PRECOG: Predictable Robot Evolves via Control Optimization and GP

Amanda Bertschinger[1], Piper Welch[1], James Bagrow[2], and Josh Bongard[1]

[1]Department of Computer Science, University of Vermont, USA
[2]Department of Mathematics and Statistics, University of Vermont, USA
ambertsc@uvm.edu

Abstract

Evolutionary robotics methods optimize the physical and/or neural aspects of robots against some desired task. Generally, these robots are only optimized for efficacy: how well they perform the task. How they perform the task is usually ignored (with the exception of task-specific constraints). However, this can lead to efficacious but complex and/or unpredictable behaviors. To remedy this issue, we show here for the first time that we can obtain robots of equal behavioral task efficacy and better predictability than robots that are only evolved for task efficacy, if robots are bi-objectively evolved to (1) exhibit a desired behavior and (2) yield behavioral data compressible by symbolic regression into increasingly predictive equations. Our results demonstrate that the potential tradeoff between behavioral task efficacy and predictability can be ameliorated through optimization against both desiderata. In scenarios where accuracy of prediction is critical, this approach may provide an advantage over robots only optimized for task efficacy. Further, the readability of symbolically regressed equations may, in future, make evolved behaviors understandable as well. This could in turn render future machines not just autonomous, useful, and predictable, but also trustworthy.

Data/Code available at: `https://github.com/mec-lab/PRECOG`

Introduction

Evolutionary robotics is the use of evolutionary algorithms to optimize the morphology and/or neural control policy of robots (Bongard, 2013; Doncieux et al., 2015; Nolfi et al., 2016). These evolutionary algorithms take inspiration from natural evolution, making random changes to the body and/or brain of a population of individuals. Then, the best performing population members on some specified task are selected to survive in the population, giving rise to the next generation. Through this, increasingly better performance in the population is optimized over the generations. As making such random updates to an entire population of robots can be difficult or impossible *in materia* (Floreano et al., 1994; Floreano and Mondada, 1998; Zykov et al., 2004), simulation is often employed to allow for more efficient evolution. Through use of physics simulators, an *in silico* robot can be evaluated on a virtual task that can mimic the important aspects of the real environments that a physical version of the robot may one day be in. Modifying both the morphology and the controller of an *in silico* robot is much simpler and more efficient than performing modifications on their *in materia* counterparts (Doncieux et al., 2009; Glette et al., 2012; Gongora et al., 2009).

However, evolved robots can exhibit unpredictable behaviors when exposed to unexpected scenarios (Schoppers, 1987; Wang et al., 2012). These behaviors can be harmful to an *in materia* robot with the evolved *in silico* morphology and/or controller. Unknown factors combined with the unpredictable movement can be harmful to *in materia* robots or their environments. For example, in 2022 a chess-playing robot broke a child's finger. The robot required time to respond to the moves of the human player, and the child made an unexpectedly quick motion. This led to the robot grabbing the child's finger and breaking it. As the ultimate goal of evolutionary robotics is to produce *in materia* robots that will interact with humans and the world around us, unpredictable behaviors can also be harmful to humans' opinions of and interaction with robots (Liberman-Pincu and Oron-Gilad, 2024; Mubin and Bartneck, 2015; Schadenberg et al., 2021). Thus, it is important for a robot to be predictable in its behaviors. Due to this, more emphasis has recently been placed on the explainability of robots (Cruz et al., 2023; Sado et al., 2023; Zhou et al., 2024). These studies, however, tend to focus on explaining why the robots behave the way they do through explainable artificial intelligence. Such methods focus on the ability to explain the robot's behaviors in human-understandable ways through providing technical information and/or more general explanations for non-expert users. This is distinct from our focus, which is the predictability of the behaviors themselves.

Part of this lack of predictability is due to how the robots are typically evolved. During evolution, robots that survive in the population are chosen by some fitness metric related to a task they perform. The robots are then evaluated on how well they have performed this task during simulation. However, the task that the robots are evolved to perform is

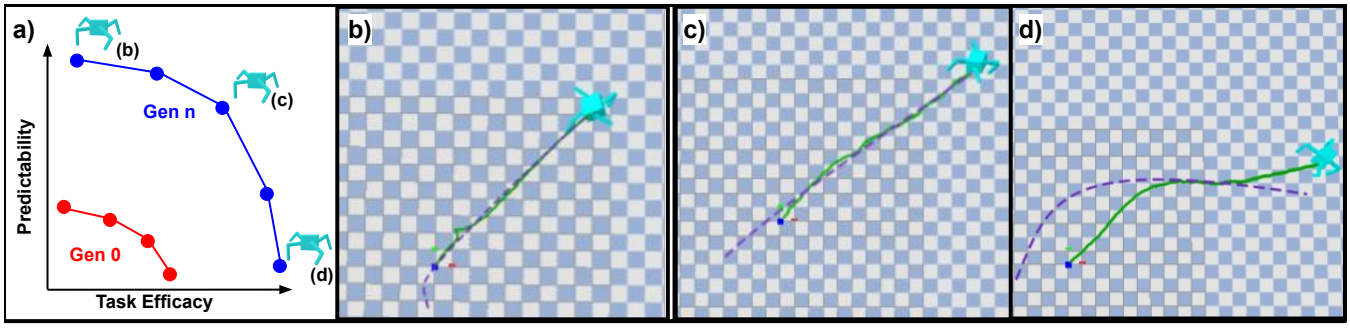

Figure 1: Predictable Robot Evolves via Control Optimization and GP (PRECOG). (a) Pareto optimization of a population of quadrupedal robots (teal) for task efficacy (x axis) and predictability (y axis). From generation 0 (red) to generation n (blue) there is an improvement in both fitnesses. The Pareto front shows robots that are (b) highly predictable but not very efficacious, (c) both predictable and efficacious, (d) are not very predictable but highly efficacious. The task efficacy of robots are shown in the distance traveled in the x direction. Predictability is shown in the comparison between simulated (green solid) and predicted (purple dashed) trajectories, with higher predictability being less difference between the two trajectories.

