Evolving Neural Networks

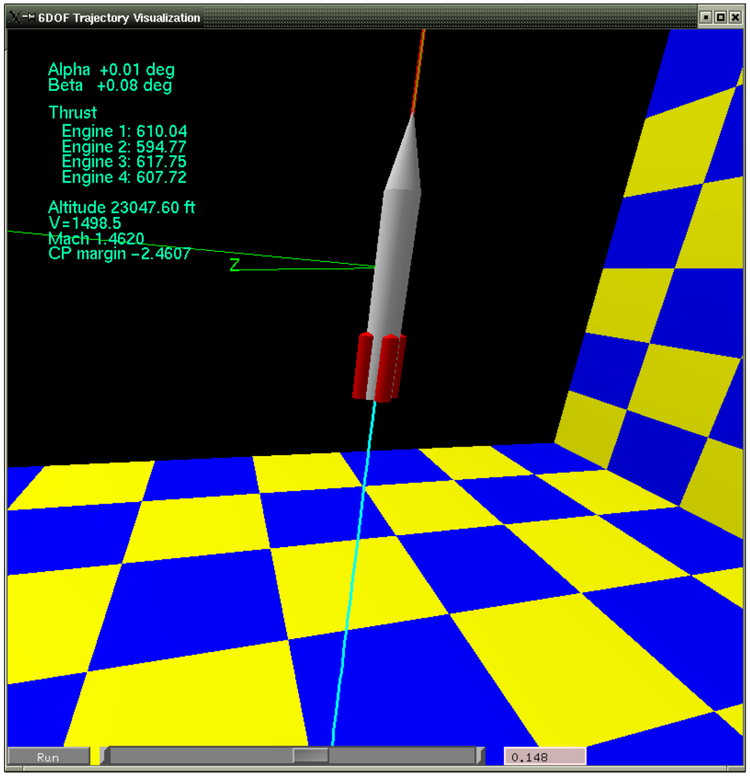
Risto Miikkulainen

Department of Computer Sciences

The University of Texas at Austin

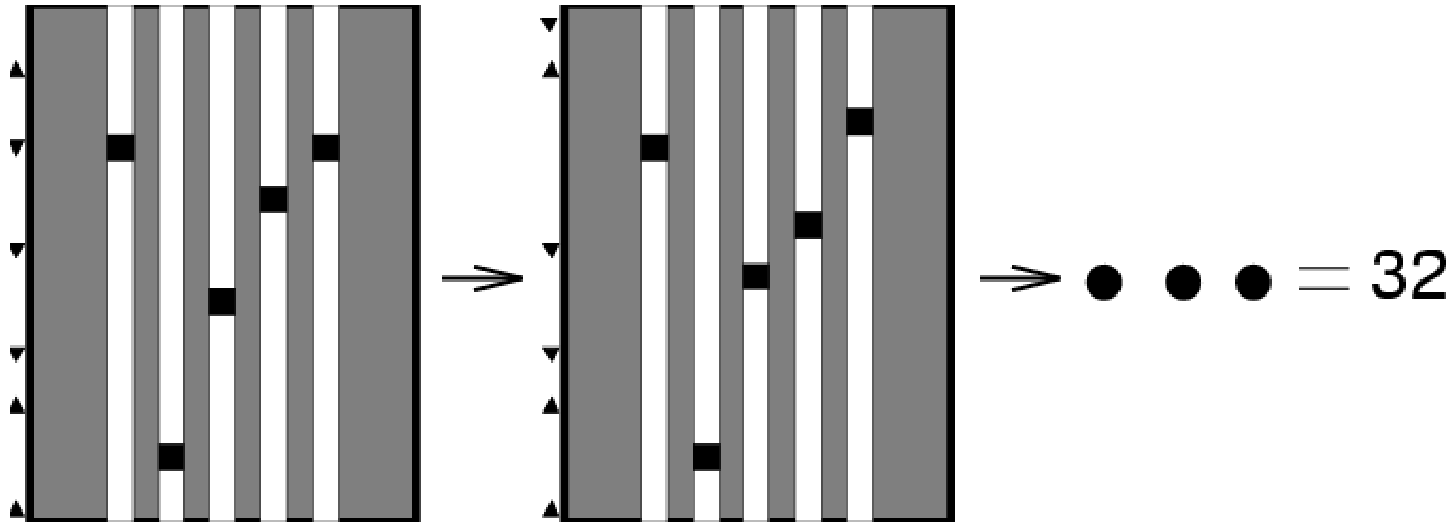
http://www.cs.utexas.edu/users/risto

Why Neuroevolution?



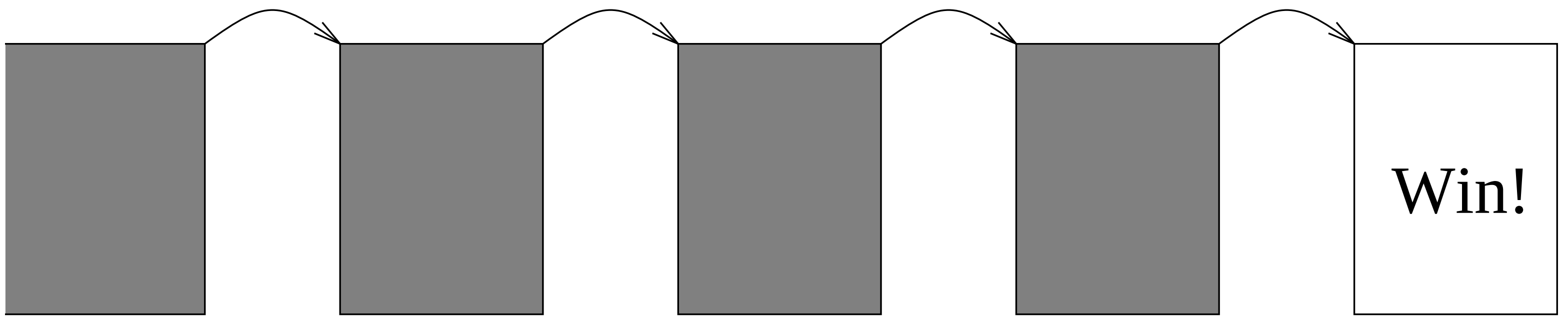
- Neural nets powerful in many statistical domains
  - E.g. control, pattern recognition, prediction, decision making
  - No good theory of the domain exists
- Good supervised training algorithms exist
  - Learn a nonlinear function that matches the examples
- What if correct outputs are not known?

Sequential Decision Tasks



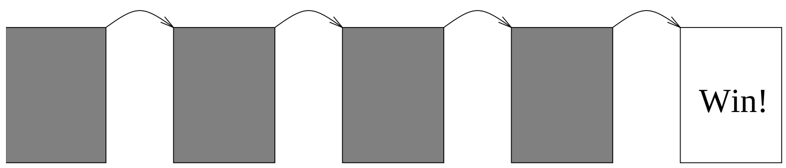
- POMDP: Sequence of decisions creates a sequence of states
- No targets: Performance evaluated after several decisions
- Many important real-world domains:
  - Robot/vehicle/traffic control
  - Computer/manufacturing/process optimization
  - Game playing

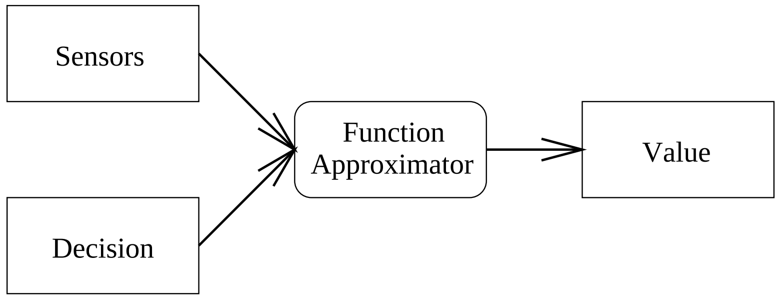
Forming Decision Strategies



- Traditionally designed by hand
  - Too complex: Hard to anticipate all scenarios
  - Too inflexible: Cannot adapt on-line
- Need to discover through exploration
  - Based on sparse reinforcement
  - Associate actions with outcomes

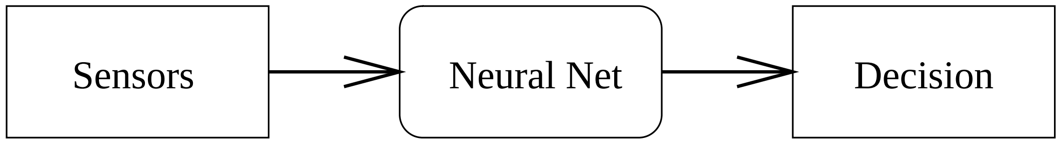
Standard Reinforcement Learning





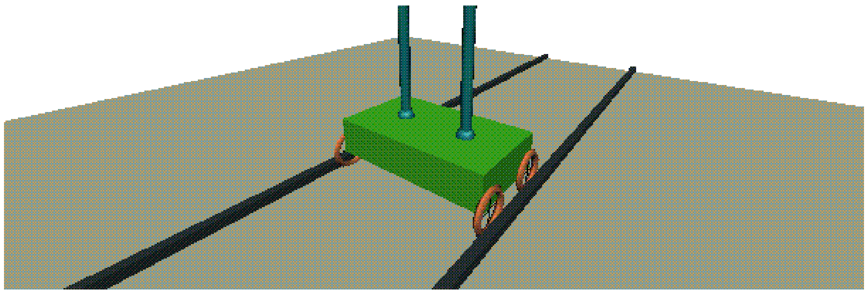
- AHC, Q-learning, Temporal Differences
  - Generate targets through prediction errors
  - Learn when successive predictions differ
- Predictions represented as a value function
  - Values of alternatives at each state
- Difficult with large/continuous state and action spaces
- Difficult with hidden states

Neuroevolution (NE) Reinforcement Learning



- NE = constructing neural networks with evolutionary algorithms
- Direct nonlinear mapping from sensors to actions
- Large/continuous states and actions easy
  - Generalization in neural networks
- Hidden states disambiguated through memory
  - Recurrency in neural networks

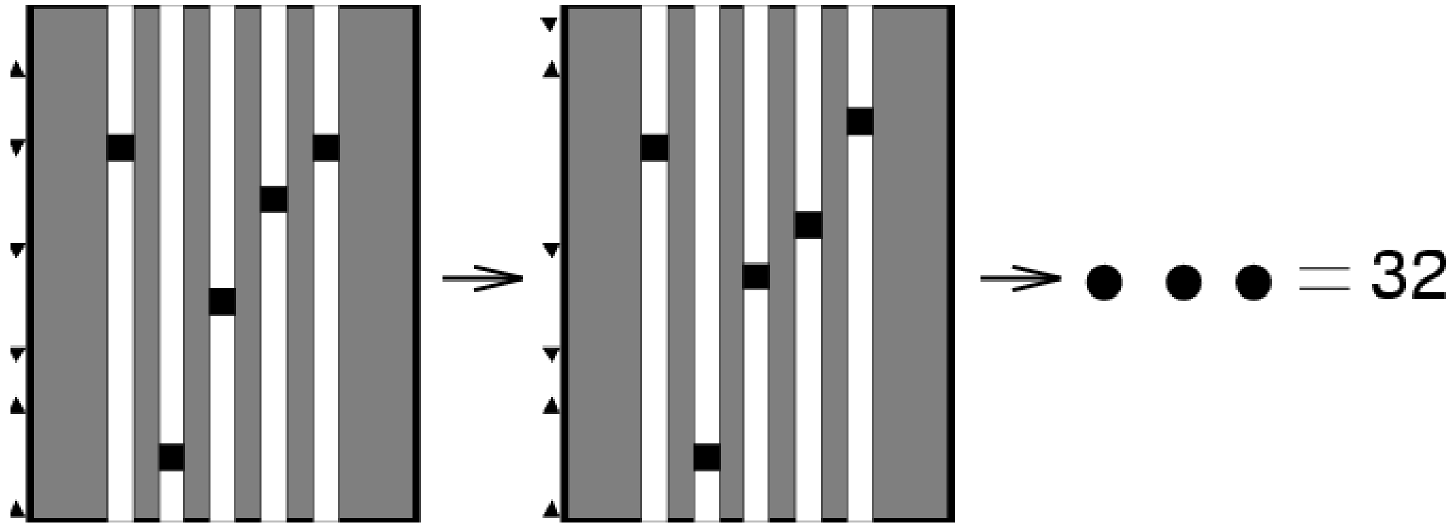
How well does it work?



Poles	Method	Evals	Succ.
One	VAPS	500,000	0%
	SARSA	13,562	59%
	Q-MLP	11,331	
	NE	589	
Two	NE	24,543	

- Difficult RL benchmark: Non-Markov Pole Balancing
- NE 2 orders of magnitude faster than standard RL
- NE can solve harder problems

Role of Neuroevolution



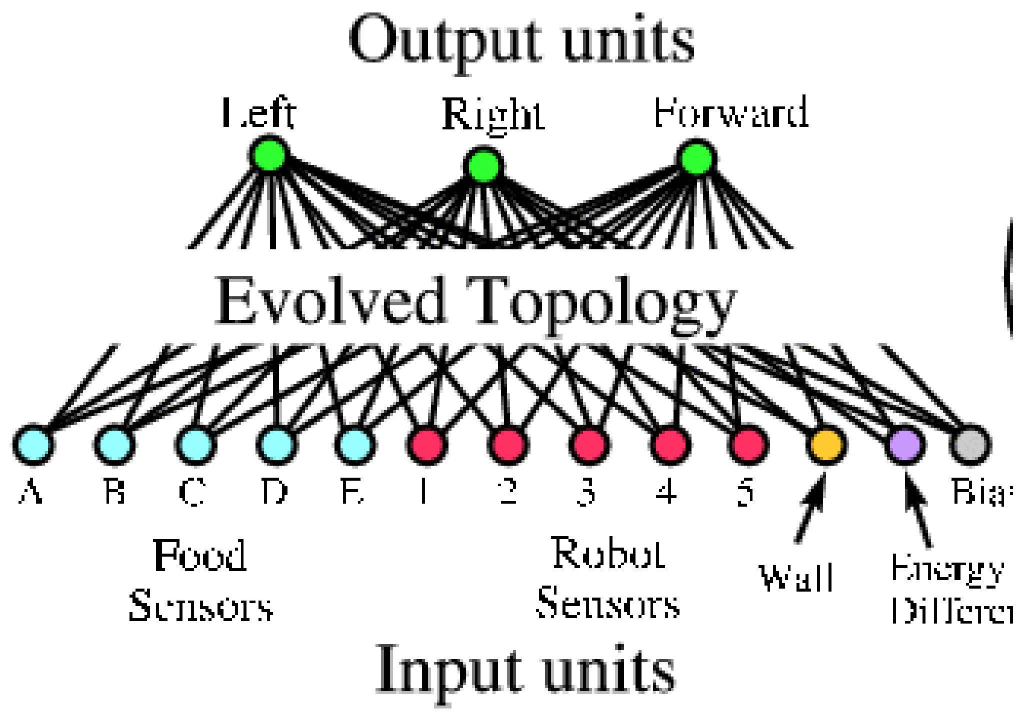
- Powerful method for sequential decision tasks [17,36,65]
  - Optimizing existing tasks
  - Discovering novel solutions
  - Making new applications possible
- Also may be useful in supervised tasks [31,41]
  - Especially when network topology important
- Unique model of biological adaptation and development [37,44,61]

Outline

- Basic neuroevolution techniques
- Advanced techniques
  - E.g. combining learning and evolution
- Extensions to applications
- Application examples
  - Control, Robotics, Artificial Life, Games

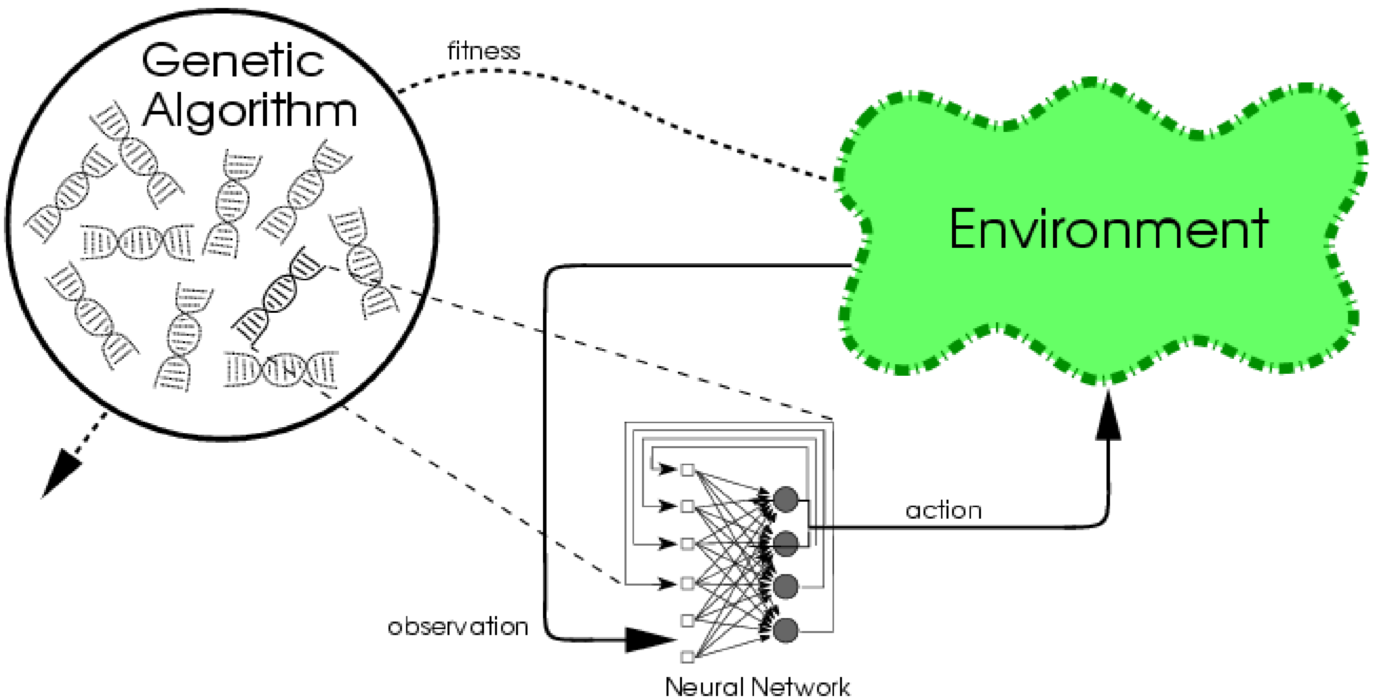
Neuroevolution Decision Strategies

- Input variables describe the state
- Output variables describe actions
- Network between input and output
  - Hidden nodes
  - Weighted connections



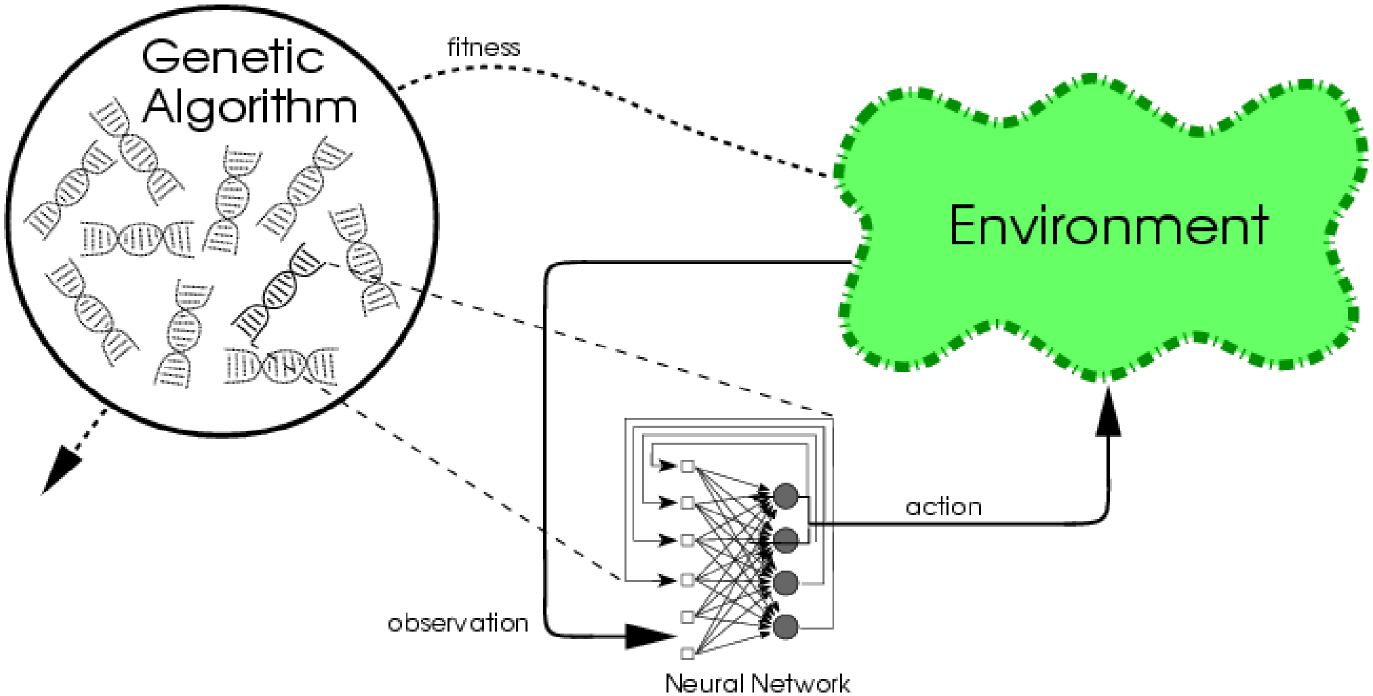
- Execution:
  - Numerical activation of input
  - Nonlinear weighted sums
- Performs a nonlinear mapping
  - Memory in recurrent connections
- Connection weights and structure evolved

Conventional Neuroevolution (CNE)



