Integrated Design, Deployment and Inference for Robot Ecologies

Josh Bongard Hod Lipson
[joshua.bongard | hod.lipson]@cornell.edu
Computational Synthesis Laboratory
Sibley School of Mechanical and Aerospace Engineering
Ithaca, NY 14853, USA

November 7, 2004

Introduction

Evolutionary computation (EC) has been used extensively for the automated design of robot controllers (see [4] for an overview); more recently, robots themselves have been automatically designed using evolutionary techniques [5], [1]: the morphology and controller of the robot is evolved together to produce robots that exhibit increasing efficacy at performing the desired task. Evolutionary techniques are a kind of stochastic search: hundreds or thousands of candidate solutions to a problem are tested (in the case of robotics, a robot controller or the robot itself), and those that solve the given problem better (such as a robot that performs a desired task well) are retained and modified, and those that are inferior are deleted. Because of the need to perform many evaluations, robot simulators are used to evolve controllers, which can then be downloaded on to the physical robot.

With the maturation of three-dimensional printing technologies, it has been demonstrated [3] that not only can robots be automatically designed, but the prospect of automated robot fabrication will become possible in the near future: robot morphologies may be evolved in simulation, and then the entire robot may be automatically printed.

We argue that integrated robot design and robot manufacture machines could be of use in planetary exploration for two main reasons: robot groups would not have to be flown from Earth to their target site, thus minimizing payload; and the design and behaviors of robots could be altered on site in response to unforseen environmental factors. This latter point could be achieved by deploying robots into the remote environment, and having them communicate sensor data back to the design/manufacture machine. The machine could then use this data to automatically improve simulations reflecting the states of the deployed robots and their task environments. These improved simulations could then be used to design more effective robots that are then printed and deployed, and so on.

In respect to this concept of an integrated design, manufacture and deployment cycle, we have recently developed an inference method—the estimation-exploration algorithm—that acts as a continual bidirectional conversation between a physical robot operating in an uncertain environment, and a set of constantly updating simulators of the robot itself and its environment.

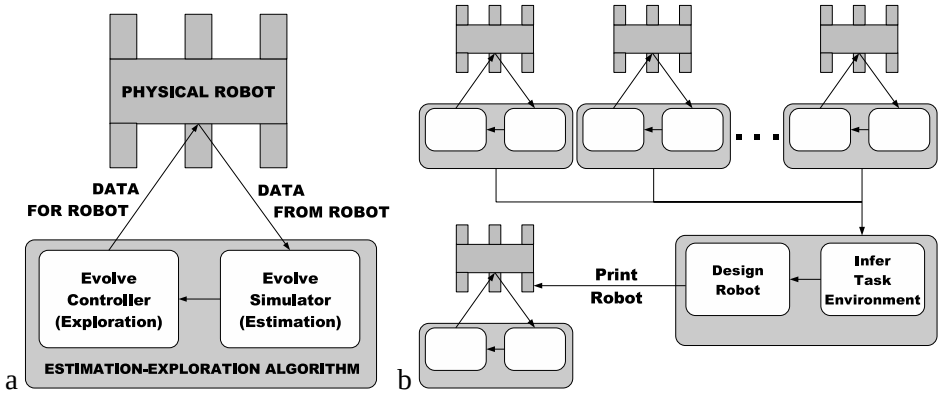

Figure 1: **Integrated robot design, manufacture and behavior generation. a:** A single robot and its associated inference engine (the estimation-exploration algorithm). The algorithm takes in sensor data, updates a simulation of the robot and its environment, uses that new simulator to evolve a better controller for the robot, and then downloads the controller on to the physical robot. **b:** In a collective robotics setting, an integrated design/manufacture machine produces robots, using data collected by already deployed robots.

The Estimation-Exploration Algorithm

The estimation-exploration algorithm imports sensor data from a physical robot, and uses that data to automatically evolve a simulator describing the state of the robot and its surrounding environment. The improved simulator is then used to evolve a new controller for the physical robot, which is then downloaded to the physical robot. By continually repeating this cycle, the physical robot can continuously adapt its mode of operation in the face of unanticipated changes to its own state, or to changes in the environment. Figure 1a shows schematically the flow of data between the algorithm and the physical robot.