rarely associated with the entirety of their movement behavior. (Nelson et al., 2009) Although some tasks are related to movement, it is generally movement in a particular direction or a certain type of movement (e.g. jumping, skipping, etc.). Therefore, these tasks do not take into account all of the details of the behavior of the robot as it goes about its task. The robot can be optimized instead for predictability of behavior instead of task efficacy, however this may lead to a robot that is very poor at performing its designated task. One of the simplest things to predict is a completely stationary robot, and therefore optimizing purely for predictability could easily lead to such solutions. There has been work in minimizing surprise in robot swarms, defined as the ability of a predictive network to predict the next action that the controller network will take for individuals in a swarm (Hamann, 2014; Kaiser and Hamann, 2019, 2022b,a). However, although some of these works discussed the effect of evolving for predictability on particular task performance, in these papers the swarm fitness was purely the predictive performance and did not involve a task fitness in the evolution. Kaiser and Hamann (2020) compare evolution for purely predictability to evolution for purely task efficacy and novelty, but do not jointly evolve for any combination therein. Kaiser et al. (2023) do include sensor input in their reward function, but they are testing for the emergence of dynamic behaviors not task efficacy as it relates to the relevant sensor values. To the best of our knowledge, no one to date has evolved for both task efficacy and predictability.

Thus, the solution seems to be to not just automatically generate behaviors through evolution, but also to make these behaviors predictable. However, the definition of predictability can vary. With unlimited compute, a perfect 1-to-1 simulation of the *in materia* environment could be near-perfectly recreated, and a *in silico* twin of the robot simulated to study its behaviors. This "perfect" predictability would require details such as the explicit simulation of subatomic particles which would be so computationally expensive that it is functionally impossible. Additionally, there will always be unexpected factors that are not simulated, such as the child's quick movement discussed above. Therefore, we here define predictability as how accurately an *in silico* robot's motion can be described using an equation. An equation can implicitly represent the details of a robot's motion in an interpretable way. The evaluation of an equation will give a predicted trajectory, and this can be compared to a simulated trajectory in some time domain to then quantify how predictable the trajectory was. We chose equations for their interpretability, as other models such as neural networks tend to be black-box solutions where it can be near impossible to understand why a certain prediction is made. There have been works on self-modeling in neural networks for predictive ability (Premakumar et al., 2024; Watson et al., 2011). However, this does not overcome the aforementioned issue in black-box models. Equations, on the other hand, are generally simpler and more understandable at a glance, using well known mathematical operations to show relationships in the data.

To allow the use of equations to describe the trajectories of the *in silico* robots, we propose using symbolic regression. Symbolic regression is an algorithm to automatically distill equations from data without presupposing the form of the equation (Koza, 1992). As it is able to build an equation from scratch using a set of mathematical operators, it has an advantage over techniques like linear regression. Linear regression can fit data to a predetermined skeleton equation using constants, but will always be restricted by the operators that were chosen for the skeleton. Symbolic regression, on the other hand, has full freedom to build its own skeleton equation and can therefore more accurately represent the relationships in the data it fits.

There are three main types of symbolic regression used in literature: genetic programming symbolic regression (GPSR), neural symbolic regression (NSR) and generalized neural symbolic regression (GNSR). GPSR is the most traditional form of symbolic regression, evolving a population of equations to fit a particular dataset (Arnaldo et al., 2014; Koza, 1992, 1994; Schmidt and Lipson, 2008, 2009; Virgolin et al., 2021). Neural symbolic regression uses neural networks to discover equations, generally with the structure of the network and activation functions of the neurons being used to represent an equation (Costa et al., 2020; Martius and Lampert, 2016; Petersen et al., 2019; Sahoo et al., 2018). GNSR uses large-scale pretraining to train a transformer network on a corpus of data-equation pairs in order to obtain a transformer that can take as input a dataset and output a fitted equation (Bendinelli et al., 2023; Biggio et al., 2021; Kamienny et al., 2022; Li et al., 2022; Valipour et al., 2021; Vastl et al., 2022). Both GPSR and NSR are methods optimized per dataset, with NSR generally being more efficient than some GPSR methods, as evolutionary computation is generally less efficient than training through backpropagation. However, highly optimized GPSR methods exist that are competitive with NSR methods in terms of efficiency (La Cava et al., 2021). GNSR has the advantage of being generalized to produce any equation for any given dataset, not needing to be retrained per dataset. However, GNSR networks tend to show trouble with equation accuracy (Bertschinger et al., 2023) whereas many GPSR and NSR methods have been shown to be highly accurate (La Cava et al., 2021). Here, we use GPSR as our chosen symbolic regression method as we are concerned primarily with having high equation accuracy and prefer to use an evolutionary method to echo what we are employing for our methodology.

Symbolic regression has often historically been used to predict scientific equations to a high degree of accuracy (Angelis et al., 2023; Kim et al., 2020; Quade et al., 2016; Schmidt and Lipson, 2009; Wang et al., 2019). In fact, using symbolic regression to describe robot behaviors has been done in evolutionary robots. acero2024 use symbolic regression to describe the behaviors of robots with various gaits. Specifically, they fit equations to the robot behaviors after the robot controllers have already been evolved. In doing so, they seek to increase the interpretability of their robots' behaviors after optimization has occurred. However, for the symbolic regression model, the accuracy of their equations is consistently lower than the other models they employ for prediction of robot behavior.

The solution to this lack of predictive accuracy can be accomplished by optimizing robots for both task efficacy and predictability rather than evolving for task efficacy alone and fitting equations for predictability after optimization. Although it may first seem like adding an additional predictability objective to the optimization will harm the primary task efficacy objective, this is not necessarily the case. In fact, there cases where adding an objective to optimization will ultimately help improve the primary objective (Grasso and Bongard, 2022, 2023). In this work, however, we aim only for maintaining the task efficacy of the robots while increasing their predictability, as we concern ourselves more with the explainability and interpretability of the robot behaviors instead of their task efficacy. In literature, robots have been evolved multi-objectively for both primary task efficacy and another behavior like reactivity (Lehman et al., 2013), but to date none have been evolved for task efficacy and predictability.

Here, we show for the first time that a robot can be successfully evolved for both task efficacy and predictability of movement, such that a set of equations fit to the simulated robot movement trajectory during evolution can have higher prediction accuracy than equations fit to simulated robot movement trajectory after evolution. Additionally, we show that there is no significant loss in primary task efficacy with bi-objective evolution for both task efficacy and predictability as compared to a robot evolved for only task efficacy.

