Co-Imagination of Behaviour & Morphology of Agents

Maria Sliacka[1], Michael Mistry[2 [0000−0003−3979−922X]],
Roberto Calandra[3,4 [0000−0001−9430−8433]],
Ville Kyrki[1 [0000−0002−5230−5549]], and
Kevin Sebastian Luck[1,5,6 ✉[0000−0003−2228−203X]]

[1] Department of Electrical Engineering and Automation (EEA), Aalto University, Espoo, Finland
m.sliacka12@gmail.com, ville.kyrki@aalto.fi
[2] University of Edinburgh, Edinburgh, UK
mmistry@ed.ac.uk
[3] Learning, Adaptive Systems, and Robotics (LASR) Lab, TU Dresden, Germany
roberto.calandra@tu-dresden.de
[4] The Centre for Tactile Internet with Human-in-the-Loop (CeTI), Dresden, Germany
[5] Finnish Center for Artificial Intelligence, Espoo, Finland
[6] Vrije Universiteit Amsterdam, Amsterdam, Netherlands
k.s.luck@vu.nl

Abstract. The field of robot learning has made great advances in developing behaviour learning methodologies capable of learning policies for tasks ranging from manipulation to locomotion. However, the problem of combined learning of behaviour and robot structure, here called co-adaptation, is less studied. Most of the current co-adapting robot learning approaches rely on model-free algorithms or assume to have access to an a-priori known dynamics model, which requires considerable human engineering. In this work, we investigate the potential of combining model-free and model-based reinforcement learning algorithms for their application on co-adaptation problems with unknown dynamics functions. Classical model-based reinforcement learning is concerned with learning the forward dynamics of a specific agent or robot in its environment. However, in the case of jointly learning the behaviour and morphology of agents, each individual agent-design implies its own specific dynamics function. Here, the challenge is to learn a dynamics model capable of generalising between the different individual dynamics functions or designs. In other words, the learned dynamics model approximates a multi-dynamics function with the goal to generalise between different agent designs. We present a reinforcement learning algorithm that uses a learned multi-dynamics model for co-adapting robot's behaviour and morphology using imagined rollouts. We show that using a multi-dynamics model for imagining transitions can lead to better performance for model-free co-adaptation, but open challenges remain.

Keywords: Evolutionary Robotics · Co-Adaptation · Co-Design · Reinforcement Learning.

1 Introduction

Co-adaptation is a process that is present everywhere on Earth. From tiny insects adapting to human houses [13] to rats adapting their diets in cities [11], to the large whales that

adapt to fight human-produced noise in the ocean [18], it has proven to be crucial for survival in a changing environment. The idea of co-adaptation brings together the two different ways that organisms adapt – behavioural and morphological. Behavioural adaptation to new tasks happens on short timescales and is not difficult for humans and animals, whereas, adapting morphological traits is often not possible and is a process operating on long timescales [19, 17].

When it comes to robotics, we usually only optimise the behaviour of our robots given a new task. This leads to robots being designed by human engineers to be multi-purpose and easy to control, such that they can be used for a wide range of tasks. However, as nature has shown repeatedly, having evolved a specialised body morphology can lead to vastly improved behavioural policy, performance and excellence in a low number of essential tasks. Even humans tend to complement or change their morphology for improving their performance, for example, by using artificial modifications to their bodies, such as bodysuits for diving.

This leads to the idea of co-adaptation of the behaviour and morphology in robots [16, 24, 6, 25, 15, 21, 22, 26]. The goal is to jointly optimise the control and design parameters of a robot given its task. The challenge of co-adaptation comes primarily from the design search space: especially with a high number of continuous design parameters, it is impossible to evaluate all possible body shapes. One possible alternative option to reduce these costs drastically is by utilising simulations to build and evaluate robots' morphologies [25, 6, 21]. However, creating, evaluating and mutating possible robot candidates in a simulation is not only computationally demanding but also suffers from the simulation-to-reality gap. Designs and behaviours found to be optimal in simulation may not be optimal in the real world [15, 23]. However, to allow for the co-adaptation of robots in the real world, data-efficient co-adaptation methodologies are required. Especially methods, which are able to optimise robot morphologies within a low number of design iterations, as manufacturing robots in the real world requires many resources and person-hours. Prior work tackling this problem has largely focused on model-free reinforcement learning approaches [24, 16, 8, 21]. In this work, we will explore the possibility to utilise model-based reinforcement learning techniques [20, 10] to further improve the performance and sample-efficiency of model-free co-adaptation algorithms [16]. This requires us to learn not only one single forward-dynamics function for a specific agent morphology, but also the dynamics of multiple, if not infinitely many, agents. In our work, we will investigate the benefit of training a forward-dynamics function parameterised by the known design parameters of agents, and evaluate their generalisation ability across known and unseen agent morphologies. Furthermore, we will incorporate this model-based learning approach into state-of-the-art model-free co-adaptation methods to further increase the data- and training-efficiency of the co-adaptation using imagination-based [10, 20] training data augmentation for a co-adapting deep reinforcement learning method. We show that by adding artificial data in known design space, we are able to improve the performance of the co-adaptation algorithm in terms of total cumulative rewards collected.

