Resilient Machines Through Continuous Self-Modeling

Josh Bongard,[1]*† Victor Zykov,[1] Hod Lipson[1,2]

Animals sustain the ability to operate after injury by creating qualitatively different compensatory behaviors. Although such robustness would be desirable in engineered systems, most machines fail in the face of unexpected damage. We describe a robot that can recover from such change autonomously, through continuous self-modeling. A four-legged machine uses actuation-sensation relationships to indirectly infer its own structure, and it then uses this self-model to generate forward locomotion. When a leg part is removed, it adapts the self-models, leading to the generation of alternative gaits. This concept may help develop more robust machines and shed light on self-modeling in animals.

Robotic systems are of growing interest because of their many practical applications as well as their ability to help understand human and animal behavior (*1*–*3*), cognition (*4*–*6*), and physical performance (*7*). Although industrial robots have long been used for repetitive tasks in structured environments, one of the long-standing challenges is achieving robust performance under uncertainty (*8*). Most robotic systems use a manually constructed mathematical model that captures the robot's dynamics and is then used to plan actions (*9*). Although some parametric identification methods exist for automatically improving these models (*10*–*12*), making accurate models is difficult for complex machines, especially when trying to account for possible topological changes to the body, such as changes resulting from damage.

[1]Mechanical and Aerospace Engineering, Cornell University, Ithaca, NY 14853, USA. [2]Computing and Information Science, Cornell University, Ithaca, NY 14853, USA.

*Present address: Department of Computer Science, University of Vermont, Burlington, VT 05405, USA.

†To whom correspondence should be addressed. E-mail: josh.bongard@uvm.edu

Although much progress has been made in allowing robotic systems to model their environment autonomously (*8*), relatively little is known about how a robot can learn its own morphology, which cannot be inferred by direct observation or retrieved from a database of past experiences (*13*). Without internal models, robotic systems can autonomously synthesize increasingly complex behaviors (*6*, *14*–*16*) or recover from damage (*17*) through physical trial and error, but this requires hundreds or thousands of tests on the physical machine and is generally too slow, energetically costly, or risky.

Here, we describe an active process that allows a machine to sustain performance through an autonomous and continuous process of self-modeling. A robot is able to indirectly infer its own morphology through self-directed exploration and then use the resulting self-models to synthesize new behaviors. If the robot's topology unexpectedly changes, the same process restructures its internal self-models, leading to the generation of qualitatively different, compensatory behavior. In essence, the process enables the robot to continuously diagnose and recover from damage. Unlike other approaches to damage recovery, the concept introduced here does not presuppose built-in redundancy (*18*, *19*), dedicated sensor arrays, or contingency plans designed for anticipated failures (*20*). Instead, our approach is based on the concept of multiple competing internal models and generation of actions to maximize disagreement between predictions of these models.

The process is composed of three algorithmic components that are executed continuously by the physical robot while moving or at rest (Fig. 1): Modeling, testing, and prediction. Initially, the robot performs an arbitrary motor action and records the resulting sensory data (Fig. 1A). The model synthesis component (Fig. 1B) then synthesizes a set of 15 candidate self-models using stochastic optimization to explain the observed sensory-actuation causal relationship. The action synthesis component (Fig. 1C) then uses these models to find a new action most likely to elicit the most information from the robot. This is accomplished by searching for the actuation pattern that, when executed on each of the candidate self-models, causes the most disagreement across the predicted sensor signals (*21*–*24*). This new action is performed by the physical robot (Fig. 1A), and the model synthesis component now reiterates with more available information for assessing model quality. After 16 cycles of this process have terminated, the most accurate model is used by the behavior synthesis component to create a desired behavior (Fig. 1D) that can then be executed by the robot (Fig. 1E). If the robot detects unexpected sensor-motor patterns or an external signal as a result of unanticipated morphological change, the robot reinitiates the alternating cycle of modeling and exploratory actions to produce new models reflecting the change. The new most accurate model is now used to generate a new, compensating behavior to recover functionality. A complete sample experiment is shown in Fig. 2.