Methods

Here, we explain the methodology we use for our experiments. We describe in order the details of: our robots, the movement behaviors we evolve for, the equations used to describe said behaviors, our evolutionary optimization strategy, and finally the predictions of the movement behaviors. As a control method, we use use age-fitness Pareto optimization (AFPO) (Schmidt and Lipson, 2010) for primary task efficacy only. We select AFPO as an appropriate comparison as it is a single-objective optimization method that uses Pareto front optimization, allowing for similar methodology between PRECOG and AFPO for optimization, detailed below.

Robots

Morphology and Controller For all of our experiments, we used the same robot morphology and controller architecture, chosen for the simplicity of form that can lead to complex behaviors. The robot morphology we use is a simple quadrupedal robot with a cube body. Each of the four legs is attached to a different side of the robot (with none on the top or the bottom of the robot). Each leg has two hinge joints capable of moving: one to attach it to the robot body and one "knee" joint. Each joint has a motor which allows that joint to move as instructed by the controller at each timestep. The maximum force the motors can apply is 30 Newton-metres, while the amplitude and frequency used to determine the angle the motor should move to is $\frac{\pi}{2}$ and $\frac{1}{1000}$ respectively. The bottom of each leg has a sensor on it which senses contact with the ground. The robot also has a

positional sensor which detects the x position of the robot at each timestep.

All robot controllers consist of a simple neural network, with each sensor being assigned an input neuron and each motor being assigned an output neuron. All neurons in the network are connected with synapses to all other neurons. The synaptic weights are randomly internalized between $[-1, 1]$. Synapse weights are mutated over evolution, details of which are provided below.

Physics Simulator For simulating the robots in our experiments, we use PyBullet (Coumans and Bai, 2021), a Python library for the Bullet physics engine. We specifically use the Pyrosim package for Pybullet for ease of use, as Pyrosim simplifies several important Pybullet functions for simple robots like the quadrupedal robots that we use here. For our training simulations during evolution, we simulate the robots for 10,000 timesteps, with each timestep being $\frac{1}{240}$th of a second. We use the standard gravity of -9.81, and all other PyBullet hyperparameters are the Pyrosim defaults.

Robot Behaviors

Task Efficacy Fitness The task we evolve our robots for is a basic one: how far in the positive x direction the robot can move during the training time domain. Thus, the task efficacy fitness is the x position of the robot during the final training timestep. We choose this task for its simplicity, showing that even for simple tasks and robots there are improvements to be made in predictability of behaviors.

Predictability Fitness The predictability of robot motion is described using the R^2 value between the simulated trajectories and trajectories predicted by equations fit by symbolic regression (algorithm described below), given by

$$R^2(P, \hat{P}) = 1 - \frac{\sum_{t=0}^{T}(p_t - \hat{p}_t)^2}{\sum_{t=1}^{T}(p_t - \bar{p})^2} \tag{1}$$

where P and $\hat{P}$ are the true and predicted trajectory (X, Y) values respectively, $\bar{p}$ is the mean of P values, and T is the number of timesteps.

Equations

To obtain equations that describe robot trajectories, we use the genetic programming symbolic regression algorithm PySR (Cranmer, 2023). For PySR, we use a mathematical operator set of $[t, +, -, *, /, pow, \sin, \cos, \tan, \exp, \log]$, a maximum equation size of 20 terms, a maximum equation tree depth of 7, 15 iterations of evolutionary fitting per equation, and a timeout of 90 seconds to best fit an equation. The timeout is in place to ensure computational efficiency. After 90 seconds or 15 iterations, the best performing equations by PySR's default fitness score will be considered the fitted equations. For each robot (X, Y) trajectory, two infix ordered equations are saved. In PRECOG, the equations are fit during evolution to each population member's entire simulated trajectory. In AFPO, the entire simulated trajectories saved and the equations are fit after evolution ends for the best performing population member (obtained by finding the knee of the Pareto front) for each independent run.

Optimization

Here, we discuss details of our evolutionary algorithm, including the evaluation of population members, selection of population members during evolution, and details of how population members are mutated to produce offspring.

For our evolutionary algorithm, we use bi-objective pareto optimization, with the designated task efficacy and the predictability of the robots' motion being the two objectives we optimize for in PRECOG. For AFPO, following the standard, robot task efficacy and age are the two objectives optimized for. We use a population size of 50 (25 parents and 25 children) and evolve for 50 generations for both PRECOG and AFPO.

Selection During evolution, at each generation we use a bi-objective selection strategy, with our two fitnesses being task efficacy fitness and predictability fitness in PRECOG and task efficacy fitness and age in AFPO respectively. We use survivor selection, where two population members are randomly chosen to compete. They are compared for task efficacy fitness and predictability/age, and if either of the robots are dominated, that robot is then removed from the population. In PRECOG, domination is when both task efficacy fitness and predictability fitness are higher. In AFPO, domination is when task efficacy fitness is higher and age is lower. In the case that the two robots have the same task and predictability fitness, the younger robot is chosen to survive. This continues until the population size matches the assigned number of parents.

After evolution, the best-performing population members at the final generation are selected by Pareto domination. The kneedle algorithm (Satopaa et al., 2011) is then used to select a robot from the Pareto front for each independent trial for both PRECOG and AFPO. These robots are combined into their respective best-performing population for evaluation.

Mutation We only mutate the controllers of the robots in this experiment, leaving the morphology consistent across all robots. Specifically, the synapse weights of the neural network controllers are what is mutated during evolution. At each generation, for each parent, a random synapse is chosen to be mutated, randomly assigning a new value for the synaptic weight between $[-1, 1]$, to produce the children. All children are then simulated and evaluated on their performance, and the new parents are chosen using the selection strategy described above.

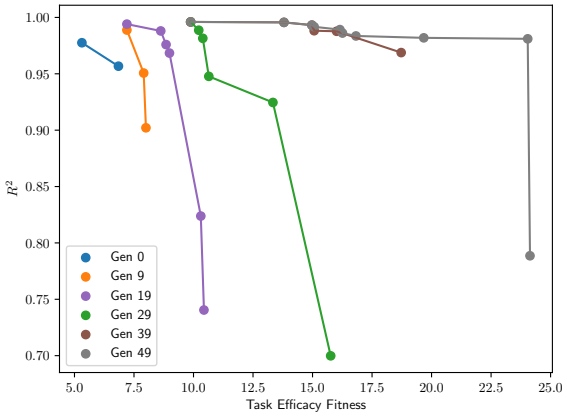

Figure 2: PRECOG Pareto fronts every 10 generations for a chosen sample trial out of the 30 independent trials. X axis is the task efficacy fitness and Y axis is the predictability fitness (R^2).