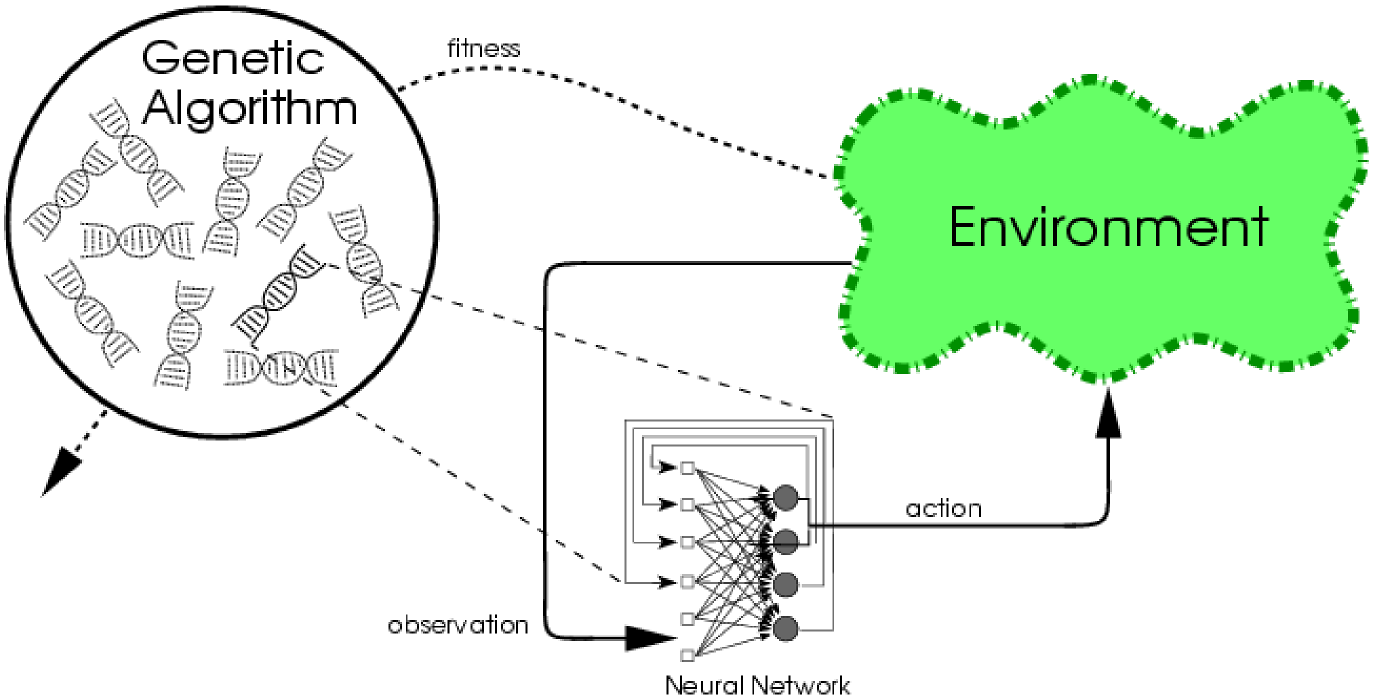
- Evolving connection weights in a population of networks [31,65,66]
- Chromosomes are strings of weights (bits or real)
  - E.g. 10010110101100101111001
  - Usually fully connected, fixed topology
  - Initially random

Conventional Neuroevolution (2)



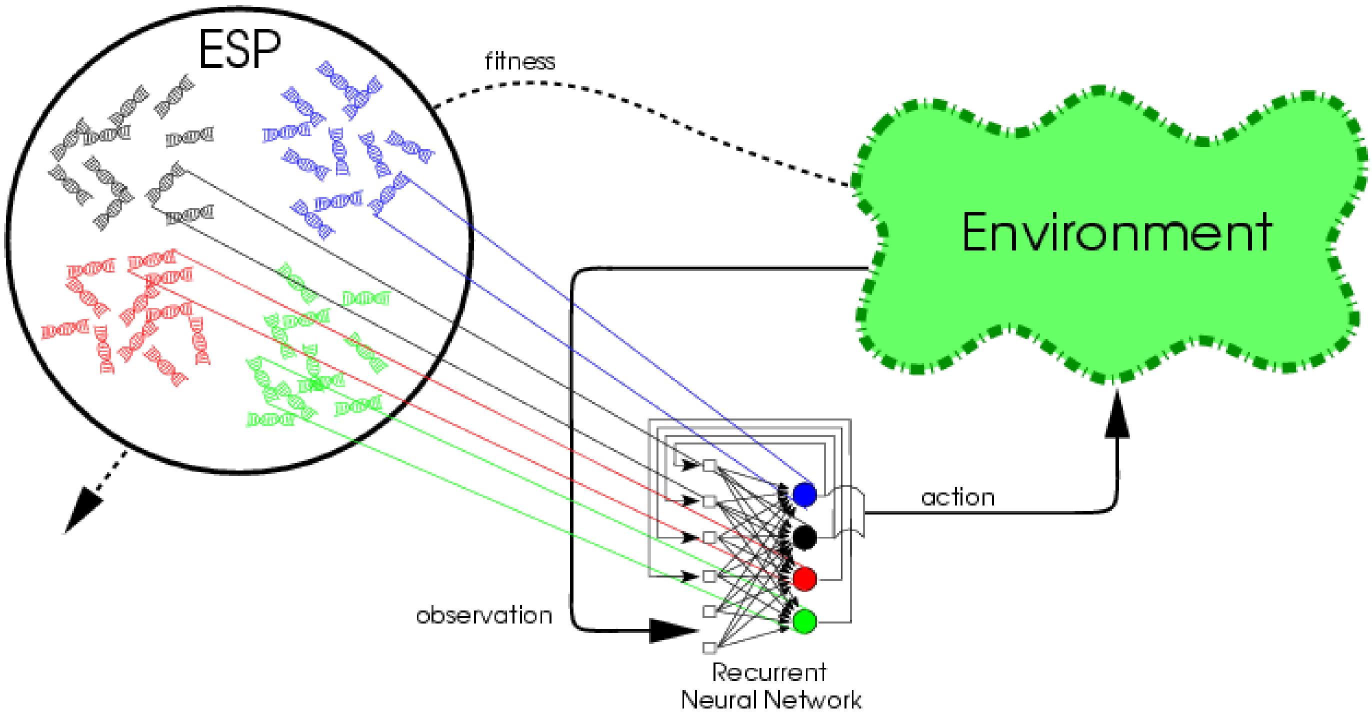
- Each NN evaluated in the task
  - Good NN reproduce through crossover, mutation
  - Bad thrown away
  - Over time, NNs evolve that solve the task
- Natural mapping between genotype and phenotype
- GA and NN are a good match!

Problems with CNE



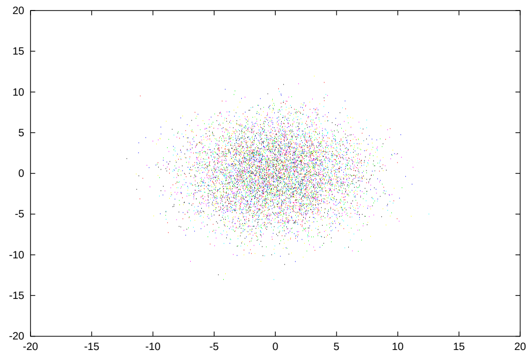
- Evolution converges the population (as usual with EAs)
  - Diversity is lost; progress stagnates
- Competing conventions
  - Different, incompatible encodings for the same solution
- Too many parameters to be optimized simultaneously
  - Thousands of weight values at once

Advanced NE 1: Evolving Neurons

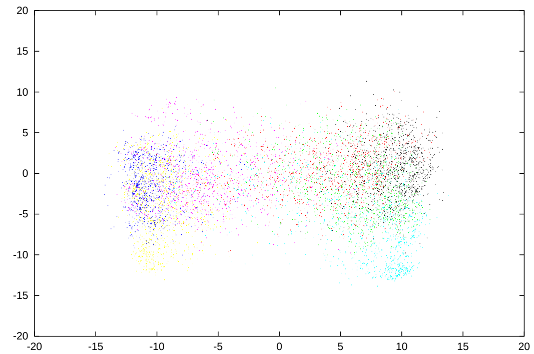


- Evolving individual neurons to cooperate in networks [1,35,41] (Agogino GECCO'05)
- E.g. Enforced Sub-Populations (ESP [17])
  - Each (hidden) neuron in a separate subpopulation
  - Fully connected; weights of each neuron evolved
  - Populations learn compatible subtasks

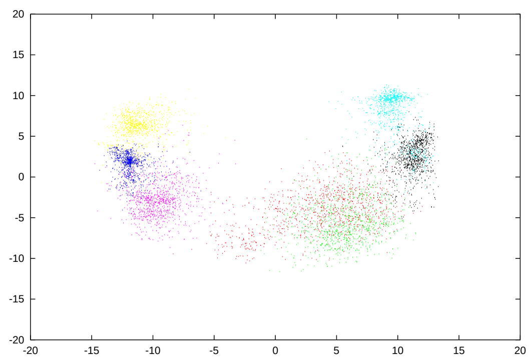
Evolving Neurons with ESP



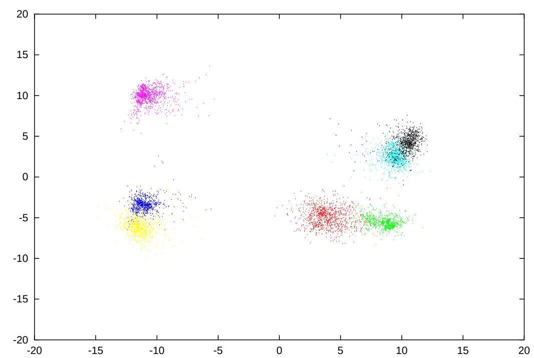
Generation 1



Generation 20



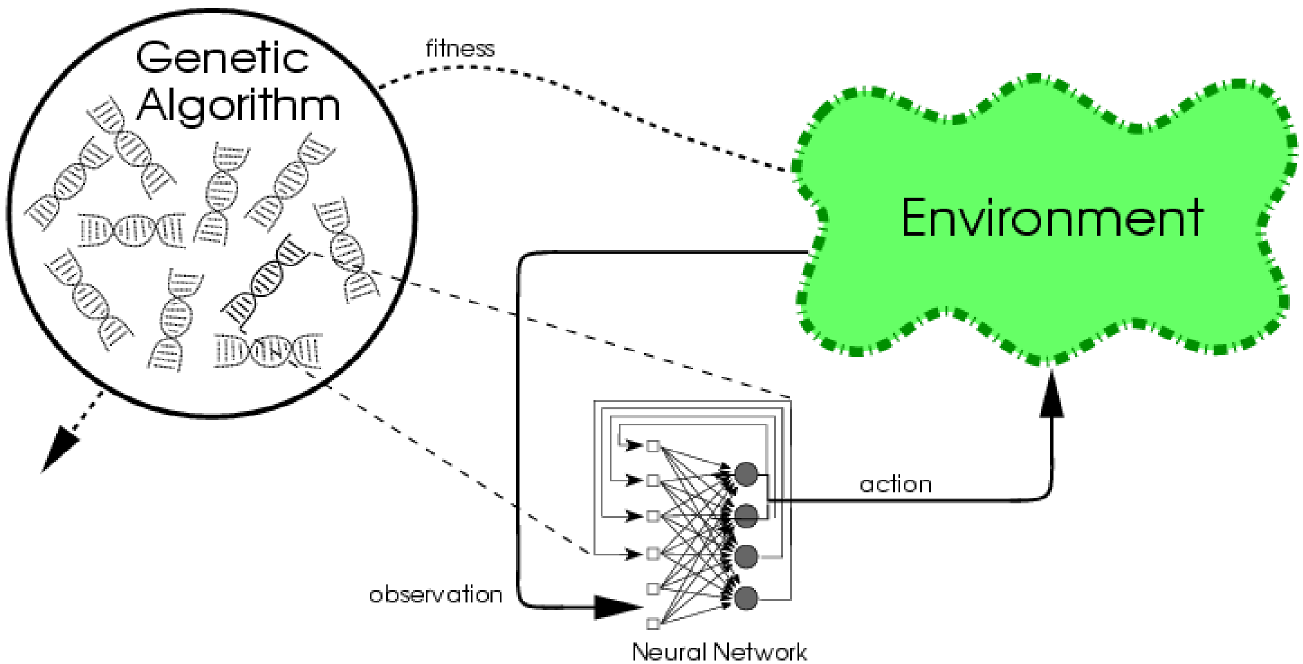
Generation 50



Generation 100

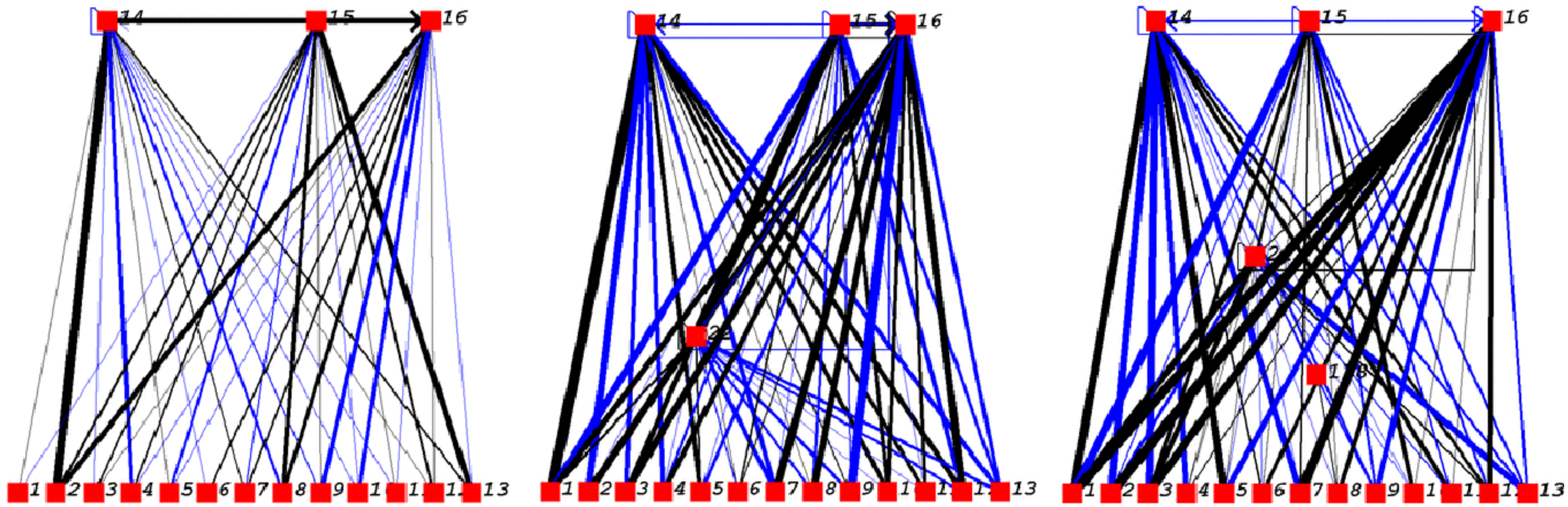
- Evolution encourages diversity automatically
  - Good networks require different kinds of neurons
- Evolution discourages competing conventions
  - Neurons optimized for compatible roles
- Large search space divided into subtasks
  - Optimize compatible neurons

Advanced NE 2: Evolutionary Strategies



- Evolving complete networks with ES (CMA-ES[25])
- Small populations, no crossover
- Instead, intelligent mutations
  - Adapt covariance matrix of mutation distribution
  - Take into account correlations between weights
- Smaller space, less convergence, fewer conventions

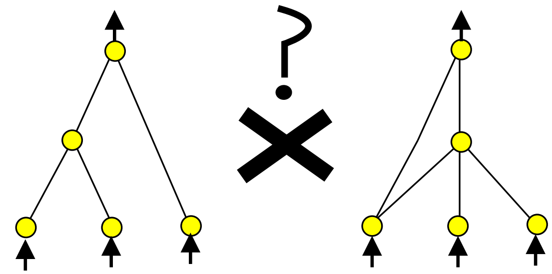
Advanced NE 3: Evolving Topologies



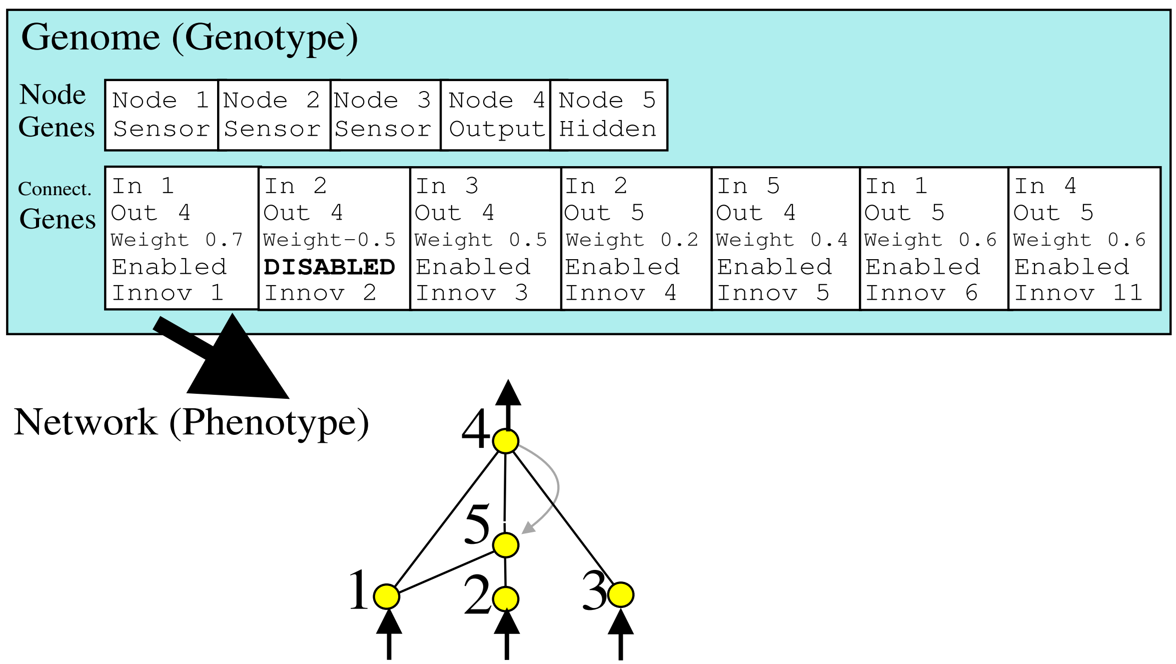
- Optimizing connection weights and network topology [16,67]
- E.g. Neuroevolution of Augmenting Topologies (NEAT [48,52])
- Based on *Complexification*
- Of networks:
  - Mutations to add nodes and connections
- Of behavior:
  - Elaborates on earlier behaviors

How Can Crossover be Implemented?

- Problem: Structures do not match

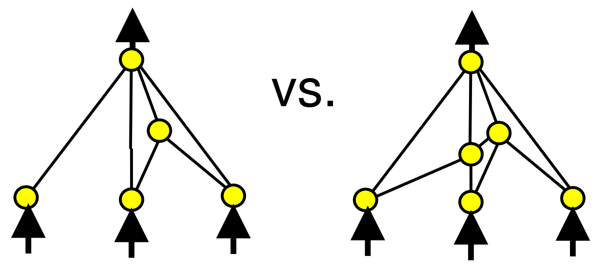


- Solution: Utilize historical markings



How can Innovation Survive?

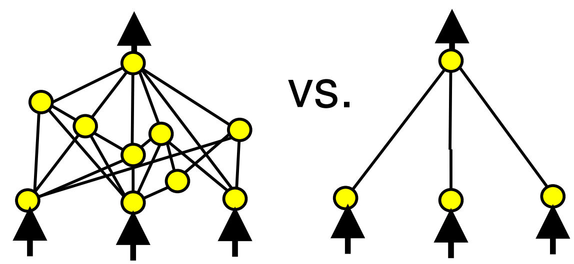
- Problem: Innovations have initially low fitness



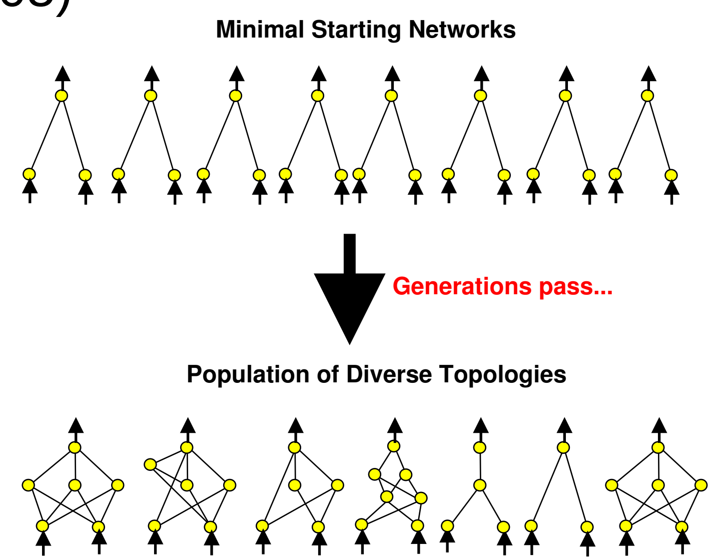
- Solution: Speciate the population
  - Innovations have time to optimize
  - Mitigates competing conventions
  - Promotes diversity

How Can We Search in Large Spaces?

