Adaptive Behavior for Fighting Game Characters

May 24, 2005
Leo Lee
Committee Members:
Dr. Chris Pollett
Dr. Rudy Rucker
Dr. Jeffrey Smith
Outline

- Motivation and Goal
- Requirements
- Game Architecture
- AI System
  - The Three Layers
  - The Two AI Model Types
- Conclusion and Future Work
- Demo Alpha Fighter
Motivation: Why Games?

- Games are relevant financially
  - Huge industry rivaling film industry
- Games are relevant academically
  - It’s difficult!
    - Games are large projects
    - Games cover a wide area of CS
      - Graphics, AI, Physics, S/W Engineering, UI
    - Games must run in real-time
Motivation: Why Game AI?

- Common game AI techniques[3]:
  - FSMs, decision trees, A* path-finding
  - Developer defines all behavior
  - Leads to a static world

- Promising game AI techniques[4]:
  - Neural networks, genetic algorithms, etc.
  - Next step = machine learning
  - Developer defines rules
  - Emergent behavior, adaptation, dynamic world
Goal: 3D Fighting Game
Requirements

- Very simple game mechanics
  - Two playable characters
  - Simple and small set of fighting moves
  - Goal is just to beat your opponent

- Provides context to showcase adaptive AI
  - Non-player character (NPC) adapts to player
  - Provide extensibility
Game Architecture: Major Modules

- AI
- Physics
- Game logic
- Graphics
- Sound
- Controls
Game Architecture:
Graphics, Sound, Controls

- Graphics: Direct3D
  - CGraphics class, CThing::render()
- Sound: DirectMusic and DirectSound
  - playSound(), playMusic()
- Controls: DirectInput
  - keyPressed(), keySingle(), etc.
  - processInput()
Game Architecture: Game Logic, Physics

- Game Flow

- Physics
  - Simple kinematics (pos, vel, acc)
  - Collision detection – spheres & capsules
  - Collision reaction – body part state
Game Architecture: Collision Detection/Reaction

- **Blocking**
- **Neutral**
- **Attacking**
- **Sphere**
- **Capsule**
Game AI: The Three Layers

- **Strategic Layer**
  - Choose attack set or defense set

- **Tactical Layer**
  - Choose a tactic from set decided above

- **Operational Layer**
  - Execute the tactic
Game AI: Strategic Layer

- The only non-adaptive layer
- Normally offensive (regular tactics)
- Defense (counter) when “see” attack coming.
  - NPC does not know which attack it is.
  - Mimic reactionary behavior of human player.
AI models choose tactic from given set

Based on a Matrix of Production Sets (described later)

Note each takes two inputs as indices into the matrix
A Tactic is a sequence of Steps

Ex 1: Move within kicking range and attack
BEGIN_TACTIC Long_Attack 0(regular) 1(init points)
MoveWithinRange 1(kicking)
Attack
END_TACTIC

Ex 2: Block for at most 2 seconds then attack
BEGIN_TACTIC Block_2_Attack 1(counter) 1
Block 2 (max seconds, or until attacked)
Attack
END_TACTIC
Game AI: Operational Layer

- Carries out details of a Step
  - Attack – Which attack to use?
    - Based on NPC Attack AI Model
  - Block – Block high or low?
    - Based on Player Attack AI Model
  - MoveWithinRange – How?
    - Based on simple conditional logic
Game AI: The Two Model Types

- HMM Tree Array
  - Used by Player Attack Model
  - Used for prediction

- Matrix of Sets
  - Used by Regular Tactics Model, Counter Tactics Model, and NPC Attack Model
  - Used for production
Game AI: HMM Tree Array[1]

- Level $i$ contains $n$-gram of degree $i$.
- To predict: traverse to level $n - 1$ and pick most probable child.
- To learn: traverse to node at level $n$ and add points.
Game AI: Matrix of Sets

- To produce (probabilistic production):
  - Find the set.
  - Pick random number $r$ [0, sum of points in set].
  - Iterate through set until sum of points $\geq r$. 
Game AI: Matrix of Sets

- To learn (reinforcement with discount):
  - Logs are kept of recent tactics/actions
  - On a reinforcing event:
    1. Adjust points of newest logged element by $x$
    2. Discount $x$ by discount factor
    3. If $x \neq 0$ and more in log repeat (1) with next
    - Note points are integers so $x \neq 0$ makes sense.

Initial amount = -8
Falloff factor = 0.5
## Game AI: Reinforcing Events

<table>
<thead>
<tr>
<th>Event</th>
<th>Reinforcement</th>
</tr>
</thead>
<tbody>
<tr>
<td>NPC hurts player</td>
<td>+ +</td>
</tr>
<tr>
<td>Player misses NPC</td>
<td>+</td>
</tr>
<tr>
<td>NPC misses player</td>
<td>−</td>
</tr>
<tr>
<td>Player hurts NPC</td>
<td>− −</td>
</tr>
</tbody>
</table>
References


