Artificial Intelligence: Agents

Artificial Intelligence

- The quality of a video game experience depends on the challenge presented to the player
- There is a need for intelligent opponents or allies
- AI describes the intelligence embodied in a man-made device
- Human level AI still unobtainable
  - Achieve "Reasonable and believable" AI

Game Artificial Intelligence: What is considered Game AI?

- Is it any NPC behavior?
  - A single "if" statement?
  - Scripted behavior?
- Pathfinding?
- Animation selection?
- Automatically generated environment?
- Best shot at a definition of game AI?

Possible Game AI definition

Inclusive view of game AI:

"Game AI is anything that contributes to the perceived intelligence of an entity, regardless of what's under the hood."

Goals of an AI Game Programmer

1. AI must be intelligent, yet purposely flawed

Opponents must challenge, entertain & lose

2. AI must have no unintended weaknesses

No easy defeat or dumb comportment

3. AI must perform within the constraints

CPU and memory constraints in real time (10-20%)

4. AI must be configurable by game designers or players

Adjustable difficulty levels

5. AI must not keep the game from shipping

Experimental technique must be proved early

Specialization of Game AI Developer

- No one-size fits all solution to game AI
  - Results in dramatic specialization
- Strategy Games
  - Battlefield analysis
  - Long term planning and strategy
- First-Person Shooter Games
  - One-on-one tactical analysis
  - Intelligent movement at footstep level
- Real-Time Strategy games the most demanding, with as many as three full-time AI game programmers

Game Agents

An agent may act as an

- Opponent
- Ally
- Neutral character

Continually loops through the cycle:

- Sense
- Think
- Act
- Optional learning or remembering step

Sense-Think-Act Cycle: Sensing

- Agent can have access to perfect information of the game world
  - Expensive/difficult to tease out useful info
- Game World Information
  - Complete terrain layout
  - Location and state of every game object
  - Location and state of player
- But this is considered cheating!
  - There should be limitations...

Sensing: Enforcing Limitations

- Human limitations?
- Limitations such as
  - Not knowing about unexplored areas
  - Not seeing through walls
  - Not knowing location or state of player
- Can only know about things seen, heard, or told about
- Must create a sensing model

Sensing: Human Vision Model for Agents

Get a list of all objects or agents; for each:

1. Is it within the viewing distance of the agent?
   - How far can the agent see?
   - What does the code look like?
2. Is it within the viewing angle of the agent?
   - What is the agent's viewing angle?
   - What does the code look like?
3. Is it unobscured by the environment?
   - Most expensive test, so it is purposely last
   - What does the code look like?

Sensing: Vision Model

Isn't vision more than just detecting the existence of objects?

What about recognizing interesting terrain features?

- What would be interesting to an agent?

Sensing: Human Hearing Model

- Humans can hear sounds
  - Can recognize sounds
    - Knows what emits each sound
  - Can sense volume
    - Indicates distance of sound
  - Can sense pitch
    - Sounds muffled through walls have more bass
  - Can sense location
    - Where sound is coming from

Sensing: Modeling Hearing

How do you model hearing efficiently?

- Do you model how sounds reflect off every surface?
- How should an agent know about sounds?

Sensing: Modeling Hearing Efficiently

- Event-based approach
  - When sound is emitted, it alerts interested agents
- Use distance and zones to determine how far sound can travel

Sensing: Communication

- Agents might talk amongst themselves!
  - Guards might alert other guards
  - Agents witness player location and spread the word
- Model sensed knowledge through communication
  - Event-driven when agents within vicinity of each other

Sensing: Reaction Times

- Agents shouldn't see, hear, communicate instantaneously
- Players notice!
- Build in artificial reaction times
  - Vision: ¼ to ½ second
  - Hearing: ¼ to ½ second
  - Communication: > 2 seconds

Sense-Think-Act Cycle: Thinking

- Sensed information gathered
- Must process sensed information
- Two primary methods
  - Process using pre-coded expert knowledge
    - If-then rules with some randomness
  - Use search to find an optimal solution

Thinking: Expert Knowledge

- Many different systems
  - Finite-state machines
  - Production systems
  - Decision trees
  - Logical inference
- Encoding expert knowledge is appealing because it's relatively easy
  - Can ask just the right questions
  - As simple as if-then statements
- Problems with expert knowledge
  - Not very scalable

Thinking: Search

Employs search algorithm to find an optimal or near-optimal solution

Given possible moves and rules that govern moves

A* pathfinding common use of search

Thinking: Machine Learning

- If imparting expert knowledge and search are both not reasonable/possible, then machine learning might work
- Examples:
  - Reinforcement learning
  - Neural networks
  - Decision tree learning
- Not often used by game developers
  - Why?

Thinking: Flip-Flopping Decisions

- Must prevent flip-flopping of decisions
- Reaction times might help keep it from happening every frame
- Must make a decision and stick with it
  - Until situation changes enough
  - Until enough time has passed

Sense-Think-Act Cycle: Acting

Sensing and thinking steps invisible to player

Acting is how player witnesses intelligence

Numerous agent actions, for example:

- Change locations
- Pick up object
- Play animation
- Play sound effect
- Converse with player
- Fire weapon

Acting: Showing Intelligence

- Proficiency and subtlety of actions impact perceived level of intelligence
- Enormous burden on asset generation
- Agent can only express intelligence in terms of vocabulary of actions
- Current games have huge sets of animations/assets
- Must use scalable solutions to make selections
  - animation selection given to designers and artists (data-driven design)

Extra Step in Cycle: Learning and Remembering

- Optional 4th step
- Not necessary in many games
  - Agents don't live long enough
  - Game design might not desire it

Learning

- Remembering outcomes and generalizing to future situations
- Simplest approach: gather statistics
  - If 80% of time player attacks from left
  - Then expect this likely event
- Adapts to player behavior

Remembering

- Remember hard facts
  - Observed states, objects, or players
- For example
  - Where was the player last seen?
  - What weapon did the player have?
  - Where did I last see a health pack?
- Memories should fade
  - Helps keep memory requirements lower
  - Simulates poor, imprecise, selective human memory

Remembering within the World

All memory doesn't need to be stored in the agent – can be stored in the world

For example:

- Agents get slaughtered in a certain area
- Area might begin to "smell of death"
- Agent's path planning will avoid the area
- Simulates group memory

Making Agents Stupid

- Sometimes very easy to trounce player
  - Make agents faster, stronger, more accurate
- Sometimes necessary to dumb down agents, for example:
  - Make shooting less accurate
  - Make longer reaction times
  - Engage player only one at a time
  - Change locations to make self more vulnerable

Agent Cheating

- Players don't like agent cheating
  - When agent given unfair advantage in speed, strength, or knowledge
- Sometimes necessary
  - For highest difficultly levels
  - For CPU computation reasons
  - For development time reasons
- Don't let the player catch you cheating!
  - Consider letting the player know upfront