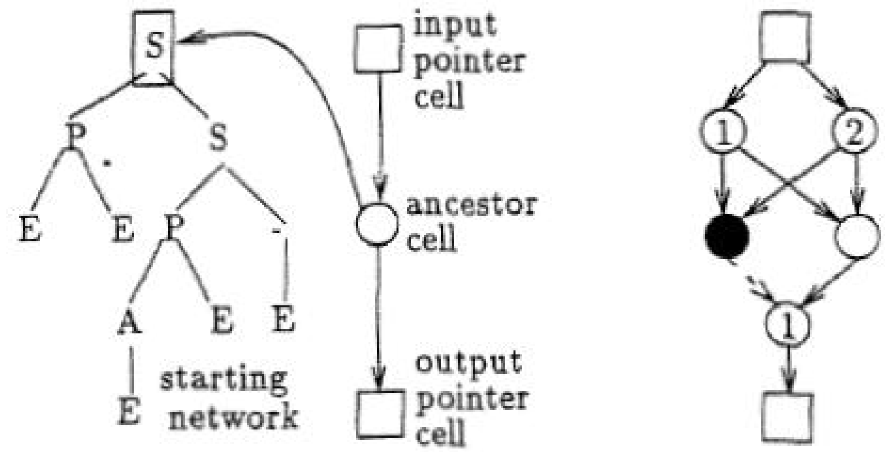
- Need to optimize not just weights but also topologies



- Solution: Start with minimal structure and complexify
  - Hidden nodes, connections, input features
    (Whiteson GECCO'05)

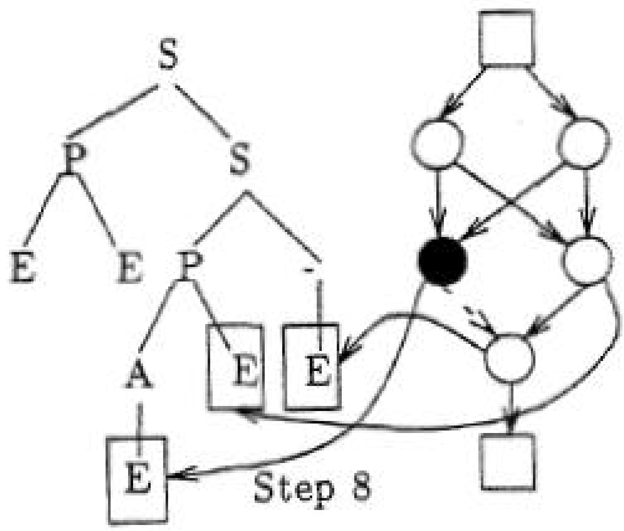


Advanced NE 4: Indirect Encodings



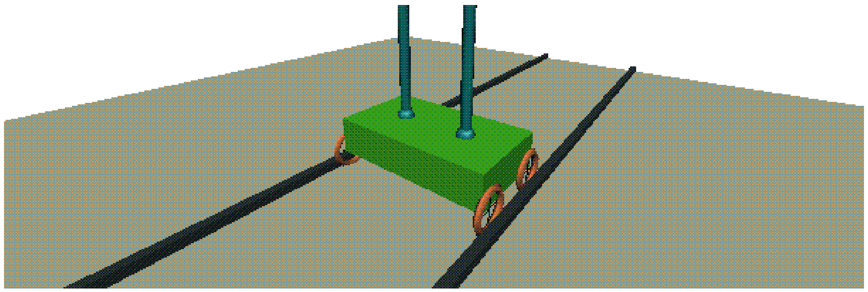
- Instructions for constructing the network evolved
  - Instead of specifying each unit and connection [2,30,46,67]
- E.g. Cellular Encoding (CE [22])
- Grammar tree describes construction
  - Sequential and parallel cell division
  - Changing thresholds, weights
  - A "developmental" process that results in a network

Properties of Indirect Encodings



- Smaller search space
- Avoids competing conventions
- Describes classes of networks efficiently
- Modularity, reuse of structures
  - Recurrency symbol in CE: XOR $\longrightarrow$ parity
  - Useful for evolving morphology
- Not all that powerful (yet)
- Much future work needed
  - More general L-systems
  - Developmental codings, embryogeny

How Do the NE Methods Compare?



Poles	Method	Evals
Two-1	CE	(840,000)
	CNE	87,623
	ESP	26,342
	NEAT	24,543
Two-2	CMA-ES	6,061 - 25,254
	NEAT	6,929

Two poles, no velocities, 2 different setups:

- Advanced methods better than CNE
- Advanced methods about equal
- Indirect encodings future work
- DEMO

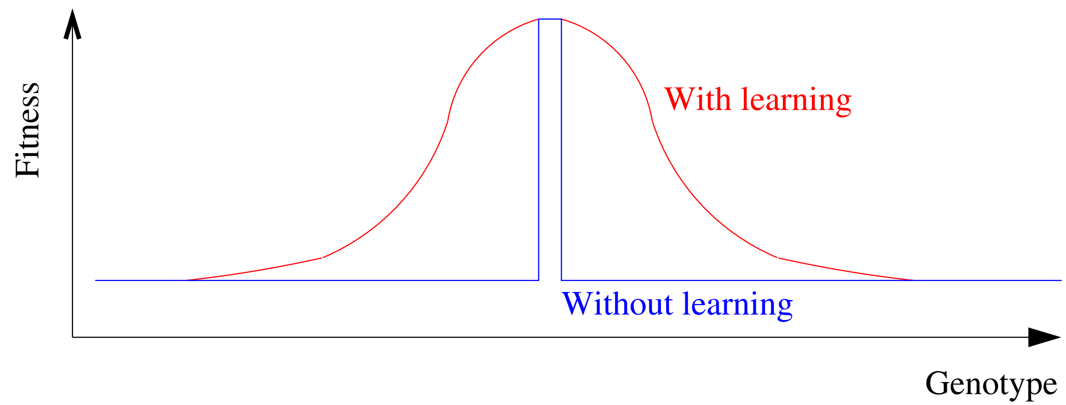
Further NE Techniques

- Incremental evolution [19,57,66]
- Utilizing population culture [4,29]
- Evolving ensembles of NNs [26,40,63]
  (Pardoe GECCO'05)
- Evolving neural modules [42]
- Evolving transfer functions and learning rules [6,43,56]
- Combining learning and evolution

Combining Learning and Evolution

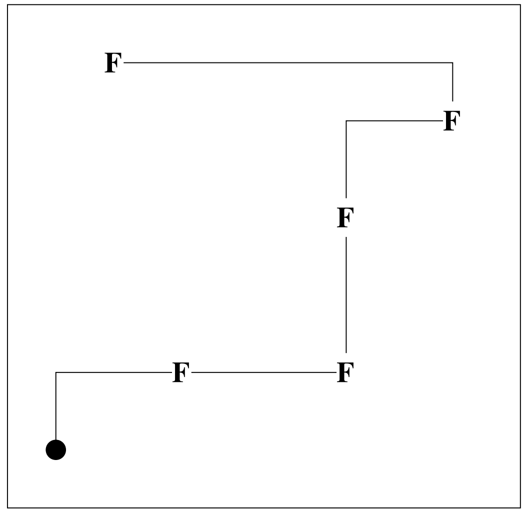
- Good learning algorithms exist for NN
  - Why not use them as well?
- Evolution provides structure and initial weights
- Fine tune the weights by learning
- Lamarckian evolution is possible
  - Coding weight changes back to chromosome
- Difficult to make it work
  - Diversity reduced; progress stagnates

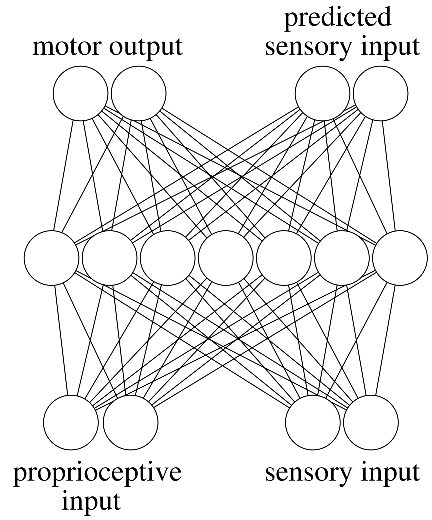
Baldwin Effect



- Learning can guide Darwinian evolution[3,23]
  - Makes fitness evaluations more accurate
- With learning, more likely to find the optimum if close
- Can select between good and bad individuals better
  - Lamarckian not necessary
- How can we implement it?
  - How to obtain training targets?

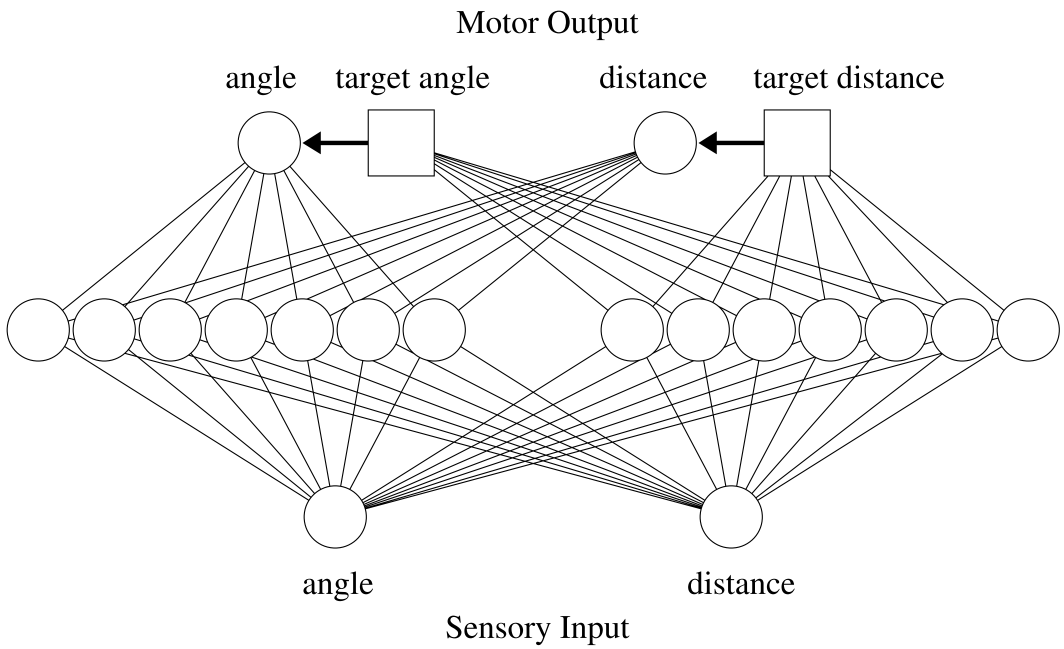
Targets from a Related Task





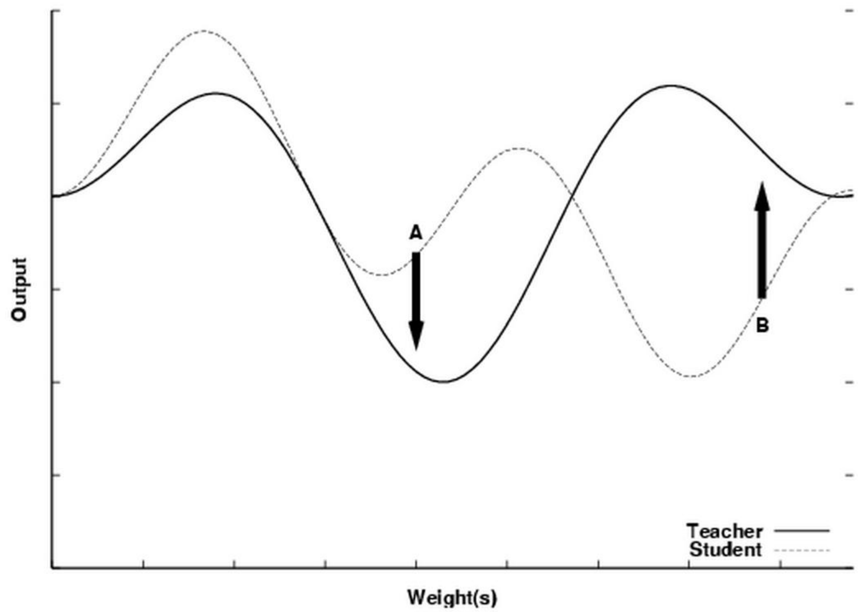
- Learning in a related task is sufficient
- E.g. foraging for food in a microworld[37]
  - Network sees the state, outputs motor commands
  - Trained with backprop to predict the next input
  - Training emphasizes useful hidden-layer representations
  - Allows more accurate evaluations

Evolving the Targets



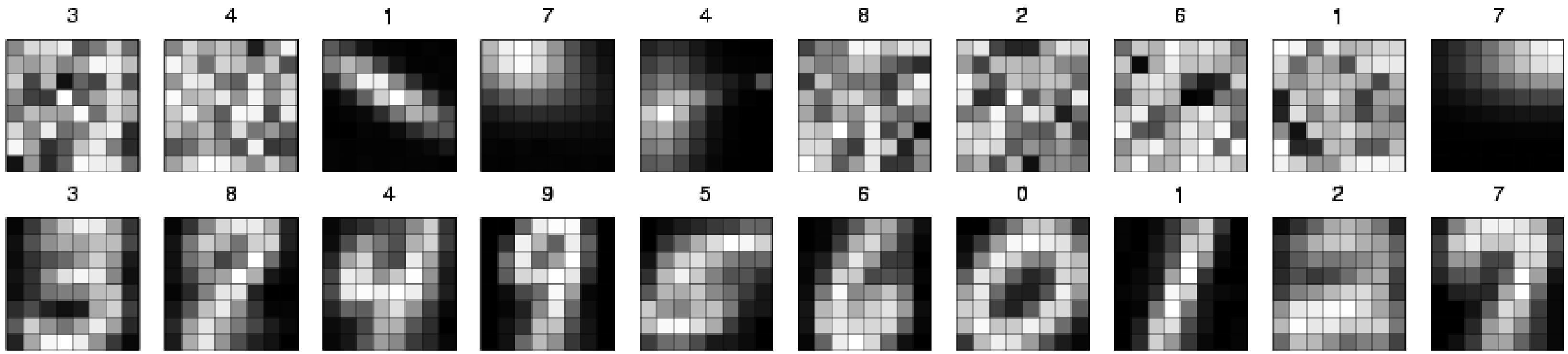
- Evolve extra outputs to provide targets
- E.g. in the foraging task[39]
  - Motor outputs and targets with separate hidden layers
  - Motor weights trained with backprop, targets evolved
  - Targets do not correspond to optimal performance: Direct system towards useful learning experiences

Targets from the Population



- Train new offspring to imitate parents/champion
  - Trained in population "culture"
- Local search around good individuals
  - Limited training: 8-20 backprop iterations
- Becomes part of the evaluation
  - Individuals evolve to anticipate training
  - Perform poorly at birth, well after training
- Evolution discovers optimal starting points for learning!

No Targets: Unsupervised Learning

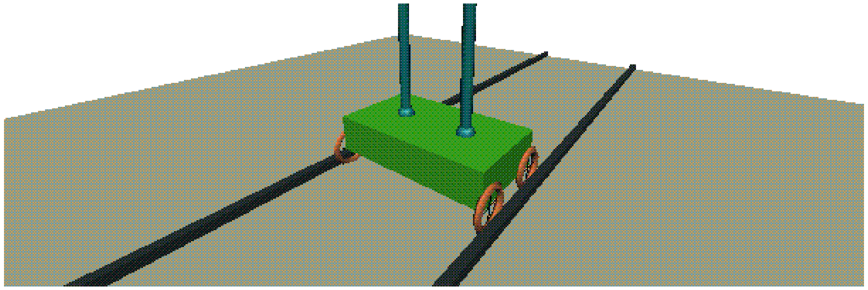


- Hebbian adaptation during performance[13,50]
- E.g. handwritten character recognition[60] (Valsalam GECCO'05)
  - Evolution determines the starting point
  - Competitive learning finishes the design
- Starting points are poor recognizers
  - Only bias learning away from local minima
- Synergetic effect: Evolution utilizes learning
- Future work: Constructing developmental systems

Extending NE to Applications

- Evolving composite decision makers [63]
- Evolving teams of agents [5,49,68]
- Utilizing coevolution [54]
- Real-time neuroevolution [49]
- Combining human knowledge with evolution [11]

Applications to Control



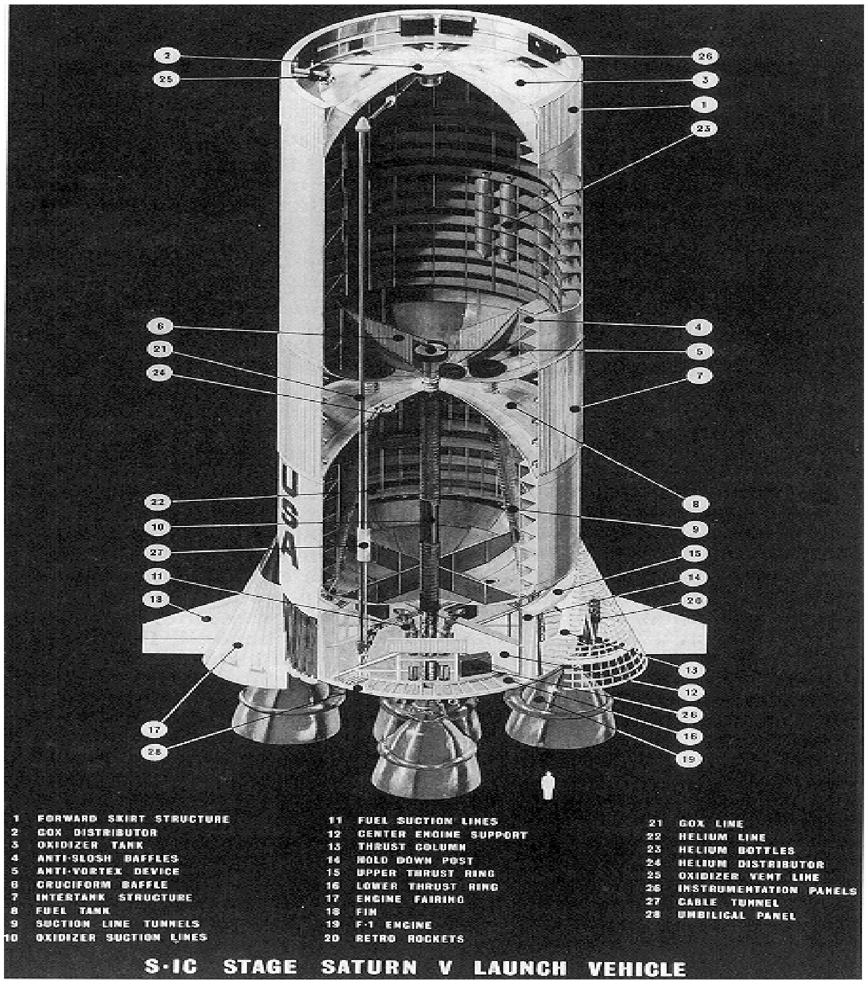
- Pole-balancing benchmark
  - Originates from the 1960s
  - Original 1-pole version too easy
  - Several extensions: acrobat, jointed, 2-pole, particle chasing [40]
- Good surrogate for other control tasks
  - Vehicles and other physical devices
  - Process control [58]

Controlling a Finless Rocket

Task: Stabilize a finless version of the Interorbital Systems RSX-2 sounding rocket[20]

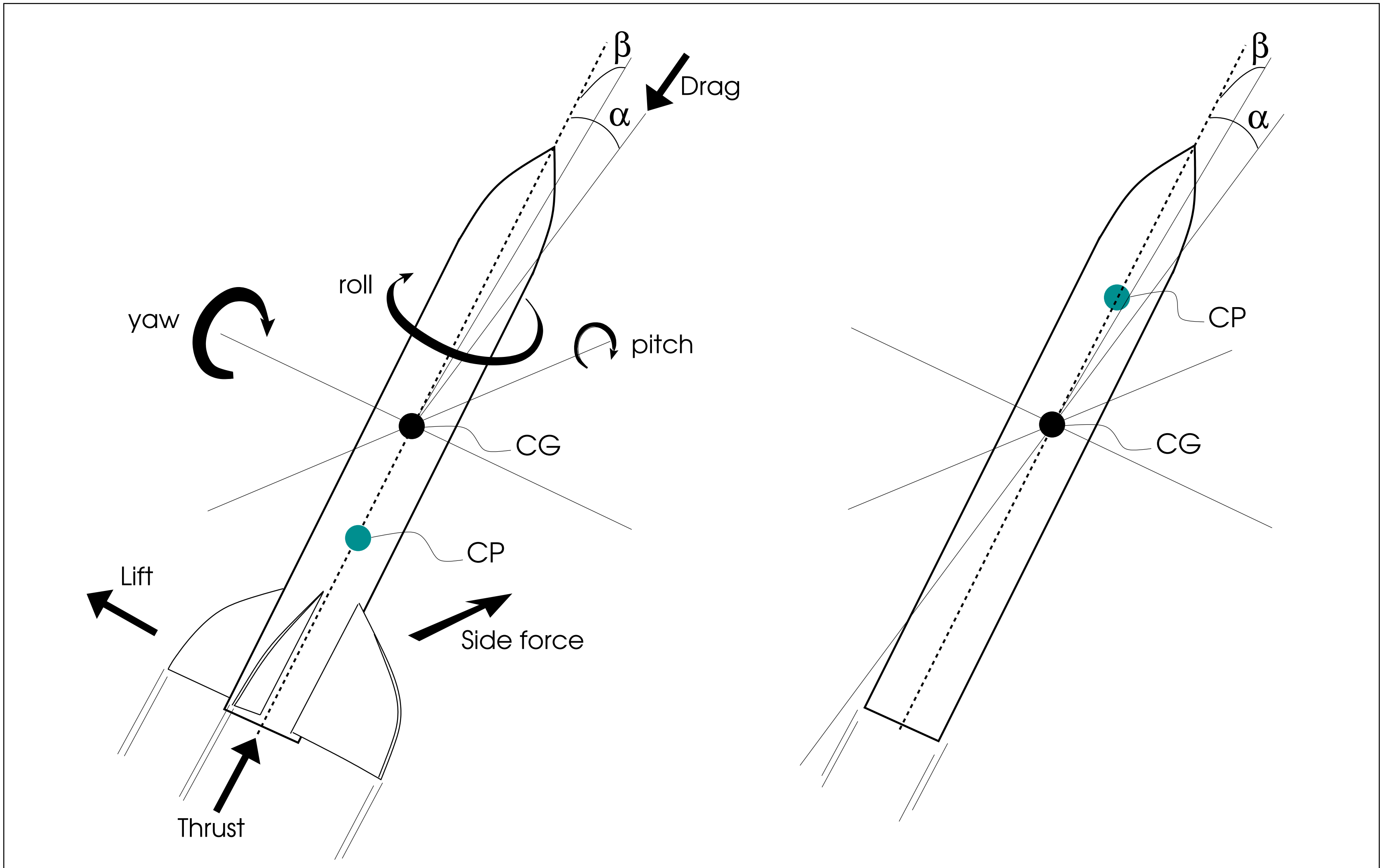
- Scientific measurements in the upper atmosphere
- 4 liquid-fueled engines with variable thrust
- Without fins will fly much higher for same amount of fuel

Active Rocket Guidance



- Used on large scale launch vehicles (Saturn, Titan)
- Typically based on classical linear feedback control
- High level of domain knowledge required
- Expensive, heavy

