Modern Robots: Evolutionary Robotics

Jeff Clune
Assistant Professor
Evolving Artificial Intelligence Laboratory

University of Wyoming
We choose to go to the moon. We choose to go to the moon … not because they are easy, but because they are hard, because that goal will serve to organize and measure the best of our energies and skills, because that challenge is one that we are willing to accept, one we are unwilling to postpone, and one which we intend to win

- John F. Kennedy, 1962
News

• Evaluations come out this weekend
• We will do them in class
  • anonymously of course!
Transferability
Transferability
Transferability: Typical Approaches

• Add noise (Jakobi 1997)
  • But to what? How much?
  • It’s a very manual process with human domain knowledge
Transferability

- Why might it fail?
Transferability

• What would you do to improve it?
Self-Modeling

• How does a robotic brain know how many legs it has?
• How it’s body works?
• How to respond to injury?
• Test its actions mentally before trying them?

• How did/do you do it?
Comes to be called “Estimation-Exploration Algorithm” (EEA) in later papers
Continuous Self-Modeling

- robot infers its own morphology
  - through experiments
- uses model to predict which behaviors will work
  - enables ruling out bad decisions without trying them
- can detect if morphology has changed
  - e.g. damage
- updates self-model
Continuous Self-Modeling

- Three phases
  - Modeling
  - Prediction
  - Testing

---

**Fig. 1**: Outline of the algorithm. The robot continuously cycles through action execution. (A and B) Self-model synthesis. The robot physically performs an action (A). Initially, this action is random; later, it is the best action found in (C). The robot then generates several self-models to match sensor data collected while performing previous actions (B). It does not know which model is correct. (C) Exploratory action synthesis. The robot generates several possible actions that disambiguate competing self-models. (D) Target behavior synthesis. After several cycles of (A) to (C), the currently best model is used to generate locomotion sequences through optimization. (E) The best locomotion sequence is executed by the physical device. (F) The cycle continues at step (B) to further refine models or at step (D) to create new behaviors.
Continuous Self-Modeling

- Creating Self Model
  - Make arbitrary motion
    - Record sensory data
  - Evolves models to best explain the data
  - Select action that maximizes disagreement
    - aka Active Learning
  - Perform action, add new data to archive
  - Evolve new models to explain enlarged archive

Fig. 1. Outline of the algorithm. The robot continuously cycles through action execution. (A and B) Self-model synthesis. The robot physically performs an action (A). Initially, this action is random; later, it is the best action found in (C). The robot then generates several self-models to match sensory data collected while performing previous actions (B). It does not know which model is correct. (C) Exploratory action synthesis. The robot generates several possible actions that disambiguate competing self-models. (D) Target behavior synthesis. After several cycles of (A) to (C), the currently best model is used to generate locomotion sequences through optimization. (E) The best locomotion sequence is executed by the physical device. (F) The cycle continues at step (B) to further refine models or at step (D) to create new behaviors.
Continuous Self-Modeling

- **Prediction**
  - After $N$ cycles (here, 16), use best models to evolve desired behavior
    - e.g. walking
  - Thus, the simulator predicts what behaviors will work in reality
Continuous Self-Modeling

- Testing
  - Test the desired behavior

- Recovering from damage
  - Detect it (e.g. sensor readings different than expected)
  - Restart self-modeling algorithm

Fig. 1. Outline of the algorithm. The robot continuously cycles through action execution. (A and B) Self-model synthesis. The robot physically performs an action (A). Initially, this action is random; later, it is the best action found in (C). The robot then generates several self-models to match sensor data collected while performing previous actions (B). It does not know which model is correct. (C) Exploratory action synthesis. The robot generates several possible actions that disambiguate competing self-models. (D) Target behavior synthesis. After several cycles of (A) to (C), the currently best model is used to generate locomotion sequences through optimization. (E) The best locomotion sequence is executed by the physical device. (F) The cycle continues at step (B) to further refine models or at step (D) to create new behaviors.
Fig. 2. The robot continually models and behaves. The robot performs a random action (A). A set of random models, such as (B), is synthesized into approximate models, such as (C). A new action is then synthesized to create maximal model disagreement and is performed by the physical robot (D), after which further modeling ensues. This cycle continues for a fixed period or until no further model improvement is possible (E and F). The best model is then used to synthesize a behavior. In this case, the behavior is forward locomotion, the first few movements of which are shown (G to I). This behavior is then executed by the physical robot (J to L). Next, the robot suffers damage [the lower part of the right leg breaks off (M)]. Modeling recommences with the best model so far (N), and using the same process of modeling and experimentation, eventually discovers the damage (O). The new model is used to synthesize a new behavior (P to R), which is executed by the physical robot (S to U), allowing it to recover functionality despite the unanticipated change.
Continuous Self-Modeling

Robust Machines Through Continuous Self-Modeling
Josh Bongard, Victor Zykov, Hod Lipson
Computational Synthesis Laboratory
Sibley School of Mechanical and Aerospace Engineering
Cornell University
Continuous Self-Modeling: Experiments

• **Controls:**
  - all use same computation:
    - 250k model simulations & 16 physical actions
  - Control 1: 16 random actions taken, data added to archive, modeling takes place
    - takes out active learning
  - Control 2: Random action taken each cycle
    - takes out active learning, but allows incremental model building
Continuous Self-Modeling: Results

- Their algo beats the controls
  - higher probability of learning correct morphology
  - better at detecting damage

- Transfer to reality was still not perfect

**Fig. 3.** Distance traveled during optimized versus random behaviors. Dots indicate the final location of the robot’s center of mass, when it starts at the origin. Red dots indicate final positions of the physical robot when executing random behaviors. Black dots indicate final expected positions predicted by the 30 optimized behaviors when executed on the self-model (Fig. 2F). Blue dots denote the actual final positions of the physical robot after executing those same behaviors in reality. The behaviors corresponding to the circled dots are depicted in Fig. 2, G to L. Squares indicate mean final positions. Vertical and horizontal lines indicate 2 SD for vertical and horizontal displacements, respectively.
Continuous Self-Modeling

• Also would work in new environments

• Good example of a high-level cognitive function, tested in a simplified system
Potential Future Work

• Expansions
  • Model motor functions (mapping from input to result)
  • Model simulator parameters (e.g. coefficient of friction)
  • Model the entire simulator!
Other Applications

- Automatically
  - discover natural laws (PNAS paper)
  - model any data