We tested the proposed process on a four-legged physical robot that had eight motorized joints, eight joint angle sensors, and two tilt sensors. The space of possible models comprised any planar topological arrangement of eight limbs, including chains and trees (for examples, see Figs. 1 and 2). After damage occurs, the space of topologies is fixed to the previously inferred morphology, but the size of the limbs can be scaled (Fig. 2, N and O). The space of possible actions comprised desired angles that the motors were commanded to reach (*25*). Many other self-model representations could replace the explicit simulations used here, such as artificial neural or Bayesian networks, and other sensory modalities could be exploited, such as pressure and acceleration (here the joint angle sensors were used only to verify achievement of desired angles, and orientation of the main body was used only for self-model synthesis). Nonetheless, the use of implicit representations such as artificial neural networks—although more biologically plausible than explicit simulation—would make the validation of our theory more challenging, because it would be difficult to assess the correctness of the model (which can be done by visual inspection for explicit simulations). More important, without an explicit representation, it is difficult to reward a model for a task such as forward locomotion (which requires predictions about forward displacement) when the model can only predict orientation data.

The proposed process was compared with two baseline algorithms, both of which use random rather than self-model–driven data acquisition. All three algorithm variants used a similar amount of computational effort (~250,000 internal model simulations) and the same number (16) of physical actions (Table 1). In the first baseline algorithm, 16 random actions were executed by the physical robot (Fig. 1A), and the resulting data were supplied to the model synthesis component for batch training (Fig. 1B). In the second baseline algorithm, the action synthesis component output a random action, rather than searching for one that created disagreement among competing candidate self-models. The actions associated with Fig. 1, A to C,

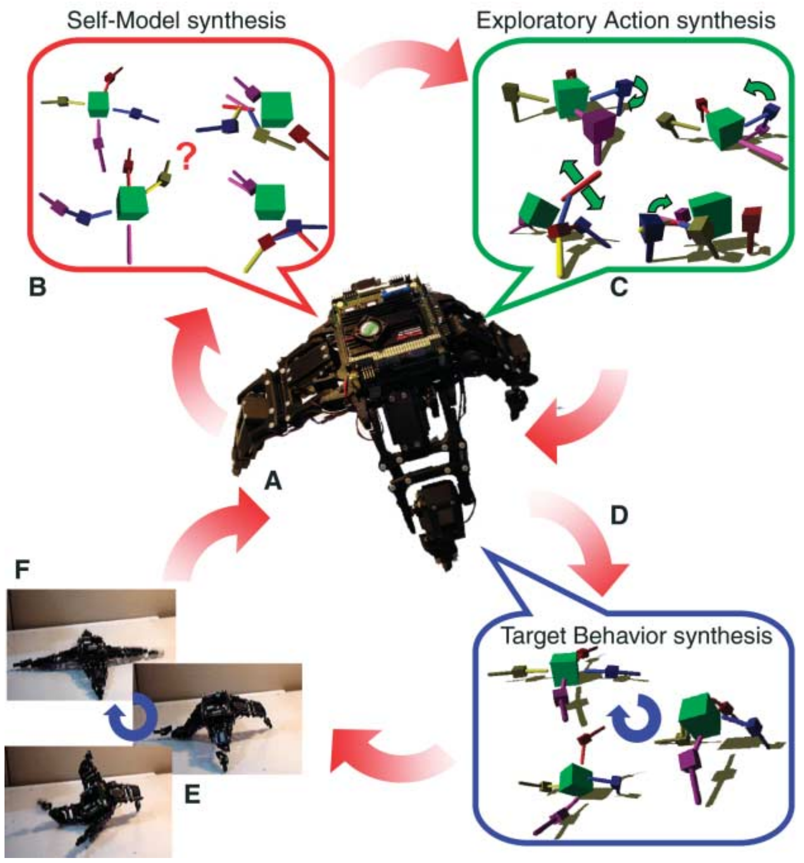

Fig. 1. Outline of the algorithm. The robot continuously cycles through action execution. (**A** and **B**) Self-model synthesis. The robot physically performs an action (A). Initially, this action is random; later, it is the best action found in (C). The robot then generates several self-models to match 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.

were cycled as in the proposed algorithm, but Fig. 1C output a random action, rather than an optimized one.

Thirty experiments of each of the three algorithm variants were conducted, both before and after the robot suffered damage. Before damage, the robot began each experiment with a set of random models; after damage, the robot began with the best model produced by the model-driven algorithm (Fig. 2F). We found that the probability of inferring a topologically correct model was notably higher for the model-driven algorithm than for either random baseline algorithm (Table 1) and that the final models were more accurate on average in the model-driven algorithm than in either random baseline algorithm (Table 1). Similarly, after damage, the robot was better able to infer that one leg had been reduced in length using the model-driven algorithm than it could using either baseline algorithm (Table 1). This indicates that alternating random actions with modeling, compared with simply performing several actions first and then modeling, does not improve model synthesis (baseline 2 does not outperform baseline 1), but a robot that actively chooses which action to perform next on the basis of its current set of hypothesized self-models has a better chance of successfully inferring its own morphology than a robot that acts randomly (the model-driven algorithm outperforms baseline algorithms 1 and 2).