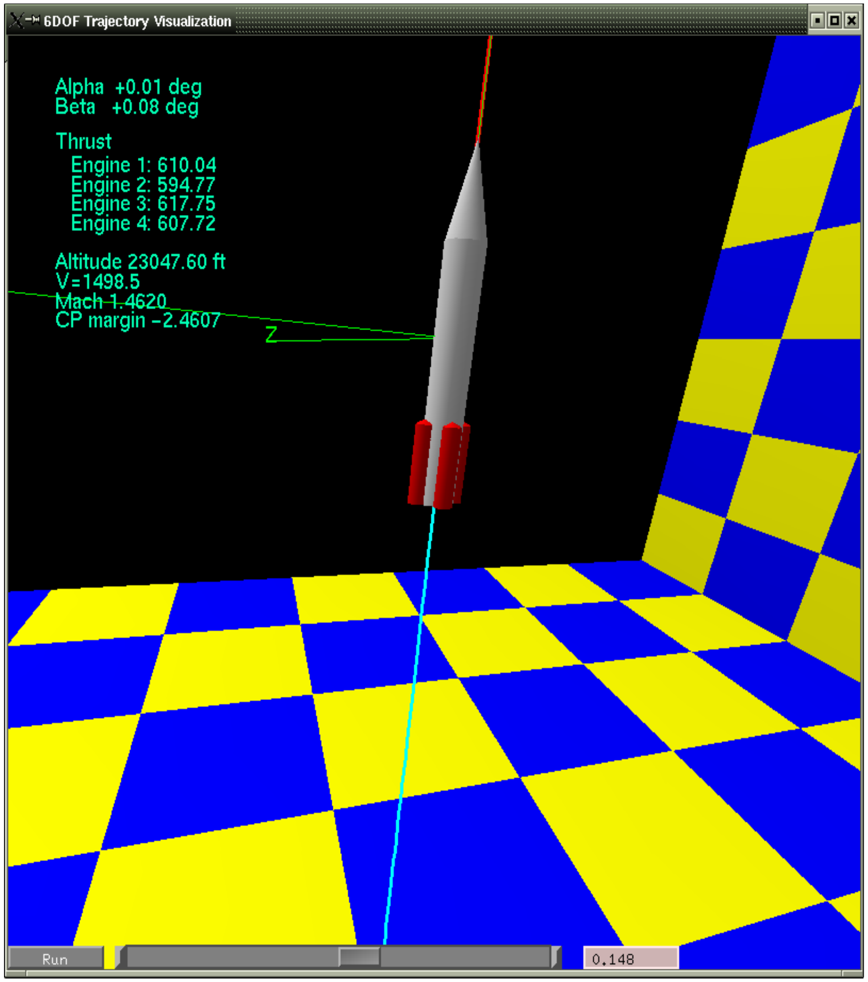
Rocket Stability



(a) Fins: stable

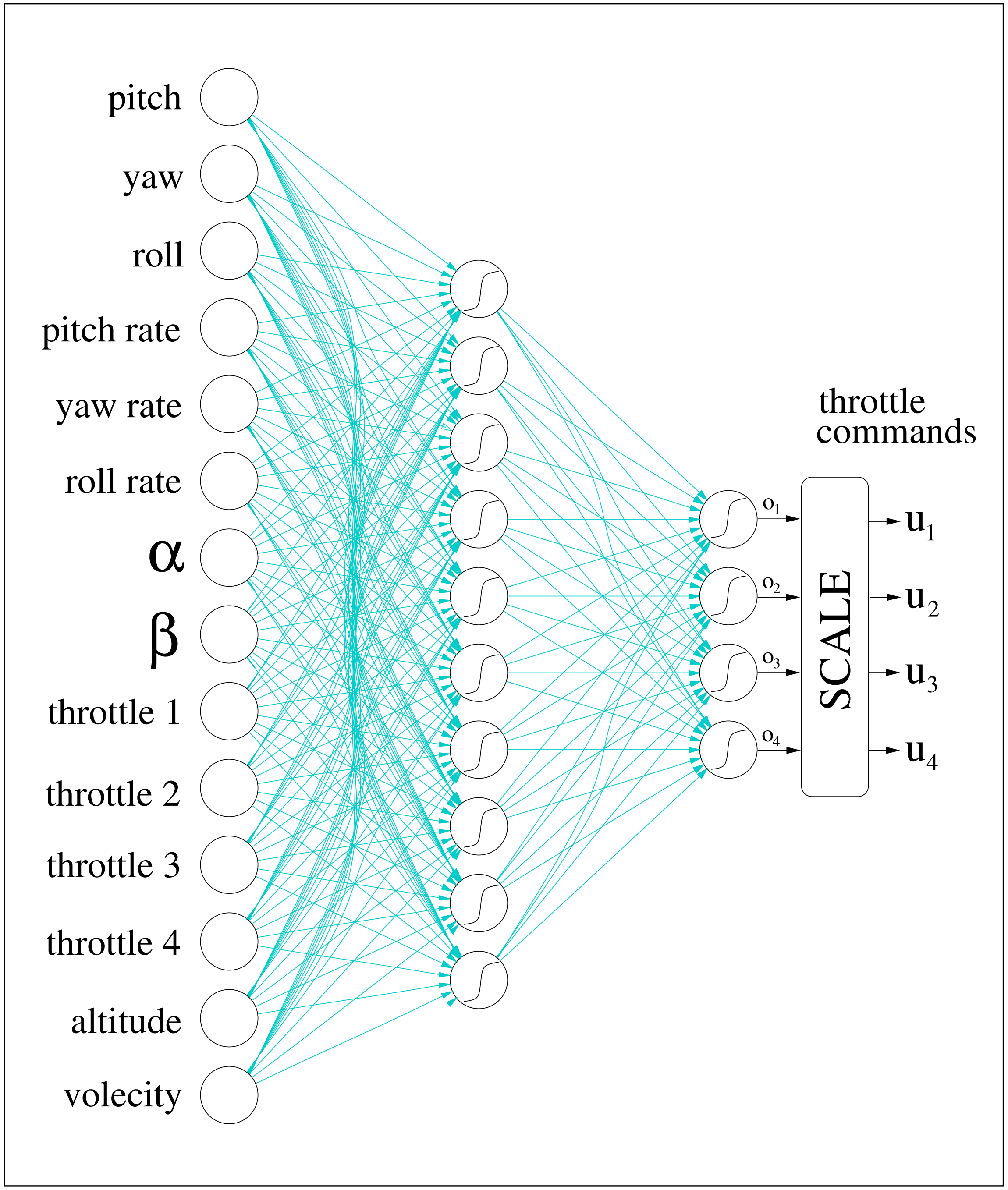
(b) Finless: unstable

Simulation Environment: JSBSim

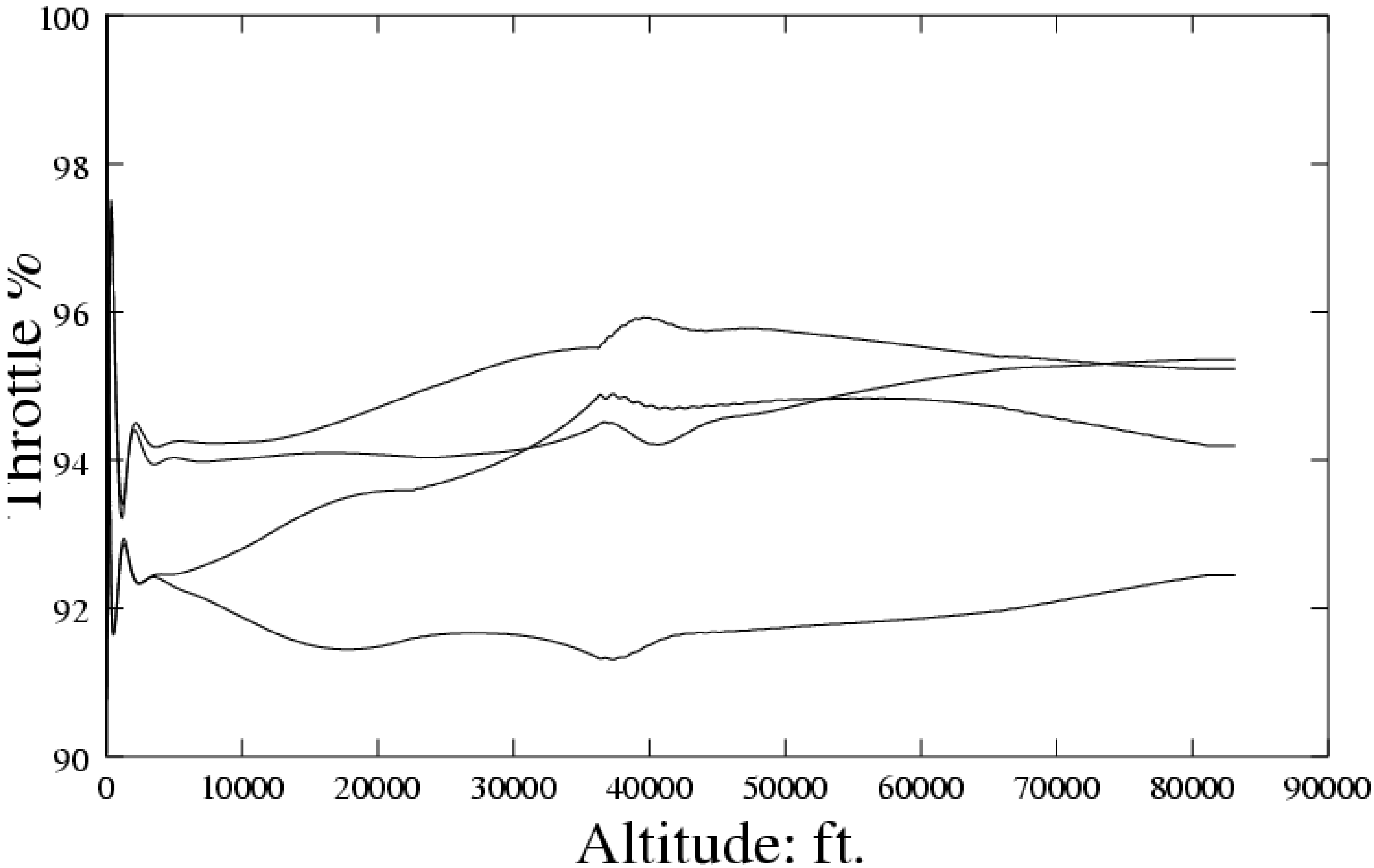


- General rocket simulator
- Models complex interaction between air-frame, propulsion, aerodynamics, and atmosphere
- Used by IOS in testing their rocket designs
- Accurate geometric model of the RSX-2

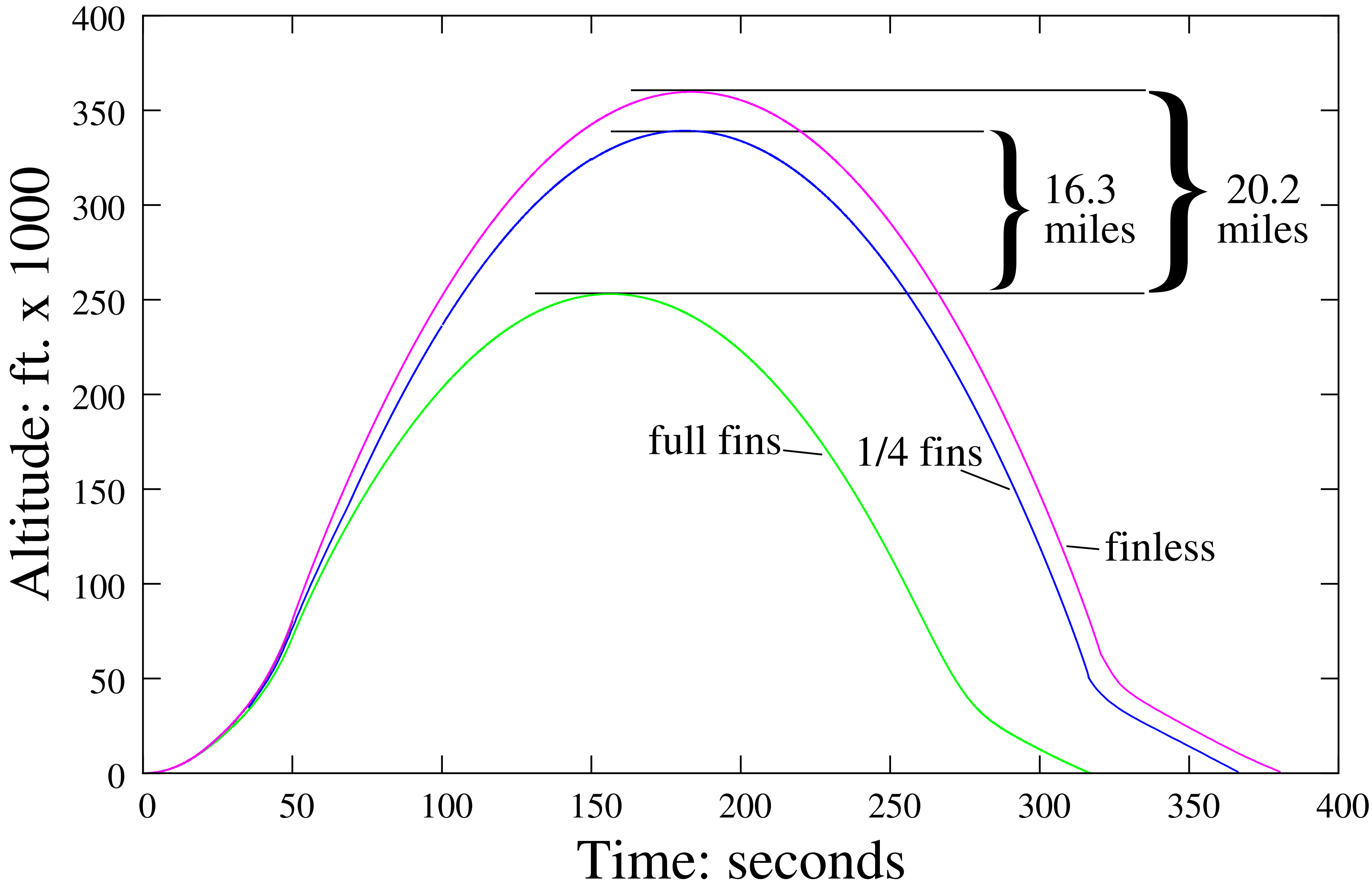
Rocket Guidance Network



Results: Control Policy



Results: Apogee

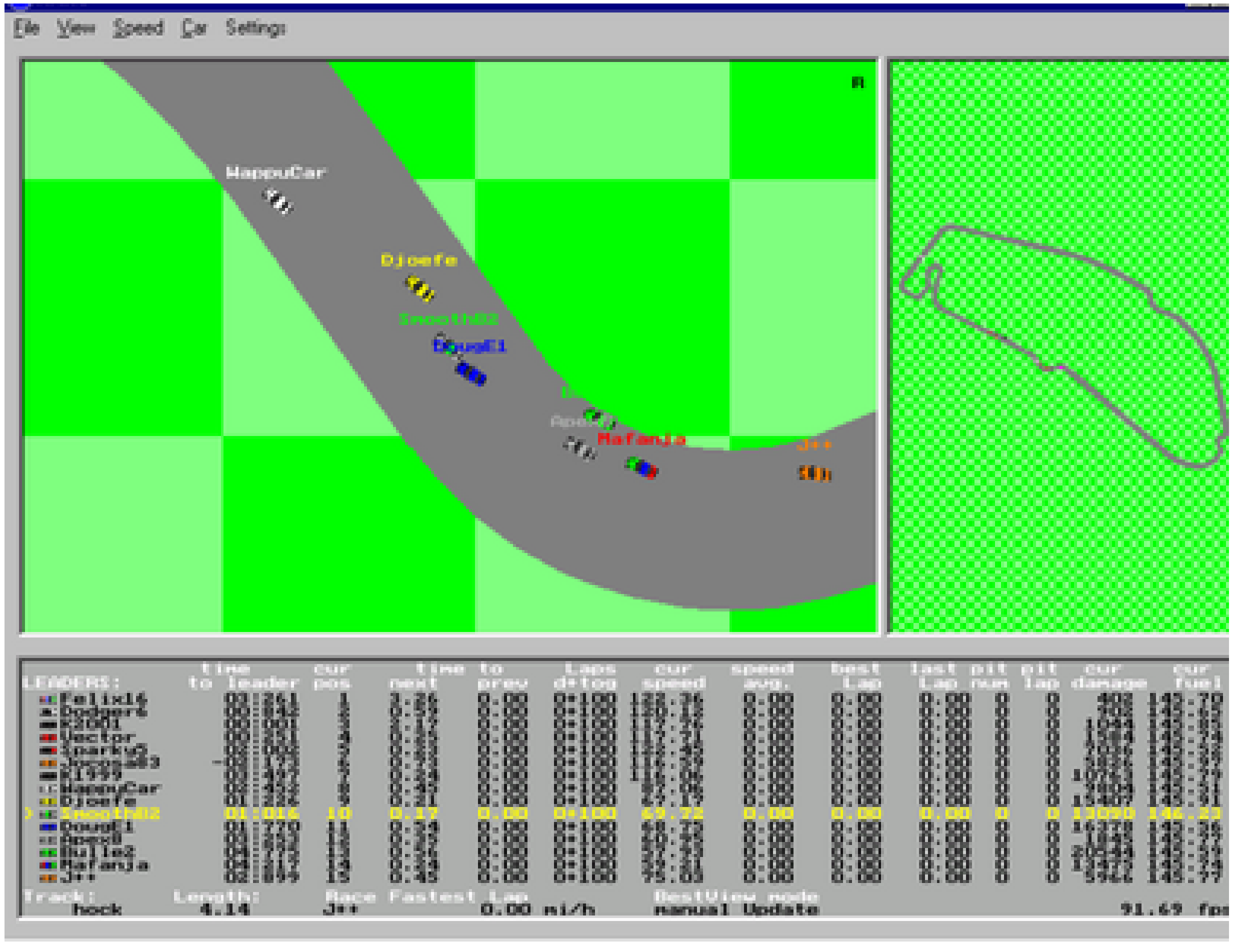


- DEMO

Driving and Collision Warning

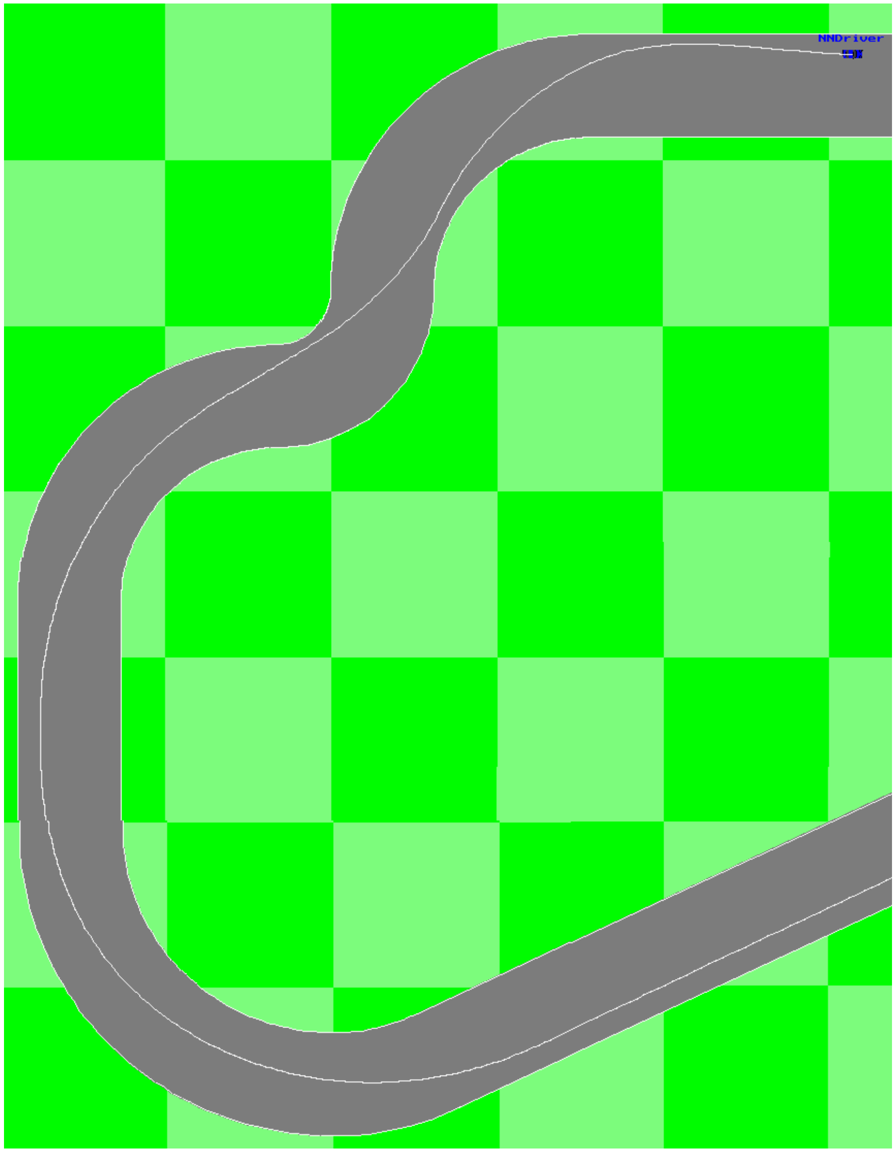
- Goal: evolve a collision warning system
  - Looking over the driver's shoulder
  - Adapting to drivers and conditions
  - Collaboration with Toyota (Stanley GECCO'05)

The RARS Domain



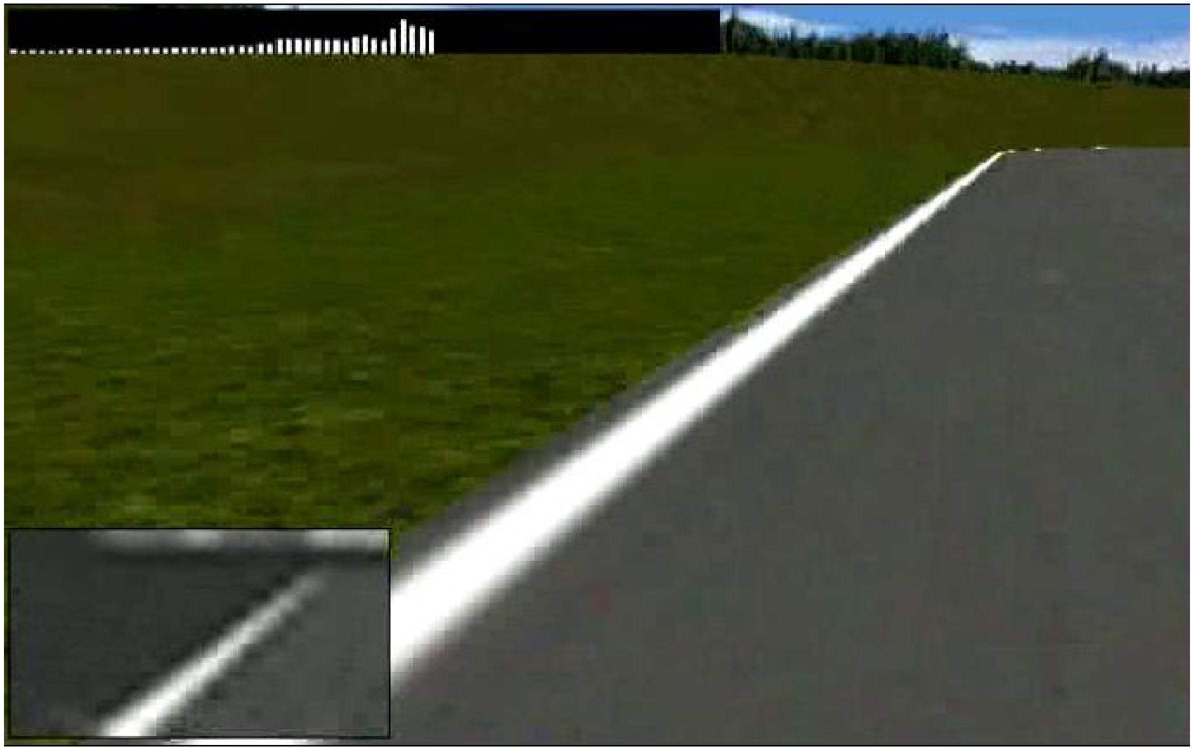
- RARS: Robot Auto Racing Simulator
  - Internet racing community
  - Hand-designed cars and drivers
  - First step towards real traffic