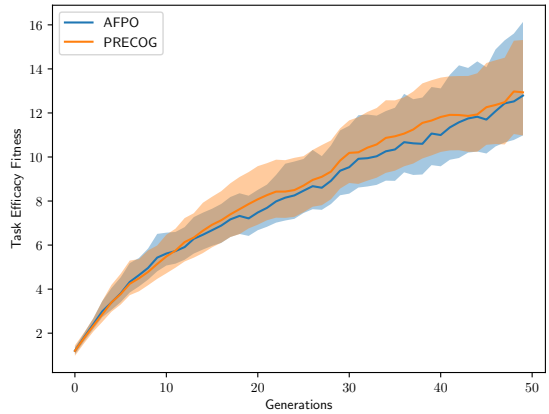

Figure 3: Average task efficacy fitness over generations for AFPO (blue) and PRECOG (orange). Shaded areas are the 95% confidence intervals for each method.

Prediction

In order to obtain and evaluate the predictions of robot behavior for obtaining the R^2 fitness, we evaluate the equations obtained through symbolic regression. The two equations $x(t)$ and $y(t)$ are evaluated for each timestep $t = [0, ..., 9999]$ to obtain the predicted behavior of the robot. In PRECOG, this evaluation happens during evolution in order to obtain the R^2 fitness of the prediction in comparison to the simulated trajectory. For the AFPO control, this equation evaluation happens after evolution.

Results

In this section, we present the results of our optimization, robot behaviors, and the predictions made from the equations fitted to the robot behaviors. All data presented here is from 30 independent trials of PRECOG and 30 independent trials of AFPO, following the above methodology. All single-trial results presented are from one randomly selected trial out of the 30 PRECOG trials.

Evolutionary Results

Here, we discuss the results of our evolutionary optimization through a sample Pareto front of one of the 30 independent trials and the fitness over time of the task efficacy of both PRECOG and the AFPO control.

Pareto Fronts Figure 2 shows the Pareto fronts over evolutionary time for a sample trial from the 30 independent trials. The Pareto fronts every 10 generations show the increasing fitness for both task efficacy and predictability over generations.

Fitness Over Time Figure 3 shows the average task efficacy fitness over time of both our 30 independent PRECOG trials as well as the 30 independent AFPO trials. The average of PRECOG task efficacy closely follows the average of AFPO task efficacy, with the average PRECOG task efficacy lying within the 95% confidence interval of the average AFPO task efficacy.

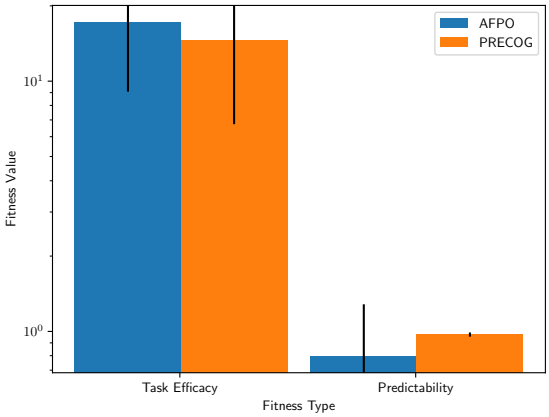

Figure 4: Left: Mean task efficacy fitness of the best-performing population of AFPO (blue) and PRECOG (orange) respectively. Right: Mean predictability fitness of the best-performing population of AFPO (blue) and PRECOG (orange) respectively. Error bars show one standard deviation for each respective population.

Task Efficacy and Predictability

Figure 4(left) shows the mean task efficacy fitness of both the AFPO control and PRECOG best-performing populations. The mean task efficacy of AFPO is 17.21 and the mean task efficacy of PRECOG is 14.55. In spite of the higher mean task efficacy of AFPO, the PRECOG and AFPO best-performing populations task efficacy fitness have a $p > 0.05$ as calculated by the Mann-Whitney U

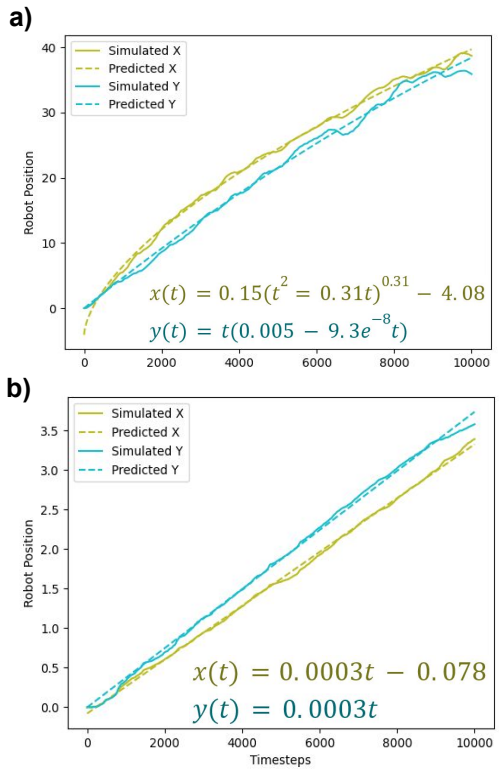

Figure 5: For the meta-Pareto front of best performing population members over 30 independent trials: (a) simulated (solid) and predicted (dashed) X (green) and Y (blue) trajectories (top) and corresponding equations (bottom) for the best performing robot by task efficacy. (b) Simulated and predicted X and Y trajectories (top) and corresponding equations (bottom) for the best performing robot by predictability.

test (Mann and Whitney, 1947), showing no statistically significant difference between our PRECOG method and the AFPO control for task efficacy.

Figure 4(right) shows the mean predictability fitness of the best-performing population for both PRECOG and AFPO. The PRECOG and AFPO best-performing populations have a mean R^2 value of 0.972 and 0.797 respectively. There is a p-value of 0.02 as calculated by the Mann-Whitney U test that the PRECOG population's predictability is greater than the AFPO population's.

