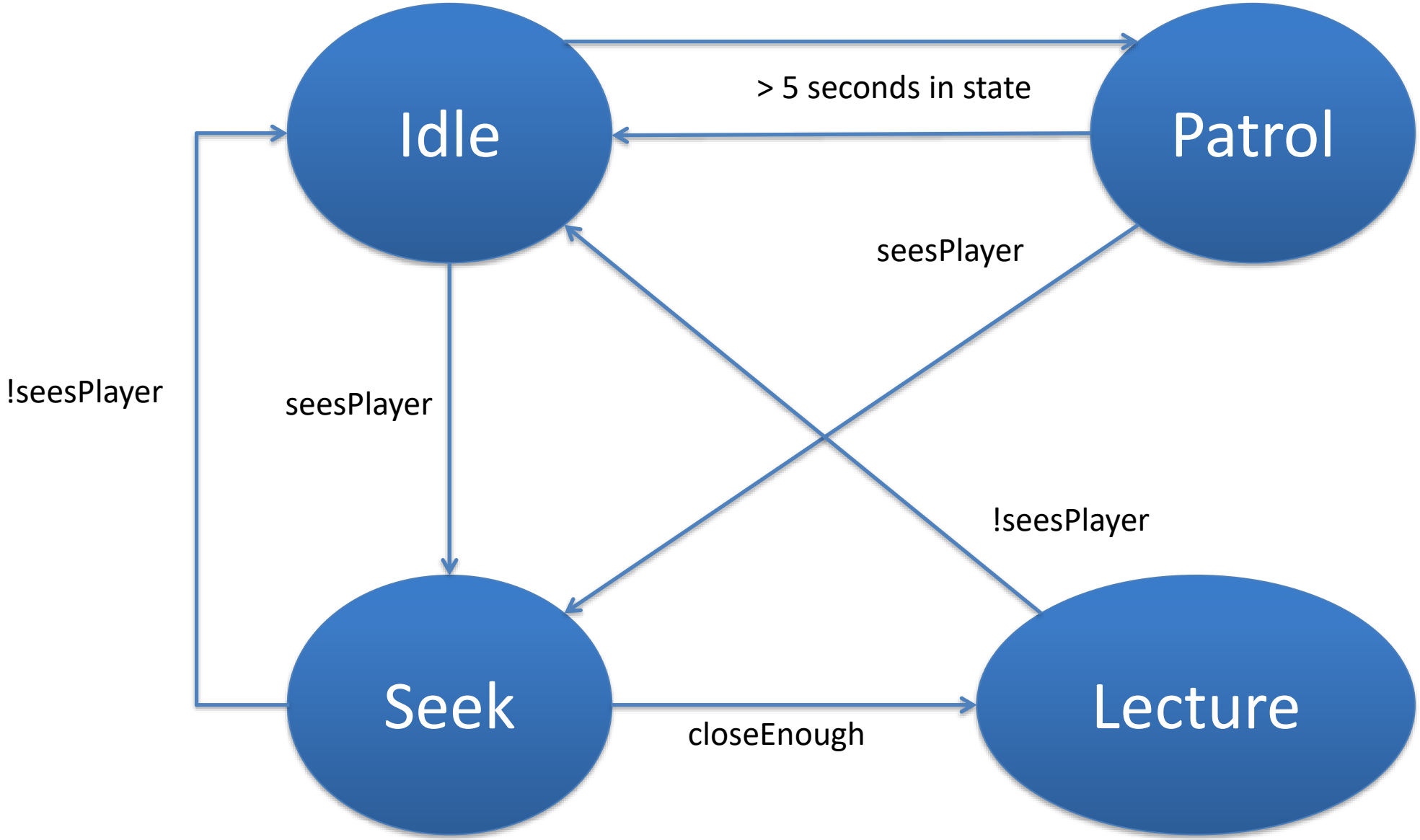
Jarret Raim

Question 1

Using the graph form, design/visualize an FSM for a security guard that patrols a school after hours. If it sees the player it needs to seek the player and give them a lecture till the player can slip away. Otherwise switch between Idle and Patrol.

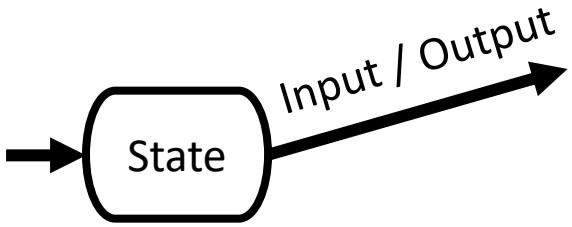
States = {Idle, Patrol, Seek, Lecture}

Feel free to make up variables.