Evolving Good Drivers



- Evolving to drive fast without crashing (off road, obstacles)
- Discovers optimal driving strategies (e.g. how to take curves)
- Works from range-finder & radar inputs
- Works from raw visual inputs
- DEMO

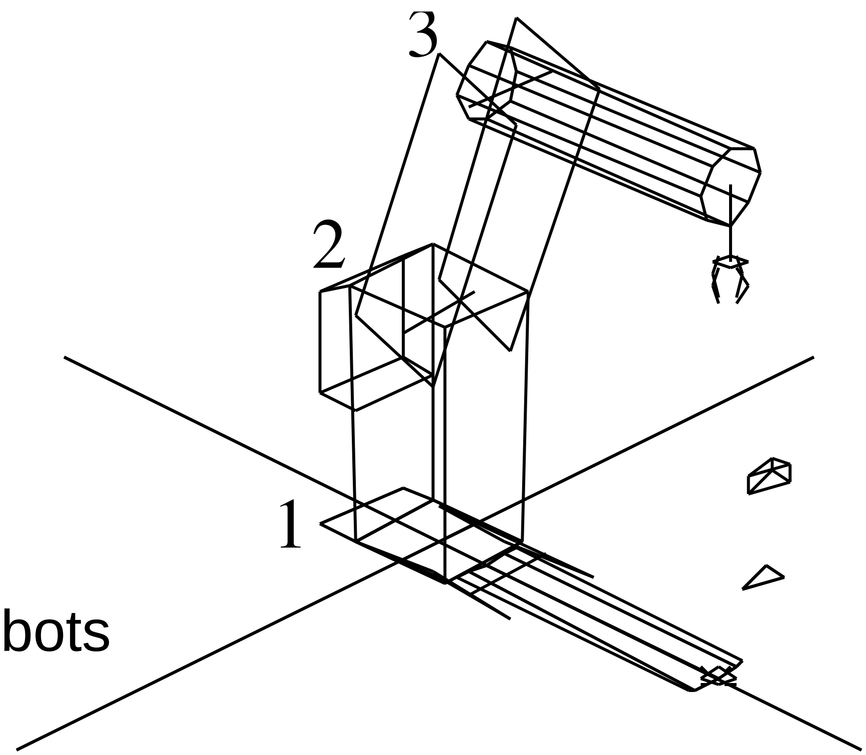
Evolving Warnings



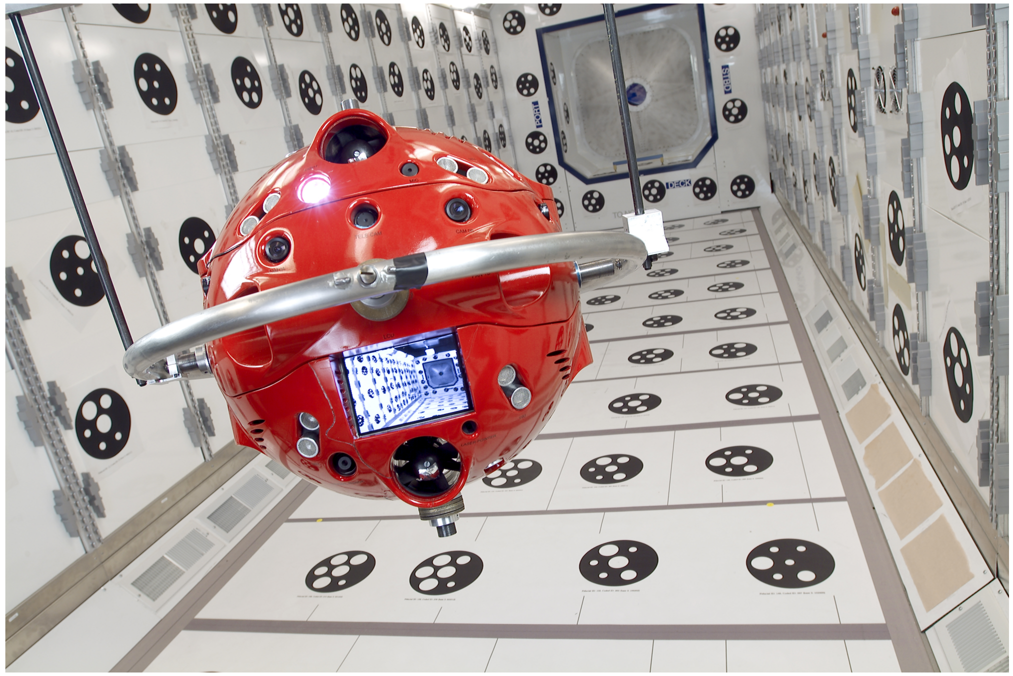
- Evolving to estimate probability of crash
- Predicts based on subtle cues (e.g. skidding off the road)
- Compensates for disabled drivers
- Human drivers learn to drive with it!
- DEMO

Applications to Robotics

- Controlling a robot arm[34]
  - Compensates for an inop motor
- Robot walking[24,45]
  - Various physical platforms
- Mobile robots[9,12,38]
  - Transfers from simulation to physical robots
  - Evolution possible on physical robots

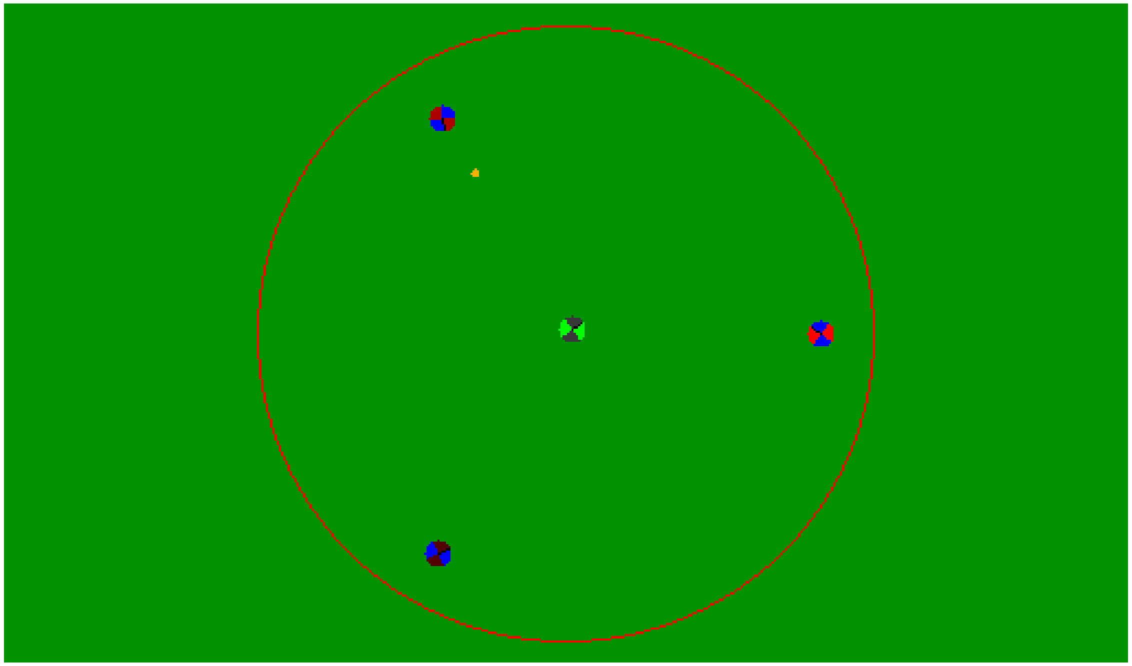


Personal Satellite Assistant



- Floating robot assistant to an astronaut
  - Needs to stay close but not crash
  - Two thrusters: Difficult to control
- Novel control strategies can be evolved
  - Stop on a spot by making a circle! 47 (Sit GECCO'05)
- DEMO

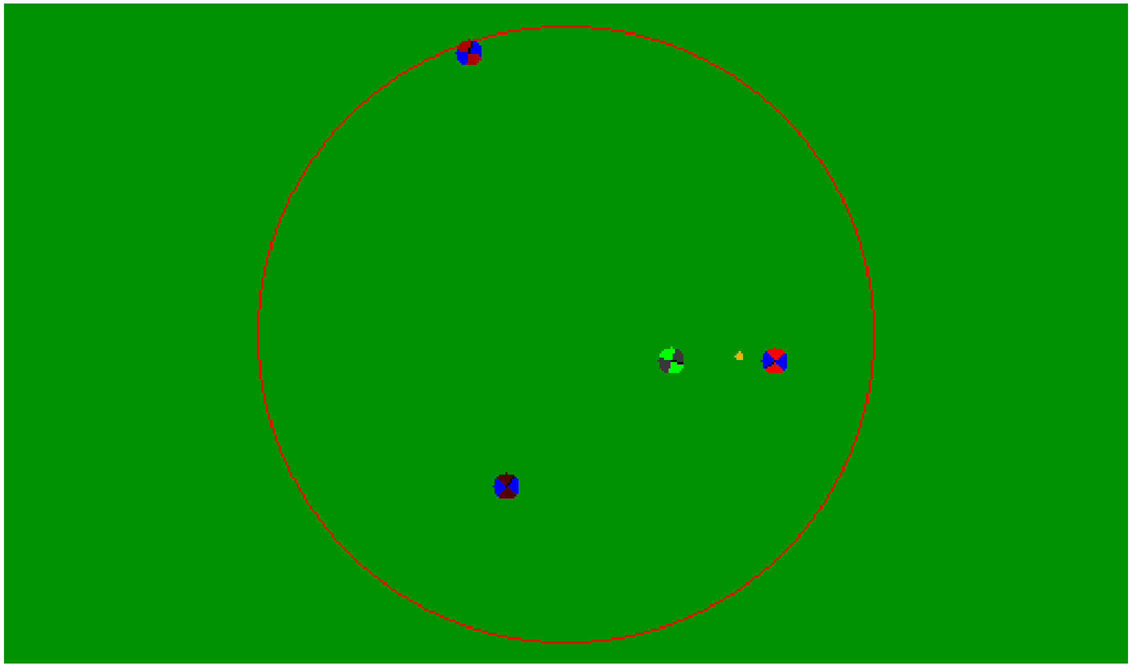
Robotic Soccer



- E.g. robocup soccer “Keepaway” task[63]
- Three keepers, one (algorithmic) taker
- Includes many behaviors:

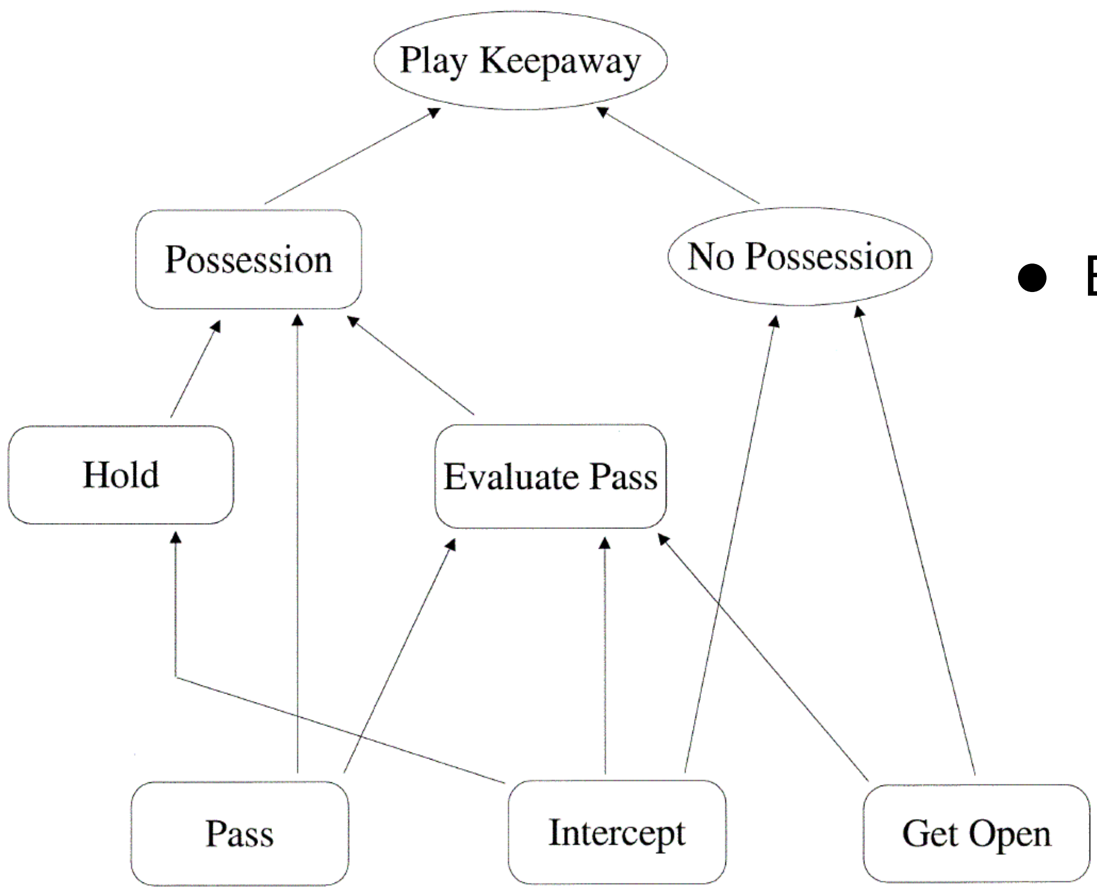
  Get-Open, Intercept, Evaluate-Pass, Pass...

Direct Evolution



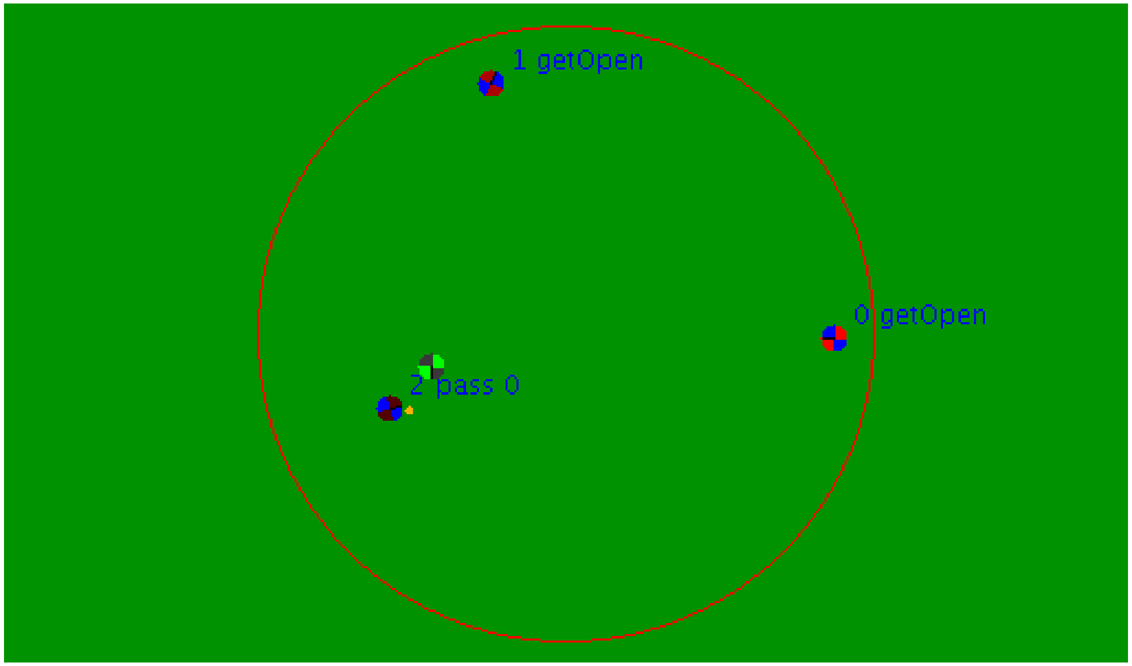
- Mapping sensors directly to actions
  - Difficult to separate behaviors
  - Ineffective combinations evolve
- DEMO

Cooperative Coevolution



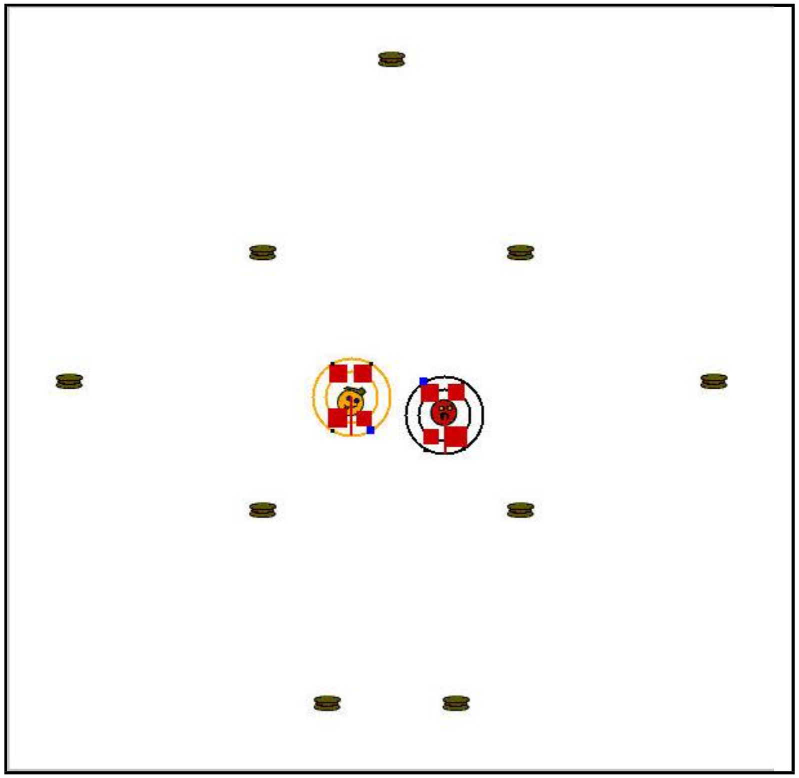
- Evolve multiple actions
  - Each one in a separate network
  - Decision tree to decide on actions
  - Or evolve a decision network

Cooperative Coevolution (2)



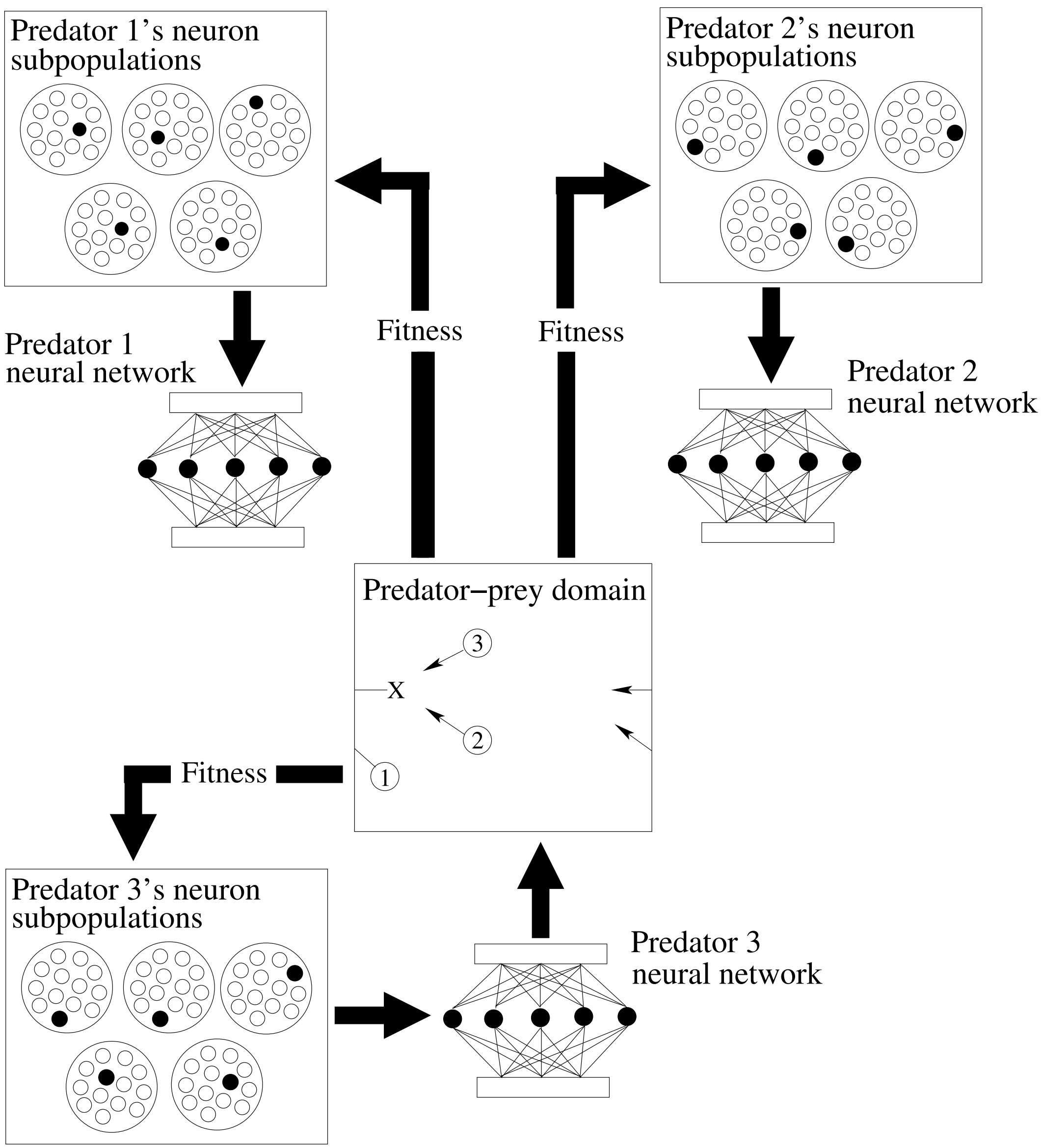
- Networks learn individual tasks
- Learn to anticipate other tasks
  - Lining up for a pass
- Cooperative coevolution of composite behavior
- DEMO