Equations and Predictions

Figure 5 shows the simulated and predicted trajectories of the two robots in the meta-Pareto front, as well as the corresponding equations from which the predicted trajectories are gotten. The two members of the meta-Pareto show the two cases of higher task efficacy and lower predictability as well as lower task efficacy and higher predictability. In this meta-Pareto, there is no knee robot that shows both high task efficacy and predictability. Both robots in the meta-Pareto have a high prediction accuracy of $R^2 > 0.99$. The robot with a higher task efficacy has an task efficacy fitness of 38.69 and a predictability fitness of 0.996. The robot with the lower task efficacy has a lower task efficacy fitness of only 3.39 and a slightly higher predictability fitness of 0.998. The robots in both Figure 5(a) and (b) have a predicted trajectory that closely matches the simulated trajectory, matching the high $R^2 > 0.99$ that both have as a predictability fitness.

Figure 6(a)(b)(c) show the simulated and predicted trajectories for the highest task efficacy fitness, kneedle algorithm selected Pareto-knee, and highest predictability fitness robots from the sample trial respectively. The robot in Figure 6(a) has an task efficacy fitness of 24.13 and a predictability fitness of 0.789. This is reflected in how the predicted trajectory does not closely match the simulated trajectory throughout the Y trajectory especially. The robot in Figure 6(b) has an task efficacy fitness of 9.88 and a predictability fitness of 0.996. The robot in Figure 6 has a task efficacy fitness of 16.15 and a predictability fitness of 0.989. In both cases, the higher predictability fitness R^2 are reflected in the predicted trajectories closely matching the simulated trajectories.

Discussion

Figure 2 shows that robots can be bi-objectively evolved for both task efficacy and predictability. Over increasing generations, both the task efficacy and predictability of each generations' Pareto fronts increase. However, the predictability fitness does increase faster, with high predictability fitness of $R^2 > 0.99$ by achieved by generation 19. Generations 29, 39, and 49 still show that more of the Pareto front is consistently achieving higher predictability fitnesses over generations. The larger increase in task efficacy fitness over the successive generations can be explained by the fact that task efficacy fitness has no upper limit whereas the maximum R^2 score is 1.0.

Figure 4 shows that this bi-objective evolution does not come at the cost of decreased task efficacy fitness. Compared to the AFPO control that is evolved for only task efficacy and not predictability, the PRECOG best-performing population is not statistically significantly lower, backed up by a $p > 0.5$. This shows that although the mean AFPO task efficacy is higher than the mean PRECOG task efficacy, the bi-objective evolution does not significantly lower the task efficacy. Further, PRECOG achieves a significantly better mean predictability fitness than AFPO. Together, these figures support - for the tasks studied here - the main claim that robots can be evolved for higher predictability without sacrificing task efficacy.

PRECOG does require additional computational resources as compared to AFPO, given the necessity of fitting equations with a GPSR method. However, Figure 7 (obtained from 30 robots from independent trials) shows that the additional compute is small when compared to the

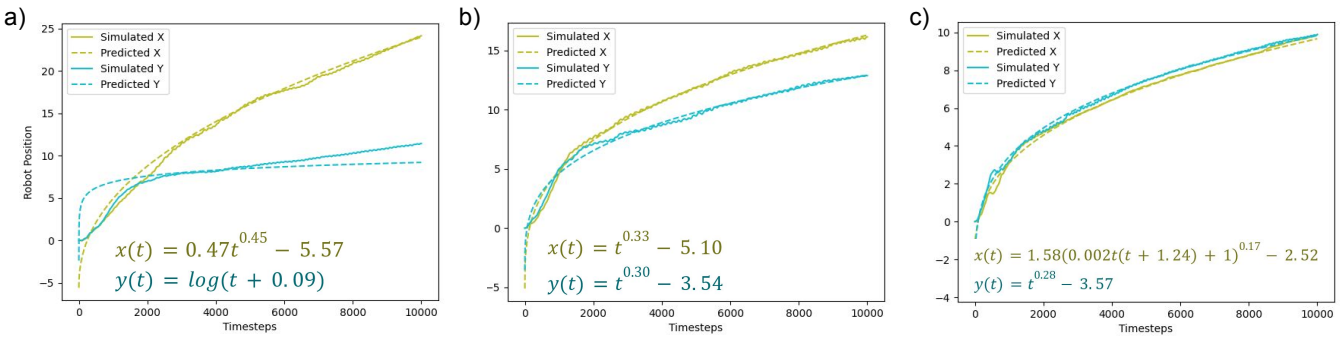

Figure 6: For a sample trial picked from the 30 independent trials: (a) simulated (solid) and predicted (dashed) X (green) and Y (blue) trajectories (top) and corresponding equations (bottom) for the best performing robot by task efficacy drawn from the final pareto front. (b) Simulated and predicted X and Y trajectories (top) and corresponding equations (bottom) for the robot drawn from the knee of the same pareto front. (c) Simulated and predicted X and Y trajectories (top) and corresponding equations (bottom) for the most predictable robot drawn from the same pareto front.

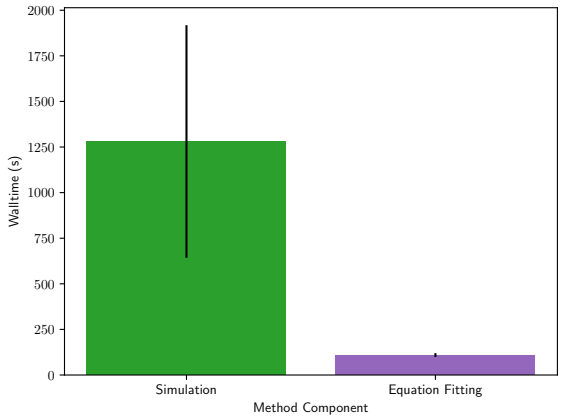

Figure 7: Mean wall-time of PyBullet simulation (green) and PySR equation fitting (purple).

amount of compute required by the simulation. The compute required by simulation is an average of 1280.29 seconds per simulation, whereas PySR equation fitting is only 109.30 seconds per set of two equations. This means that the additional compute required by the GPSR equation fitting is on average only 8.5% of the compute required by the simulation. Due to the closeness of the AFPO and PRECOG task efficacy over time seen in Figure 3, an additional approximately 10% compute given to AFPO in the form of extra generations for evolution would likely not be significant in improving task efficacy over what PRECOG achieves.