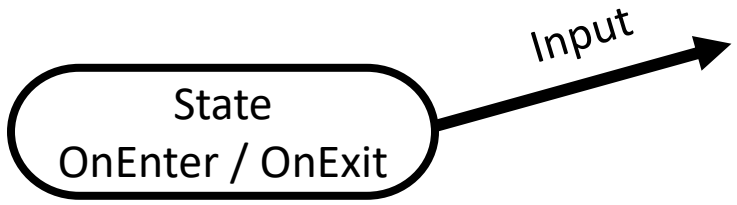

Idle
Patrol
Seek
Lecture
> 5 seconds in state
seesPlayer
seesPlayer
!seesPlayer
!seesPlayer
closeEnough

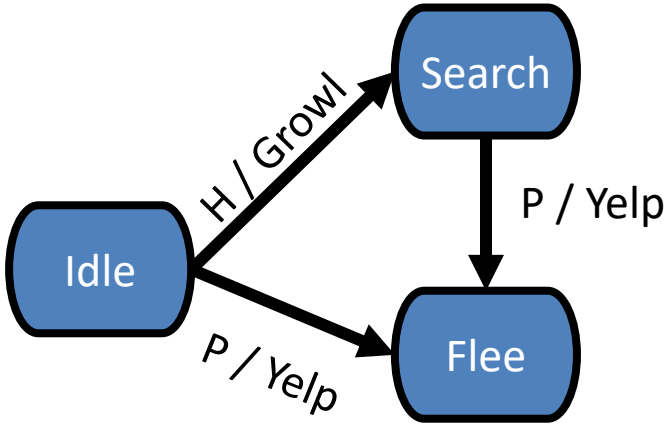
Mealy & Moore

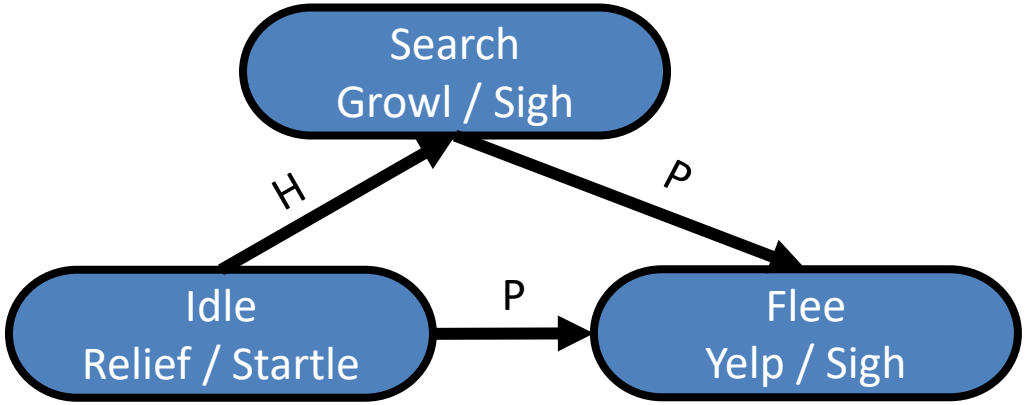
Mealy Output =
F(state, input)



Moore Output =
F(state)

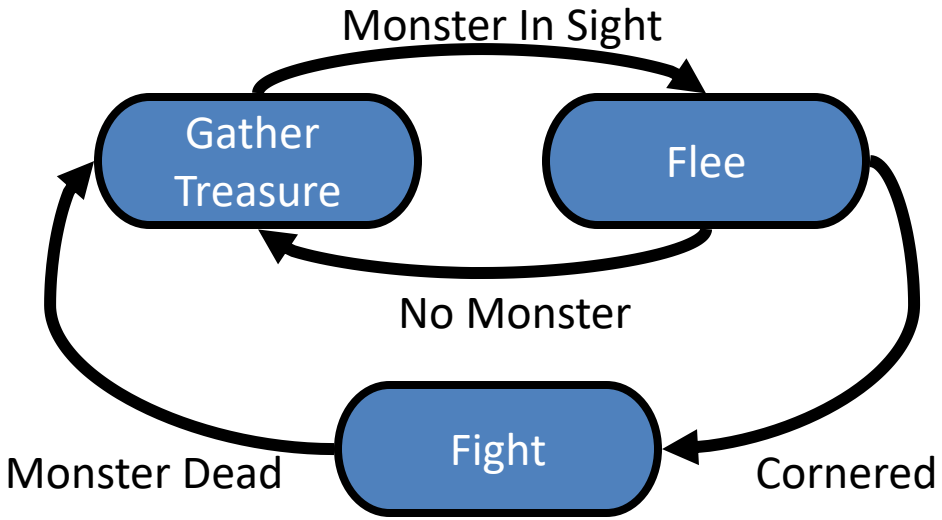






State Transition Table

Current State	Condition	State Transition
Gather Treasure	Monster	Flee
Flee	Cornered	Fight
Flee	No Monster	Gather Treasure
Fight	Monster Dead	Gather Treasure