Applications to Artificial Life



- Gaining insight into neural structure
  - E.g. evolving a command neuron [44]
- Emergence of behaviors
  - Signaling, herding, hunting... [61,62,68]
- Future challenges
  - Emergence of language
  - Emergence of community behavior

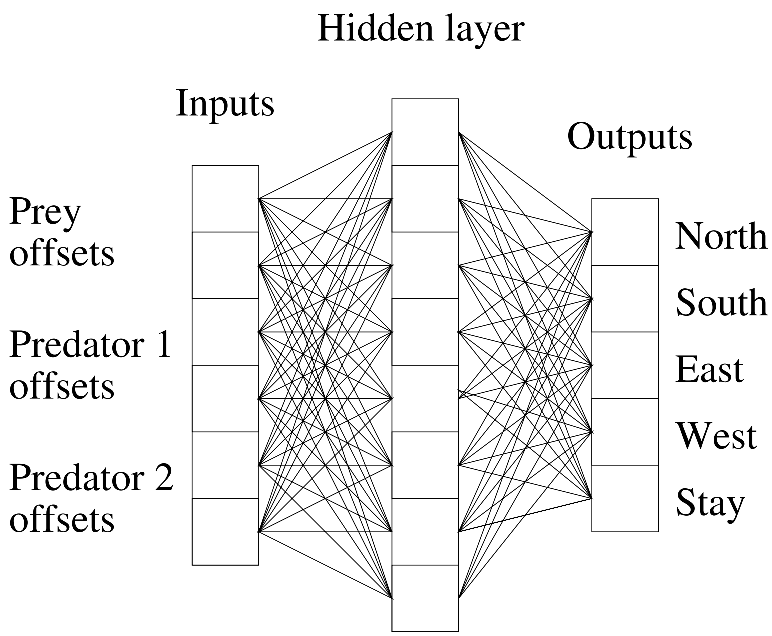
Emergence of Cooperation

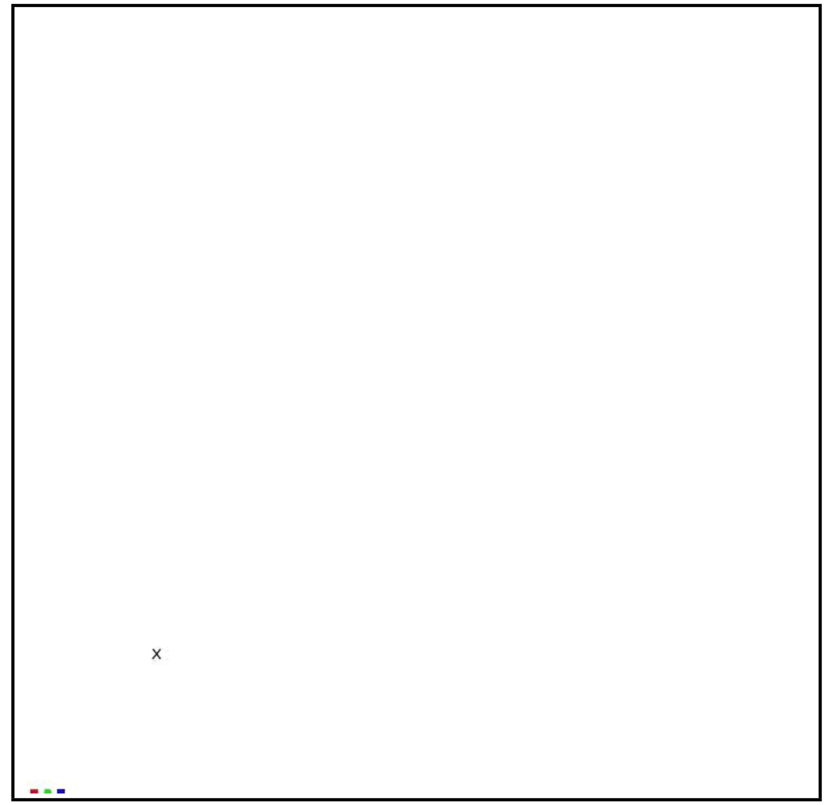


Multi-Agent ESP[68]

- Natural extension of ESP to multiple networks
- Each network constructed from its own subpopulations
- Example: A team catching a fast prey
  - 3 predators, toroidal world
  - Prey as fast, runs away from nearest agent
  - Need to coordinate an approach

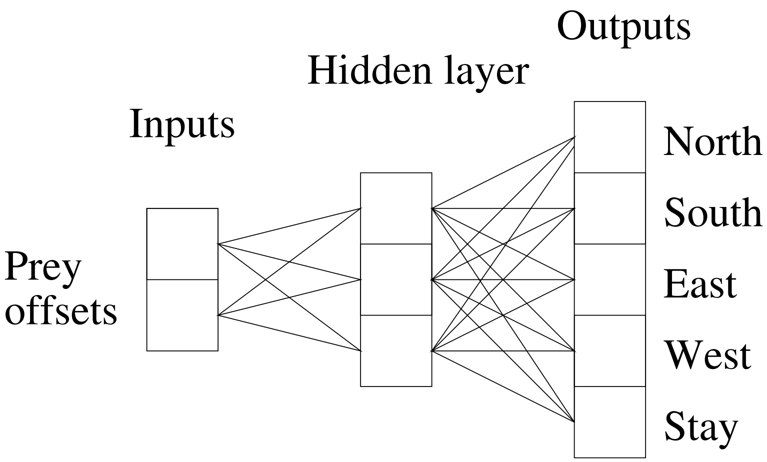
Communication-based Cooperation

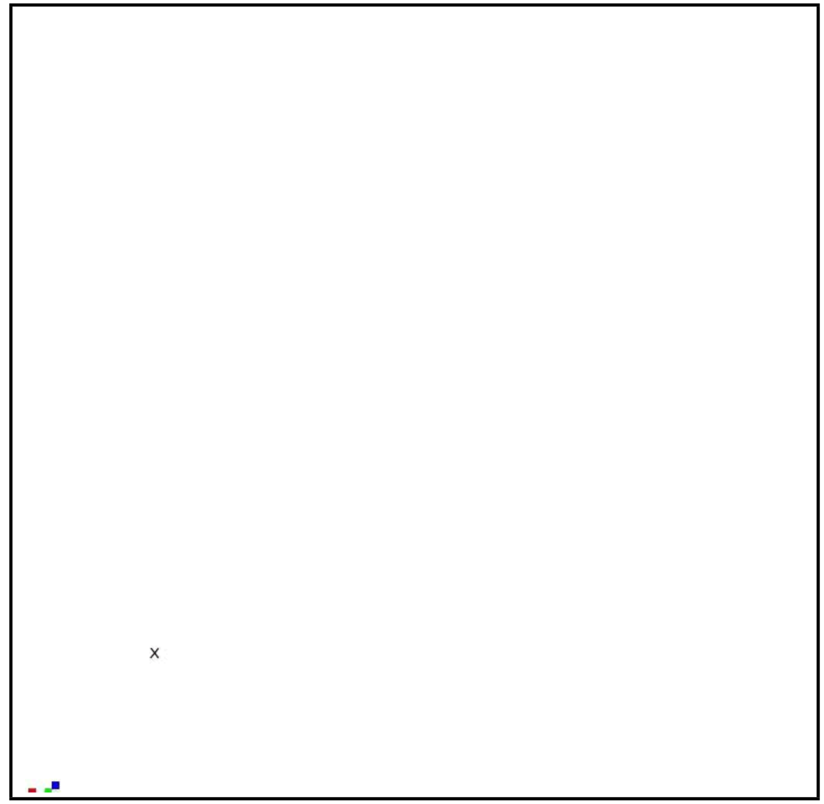




- Individual controllers for each agent
  - Observe the prey and the other predators
  - Develop flexible roles
- Distributed control works better than central control
  - Subtasking through global fitness

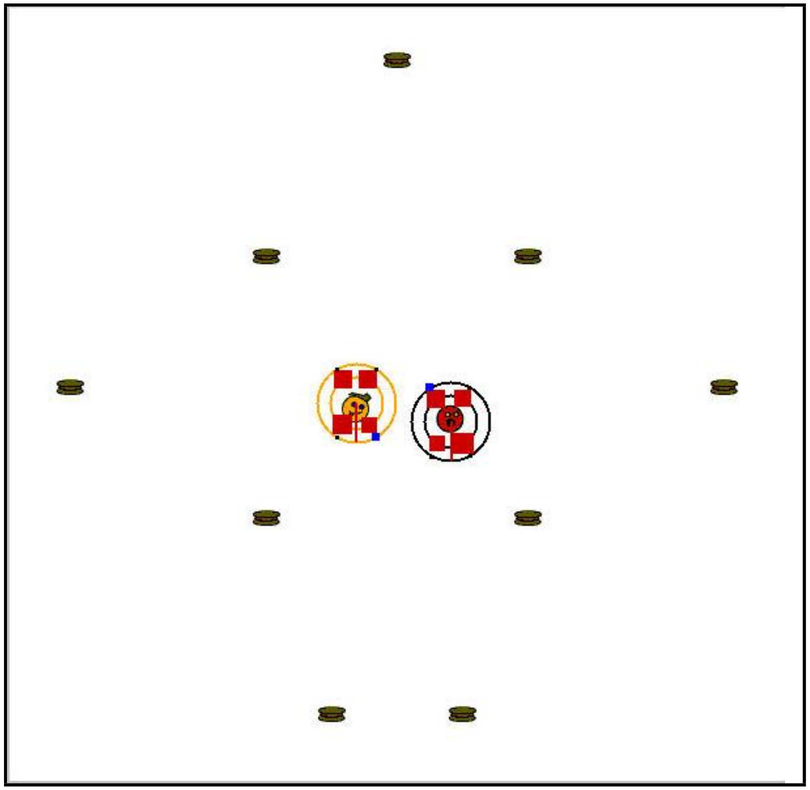
Role-Based Cooperation





- Each controller only sees the prey
  - Coordination through stigmergy
  - Develop efficient roles
- More effective than communication-based
  - Works like a well-practiced soccer team!
- Multiagent NE powerful in discovering team behaviors

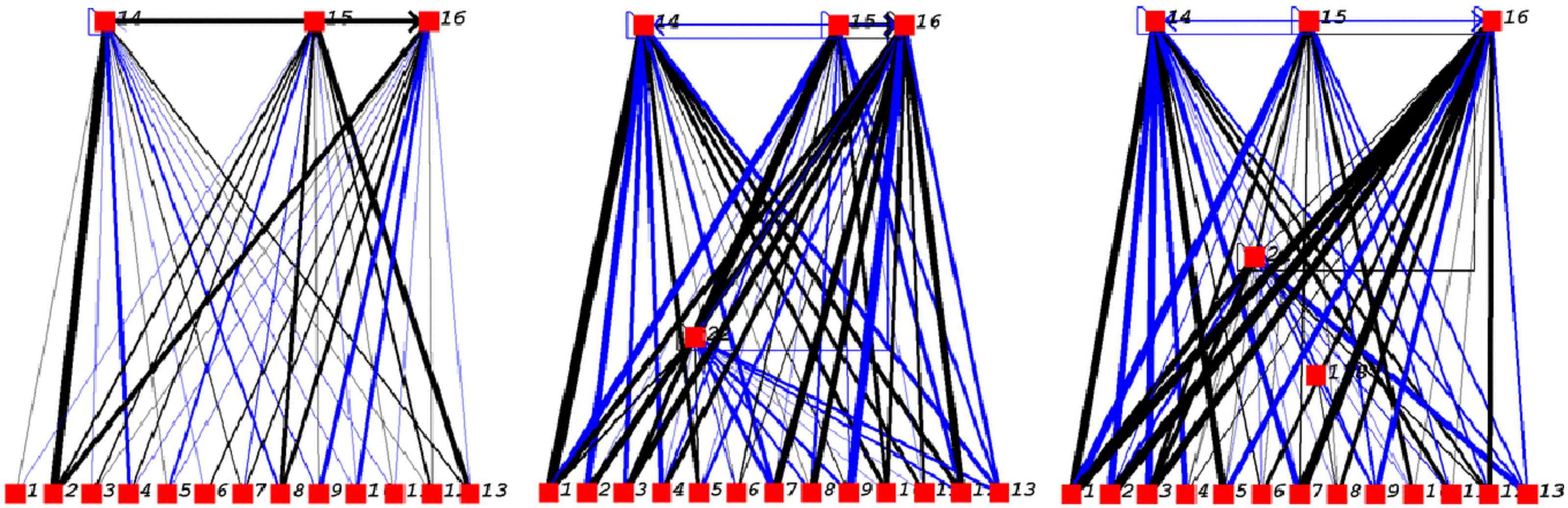
Competitive Coevolution



- Evolution requires an opponent to beat
- Such opponents are not always available

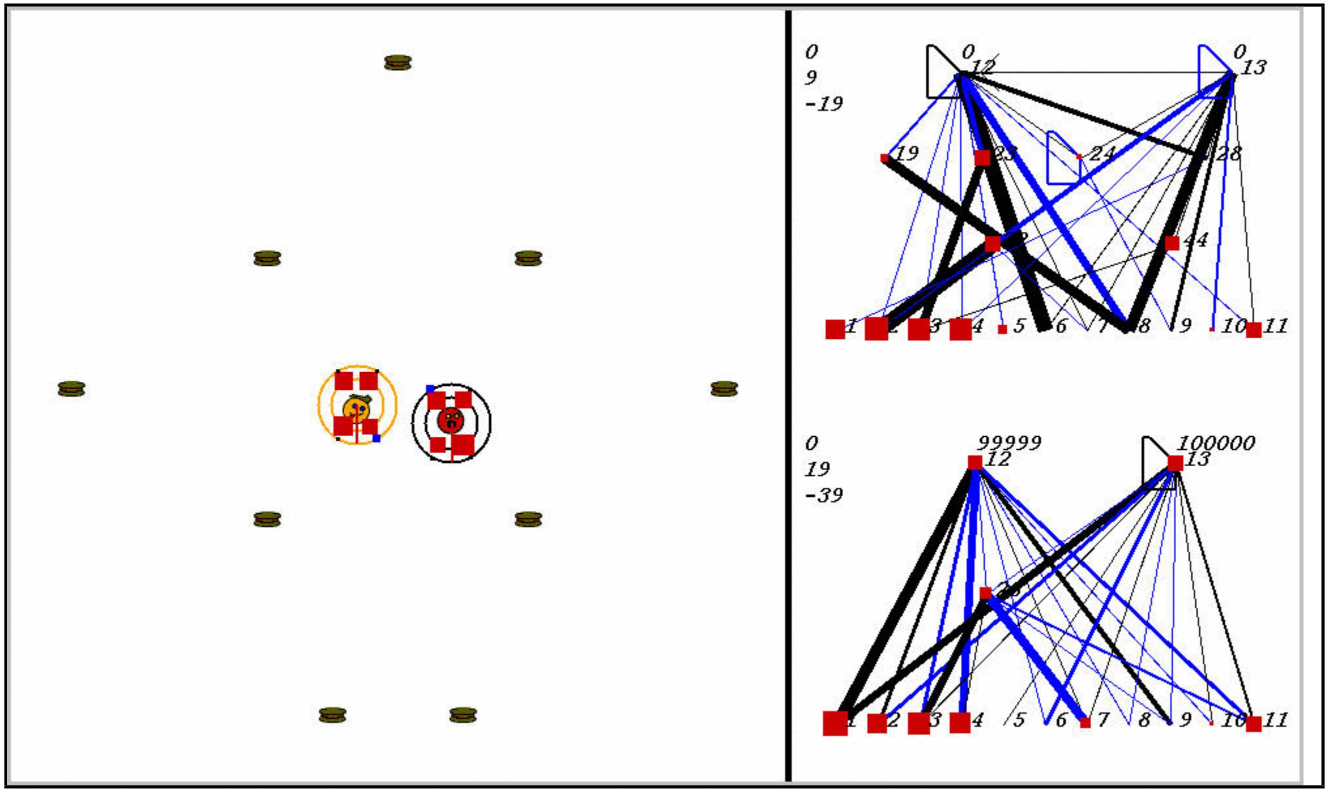
- Co-evolve two populations to outdo each other
- How to maintain an arms race?

Competitive Coevolution with NEAT



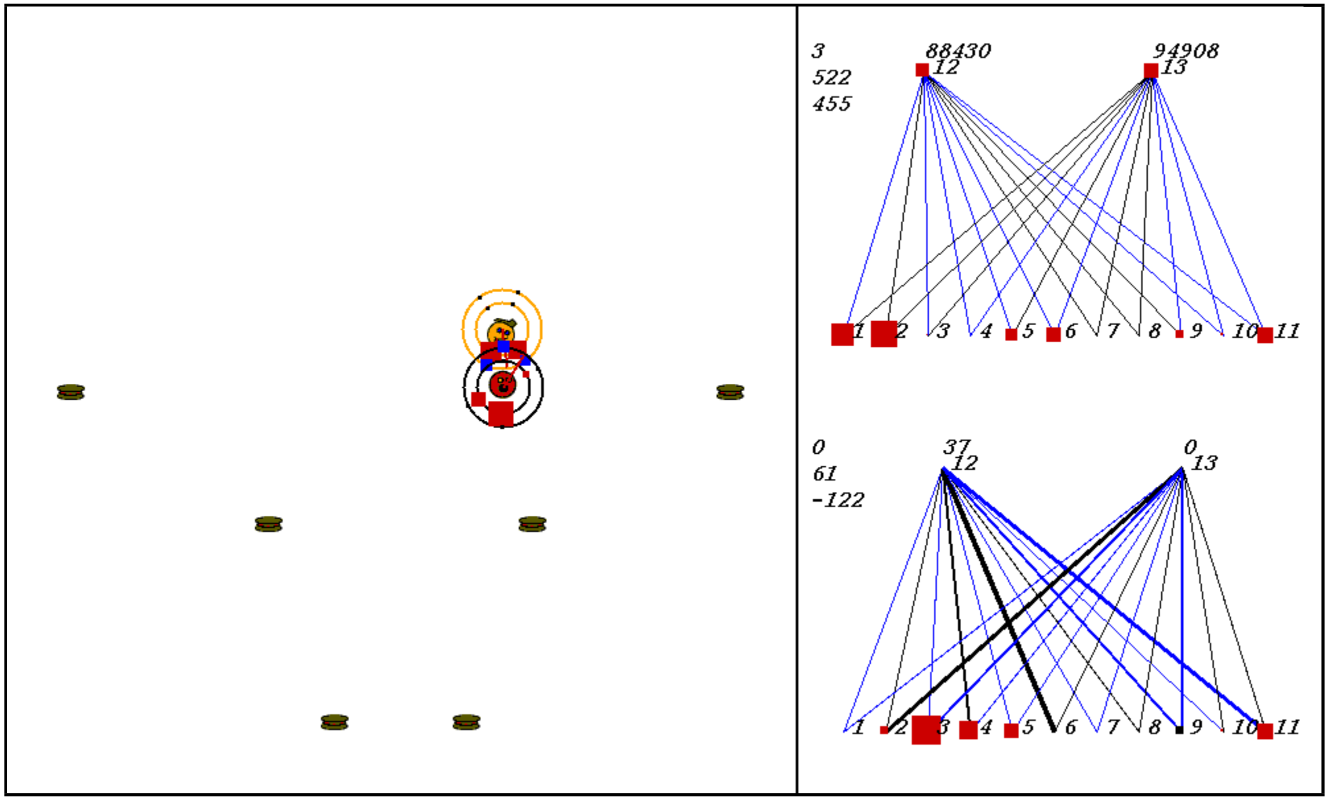
- Complexification elaborates instead of alters
  - Adding more complexity to existing behaviors
- Can establish a coevolutionary arms race
  - Two populations continually outdo each other
  - Absolute progress, not just tricks

Robot Duel Domain



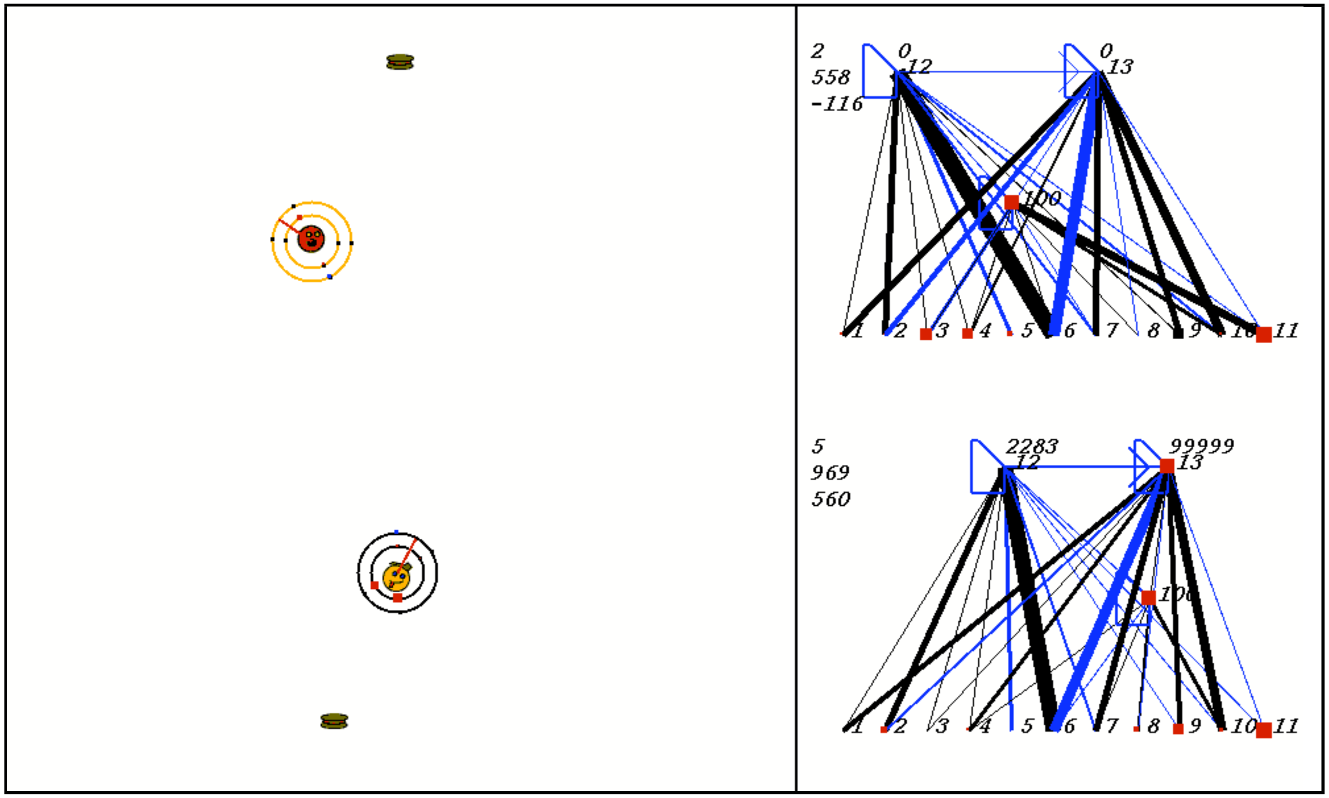
- Two Khepera-like robots forage, pursue, evade
  - Collect food to gain energy
  - Win by crashing to a weaker robot