Overall, for the meta-Pareto front detailed in Figure 5, both the low-task efficacy and high-task efficacy robots have a high predictability, confirmed by their R^2 scores as well as a visual inspection of the simulated vs predicted trajectories in Figure 5. Both robots' predicted trajectories are able to match the overall behavior of the robot well. The similar predictability of the high task efficacy robot further shows that there does not have to be dramatic decrease in predictability for high task efficacy when the robot is explicitly optimized for both.

The three robots shown in Figure 6 show the potential tradeoff between task efficacy and predictability in bi-objective optimization. The high task efficacy robot (Fig. 6(a)) is able to go much further in the positive x direction than either the knee (Fig. 6(b)) or high-predictability (Fig. 6(c)) robot. This robot confirms intuition that a robot with the highest task efficacy can be harder to predict through equations, with the simulated and predicted trajectories not matching very closely for the Y trajectory. Although the predicted X trajectory is closer to its simulated counterpart than the Y trajectory, clearly neither the fitted $x(t)$ or $y(t)$ have captured the full intricacies of the robot's behavior. In comparison, the high predictability robot in Fig. 6(c) somewhat defies intuition on what a high-predictability but low-task efficacy robot's behavior would be. Intuitively, the simplest way for a robot to achieve a very high predictability fitness would be for it to learn not to move, which gives rise to the very simple trajectory equations of $x(t) = 0$ and $y(t) = 0$. However, Fig. 6(c) trajectories and equations show that the robot did move in both the x and y directions. This can likely be explained by the difficulty in creating a robot that does not move at all in our methodology, due to only one synaptic weight being randomly modified for each child. It is possible that gradient descent methods, which are able to continuously adjust network weights in small increments would more easily find the solution of a completely static robot. Importantly, the knee robot (Fig. 6(b)) confirms that a good task efficacy can be achieved with a high predictability, where the equations capture the overall behavior of the robot well although some of the fine intricacies of the motion are not represented.

Further analysis of the equations fit to evolved behaviors in the chosen sample trial reveals interesting details of what

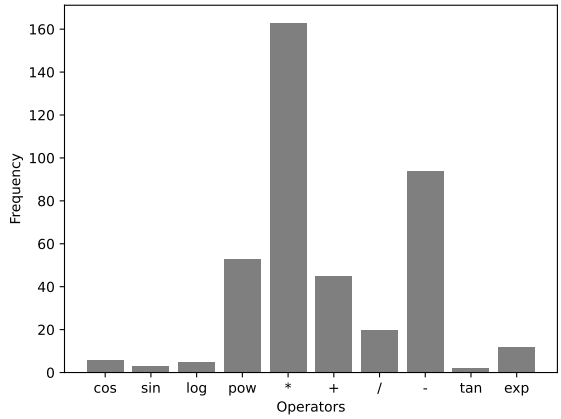

Figure 8: Distribution of operators over the Pareto front equations in the chosen sample trial for generations 0, 9, 19, 29, 39, 49.

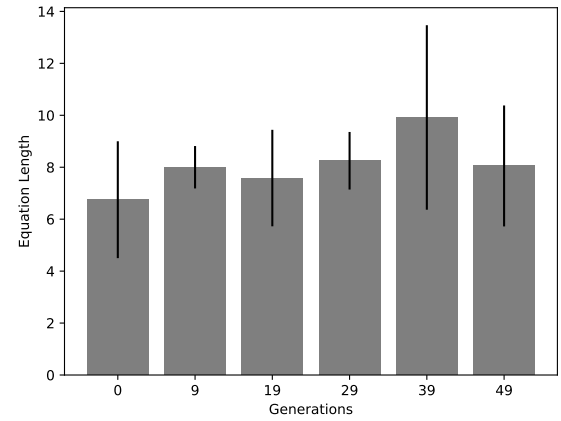

Figure 9: Mean lengths of the preorder traversal of the predicted equations for Pareto front robots every 10 generations of a sample trial. Error bars show one standard deviation for each respective generation.

the GPSR algorithm focuses on when fitting the equations. Figure 8 shows that the GPSR algorithm uses more basic operators such as $[+, *, pow]$ over trigonometric operators. The GPSR algorithm is not fitting oscillatory behavior, even when it is present (such as in Fig. 6(b)). The oscillations in the trajectories for this robot are small in amplitude, but visibly present. However, as evidenced by the lack of any sine or cosine operators in the relevant equations, the GPSR algorithm does not include the small oscillatory behavior, as it is not as dominant as the curving trajectory captured by the $t^{1/3}$ term. Given the overall low frequency of trigonometric operators in the analyzed Pareto equations, it can be assumed that this tendency to ignore small oscillatory behavior is systematic in the fitted equations and their predicted trajectories. This implies that it is the larger movement behaviors that are more important to overall predictability of the robots than the smaller, more detailed behaviors.

Figure 9 shows the mean equation lengths of the Pareto front robots' fitted equations every 10 generations for the selected sample trial run. Here, equation length is the length of the preorder traversal of an equation tree, and is being used as a measure of equation complexity. As can be seen, there is a slight trend toward an increase in complexity until generation 39, after which the complexity drops again. This indicates a possible trend toward predicting more complex equations as trajectories get longer and more complex. However, the effect is small and somewhat inconsistent and as such causation between increased efficacy/predictability and higher equation complexity cannot be proven.

These equations that describe the robots' behaviors are a powerful tool beyond simply what behaviors the GPSR algorithm finds important. Generally, robot behaviors are controlled by neural networks, broadly considered to be black-box models. Equations, on the other hand, have a very high degree of interpretability. A robot's neural network controller generally must be simulated and this simulation viewed to understand the behaviors it controls, whereas those behaviors can be easily understood from inspecting the robot's equations, and potentially plotting the trajectories that result from evaluating said equations. As interpretability of methods is an increasingly large concern dealt with in not just evolutionary robots but all AI fields, having highly interpretable equations to describe robot behaviors can potentially be highly beneficial. However, the use of equations as a measure of predictability is a mathematical and not necessarily intuitively predictable by the average person. In theory, a simpler equation should lead to more intuitively understandable and predictable behavior. As such, in future work, we would like to explore evolving for equations that are not just accurate but also simple.

Overall, we have here shown that bi-objectively optimizing for both task efficacy and behavioral predictability can be used to obtain robots that are equal in behavioral task efficacy to robots optimized for only task efficacy while also having higher predictability. This is important for evolutionary robots, as the predictability of robot behaviors is a key safety issue, and these results represent a step toward more predictable robots.