Advantages

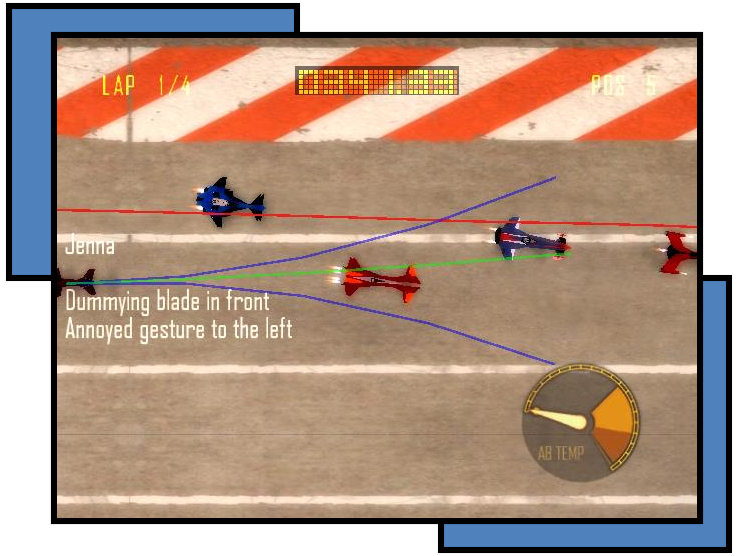
- Ubiquitous (not only in digital games)
- Quick and simple to code
- (can be) Easy* to debug
- Fast: Small computational overhead
- Intuitive
- Flexible
- Easy for designers without coding knowledge

Disadvantages

- When it fails, fails hard:
  - A transition from one state to another requires forethought (get stuck in a state or can't do the "correct" next action)
- Number of states can grow fast
  - Exponentially with number of events in world (multiple ways to react to same event given other variables)
- Number of transitions can grow even faster
- Doesn't work with sequences of actions/memory

Debugging FSM's

- Offline Debugging
  - Logging
  - Verbosity Levels
- Online Debugging
  - Graphical representation is modified based on AI state
  - Command line to modify AI behavior on the fly.