Inferring Damage and the Environment

We have previously shown that the estimation-exploration algorithm can automatically diagnose unanticipated failures suffered by a target robot [2], and evolve a compensatory neural network controller that allows the damaged robot to continue with its task, which in that instance was fast forward locomotion. In those experiments, a damaged simulated robot was used instead of a physical robot. Figure 2 shows the behavior of the target robot using an evolved controller before failure; behavior using the same controller just after failure; and behavior using a compensatory controller produced after four cycles through the algorithm, respectively. The algorithm correctly diagnosed that one of the lower legs had broken off, and was able to automatically generate a compensatory controller that provided 50% of the original functionality.

In addition to inferring changes in its own state, we have demonstrated that the estimation-exploration algorithm also can infer the state of the robot's environment. In [2], a robot originally evolved to travel over flat terrain encounters a canted floor, and the algorithm again, using only sensor data, is able to update the simulator to reflect this change, and re-evolve a compensating controller.

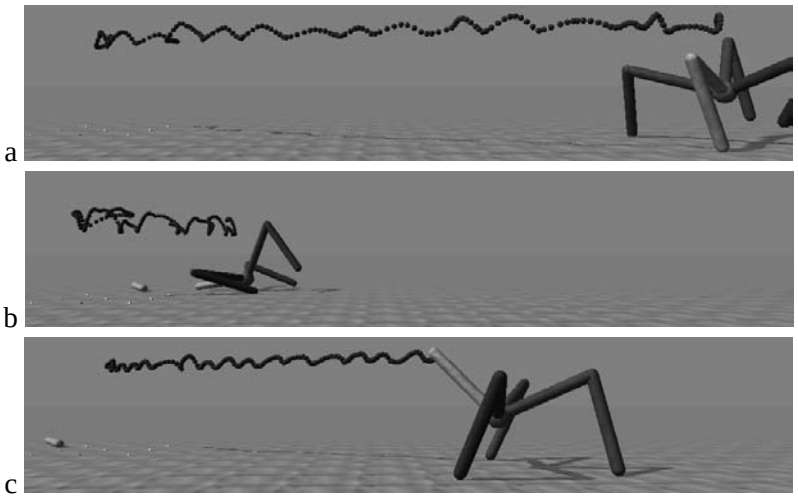

Figure 2: **Automated recovery after unanticipated failure. a:** A controller for this robot was originally evolved in a robot simulator. **b:** The robot suffers a separated lower leg. **c:** The behavior of the damaged robot using the fourth controller output by the estimation-exploration algorithm.

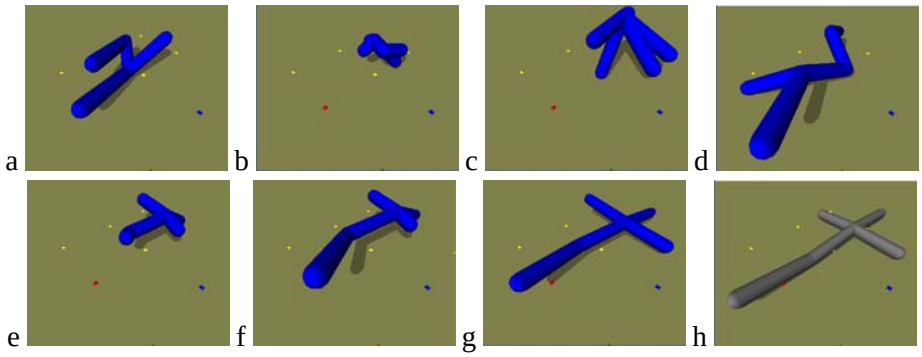

Figure 3: **Evolving robot morphologies using sensor data. a-d:** Random robot morphologies produced during the first pass through the estimation phase. **e-g:** The best robot models produced by the first, second and six iterations through the estimation phase, respectively. **h:** The hidden target robot morphology.

Inferring Morphology

More recently, we have performed initial experiments in which the algorithm can reconstruct, in simulation, the robot's morphology, and sensor and motor distribution, only using sensor data. An example run is shown in figure 3. Figure 3h shows the morphology of the target robot; figure 3a-d show the random robot simulators generated during the initial cycle of the algorithm; and figure 3e-g show the best robot simulators output by the algorithm after the first, second and sixth cycles of the algorithm.