Early Strategies



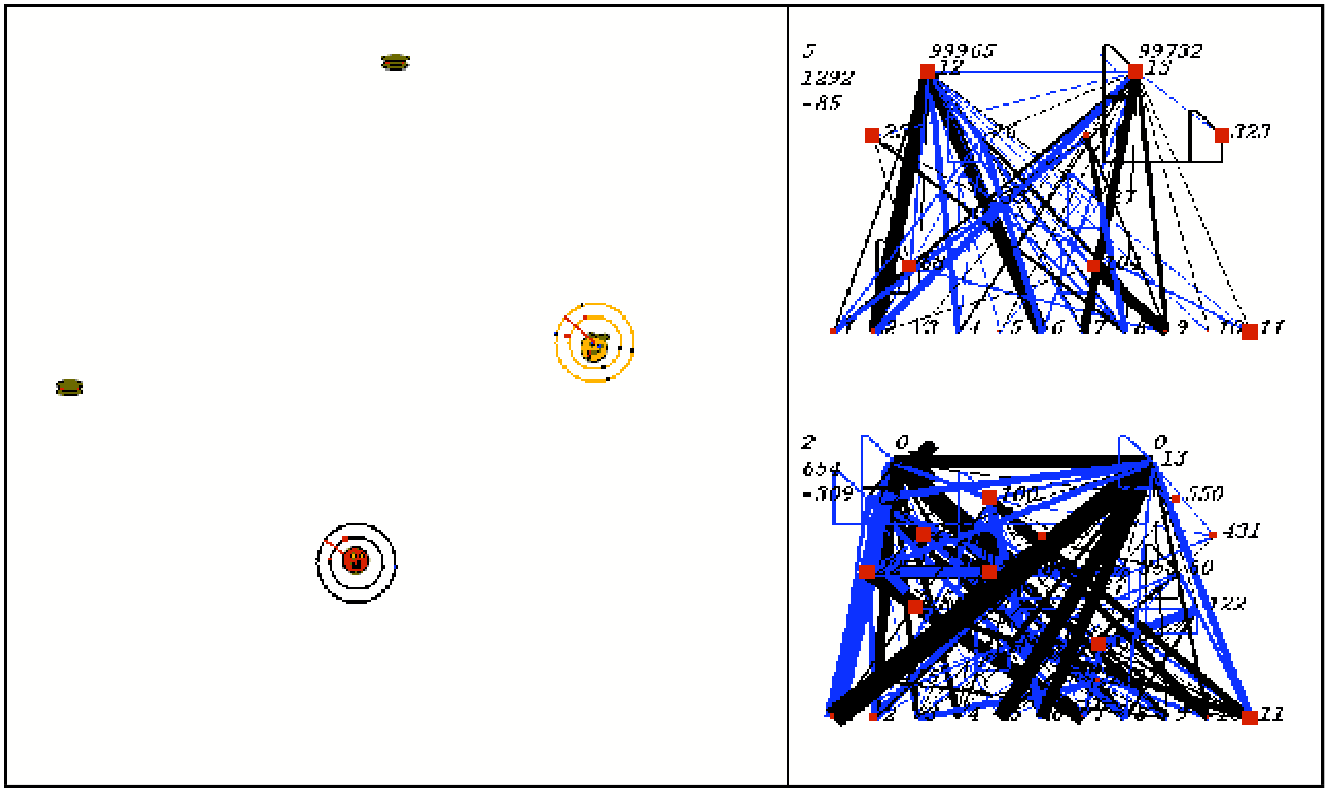
- Crash when higher energy
- Collect food by accident
- DEMO

Mature Strategies



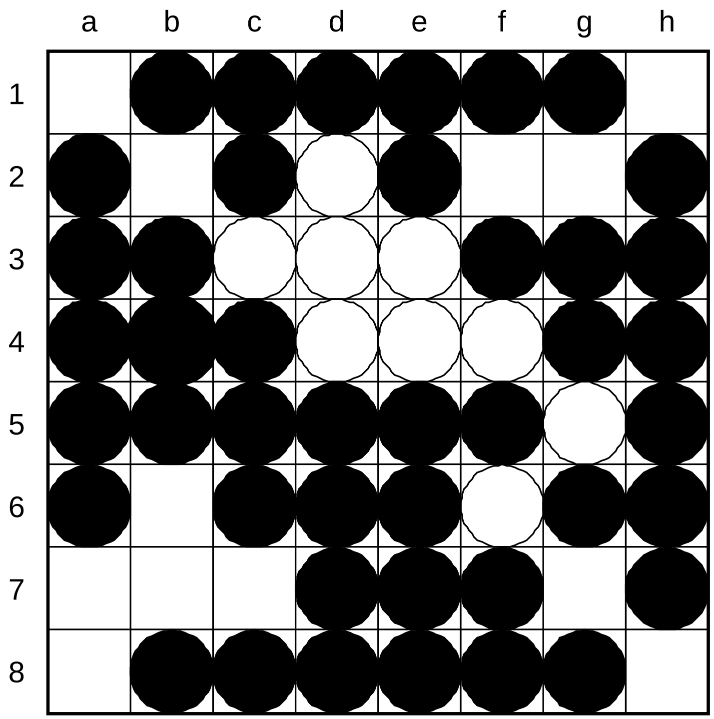
- Collect food to gain energy
- Avoid moving to lose energy
- Standoff: Difficult to predict outcome
- DEMO

Sophisticated Strategy



- “Fake” a move up, force away from last piece
- Win by making a dash to last piece
- Complexification ⟶ arms race
- DEMO

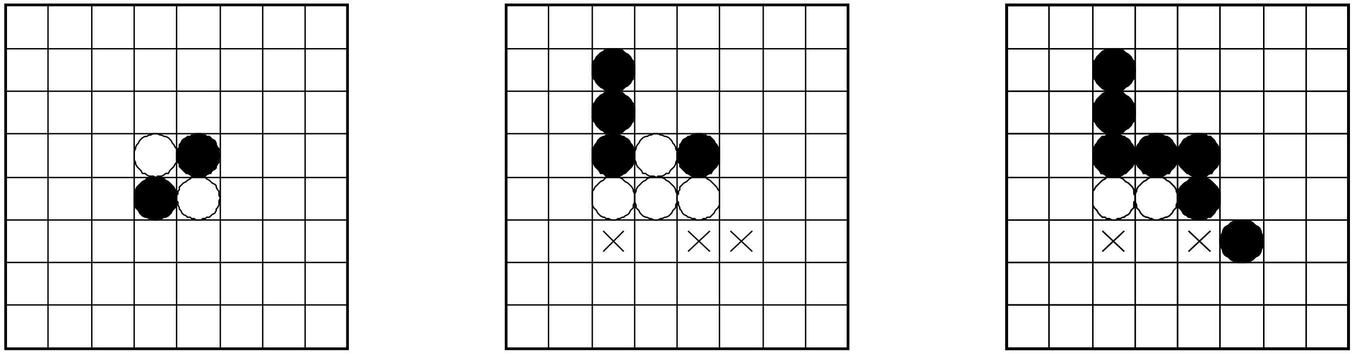
Applications to Games



- Good research platform
  - Controlled domains, clear performance, safe
  - Economically important; training games possible

- Board games: beyond limits of search
  - Evaluation functions in checkers, chess [7,14,15]
  - Filtering information in go, othello [32,55]

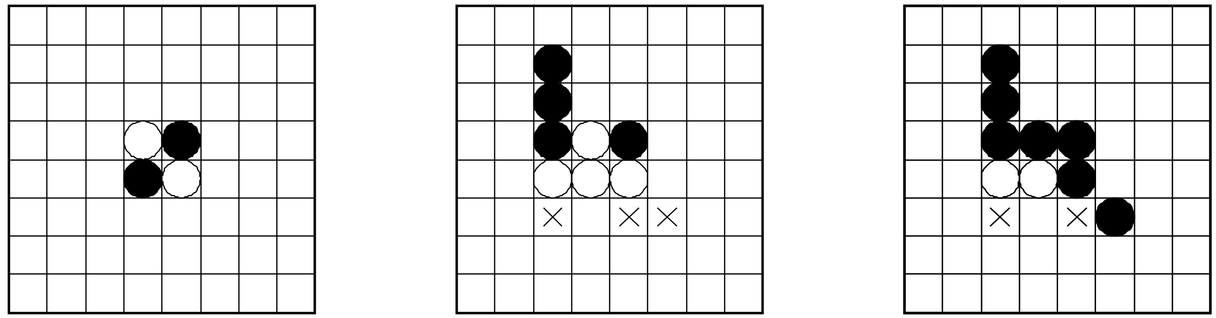
Discovering Novel Strategies in Othello



(a) (b) (c)

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

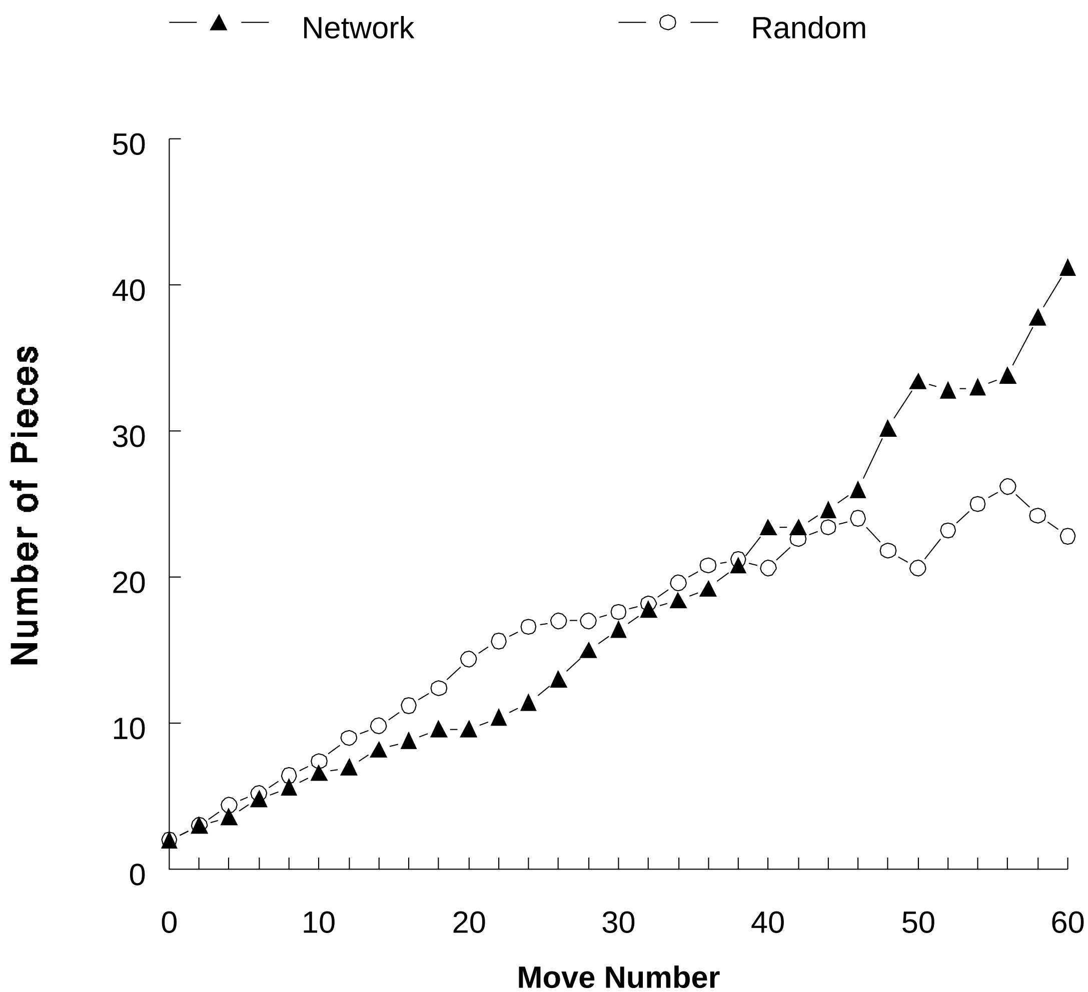
Strategies in Othello



(a) (b) (c)

- 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

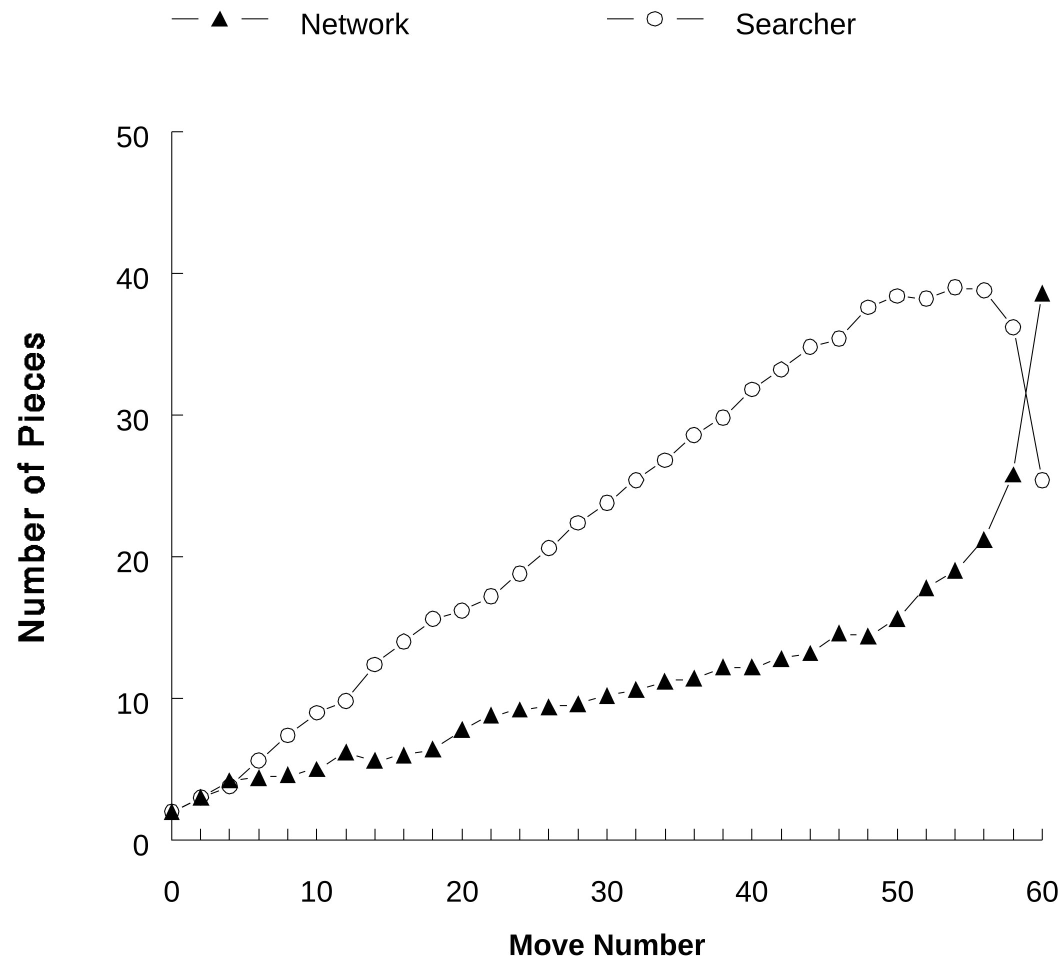
Evolving Against a Random Player



- Network sees the board, suggests moves by ranking

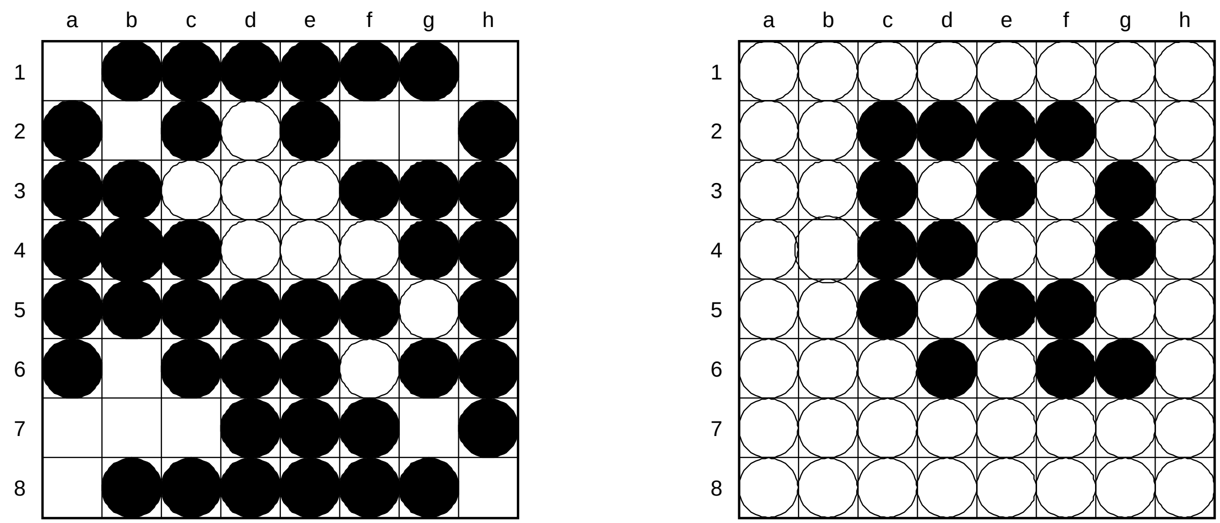
- Networks maximize piece counts throughout the game
- A positional strategy emerges
- Achieved 97% winning percentage

Evolving Against an α-β Program



- Iago’s positional strategy destroyed networks at first
- Evolution turned low piece count into an advantage
- Mobility strategy emerged!
- Achieved 70% winning percentage

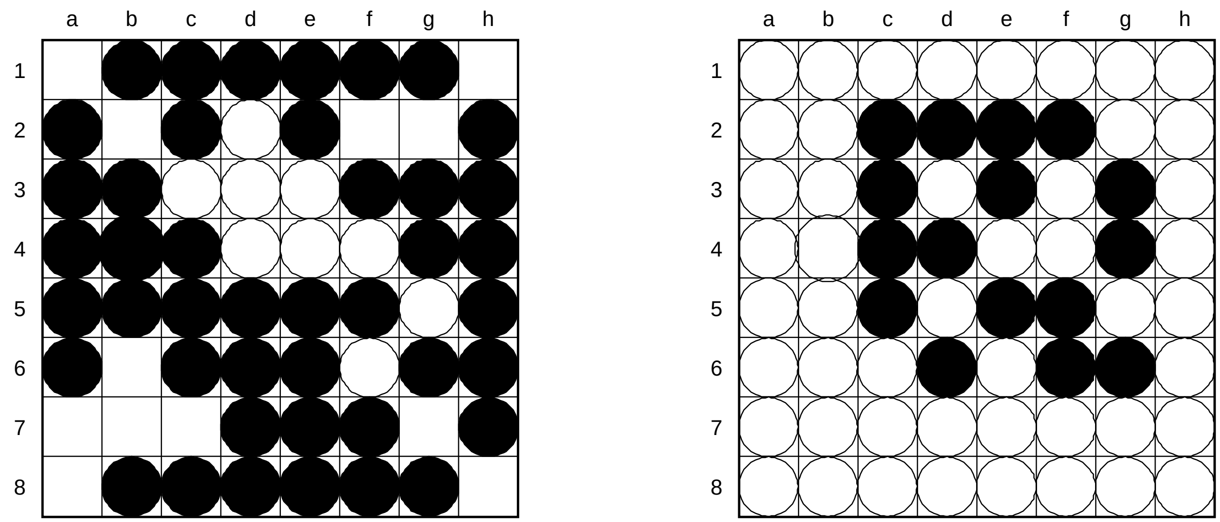
Example game



(a) (b)

- 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

Discovering Novel Strategies



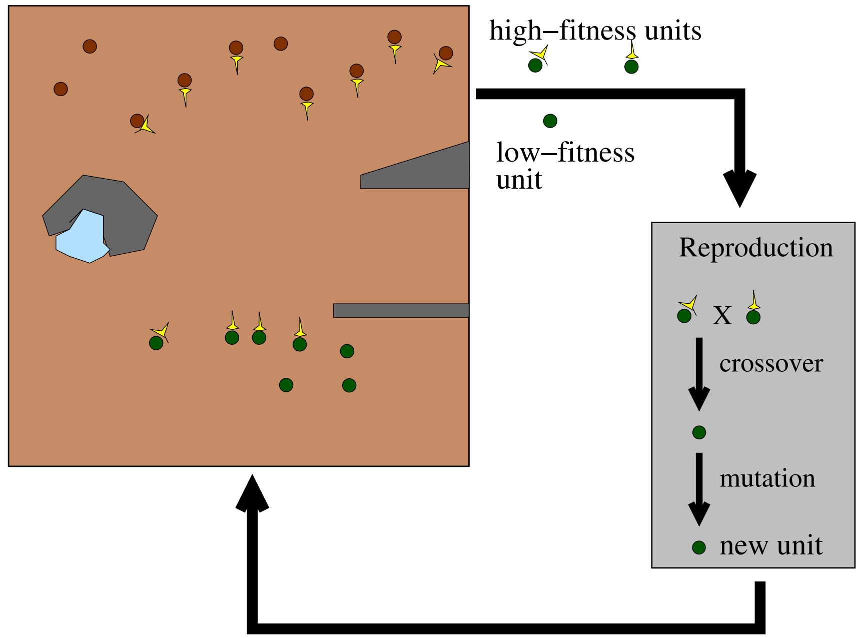
(a) (b)

- Neuroevolution discovered a strategy novel to us
- “Evolution works by tinkering”
  - So does neuroevolution
  - Initial disadvantage turns into novel advantage

Video Games

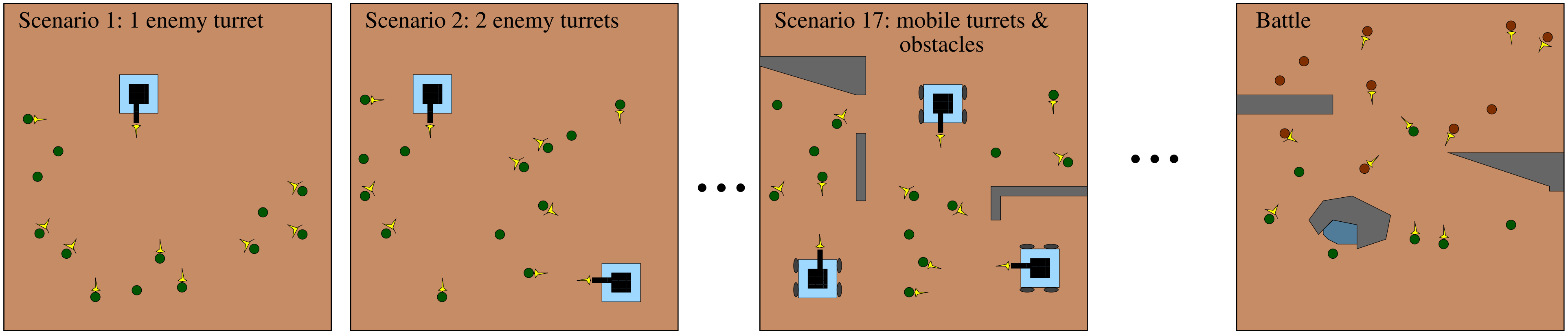
- Economically and socially important
- Adaptation an important future goal
  - More challenging, more fun games
  - Possible to use for training people
- How to make evolution run in real time?