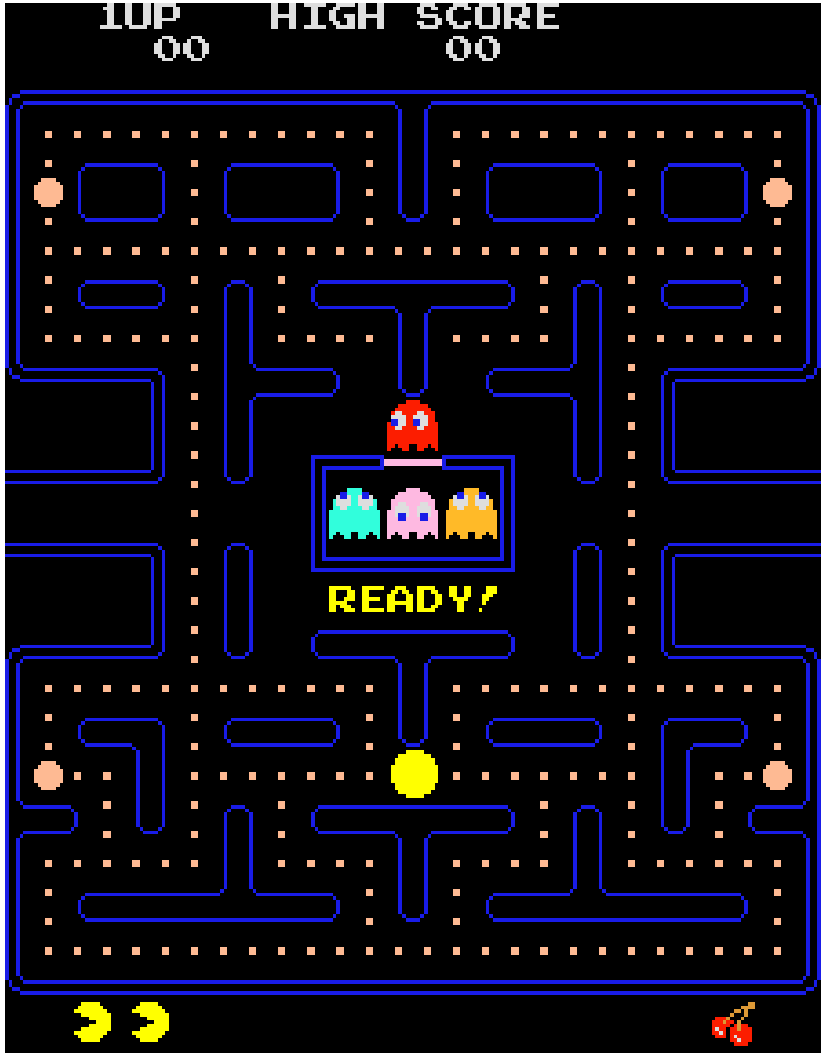


Where we see that they’re incredibly common, effective, and useful

FSM EXAMPLES

Activity

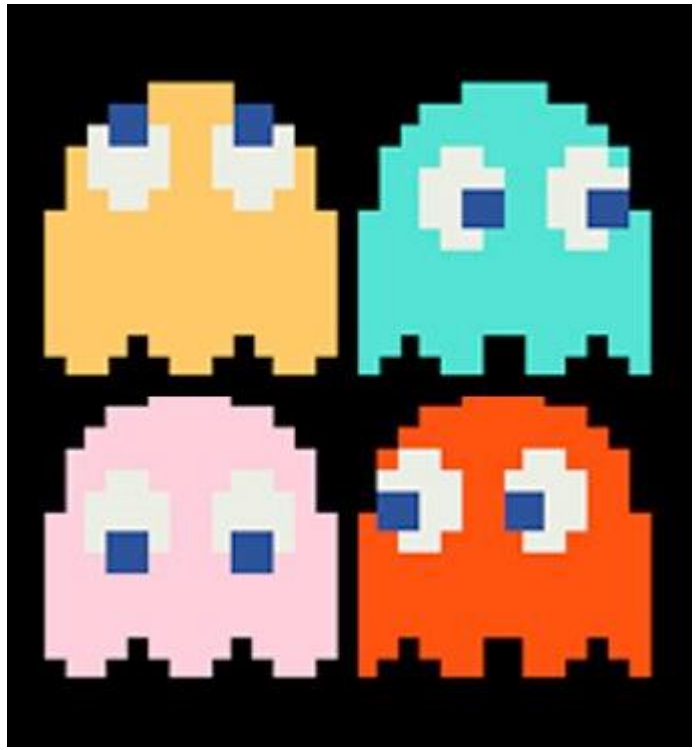
- States:
- Transitions:



See also https://www.gamasutra.com/view/feature/3938/the_pacman_dossier.php?print=1