In future, we would like to further extend the multi-objective optimization explored here. In both biology and evolutionary robots, there is an accepted tradeoff in legged motion between task efficacy of motion and energetic efficiency (Sellers et al., 2003). Further, legged motion requires more complex motions than peristaltic motion or wheeled travel, suggesting legged locomotion may be more unpredictable than non-legged motion. As such, we believe employing tri-objective evolution between task efficacy, predictability, and energetic efficiency could result in robots that enjoy high task efficacy while also being predictable and efficient.

Acknowledgments

This material is based upon work supported by the National Science Foundation Graduate Research Fellowship Program under Grant No. 2235204. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

Computations were performed, in part, on the Vermont Advanced Computing Center.

References

Angelis, D., Sofos, F., and Karakasidis, T. E. (2023). Artificial intelligence in physical sciences: Symbolic regression trends and perspectives. *Archives of Computational Methods in Engineering*, 30(6):3845–3865.

Arnaldo, I., Krawiec, K., and O'Reilly, U.-M. (2014). Multiple regression genetic programming. In *Proceedings of the 2014 Annual Conference on Genetic and Evolutionary Computation*, pages 879–886.

Bendinelli, T., Biggio, L., and Kamienny, P.-A. (2023). Controllable neural symbolic regression. *arXiv preprint arXiv:2304.10336*.

Bertschinger, A., Davis, Q. T., Bagrow, J., and Bongard, J. (2023). The metric is the message: Benchmarking challenges for neural symbolic regression. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 161–177. Springer.

Biggio, L., Bendinelli, T., Neitz, A., Lucchi, A., and Parascandolo, G. (2021). Neural symbolic regression that scales. In *International Conference on Machine Learning*, pages 936–945. PMLR.

Bongard, J. C. (2013). Evolutionary robotics. *Communications of the ACM*, 56(8):74–83.

Costa, A., Dangovski, R., Dugan, O., Kim, S., Goyal, P., Soljačić, M., and Jacobson, J. (2020). Fast neural models for symbolic regression at scale. *arXiv preprint arXiv:2007.10784*.

Coumans, E. and Bai, Y. (2016–2021). Pybullet, a python module for physics simulation for games, robotics and machine learning. `http://pybullet.org`.

Cranmer, M. (2023). Interpretable Machine Learning for Science with PySR and SymbolicRegression.jl. arXiv:2305.01582 [astro-ph, physics:physics].

Cruz, F., Dazeley, R., Vamplew, P., and Moreira, I. (2023). Explainable robotic systems: Understanding goal-driven actions in a reinforcement learning scenario. *Neural Computing and Applications*, 35(25):18113–18130.

Doncieux, S., Bredeche, N., Mouret, J.-B., and Eiben, A. E. (2015). Evolutionary robotics: what, why, and where to. *Frontiers in Robotics and AI*, 2:4.

Doncieux, S., Mouret, J.-B., and Bredeche, N. (2009). Exploring new horizons in evolutionary design of robots. In *Workshop on Exploring new horizons in Evolutionary Design of Robots at IROS 2009*, pages 5–12.

Floreano, D. and Mondada, F. (1998). Evolutionary neurocontrollers for autonomous mobile robots. *Neural networks*, 11(7-8):1461–1478.

Floreano, D., Mondada, F., et al. (1994). Automatic creation of an autonomous agent: Genetic evolution of a neural-network driven robot. *From animals to animats*, 3:421–430.

Glette, K., Klaus, G., Zagal, J. C., Torresen, J., et al. (2012). Evolution of locomotion in a simulated quadruped robot and transferral to reality. In *Proceedings of the Seventeenth International Symposium on Artificial Life and Robotics*, pages 1–4.

Gongora, M. A., Passow, B. N., and Hopgood, A. A. (2009). Robustness analysis of evolutionary controller tuning using real systems. In *2009 IEEE Congress on Evolutionary Computation*, pages 606–613. IEEE.

Grasso, C. and Bongard, J. (2022). Empowered neural cellular automata. In *Proceedings of the Genetic and Evolutionary Computation Conference Companion*, pages 108–111.

Grasso, C. and Bongard, J. (2023). Selection for short-term empowerment accelerates the evolution of homeostatic neural cellular automata. In *Proceedings of the Genetic and Evolutionary Computation Conference*, pages 147–155.

Hamann, H. (2014). Evolution of collective behaviors by minimizing surprise. In *ALIFE 14 : Proceedings of the Fourteenth International Conference on the Synthesis and Simulation of Living Systems*, pages 344–351, Cambridge, Massachusetts. MIT Press.

Kaiser, T. K. and Hamann, H. (2019). Engineered self-organization for resilient robot self-assembly with minimal surprise. *Robotics and Autonomous Systems*, 122:103293.

Kaiser, T. K. and Hamann, H. (2020). Evolution of diverse swarm behaviors with minimal surprise. volume ALIFE 2020: The 2020 Conference on Artificial Life of *ALIFE 2022: The 2022 Conference on Artificial Life*, pages 384–392.

Kaiser, T. K. and Hamann, H. (2022a). Innate motivation for robot swarms by minimizing surprise: From simple simulations to real-world experiments. *IEEE Transactions on Robotics*, 38(6):3582–3601.

Kaiser, T. K. and Hamann, H. (2022b). Minimize surprise map-elites: a task-independent map-elites variant for swarms. In *Proceedings of the Genetic and Evolutionary Computation Conference Companion*, GECCO '22, page 116–119, New York, NY, USA. Association for Computing Machinery.

Kaiser, T. K., Kluth, C., and Hamann, H. (2023). Evolving dynamic collective behaviors by minimizing surprise. volume ALIFE 2023: Ghost in the Machine: Proceedings of the 2023 Artificial Life Conference of *ALIFE 2022: The 2022 Conference on Artificial Life*, page 50.

Kamienny, P.-A., d'Ascoli, S., Lample, G., and Charton, F. (2022). End-to-end symbolic regression with transformers. *arXiv preprint arXiv:2204.10532*.

Kim, S., Lu, P. Y., Mukherjee, S., Gilbert, M., Jing, L., Čeperić, V., and Soljačić, M. (2020). Integration of neural network-based symbolic regression in deep learning for scientific discovery. *IEEE transactions on neural networks and learning systems*, 32(9):4166–4177.