Real-time NEAT



- A parallel, continuous version of NEAT[49]
- Individuals created and replaced every n ticks
- Parents selected probabilistically, weighted by fitness
- Long-term evolution equivalent to generational NEAT

NERO: A Complex Game Platform

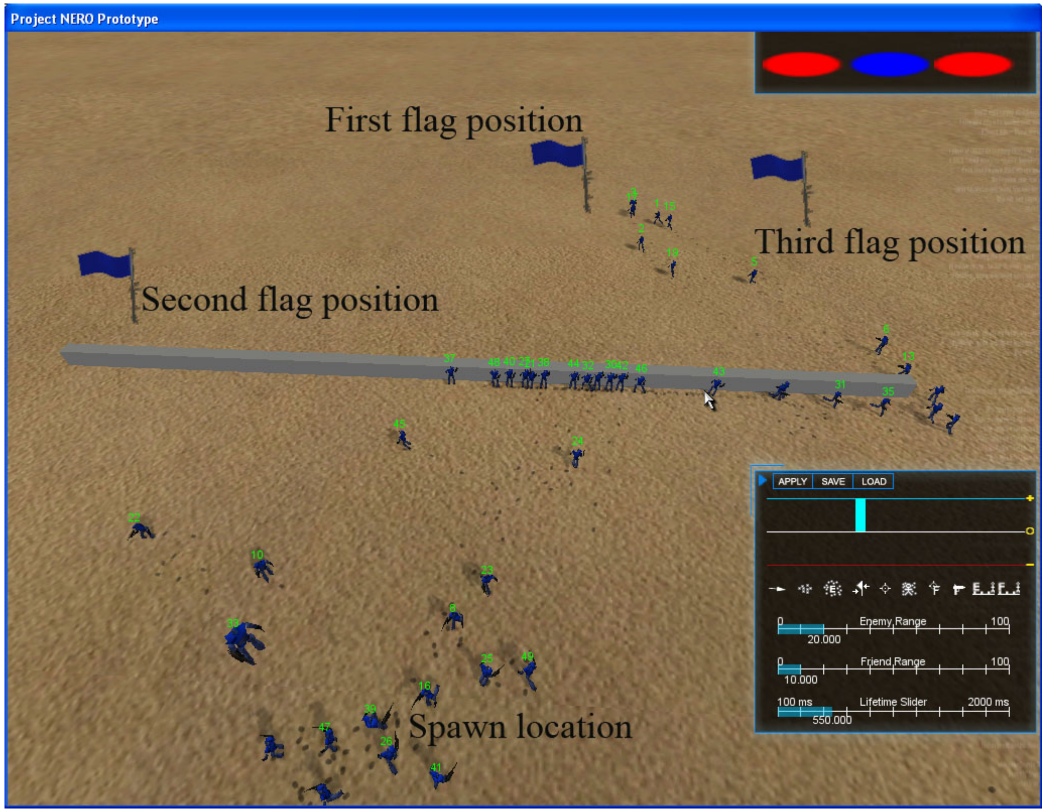


- Teams of agents trained to battle each other
  - Player trains agents through excercises
  - Agents evolve in real time
- New genre: Learning *is* the game
- Challenging platform for reinforcement learning
  - Real time, open ended, requires discovery
- DEMO

Future Challenge: Utilizing Knowledge

- Given a problem, NE discovers a solution by exploring
  - Sometimes you already know (roughly) what works
  - Sometimes random initial behavior is not acceptable
- How can domain knowledge be utilized?
  - By incorporating rules
  - By learning from examples

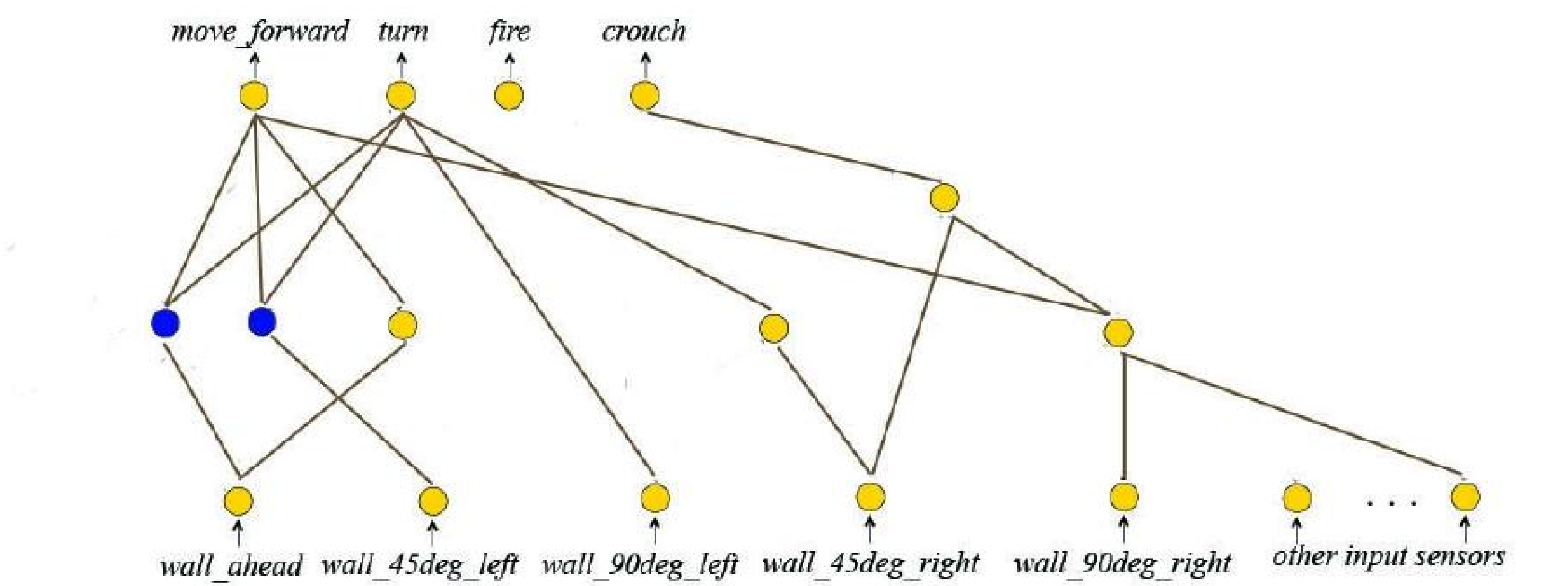
Incorporating Rules into NE



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

Incorporating Rules into NE (2)



- KBANN network added to NEAT networks
  - Treated as complexification
  - Continues to evolve
  - If advice is useful, it stays
- Initial behaviors, on-line advice
- Injecting human knowledge as rules
- DEMO

Lessons from NERO

- NEAT is a strong method for real-time adaptation
  - Complex team behaviors can be constructed
  - Novel strategies can be discovered
- Problem solving with human guidance
- Coevolutionary arms race
- NE makes a new genre of games possible!

Numerous Other Applications

- Creating art, music[8]
- Theorem proving[10]
- Time-series prediction[28]
- Computer system optimization[18]
- Manufacturing optimization[21]
- Process control optimization[58,59]
- Etc.

Evaluation of Applications

- Neuroevolution strengths
  - Can work very fast, even in real-time
  - Potential for arms race, discovery
  - Effective in continuous, non-Markov domains
- Requires many evaluations
  - Requires an interactive domain for feedback
  - Best when parallel evaluations possible
  - Works with a simulator & transfer to domain

Conclusion

- NE is a powerful technology for sequential decision tasks
  - Evolutionary computation and neural nets are a good match
  - Lends itself to many extensions
  - Powerful in applications
- Easy to adapt to applications
  - Control, robotics, optimization
  - Artificial life, biology
  - Gaming: entertainment, training
- Lots of future work opportunities
  - Theory not well developed
  - Indirect encodings
  - Learning and evolution
  - Knowledge and interaction

References

[1] A. Agogino, K. Tumer, and R. Miikkulainen, Efficient credit assignment through evaluation function decomposition, in: *Proceedings of the Genetic and Evolutionary Computation Conference* (2005).

[2] P. J. Angeline, G. M. Saunders, and J. B. Pollack, An evolutionary algorithm that constructs recurrent neural networks, *IEEE Transactions on Neural Networks*, 5:54–65 (1993).

[3] J. M. Baldwin, A new factor in evolution, *The American Naturalist*, 30:441–451, 536–553 (1896).

[4] R. K. Belew, Evolution, learning and culture: Computational metaphors for adaptive algorithms, *Complex Systems*, 4:11–49 (1990).

[5] B. D. Bryant and R. Miikkulainen, Neuroevolution for adaptive teams, in: *Proceedings of the 2003 Congress on Evolutionary Computation* (2003).

[6] D. J. Chalmers, The evolution of learning: An experiment in genetic connectionism, in: *Connectionist Models: Proceedings of the 1990 Summer School*, D. S. Touretzky, J. L. Elman, T. J. Sejnowski, and G. E. Hinton, eds., 81–90, San Francisco: Kaufmann (1990).

[7] K. Chellapilla and D. B. Fogel, Evolution, neural networks, games, and intelligence, *Proceedings of the IEEE*, 87:1471–1496 (1999).

[8] C.-C. Chen and R. Miikkulainen, Creating melodies with evolving recurrent neural networks, in: *Proceedings of the INNS-IEEE International Joint Conference on Neural Networks*, 2241–2246, IEEE, Piscataway, NJ (2001).

[9] D. Cliff, I. Harvey, and P. Husbands, Explorations in evolutionary robotics, *Adaptive Behavior*, 2:73–110 (1993).

[10] N. S. Desai and R. Miikkulainen, Neuro-evolution and natural deduction, in: *Proceedings of The First IEEE Symposium on Combinations of Evolutionary Computation and Neural Networks*, 64–69, IEEE, Piscataway, NJ (2000).

[11] J. Fan, R. Lau, and R. Miikkulainen, Utilizing domain knowledge in neuroevolution, in: *Machine Learning: Proceedings of the 20th Annual Conference* (2003).

[12] D. Floreano and F. Mondada, Evolutionary neurocontrollers for autonomous mobile robots, *Neural Networks*, 11:1461–1478 (1998).

[13] D. Floreano and J. Urzelai, Evolutionary robots with on-line self-organization and behavioral fitness, *Neural Networks*, 13:431–4434 (2000).

[14] D. B. Fogel, *Blondie24: Playing at the Edge of AI*, Kaufmann, San Francisco (2001).

[15] D. B. Fogel, T. J. Hays, S. L. Hahn, and J. Quon, Further evolution of a self-learning chess program, in: *Proceedings of the IEEE Symposium on Computational Intelligence and Games*, IEEE, Piscataway, NJ (2005).

[16] B. Fullmer and R. Miikkulainen, Using marker-based genetic encoding of neural networks to evolve finite-state behaviour, in: *Toward a Practice of Autonomous Systems: Proceedings of the First European Conference on Artificial Life*, F. J. Varela and P. Bourgine, eds., 255–262, MIT Press, Cambridge, MA (1992).

[17] F. Gomez, *Robust Non-Linear Control through Neuroevolution*, Ph.D. thesis, Department of Computer Sciences, The University of Texas at Austin, Austin, TX (2003).

[18] F. Gomez, D. Burger, and R. Miikkulainen, A neuroevolution method for dynamic resource allocation on a chip multiprocessor, in: *Proceedings of the INNS-IEEE International Joint Conference on Neural Networks*, 2355–2361, IEEE, Piscataway, NJ (2001).

[19] F. Gomez and R. Miikkulainen, Incremental evolution of complex general behavior, *Adaptive Behavior*, 5:317–342 (1997).

[20] F. Gomez and R. Miikkulainen, Active guidance for a finless rocket using neuroevolution, in: *Proceedings of the Genetic and Evolutionary Computation Conference* (2003).

[21] B. Greer, H. Hakonen, R. Lahdelma, and R. Miikkulainen, Numerical optimization with neuroevolution, in: *Proceedings of the 2002 Congress on Evolutionary Computation*, 361–401, IEEE, Piscataway, NJ (2002).

[22] F. Gruau and D. Whitley, Adding learning to the cellular development of neural networks: Evolution and the Baldwin effect, *Evolutionary Computation*, 1:213–233 (1993).

[23] G. E. Hinton and S. J. Nowlan, How learning can guide evolution, *Complex Systems*, 1:495–502 (1987).

[24] G. S. Hornby, S. Takamura, J. Yokono, O. Hanagata, M. Fujita, and J. Pollack, Evolution of controllers from a high-level simulator to a high dof robot, in: *Evolvable Systems: From Biology to Hardware; Proceedings of the Third International Conference*, 80–89, Springer, Berlin (2000).

[25] C. Igel, Neuroevolution for reinforcement learning using evolution strategies, in: *Proceedings of the 2003 Congress*

on Evolutionary Computation, 2588–2595 (2003).

[26] Y. Liu, X. Yao, and T. Higuchi, Evolutionary ensembles with negative correlation learning, *IEEE Transactions on Evolutionary Computation*, 4:380–387 (2000).

[27] R. Maclin and J. W. Shavlik, Creating advice-taking reinforcement learners, *Machine Learning*, 22(1-3):251–281 (1996).

[28] J. R. McDonnell and D. Waagen, Evolving recurrent perceptrons for time-series modeling, *IEEE Transactions on Evolutionary Computation*, 5:24–38 (1994).

[29] P. McQuesten, *Cultural Enhancement of Neuroevolution*, Ph.D. thesis, Department of Computer Sciences, The University of Texas at Austin, Austin, TX (2002). Technical Report AI-02-295.

[30] E. Mjolsness, D. H. Sharp, and B. K. Alpert, Scaling, machine learning, and genetic neural nets, *Advances in Applied Mathematics*, 10:137–163 (1989).

[31] D. J. Montana and L. Davis, Training feedforward neural networks using genetic algorithms, in: *Proceedings of the 11th International Joint Conference on Artificial Intelligence*, 762–767, San Francisco: Kaufmann (1989).

[32] D. E. Moriarty, *Symbiotic Evolution of Neural Networks in Sequential Decision Tasks*, Ph.D. thesis, Department of Computer Sciences, The University of Texas at Austin (1997). Technical Report UT-AI97-257.

[33] D. E. Moriarty and R. Miikkulainen, Discovering complex Othello strategies through evolutionary neural networks, *Connection Science*, 7(3):195–209 (1995).

[34] D. E. Moriarty and R. Miikkulainen, Evolving obstacle avoidance behavior in a robot arm, in: *From Animals to Animats 4: Proceedings of the Fourth International Conference on Simulation of Adaptive Behavior*, P. Maes, M. J. Mataric, J.-A. Meyer, J. Pollack, and S. W. Wilson, eds., 468–475, Cambridge, MA: MIT Press (1996).

[35] D. E. Moriarty and R. Miikkulainen, Forming neural networks through efficient and adaptive co-evolution, *Evolutionary Computation*, 5:373–399 (1997).

[36] D. E. Moriarty, A. C. Schultz, and J. J. Grefenstette, Evolutionary algorithms for reinforcement learning, *Journal of Artificial Intelligence Research*, 11:199–229 (1999).

[37] S. Nolfi, J. L. Elman, and D. Parisi, Learning and evolution in neural networks, *Adaptive Behavior*, 2:5–28 (1994).

[38] S. Nolfi and D. Floreano, *Evolutionary Robotics*, MIT Press, Cambridge (2000).

[39] S. Nolfi and D. Parisi, Good teaching inputs do not correspond to desired responses in ecological neural networks, *Neural Processing Letters*, 1(2):1–4 (1994).

[40] D. Pardoe, M. Ryoo, and R. Miikkulainen, Evolving neural network ensembles for control problems, in: *Proceedings of the Genetic and Evolutionary Computation Conference* (2005).

[41] M. A. Potter and K. A. D. Jong, Cooperative coevolution: An architecture for evolving coadapted subcomponents, *Evolutionary Computation*, 8:1–29 (2000).

[42] J. Reisinger, K. O. Stanley, and R. Miikkulainen, Evolving reusable neural modules, in: *Proceedings of the Genetic and Evolutionary Computation Conference* (2004).

[43] T. P. Runarsson and M. T. Jonsson, Evolution and design of distributed learning rules, in: *Proceedings of The First IEEE Symposium on Combinations of Evolutionary Computation and Neural Networks*, 59–63, IEEE, Piscataway, NJ (2000).

[44] E. Ruppin, Evolutionary autonomous agents: A neuroscience perspective, *Nature Reviews Neuroscience* (2002).

[45] C. W. Seys and R. D. Beer, Evolving walking: The anatomy of an evolutionary search, in: *From Animals to Animats 8: Proceedings of the Eight International Conference on Simulation of Adaptive Behavior*, S. Schaal, A. Ijspeert, A. Billard, S. Vijayakumar, J. Hallam, and J.-A. Meyer, eds., 357–363, MIT Press, Cambridge, MA (2004).

[46] A. A. Siddiqi and S. M. Lucas, A comparison of matrix rewriting versus direct encoding for evolving neural networks, in: *Proceedings of IEEE International Conference on Evolutionary Computation*, 392–397, IEEE, Piscataway, NJ (1998).

[47] Y. F. Sit and R. Miikkulainen, Learning basic navigation for personal satellite assistant using neuroevolution, in: *Proceedings of the Genetic and Evolutionary Computation Conference* (2005).

[48] K. O. Stanley, *Efficient Evolution of Neural Networks through Complexification*, Ph.D. thesis, Department of Computer Sciences, The University of Texas at Austin, Austin, TX (2003).

[49] K. O. Stanley, B. Bryant, and R. Miikkulainen, Real-time neuroevolution in the nero video game, *IEEE Transactions on Evolutionary Computation* (2005). In press.

[50] K. O. Stanley, B. D. Bryant, and R. Miikkulainen, Evolving adaptive neural networks with and without adaptive synapses, in: *Proceedings of the 2003 Congress on Evolutionary Computation*, IEEE, Piscataway, NJ (2003).

[51] K. O. Stanley, N. Kohl, and R. Miikkulainen, Neuroevolution of an automobile crash warning system, in: *Proceedings of the Genetic and Evolutionary Computation Conference* (2005).

[52] K. O. Stanley and R. Miikkulainen, Evolving neural networks through augmenting topologies, *Evolutionary Computation*, 10:99–127 (2002).

[53] K. O. Stanley and R. Miikkulainen, A taxonomy for artificial embryogeny, *Artificial Life*, 9:93–130 (2003).

[54] K. O. Stanley and R. Miikkulainen, Competitive coevolution through evolutionary complexification, *Journal of Artificial Intelligence Research*, 21:63–100 (2004).

[55] K. O. Stanley and R. Miikkulainen, Evolving a roving eye for go, in: *Proceedings of the Genetic and Evolutionary Computation Conference* (2004).

[56] D. G. Stork, S. Walker, M. Burns, and B. Jackson, Preadaptation in neural circuits, in: *International Joint Conference on Neural Networks* (Washington, DC), 202–205, IEEE, Piscataway, NJ (1990).

[57] J. Urzelai, D. Floreano, M. Dorigo, and M. Colombetti, Incremental robot shaping, *Connection Science*, 10:341–360 (1998).

[58] A. v. E. Conradie, R. Miikkulainen, and C. Aldrich, Adaptive control utilising neural swarming, in: *Proceedings of the Genetic and Evolutionary Computation Conference*, San Francisco: Kaufmann (2002).

[59] A. v. E. Conradie, R. Miikkulainen, and C. Aldrich, Intelligent process control utilizing symbiotic memetic neuro-evolution, in: *Proceedings of the 2002 Congress on Evolutionary Computation* (2002).

[60] V. Valsalam, J. A. Bednar, and R. Miikkulainen, Constructing good learners using evolved pattern generators, in: *Proceedings of the Genetic and Evolutionary Computation Conference* (2005).

[61] G. M. Werner and M. G. Dyer, Evolution of communication in artificial organisms, in: *Proceedings of the Workshop on Artificial Life (ALIFE '90)*, C. G. Langton, C. Taylor, J. D. Farmer, and S. Rasmussen, eds., 659–687, Reading, MA: Addison-Wesley (1991).

[62] G. M. Werner and M. G. Dyer, Evolution of herding behavior in artificial animals, in: *Proceedings of the Second International Conference on Simulation of Adaptive Behavior*, J.-A. Meyer, H. L. Roitblat, and S. W. Wilson, eds., Cambridge, MA: MIT Press (1992).

[63] S. Whiteson, N. Kohl, R. Miikkulainen, and P. Stone, Evolving keepaway soccer players through task decomposition,

Machine Learning (In press).

[64] S. Whiteson, P. Stone, K. O. Stanley, R. Miikkulainen, and N. Kohl, Automatic feature selection in neuroevolution, in: *Proceedings of the Genetic and Evolutionary Computation Conference* (2005).

[65] D. Whitley, S. Dominic, R. Das, and C. W. Anderson, Genetic reinforcement learning for neurocontrol problems, *Machine Learning*, 13:259–284 (1993).

[66] A. P. Wieland, Evolving controls for unstable systems, in: *Connectionist Models: Proceedings of the 1990 Summer School*, D. S. Touretzky, J. L. Elman, T. J. Sejnowski, and G. E. Hinton, eds., 91–102, San Francisco: Kaufmann (1990).

[67] X. Yao, Evolving artificial neural networks, *Proceedings of the IEEE*, 87(9):1423–1447 (1999).

[68] C. H. Yong and R. Miikkulainen, Cooperative coevolution of multi-agent systems, Technical Report AI01-287, Department of Computer Sciences, The University of Texas at Austin (2001).