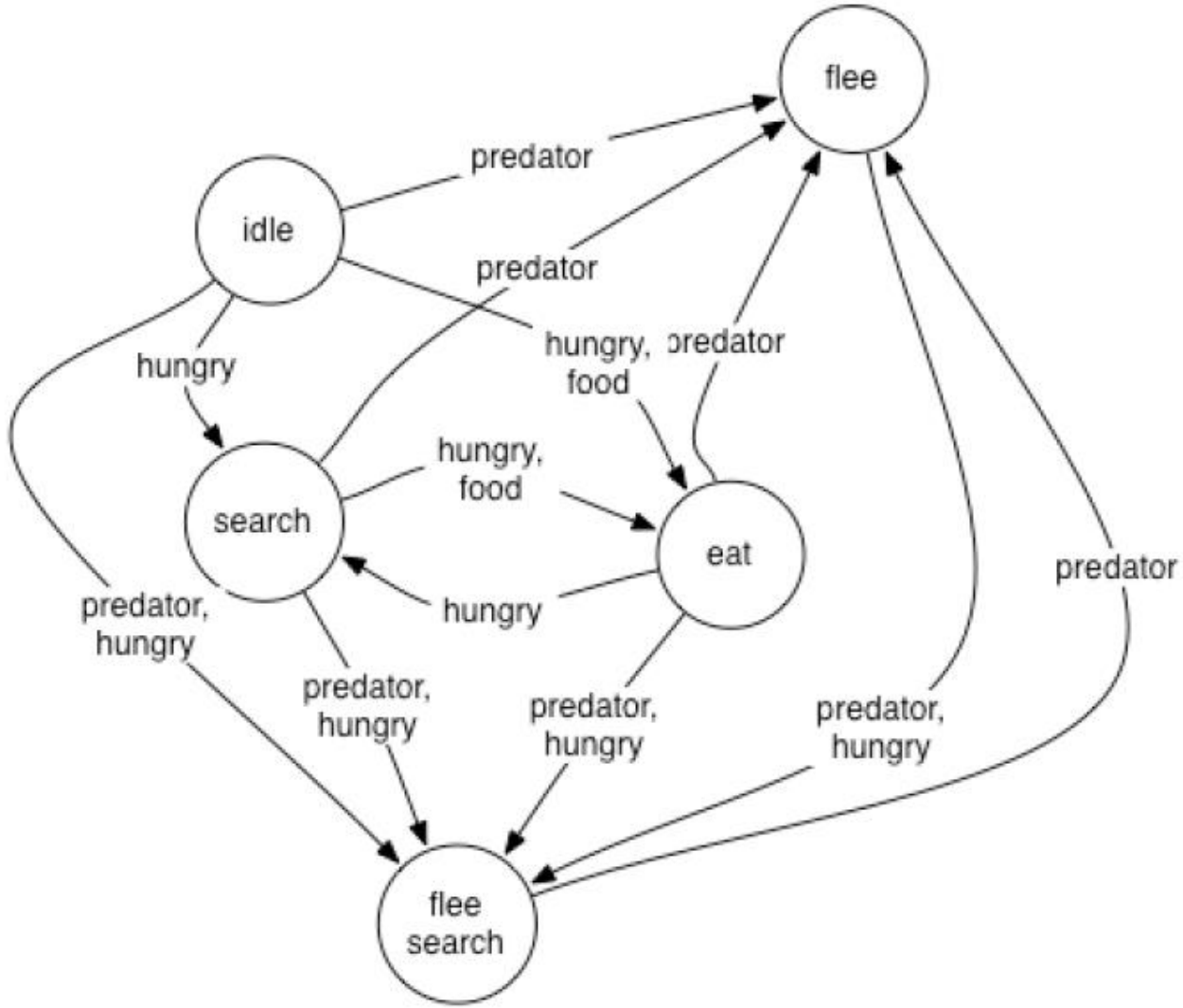
FSM Examples: Pac Man

- Red: Shadow, blinky
  - "pursuer" or "chaser"
  - Chase target: pac-man's tile
- Pink: Speedy, pinky
  - "ambusher"
  - Chase target: pac-man's tile + 4 ahead
- Blue: Bashful, inky
  - "whimsical"
