The Game Development Process: Artificial Intelligence

Introduction to AI

- Opponents that are challenging, or allies that are helpful
  - Unit that is credited with acting on own
- Human-level intelligence too hard
  - But under narrow circumstances can do pretty well
  - Ex: chess and Deep Blue
- Artificial Intelligence
  - Around in CS for some time

AI for CS different than AI for Games

- Must be smart, but purposely flawed
  - Lose in a fun, challenging way
- No unintended weaknesses
  - No "golden path" to defeat
  - Must not look dumb
- Must perform in real time (CPU)
- Configurable by designers
  - Not hard coded by programmer
- "Amount" and type of AI for game can vary
  - RTS needs global strategy, FPS needs modeling of individual units at "footstep" level
  - RTS most demanding: 3 full-time AI programmers
  - Puzzle, street fighting: 1 part-time AI programmer

AI for Games: Mini Outline

- Introduction (done)
- Agents (next)
- Finite State Machines

Game Agents (1 of 3)

- Most AI focuses around game agent
  - Think of agent as NPC, enemy, ally or neutral
- Loops through: sense-think-act cycle
  - Acting is event specific, so talk about sense+think

Game Agents (2 of 3)

- *Sensing*
  - Gather current world state: barriers, opponents, objects
  - Need limitations: avoid "cheat" of looking at game data
  - Typically, same constraints as player (vision, hearing range)
    - Often done simply by distance direction (not computed as per actual vision)
  - Model communication (data to other agents) and reaction times (can build in delay)

Game Agents (3 of 3)

- *Thinking*
  - Evaluate information and make a decision
  - As simple or elaborate as required
  - Two ways:
    - Pre-coded expert knowledge, typically hand-crafted if-then rules + randomness to make unpredictable
    - Search algorithm for best (optimal) solution

Game Agents: Thinking (1 of 3)

- Expert Knowledge
  - Finite state machines, decision trees, … (FSM most popular, details next)
  - Appealing since simple, natural, embodies common sense
    - Ex: if you see enemy weaker than you, attack. If you see enemy stronger, then flee!
  - Often quite adequate for many AI tasks
  - Trouble is, often does not scale
    - Complex situations have many factors
    - Add more rules
    - Becomes brittle

Game Agents: Thinking (2 of 3)

- Search
  - Look ahead and see what move to do next
  - Ex: piece on game board, pathfinding
- Machine learning
  - Evaluate past actions, use for future
  - Techniques show promise, but typically too slow
  - Need to learn and remember

Game Agents: Thinking (3 of 3)

- Making agents stupid
  - Many cases, easy to make agents dominate
    - Ex: bot always gets head-shot
  - Dumb down by giving "human" conditions, longer reaction times, make unnecessarily vulnerable
- Agent cheating
  - Ideally, don't have unfair advantage (such as more attributes or more knowledge)
  - But sometimes might, to make a challenge
    - Remember, that's the goal, AI lose in challenging way
  - Best to let player know how agent is doing

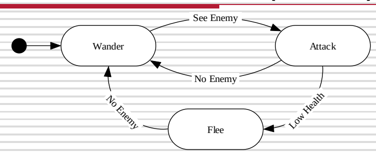
AI for Games: Mini Outline

- Introduction (done)
- Agents (done)
- Finite State Machines (next)

Group Exercise

- Consider game where hero is in a pyramid full of mummies.
  - Mummy wanders around maze
  - When hero gets close, can “sense” and moves quicker
  - When mummy sees hero and rushes to attack
  - If mummy wounded, it flees
- What “states” can you see? What are the transitions? Can you suggest appropriate code?

Finite State Machines (1 of 2)



- Abstract model of computation
- Formally:
  - Set of states
  - A starting state
  - An input vocabulary
  - A transition function that maps inputs and the current state to a next state

Finite State Machines (2 of 2)

- Most common game AI software pattern
  - Natural correspondence between states and behaviors
  - Easy to understand
  - Easy to diagram
  - Easy to program
  - Easy to debug
  - Completely general to any problem
- Problems
  - Explosion of states
  - Often created with ad-hoc structure