The algorithm was able to closely approximate the target robot only using distance sensors, which return a scalar value indicating how far each body part is from an external point source. Such virtual sensors could be instantiate on physical robots using light sensors, in which light intensity indicates distance from a light source, or infrared sensors.

This ability of the algorithm to reconstruct a robot 'from scratch' would be useful for situations in which remote robots are already operating in the field, and a new generation of replace-

ment robots are required. By incorporating the estimation-exploration algorithm into an integrated robotic design and manufacturing machine, the machine could potentially: collect data from the operating robots; reconstruct their morphologies (and possibly their task environment); use the constructed simulator to re-design an improved robot, and manufacture it for deployment in the field.

Multiple Robots

In our work so far, we have worked with individual robots, each of which maintains a copy of the estimation-exploration algorithm running onboard (figure 1a). In the near future, we envision extensions in which multiple robots work together on a collaborative task, and each maintains a local copy of the algorithm. By networking these algorithms together (either through a distributed topology or a master-slave configuration, as portrayed in figure 1b), the robots would in essence serve two separate, but interdependent functions: they would work together on a collaborative task as set by the designer; and they would accumulate knowledge about the task environment itself, and how this environment can alter their own states in unforseen ways.

Consider the following scenario: if the robots are working in a dusty environment, this may cause their motors to degrade in particular patterns that are difficult for the original designer to predict, and thus include in a robot simulator accurately. However the estimation-exploration algorithm could automatically create a correlation between such motor patterns (which would surface indirectly in recorded sensor data) and a dusty environment. The subset of robots working in dusty zones would automatically update their simulators, and broadcast these simulations (along with an appropriate compensating controller) to the other robots which have not yet encountered this environment. Then, if the recipient robots encounter a dusty environment, they can rapidly switch to a compensating controller or behavior without having to re-infer the altered environment. Finally, if the acquired knowledge is sent to a design/manufacture machine, the machine could produce replacement robots that are not as susceptible to the degradations experienced by the already-deployed robots.

Conclusions

To summarize, we have briefly outlined our work on coupling an inference algorithm to a field robot, such that the field robot can automatically update an onboard robot simulator in response to unanticipated morphological or environmental changes, and use it to evolve a compensating controller that allows the robot to continue with its mission.

We have here extrapolated from that work to a robot ecology situation, in which the robots are operating in a remote environment, and are being automatically produced by an integrated robotic design and manufacture machine. In such a scenario, robots deployed in the field would: accumulate data about the environment in which they are operating; infer probable morphological degradations they may incur as a result of that environment; evolve compensating controllers; and finally broadcast the acquired knowledge to neighboring robots and the design/manufacturing machine to design and deploy the improved robots.

The complexity of designing, manufacturing and deploying sustainable robot ecologies in remote environments cannot be overemphasized. It is the opinion of the authors that the best approach to achieving this goal is to automate as many aspects of this process as possible, some of which we have outlined in this paper.

Acknowledgments

This research was conducted using the resources of the Cornell Theory Center, which receives funding from Cornell University, New York State, federal agencies, foundations, and corporate partners. This work was supported by the U.S. Department of Energy, grant DE-FG02-01ER45902.

References

[1] J. Bongard and R. Pfeifer. Evolving complete agents using artificial ontogeny. *Morpho-functional Machines: The New Species (Designing Embodied Intelligence)*, pages 237–258, 2003.

[2] J. C. Bongard and H. Lipson. Automated robot function recovery after unanticipated failure or environmental change using a minimum of hardware trials. In *Proceedings of The 2004 NASA/DoD Conference on Evolvable Hardware*, pages 169–176, Seattle, WA, 2004.

[3] H. Lipson and J. B. Pollack. Automatic design and manufacture of artificial lifeforms. *Nature*, 406:974–978, 2000.

[4] S. Nolfi and D. Floreano. *Evolutionary Robotics: The Biology, Intelligence, and Technology of Self-Organizing Machines.* MIT Press, Boston, MA, 2000.

[5] K. Sims. Evolving 3D morphology and behaviour by competition. *Artificial Life IV*, pages 28–39, 1994.