  - Chase target: double vector from red ghost to (pac + 2 ahead)
- Orange: Pokey, Clyde
  - "feigning ignorance"
  - Chase target: dist < 8 ? scatter tile : pac-man's tile



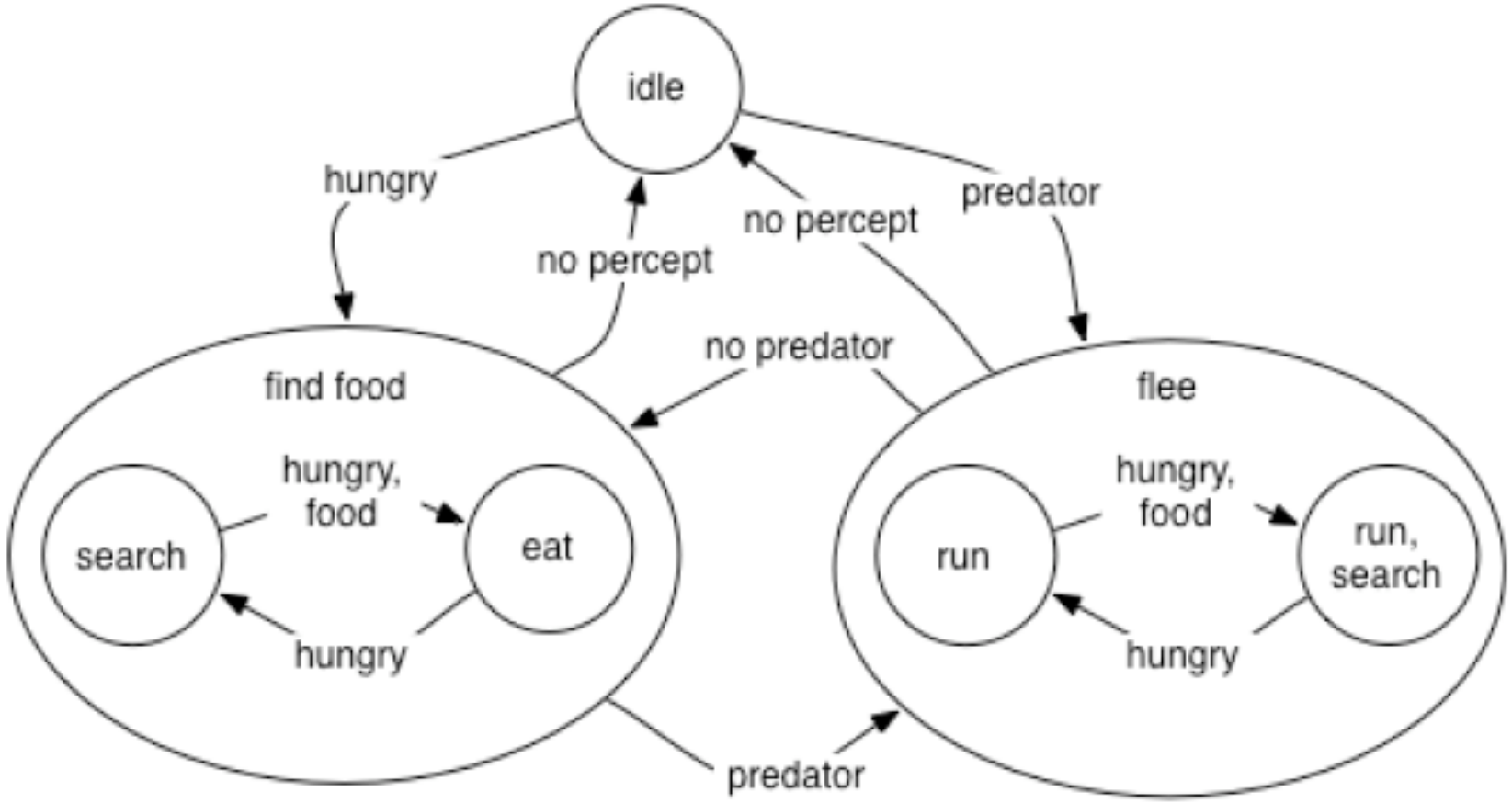
http://gameinternals.com/post/2072558330/understanding-pac-man-ghost-behavior