Because the robot is assumed not to know its own morphology a priori, there is no way for it to determine whether its current models have captured its body structure correctly. We found that disagreement among the current model set (information that is available to the algorithm) is a good indicator of model error (the actual inaccuracy of the model, which is not available to the algorithm), because a positive correlation exists between model disagreement and model error across the $n = 30$ experiments that use the model-driven algorithm (Spearman rank correlation = 0.425, $P < 0.02$). Therefore, the experiment that resulted in the most model agreement (through convergence toward the correct model) was determined to be the most successful from among the 30 experiments performed, and the best model it produced (Fig. 2F) was selected for behavior generation. This was also the starting model that the robot used when it suffered unexpected damage (Table 1).

The behavior synthesis component (Fig. 1D) was executed 30 times with this model, starting each time with a different set of random behaviors. Figure 3 reports the final positions predicted by the model robot using the best behavior produced by each experiment (black dots). Each of those 30 behaviors was then executed on the physical robot, and the resulting actual positions are reported in Fig. 3 (blue dots). As a control, 30 random behaviors were also executed on the physical robot (red dots). Although there is some discrepancy between the predicted distance and actual distance, there is a clear forward motion trend that is absent from the random behaviors. This indicates that this automatically generated self-model was sufficiently predictive to allow the robot to consistently develop forward motion patterns without further physical trials. One of the better locomotion patterns is shown in fig. S1A. The transferal from the self-model to reality was not perfect, although the gaits were qualitatively similar; differences between the simulated and physical gait (seen at 2.6 and 5.2 s) were most likely due to friction and kinematic bifurcations at symmetrical postures, both difficult to predict. Similarly, after damage, the robot was able to synthesize sufficiently accurate models (an example is given in Fig. 2O) for generating new, compensating behaviors that enabled it to continue moving forward. An example of a compensating gait is shown in fig. S1B and movie S1.

Although the possibility of autonomous self-modeling has been suggested (*26*), we demonstrated for the first time a physical system able to autonomously recover its own topology with little prior knowledge, as well as optimize the parameters of those resulting self-models after unexpected morphological change. These processes demonstrate both topological and parametric self-modeling. This suggests that future machines may be able to continually detect changes in their own morphology (e.g., after damage has occurred or when grasping a new tool) or the environment (when the robot enters an unknown or changed environment) and use the inferred models to generate compensatory behavior. Beyond robotics, the ability to actively generate and test hypotheses can lead to general nonlinear and topological system identification (*23*) in other domains, such as computational systems (*22*), biological networks (*23*), damaged structures (*24*), and even automated science (*27*).

This work may inform future investigations of cognition in animals and the development of cognition in machines. Whereas simple yet robust behaviors can be created for robots without recourse to a model (*14–17*, *28*), higher animals require predictive forward models to function, given that in many cases biological sensors are

