Constructing Intelligent Agents in Games

Risto Miikkulainen
Department of Computer Sciences
The University of Texas at Austin

Joint work with Bobby D. Bryant, Ryan Cornelius, Aliza Gold, Faustino Gomez, Igor Karpov, Kenneth O. Stanley, Chern Han Yong
Artificial Intelligence (AI)

- Ultimate goal: Building intelligent machines
- Much more difficult than expected
  - Only specialized, limited assistants exist today: “Wizards”, Internet bots, Roombas
- Real world is complex, open ended, messy
  - Scaleup is impractical, even dangerous
Intelligent Agents in Games

- Virtual worlds more tractable than the real world
  - Games are controlled, formal, measurable
  - They are a safe platform for AI
  - They provide realistic, significant challenges

- Intelligent agents can be deployed in games today
Traditional AI Technology

Final Championship Game, 1994
Chinook Red

Lafferty White

- Much of AI developed in games & for games
  - Board games
  - Good Old-Fashioned AI (GOFAI): Rules, logic, search...

- Very successful in extreme cases
  - Chinook 1994, Deep Blue 1997...
  - Largely brute force
The New World of Games

- Since the 1990s, the field has changed
- Video games have become a major industry
  - $25B worldwide (2005)
  - Sophisticated simulated worlds
  - Part of everyday life
AI in Video Games?

- Very little; Still mostly GOFAI
  - Scripting, authoring
  - A* pathfinding, finite state machine behaviors

- Sometimes impressive, but often simple and repetitive
  - Part of the challenge is to figure out the AI
Challenges for AI

- GOFAI does not work well in video games

- They are different from board games:
  - Multiple agents
  - Embedded: continuous, noisy, large-dimensional
  - Real-time, changing environments
A New Approach: “Computational Intelligence”

- Natural Computation: Neural networks, evolution, reinforcement learning
- Powerful in many statistical domains
  - E.g. pattern recognition, control, prediction, decision making
  - When hard to formulate rules, but plenty of examples
- Can learn and generalize
  - Learn a nonlinear function that matches the examples
Neural Network Agents in Video Games

- Input variables describe the state
- Output variables describe actions
- Network between input and output:
  - Nonlinear hidden nodes
  - Weighted connections
- Execution:
  - Numerical activation of input
  - Performs a nonlinear mapping
  - Memory in recurrent connections
- Learning: No targets; based on reinforcement
- Performance based on statistics, not rules (cf. HAL)
A Unique Opportunity

- CI is well suited for video games
- Research opportunity like 1980s for GOFAI
  - Adapting, embedded intelligent agents
  - Progress towards intelligent machines
- Can lead to better games
  - Reduce production cost, find bugs
  - Make games more interesting
  - Allow training games
Current CI Research in Games

(Fogel 2001)

- Initial successes with board games
  - Checkers, chess, backgammon, go, othello...
- Technique apply to video games as well
  - FPS, RTS: Unreal, Neverwinter, Quake...
- New techniques started to emerge
Neuroevolution (NE)

- Chromosomes are strings of connection weights
  - E.g. 0.1 10.5 8.8 9.3 -1.8 -9.9 0.0 -0.8 19.2

- Evolved through crossover and mutation
  - Natural mapping between genotype and phenotype

- Parallel search for a solution network
Advanced NE: Complexification

- Neuroevolution of Augmenting Topologies (NEAT) 
  (Stanley et al. 2004)
- Optimizing connection weights and network topology
- Based on Complexification
  - Of networks: Mutations add nodes and connections
  - Of behavior: Elaborate earlier behaviors
Why Complexification?

- Problem with NE: Search space is too large
- Complexification keeps the search tractable
  - Start simple, add more sophistication
- Incremental construction of intelligent agents
Applying NE to Board Games

- A good platform to develop techniques
- Different from GOFAI: Beyond limits of search
  - Pattern recognition
  - Filtering information
  - Opponent modeling
- Checkers, othello, chess, go, poker...
Discovering Novel Strategies in Othello

- Players take turns placing pieces
- Each move must flank opponent’s piece
- Surrounded pieces are flipped
- Player with most pieces wins
Strategies in Othello

- Positional
  - Number of pieces and their positions
  - Typical novice strategy

- Mobility
  - Number of available moves: force a bad move
  - Much more powerful, but counterintuitive
  - Discovered in 1970’s in Japan
Evolving Against a Random Player

- Network sees the board, suggests moves by ranking
- Networks maximize piece counts throughout the game
- A positional strategy emerges
- Achieved 97% winning percentage
Evolving Against an $\alpha$-$\beta$ Program