Prey Example



* Usually animations are linked to states, transitions, or both.

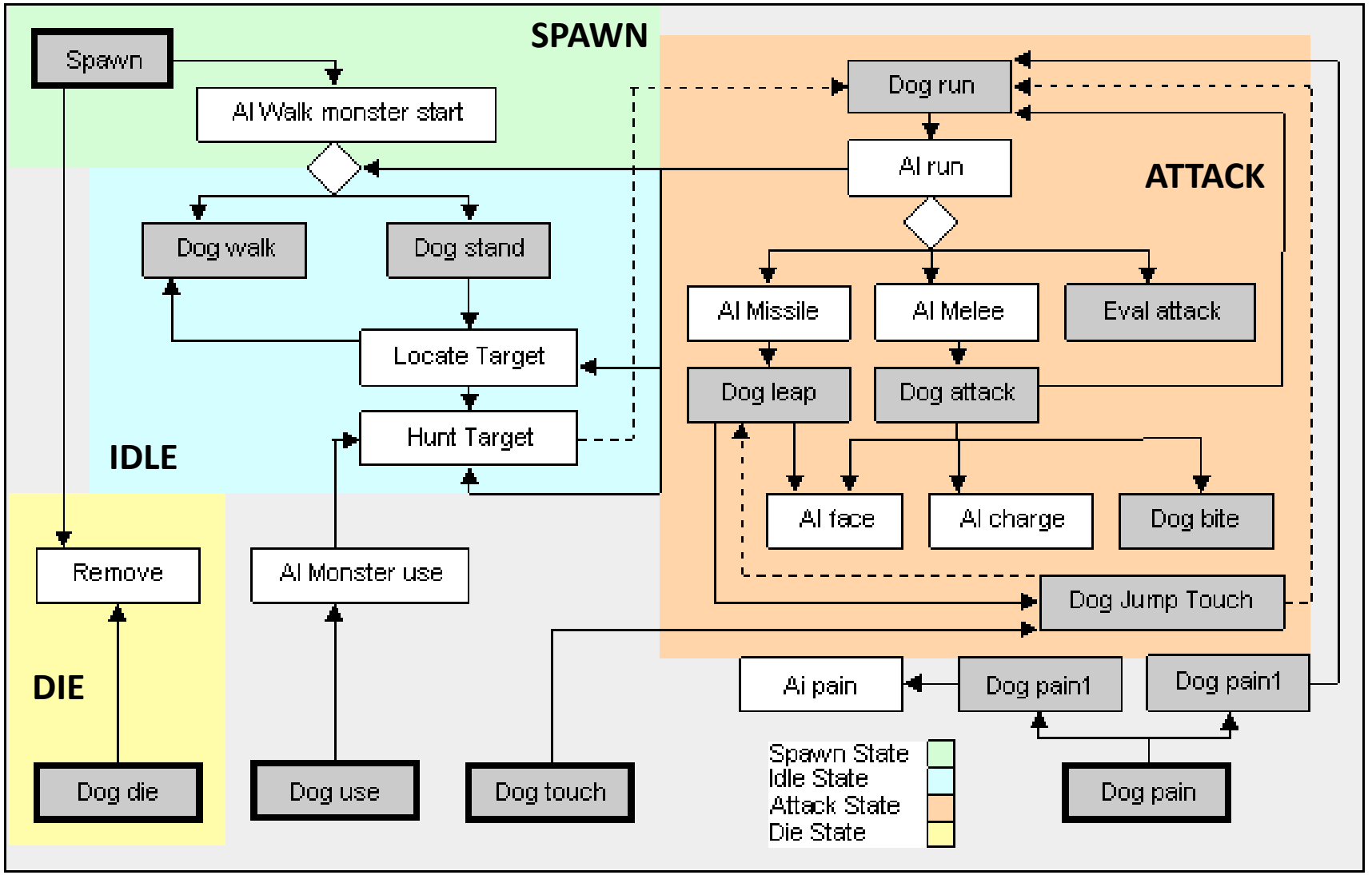
Hierarchical FSM Example



- Equivalent to regular FSMs
- Easier to think about encapsulation

FSM: Quake dog monster

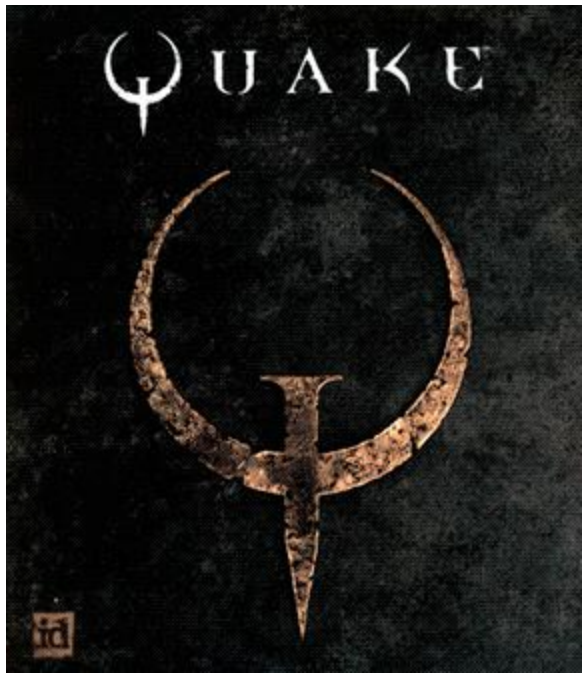
http://ai-depot.com/FiniteStateMachines/FSM-Framework.html



Main input event act. | Dog specific act. | (gen.) monster act.

FSM Examples

- Pac-Man
- FPSs
  - What might be states?
  - NPCs only?