Koza, J. (1992). *Genetic Programming: On the Programming of Computers by Means of Natural Selection*. A Bradford book. Bradford.

Koza, J. R. (1994). *Genetic programming II: automatic discovery of reusable programs*. MIT press.

La Cava, W., Burlacu, B., Virgolin, M., Kommenda, M., Orzechowski, P., de França, F. O., Jin, Y., and Moore, J. H. (2021). Contemporary symbolic regression methods and their relative performance. *Advances in neural information processing systems*, 2021(DB1):1.

Lehman, J., Risi, S., D'Ambrosio, D., and O Stanley, K. (2013). Encouraging reactivity to create robust machines. *Adaptive Behavior*, 21(6):484–500.

Li, W., Li, W., Sun, L., Wu, M., Yu, L., Liu, J., Li, Y., and Tian, S. (2022). Transformer-based model for symbolic regression via joint supervised learning. In *The Eleventh International Conference on Learning Representations*.

Liberman-Pincu, E. and Oron-Gilad, T. (2024). Arm in motion: How motion modality and erratic behavior of a robotic arm shape user perception. In *International Conference on Computer-Human Interaction Research and Applications*, pages 53–62. Springer.

Mann, H. B. and Whitney, D. R. (1947). On a test of whether one of two random variables is stochastically larger than the other. *The annals of mathematical statistics*, pages 50–60.

Martius, G. and Lampert, C. H. (2016). Extrapolation and learning equations. *arXiv preprint arXiv:1610.02995*.

Mubin, O. and Bartneck, C. (2015). Do as i say: Exploring human response to a predictable and unpredictable robot. In *Proceedings of the 2015 British HCI Conference*, pages 110–116.

Nelson, A. L., Barlow, G. J., and Doitsidis, L. (2009). Fitness functions in evolutionary robotics: A survey and analysis. *Robotics and Autonomous Systems*, 57(4):345–370.

Nolfi, S., Bongard, J., Husbands, P., and Floreano, D. (2016). *Evolutionary robotics*. Springer.

Petersen, B. K., Landajuela, M., Mundhenk, T. N., Santiago, C. P., Kim, S. K., and Kim, J. T. (2019). Deep symbolic regression: Recovering mathematical expressions from data via risk-seeking policy gradients. *arXiv preprint arXiv:1912.04871*.

Premakumar, V. N., Vaiana, M., Pop, F., Rosenblatt, J., de Lucena, D. S., Ziman, K., and Graziano, M. S. (2024). Unexpected benefits of self-modeling in neural systems. *arXiv preprint arXiv:2407.10188*.

Quade, M., Abel, M., Shafi, K., Niven, R. K., and Noack, B. R. (2016). Prediction of dynamical systems by symbolic regression. *Physical Review E*, 94(1):012214.

Sado, F., Loo, C. K., Liew, W. S., Kerzel, M., and Wermter, S. (2023). Explainable goal-driven agents and robots-a comprehensive review. *ACM Computing Surveys*, 55(10):1–41.

Sahoo, S., Lampert, C., and Martius, G. (2018). Learning equations for extrapolation and control. In *International Conference on Machine Learning*, pages 4442–4450. PMLR.

Satopaa, V., Albrecht, J., Irwin, D., and Raghavan, B. (2011). Finding a" kneedle" in a haystack: Detecting knee points in system behavior. In *2011 31st international conference on distributed computing systems workshops*, pages 166–171. IEEE.

Schadenberg, B. R., Reidsma, D., Heylen, D. K., and Evers, V. (2021). "i see what you did there" understanding people's social perception of a robot and its predictability. *ACM Transactions on Human-Robot Interaction (THRI)*, 10(3):1–28.

Schmidt, M. and Lipson, H. (2009). Distilling free-form natural laws from experimental data. *Science (American Association for the Advancement of Science)*, 324(5923):81–85.

Schmidt, M. D. and Lipson, H. (2008). Coevolution of fitness predictors. *IEEE Transactions on Evolutionary Computation*, 12(6):736–749.

Schmidt, M. D. and Lipson, H. (2010). Age-fitness pareto optimization. In *Proceedings of the 12th annual conference on Genetic and evolutionary computation*, pages 543–544.

Schoppers, M. (1987). Universal plans for reactive robots in unpredictable environments. In *IJCAI*, volume 87, pages 1039–1046. Citeseer.

Sellers, W. I., Dennis, L. A., and Crompton, R. H. (2003). Predicting the metabolic energy costs of bipedalism using evolutionary robotics. *Journal of Experimental Biology*, 206(7):1127–1136.

Valipour, M., You, B., Panju, M., and Ghodsi, A. (2021). SymbolicGPT: A generative transformer model for symbolic regression. *arXiv preprint arXiv:2106.14131*.

Vastl, M., Kulhánek, J., Kubalík, J., Derner, E., and Babuška, R. (2022). SymFormer: End-to-end symbolic regression using transformer-based architecture. *arXiv preprint arXiv:2205.15764*.

Virgolin, M., Alderliesten, T., Witteveen, C., and Bosman, P. A. (2021). Improving model-based genetic programming for symbolic regression of small expressions. *Evolutionary computation*, 29(2):211–237.

Wang, T., Guo, W., Li, M., Zha, F., and Sun, L. (2012). Cpg control for biped hopping robot in unpredictable environment. *Journal of Bionic Engineering*, 9(1):29–38.

Wang, Y., Wagner, N., and Rondinelli, J. M. (2019). Symbolic regression in materials science. *MRS communications*, 9(3):793–805.

Watson, R. A., Buckley, C. L., and Mills, R. (2011). Optimization in "self-modeling" complex adaptive systems. *Complexity*, 16(5):17–26.

Zhou, R., Bacardit, J., Brownlee, A., Cagnoni, S., Fyvie, M., Iacca, G., McCall, J., van Stein, N., Walker, D., and Hu, T. (2024). Evolutionary computation and explainable ai: A roadmap to transparent intelligent systems. *IEEE Transactions on Evolutionary Computation*.

Zykov, V., Bongard, J., and Lipson, H. (2004). Evolving dynamic gaits on a physical robot. In *Proceedings of Genetic and Evolutionary Computation Conference, Late Breaking Paper, GECCO*, volume 4, page 2004. Citeseer.