Modern Robots: Evolutionary Robotics

Jeff Clune
Assistant Professor
Evolving Artificial Intelligence Laboratory

University of Wyoming
Transferability
Transferability Approach

Mouret, Koos, & Doncieux
Transferability Approach

• If you can’t be with the simulator you want, love the one you’re with

• aka: don’t fix the simulator, just recognize where it sucks

• Imagine:
  • in simulator, fastest way around track is to drift turns
    - simulated evolution drifts perfectly and is fast
  • in reality, drifting is unreliable
    - leads to lots of crashes = slow
  • how would you fix the problem (algorithmically)?
• First ever complete map of reality vs. simulation
• Very different (4 peaks in sim, 1 or 2 in reality, major valleys in sim)
• Only one peak in sim transfers (e.g. getting stuck in top left no good)
• But there is one good area, so finding it is the goal
Transferability Approach

• High performance in sim, but not in reality w/o Trans. Approach

• Similar results with another problem
The Transferability Approach: Crossing the Reality Gap in Evolutionary Robotics (2010)

Sylvain Koos, Jean-Baptiste Mouret, Stéphane Doncieux

contact: koos@isir.upmc.fr

Institut des Systèmes Intelligents et de Robotique
Université Pierre et Marie Curie
Transferability Approach

- Same ideas apply to generalization
  - Reality is general performance
  - A few trials are the simulation
  - Learn which types of behaviors transfer/generalize well
  - Pinville et al. 2011

- Same ideas apply to damage recognition
  - we’ll talk about that next
Fast Damage Recovery in Robotics with the T-Resilience Algorithm
Koos, Cully, and Mouret, arXiv, 2013

• If damage occurs, what can you do?
  • What did Bongard do?
If damage occurs, what do you do?
- Improve the model/simulator
- What else can you do?
Fast Damage Recovery in Robotics with the T-Resilience Algorithm
Koos, Cully, and Mouret, arXiv, 2013

• If damage occurs, what do you do?
  • Improve the model/simulator
  • Learn to make motions that don’t rely on the broken component
    - “Transferability-based resilience algorithm”
Damage in Robotics

- **Traditional approaches**
  - make a robust robot (prevent damage)
  - make a robust controller (works even with damage) [hard!]
  - make contingency plans for all/likely failures, write code in case

- **Alternate approaches**
  - learn a new controller once damage happens
    - reinforcement learning can work
      - quick
      - but only in confined search spaces (tend to get stuck in local optima)
    - evolution is more open-ended
      - one option is Bongard et al. continuous self-modeling...but we’ll learn about another
Recall that Bongard et al.’s self-modeling doesn’t address transferability of behavior
  - resulting in gaits that didn’t actually transfer that well

Also, it only identified the correct model in 50% of runs
  - attempts to use it on a biped showed it didn’t work well (Zagal et al., 2009)

Both failures could work with many more physical tests
  - but that would require many more physical tests
T-Resilience

• Key idea: behaviors that rely on damaged parts will not be transferable

• Think of the undamaged simulated robot as an imperfect simulation

• Algorithm is basically Transferability Approach in new domain (damaged robot)
T-Resilience

- Maximize (multi-objective algo)
  - performance
  - transferability
  - diversity
T-Resilience is Good When...

- previously good behaviors are not good on the damaged robot
  - otherwise, why change?

- a qualitatively new behavior is needed
  - otherwise use local search
T-resilience: Experiments

- **Robot**
  - 18-DOF hexapod
  - Adapts to
    - leg removal
    - broken legs
    - motor failures

- **Controller**
  - 24 parameters total

Figure 6: (a) The 18-DOF hexapod robot is equipped with a RGB-D camera (RGB camera with a depth sensor). (b) Snapshot of the simulation used as a self-model by the robot which occurs in an ODE-based physics simulator. The robot lies on a horizontal plane and contacts are simulated as well. (c) Kinematic scheme of the robot. (d) Control function $\gamma(t, \alpha, \phi)$ with $\alpha = 1$ and $\phi = 0$.

The movement of each DOF is governed by a periodic function that computes its angular position as a function $\gamma$ of time $t$, amplitude $\alpha$ and phase $\phi$ (Fig. 6, d):

$$\gamma(t, \alpha, \phi) = \alpha \cdot \tanh \left( 4 \cdot \sin \left( 2 \cdot \pi \cdot (t + \phi) \right) \right)$$

(1)

where $\alpha$ and $\phi$ are the parameters that define the amplitude of the movement and the phase shift of $\gamma$, respectively. Frequency is fixed.
T-resilience: Experiments

• Controls
  • reference controller (no recovery learning)
  • stochastic local search
  • policy gradient
  • Bongard’s algorithm
Figure 7: Test cases considered in our experiments. (A) The hexapod robot is not damaged. (B) The left middle leg is no longer powered. (C) The terminal part of the front right leg is shortened by half. (D) The right hind leg is lost. (E) The middle right leg is lost. (F) Both the middle right leg and the front left leg are lost.
Fast Damage Recovery in Robotics with the T-Resilience Algorithm

S. Koos, A. Cully and J-B Mouret

ISIR, CNRS UMR 7222, Université Pierre et Marie Curie, F-75005, Paris, France.
Results

Figure 4: Performances obtained in each test cases (distance covered in 3 seconds). On each box, the central mark is the median, the edges of the box are the lower hinge (defined as the 25th percentile) and the upper hinge (the 75th percentile). The whiskers extend to the most extreme data point which is no more than 1.5 times the length of the box away from the box. Each algorithm has been run 5 times and distances are measured using the external motion capture system. Except for the T-Resilience, the performance of the controllers found after about 25 transfers (tests) and after about 20 minutes (time) are depicted (all T-Resilience experiments last about 20 minutes and use 25 transfers). The horizontal lines denote the performances of the reference gait, according to the COA scanner (dashed line) and according to the SLAM algorithm (solid line).

Figure 8: Typical trajectories (median performance) observed in every test case. Dashed line: reference gait. Solid line: controller with median performance value found by the T-Resilience algorithm. The poor performance of the reference controllers after any of the damages shows that adaptation is required in these situations. The trajectories obtained with the T-Resilience algorithm are not perfectly straight because our objective function does not explicitly reward straightness (see sections 3.3 and 3.5).
T-Resilience Results

(a) Distribution of duration (median duration indicated below the graph).

(b) Experimental time (experiments with the robot).
Downsides to Transferability Approach

- still takes 19 minutes
- tries to learn transferability function
  - (how all things transfer, even if low performing)
- can we go faster?
Intelligent Trial and Error

Damage Recovery in Robots via Intelligent Trial and Error

Antoine Cully
UPMC Université France

Jeff Clune
University of Wyoming USA

Jean-Baptiste Mouret
UPMC Université France
Intelligent Trial and Error

A

Mental Preparation

High-dimensional search space

Simulation (undamaged)

B

Low-dimensional search space

Confidence level

Performance

Behavior repertoire

C

Damage Recovery

Damaged robot

D

Walking robot
Repertoire and Updating It