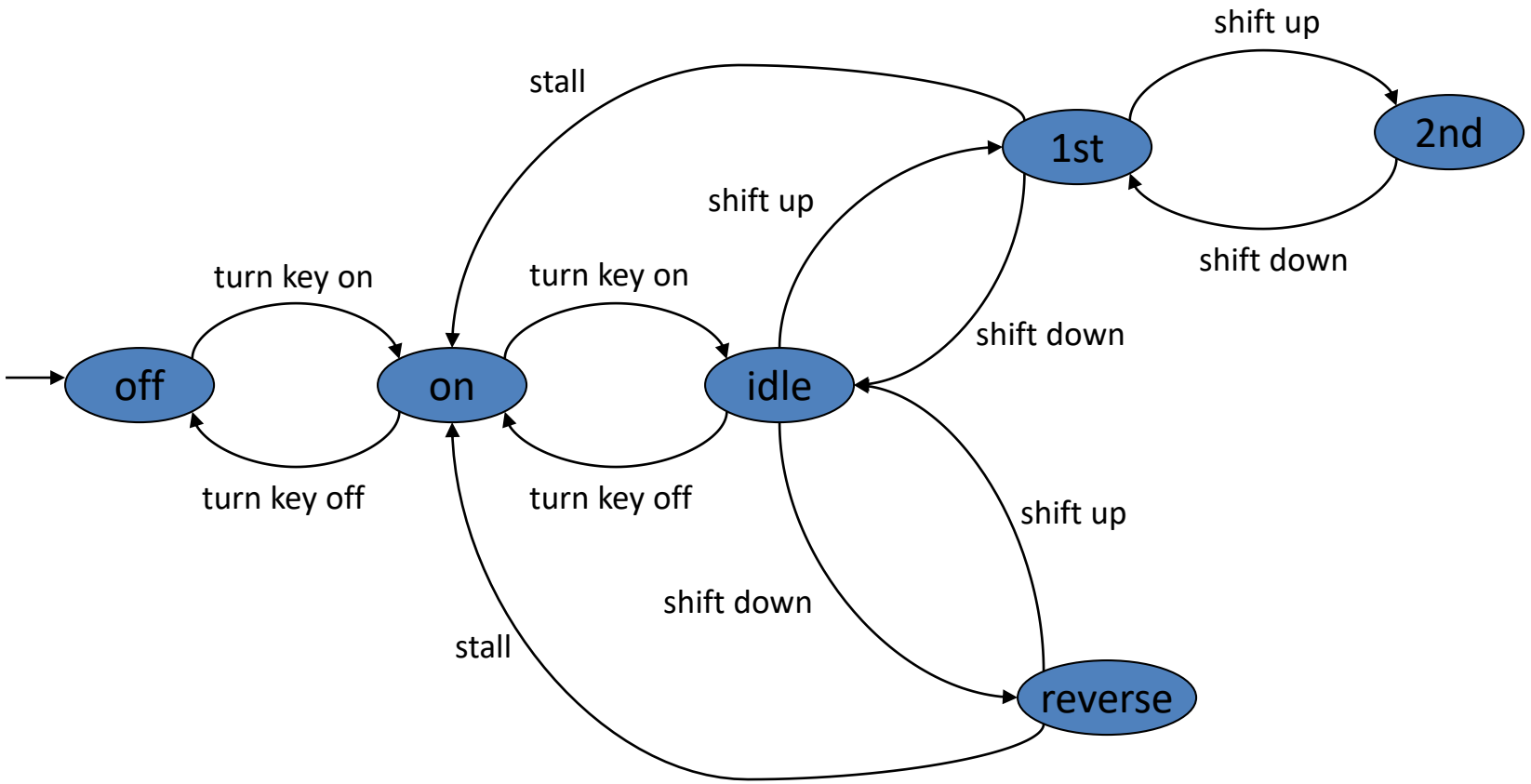
FSM Examples

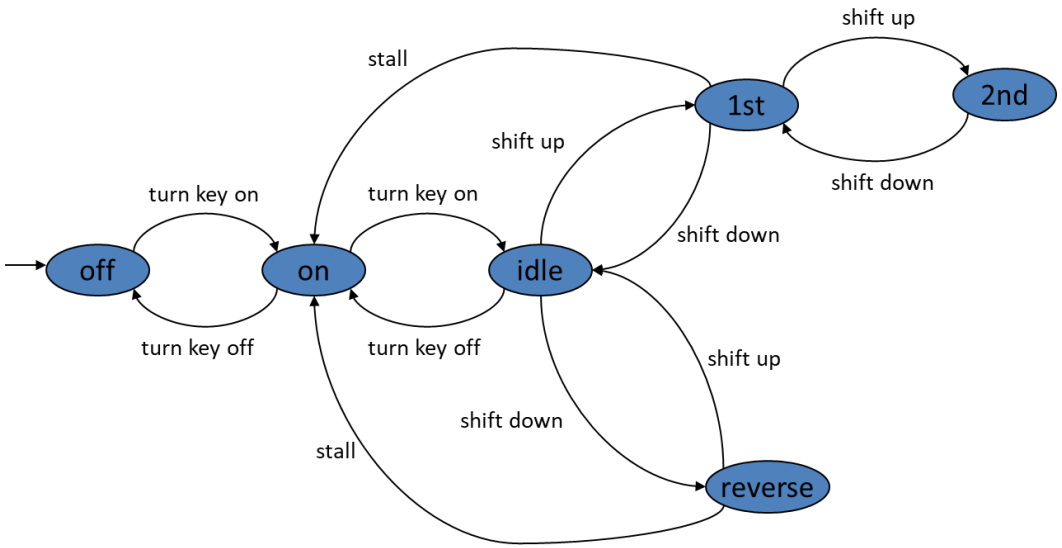
- Pac-Man
- FPSs
- Sports Simulations
  - What might be states?

FSM Examples

- Pac-Man
- FPSs
- Sports Simulations
- RTSs
  - What might be states?

UnrealScript Example






```
public void runStateMachine (Event e)
{
    switch (state) {
        case 0: //off
            if (e.isTurnOn()) { power=true; state=1;}
            break;
        case 1: //on
            if (e.isTurnOn()) { startEngine(); state=2;}
            else if (e.isTurnOff()) { power=false; state=0;}
            break;
        case 2: //idle
            makeEngineSound();
            if (e.isUpShift()) { gear=1; state=3;}
            else if (e.isDownShift()) { gear=-1; state=9;}
            else if (e.isTurnOff()) { stopEngine(); state=1;}
            break;

    ...
    }
}
```

Choices have consequences

FSM IMPLEMENTATIONS

Trajectory Update

- HW4: A*
- To come: More decision making
  - Planning
  - Decision trees
  - Behavior trees
  - Rule based systems
  - Fuzzy Logic
  - Markov Systems