- Iago’s positional strategy destroyed networks at first
- Evolution turned low piece count into an advantage
- Mobility strategy emerged!
- Achieved 70% winning percentage
Black’s positions strong, but mobility weak

White (the network) moves to f2

Black’s available moves b2, g2, and g7 each will surrender a corner

The network wins by forcing a bad move
• Neuroevolution discovered a strategy novel to us

• “Evolution works by tinkering”
  – So does neuroevolution
  – Initial disadvantage turns into novel advantage
Applying NE to Video Games

- Can be used to build “mods” to existing games
  - Adapting characters, assistants, tools
- Can also be used to build new games
- New genre: Machine Learning game
  - Gameplay involves interacting with the learning system
  - Design, training intelligent agents, training people
NERO: A Machine Learning Game

- Initially produced by *Digital Media Collaboratory*, UT Austin
- First 3 years mostly by volunteer undergraduates
- V1.0 - V2.0 using Garage Games Torque™ game engine
- V2.0 available for download at http://nerogame.org
NERO: A Machine Learning Game

- Currently funded by Undergraduate Research Initiative
- Game technology, AI, graphics, networking...
- OpenNero: Irrlicht, ODE, OpenGL, OpenAL, RakNet, CAL3D, QT, Python...
- General AI research and education platform
**NERO Gameplay**

- Teams of agents trained to battle each other
  - Player trains agents through excercises
  - Agents evolve in real time

- Challenging platform for reinforcement learning
  - Real time, open ended, requires discovery
A parallel, continuous version of NEAT

Individuals created and replaced every $n$ ticks

Parents selected probabilistically, weighted by fitness

Long-term evolution equivalent to generational NEAT
NERO Player Actions

• Player can place items on the field
e.g. static enemies, turrets, walls, rovers, flags

• Sliders specify relative importance of goals
e.g. approach/avoid enemy, cluster/disperse, hit target, avoid fire...

• Networks evolved to control the agents
NERO Neural Network Agent

- Each agent is controlled by an evolved neural network
  - Inputs: egocentric sensors
  - Outputs: simple actions

- The networks are evolved using rtNEAT

DEMO
NERO 2.0: Territory Mode

- Battle to conquer “control points”: interactive & more fun
- Player trains various “specialists”: defenders, chargers, snipers,...
- During battle, player is a high-level commander
  - Dynamically deploys specialists
  - Dynamically specifies targets
  - Real time against other players

- DEMO
Challenge: Utilizing Knowledge

- Given a problem, NE discovers a solution by exploring
  - Sometimes highly original solutions
  - Requires lots of exploration
- Designers may want to have more control
  - Seeding with initial behaviors
- Players may want to interact with learning
  - Giving advice during evolution
Incorporating Rules into NE

E.g. how to go around a wall in NERO

- Specify as a rule:
  - wall\_ahead: move\_forward, turn\_right
  - wall\_45deg\_left, move\_forward, turn\_right\_slightly

- Convert into a network with KBANN (Maclin and Shavlik 1996)
Incorporating Knowledge with KB-NEAT

- KBANN network is added to NEAT networks
  - Treated as complexification
- Continues to evolve
  - If knowledge is useful, it stays
  - If not, it is discarded or converted
- Can be given on-line as advice (Yong et al. 2006)
- DEMO
Seeding the Population with KB-NEAT

- KBANN+NEAT can also be used to seed an initial population (Cornelius et al. 2006)
- E.g. approaching a roving enemy
  - With knowledge, approach right away
  - Otherwise, learn to do so in 3mins
- Does not delay adaptation to new situations
  - E.g. Approaching an enemy that is shooting
- DEMO
Lessons from NE in Games

- NE constructs intelligent agents in games
  - Discovers effective behaviors
  - Adapts in real time
- Can add adaptation to existing games
- Can build machine-learning games
- Requires many evaluations
  - Best when parallel evaluations possible
- Best when combined with human guidance
  - E.g. examples or rules
Conclusion

• Video games for CI like board games for GOFAI
  – Research catalyst: control, decision making, optimization with uncertainty, material and time constraints
  – Feasible platform for developing intelligent agents

• Killer application
  – Huge potential economic impact
  – Entertainment, training simulators
  – Robotics, resource optimization, intelligent assistants
• After 38 years, are we ready to construct intelligent agents?
• After 38 years, are we ready to construct intelligent agents?

• Not quite, but we are getting close!

For papers, demos, software: nn.cs.utexas.edu, nerogame.org