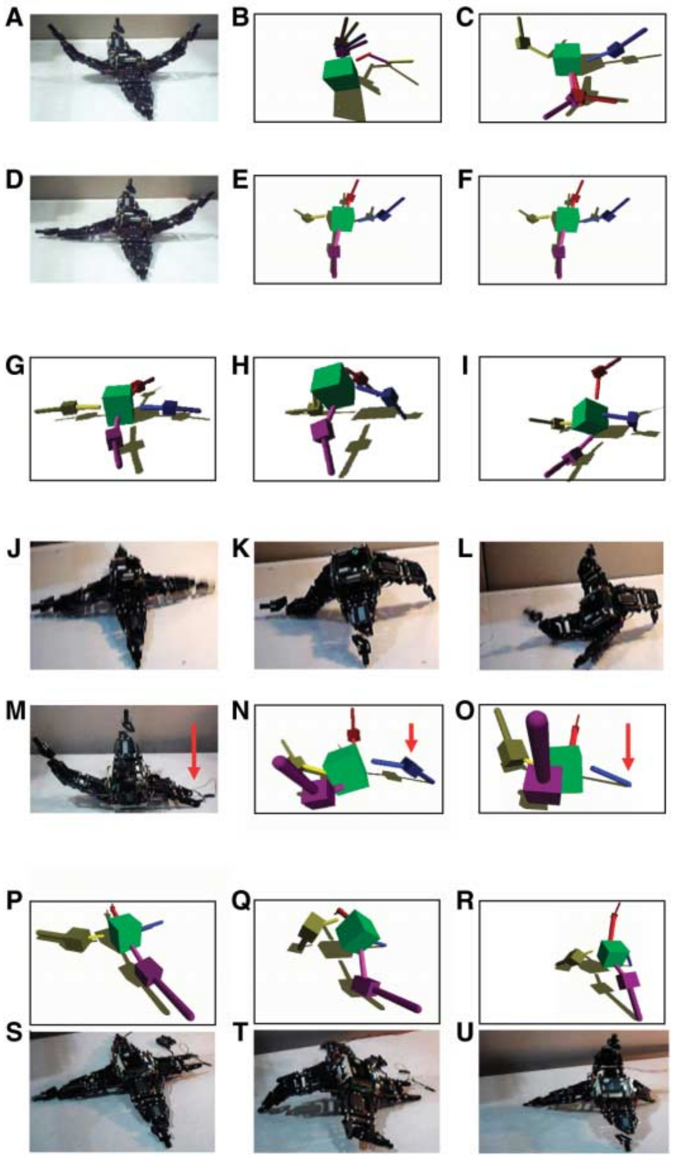

Fig. 2. The robot continually models and behaves. The robot performs a random action (**A**). A set of random models, such as (**B**), is synthesized into approximate models, such as (**C**). A new action is then synthesized to create maximal model disagreement and is performed by the physical robot (**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.

Table 1. Performance summary for the three algorithm variants. Baseline algorithms 1 and 2 disable the iterative and the model-driven nature of the learning process, respectively, while ensuring that the same computational effort and number of physical actions are used. Before damage, a successful experiment is determined as one that outputs a model with correct topology (see fig. S2 for examples of correct and incorrect topologies). Mean model error was calculated over the best model from each of the 30 experiments. Mean values are reported ± SD. An additional 90 experiments were conducted after the robot was damaged. The robot reinitiates modeling at this point using the most accurate model from the first 90 experiments (Fig. 2F). In this case, mean model error is determined as the difference between the inferred length of the damaged leg and the true damaged length (9.7 cm).

	Baseline 1	Baseline 2	Model-driven algorithm
Before damage			
Independent experiments (*n*)	30	30	30
Physical actions per experiment	16	16	16
Mean model evaluations (*n* = 30)	262,080 ± 13,859	246,893 ± 17,469	262,024 ± 13,851
Successful self-models	7	8	13
Success rate	23.3%	26.7%	43.3%
Mean model error (*n* = 30)	9.62 ± 1.47 cm	9.7 ± 1.45 cm	7.31 ± 1.22 cm
After damage			
Independent experiments (*n*)	30	30	30
Physical actions per experiment	16	16	16
Mean model evaluations (*n* = 30)	292,430 ± 44,375	278,140 ± 37,576	296,000 ± 22,351
Mean model error (*n* = 30)	5.60 ± 2.98 cm	4.55 ± 3.22 cm	2.17 ± 0.55 cm

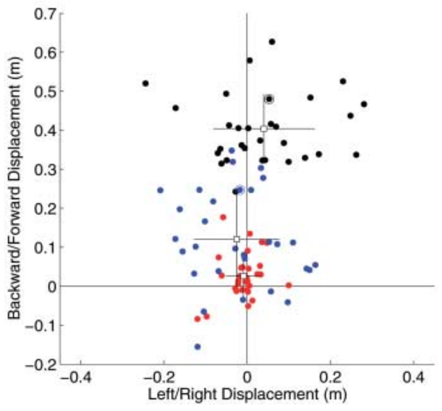

Fig. 3. Distance traveled during optimized versus random behaviors. Dots indicate the final location of the robot's center of mass, when it starts at the origin. Red dots indicate final positions of the physical robot when executing random behaviors. Black dots indicate final expected positions predicted 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.

not fast enough to provide adequate feedback during rapid and complex motion (*29*). Although it is unlikely that organisms maintain explicit models such as those presented here, the proposed method may shed light on the unknown processes by which organisms actively create and update self-models in the brain, how and which sensor-motor signals are used to do this, what form these models take, and the utility of multiple competing models (*30*). In particular, this work suggests that directed exploration for acquisition of predictive self-models (*31*) may play a critical role in achieving higher levels of machine cognition.

References and Notes

1. B. Webb, *Behav. Brain Sci.* **24**, 1033 (2001).
2. R. Arkin, *Behavior-Based Robotics* (MIT Press, Cambridge, MA, 1998).
3. R. J. Full, D. E. Koditschek, *J. Exp. Biol.* **202**, 3325 (1999).
4. R. Pfeifer, *Int. J. Cognit. Technol.* **1**, 125 (2002).
5. T. Christaller, *Artif. Life Robot.* **3**, 221 (1999).
6. S. Nolfi, D. Floreano, *Evolutionary Robotics: The Biology, Intelligence, and Technology of Self-Organizing Machines* (MIT Press, Cambridge, MA, 2000).
7. S. H. Collins, A. Ruina, R. Tedrake, M. Wisse, *Science* **307**, 1082 (2005).
8. S. Thrun, W. Burgard, D. Fox, *Probabilistic Robotics* (MIT Press, Cambridge, MA, 2005).
9. L. Sciavicco, B. Siciliano, *Modelling and Control of Robot Manipulators* (Springer-Verlag, London, 2001).
10. E. Alpaydin, *Introduction to Machine Learning* (MIT Press, Cambridge, MA, 2004).
11. K. Kozlowski, *Modelling and Identification in Robotics* (Springer-Verlag, London, 1998).
12. L. Ljung, *System Identification: Theory for the User* (Prentice-Hall, Englewood Cliffs, NJ, 1999).
13. D. Keymeulen, M. Iwata, Y. Kuniyoshi, T. Higuchi, *Artif. Life* **4**, 359 (1998).
14. P. F. M. J. Verschure, T. Voegtlin, R. J. Douglas, *Nature* **425**, 620 (2003).
15. G. S. Hornby, S. Takamura, T. Yamamoto, M. Fujita, *IEEE Trans. Robot.* **21**, 402 (2005).
16. R. Pfeifer, C. Scheier, *Understanding Intelligence* (MIT Press, Cambridge, MA, 1999).
17. S. H. Mahdavi, P. Bentley, *Auton. Robots* **20**, 149 (2006).
18. M. L. Visinsky, J. R. Cavallaro, I. D. Walker, *Reliab. Eng. Syst. Saf.* **46**, 139 (1994).
19. F. Caccavale, L. Villani, P. Ax, Eds., *Fault Diagnosis and Fault Tolerance for Mechatronic Systems* (Springer Verlag, New York, 2002).
20. S. Zilberstein, R. Washington, D. S. Benstein, A.-I. Mouaddib, *Lect. Notes Comput. Sci.* **2466**, 270 (2002).
21. H. S. Seung, M. Opper, H. Sompolinsky, in *Proceedings of the 5th Workshop on Computational Learning Theory* (ACM Press, New York, 1992), pp. 287–294.
22. J. Bongard, H. Lipson, *J. Mach. Learn. Res.* **6**, 1651 (2005).
23. J. Bongard, H. Lipson, *Trans. Evol. Comput.* **9**, 361 (2005).
24. B. Kouchmeshky, W. Aquino, J. Bongard, H. Lipson, *Int. J. Numer. Methods Eng.*, in press; published online 31 July 2006 (doi: 10.1002/nme.1803).
25. Materials and methods are available as supporting material on *Science* Online.
26. R. A. Brooks, in *Proceedings of the 1st European Conference on Artificial Life,* F. J. Varela, P. Bourgine, Eds. (Springer-Verlag, Berlin, 1992), pp. 3–10.
27. R. D. King *et al.*, *Nature* **427**, 247 (2004).
28. U. Saranli, M. Buehler, D. E. Koditschek, *Int. J. Robot. Res.* **20**, 616 (2001).
29. A. Maravita, C. Spence, J. Driver, *Curr. Biol.* **13**, R531 (2003).
30. G. Edelman, *Neural Darwinism: The Theory of Neuronal Group Selection* (Basic Books, New York, 1987).
31. F. Crick, C. Koch, *Nat. Neurosci.* **6**, 119 (2003).
32. This research was supported in part by the NASA Program for Research in Intelligent Systems under grant NNA04CL10A and the NSF grant number DMI 0547376.

Supporting Online Material
www.sciencemag.org/cgi/content/full/314/5802/1118/DC1
Materials and Methods
Figs. S1 and S2
References
Movie S1

9 August 2006; accepted 4 October 2006
10.1126/science.1133687