2 Related Work

Co-adaptation of control and morphology of an agent or robot has a long history, including the usage of evolutionary algorithms and gradient-based optimisation methods, such as reinforcement learning. Evolutionary algorithms have been used to evolve populations of agent morphologies via mutations [25, 6]. Gupta et al. [6] also added a reinforcement learning loop to optimise the controller directly. Evolutionary approaches require populations of solutions for each optimisation iteration, which often requires large amounts of data especially in the case of a high-dimensional space of morphologies. The need for such large populations means that these approaches have to rely primarily on simulations due to their high cost in real-world tasks, which makes them prone to suffer from the so-called simulation-to-reality gap.

There are multiple approaches for solving the co-adaptation problem using model-free reinforcement learning [24, 7, 2, 16]. Schaff et al. [24] use a distribution of designs that is shifted towards better-performing morphologies with a policy using the design as context. Whereas, [7] directly considers the design parameters to be learnable but also requires keeping a population of morphologies to compute the policy gradient. Similarly to the evolutionary methods, both of these model-free approaches have the same drawback of requiring a population or distribution of morphologies during the entire algorithm run, thus being limited in their real-world applicability. A different approach was proposed in [2], where the design or hardware is considered as part of the policy parameters and jointly optimised. However, this requires hand-engineering and prior knowledge as it relies on simulating the hardware with an auto-differentiable computational graph. ORCHID [12] uses model-based reinforcement learning to simultaneously optimise the hardware and control parameters of the agent. It also uses an actor-critic algorithm similarly to the proposed approach, however, it relies on a differentiable transition function which is assumed to be known a-priori. Closest to our method is [16], which is a model-free co-adaptation approach utilising the Q-value function for data-efficient evaluation of design candidates. However, it only uses model-free reinforcement learning while our method uses a learned dynamics model in addition to generate artificial data, thus combining model-free and model-based reinforcement learning. We use model-free co-adaptation [16] as a starting point and baseline for evaluating the potential of joint co-imagination of agent behaviour and design as augmented training data throughout the learning process. Another approach [5, 8] is to use function and control theory to co-optimise the morphology and control, both these methods rely on the accuracy of their models and equations, where [5] also require differentiable control planner and simulation. [14] use Bayesian optimisation to first optimise the morphology and then learn the corresponding controller. This method, however, requires a parameterised controller and would struggle to scale to high-dimensional spaces.

3 Problem Statement

We assume a Markov Decision Process (MDP) for solving the co-adaptation problem, extended with a design context ξ. The MDP is denoted by $(S, A, p, r, \gamma, \Xi)$, with state space $S \subseteq R^d$, action space $A \subseteq R^n$, design parameter space $\Xi \subseteq \mathbb{R}^i$ and a given

reward function $r(s, a)$, with $r : S \times A \mapsto \mathbb{R}$. Without loss of generalisation we will assume for the reminder of the paper a continuous but bounded design space $\Xi \subseteq \mathbb{R}^i$, with i dimensions. However, Ξ could be in principle also be a set of discrete designs. In this MDP, the underlying transition probability $p(s_{t+1}|s_t, a_t, \xi) : S \times A \times S \times \Xi \mapsto \mathbb{R}^+$ maps the current state s_t and an action a_t to the next state s_{t+1}. Importantly, the transition probability does not only depend on the current state s_t and the agent's action a_t, but also on the agent's design variable ξ which parameterises the morphology of the agent such as lengths of legs, their shape or diameter.

The general problem of co-adapting behaviour and morphology of agents with reinforcement learning is to find a policy $\pi : S \times A \mapsto \mathbb{R}^+$ and designs $\xi \in \Xi$ such that the expected discounted return is maximised as

$$\max_{\pi,\xi} \mathbb{E}_{\substack{a_t \sim \pi(s_t) \\ s_{t+1} \sim p(s_{t+1}|s_t,a_t,\xi)}} \left[\sum_{t=0}^{\infty} \gamma^t r(s_t, a_t) \right], \tag{1}$$

given a specific reward function $r(s, a)$ and discount $\gamma \in [0, 1]$. While not required, we will also assume for the reminder of the paper that the policy π and reward r are parameterised by the design variable ξ, i.e. $\pi(a_t|s_t, \xi)$ and $r(s_t, a_t, \xi)$. This allows to learn a single policy π capable of adapting to specific agent designs. We will furthermore assume that design parameters ξ are observed, which is generally true as these parameters are required for manufacturing or simulating (e.g. via urdf files) specific agents.

4 Learning Behaviour, Design and Dynamics across the Design-Space

To find an optimal combination of behaviour and design with respect to a given reward function as defined in Eq. (1) using model-based reinforcement learning we need to learn three components:

$$\begin{aligned} &\text{Policy:} && \pi(s_t, \xi)\,, \\ &\text{Design:} && \xi\,, \\ &\text{Transition model:} && p(s_{t+1}|s_t, a_t, \xi)\,. \end{aligned} \tag{2}$$

The first two, policy and design, correspond to the core ideas in the co-adaptation framework, behaviour learning and morphology optimisation, which also appear in existing model-free co-adaptation approaches. The third corresponds to our main contribution: Learning a forward-dynamics model across designs. In the following, we will discuss how we will learn each component.

Behaviour Learning: A central component of co-adaptation is the learning of an optimal behavioural policy given a design and task. As discussed in the problem statement, we will operate with an extended MDP formulation considering the effect a parameterised design has on the reward function $r(s_t, a_t, \xi)$ and dynamics $p(s_{t+1}|s_t, a_t, \xi)$. In effect, we consider an extension of the standard reinforcement learning approach optimising for Eq. (1) in which both the policy π and value $V : S \mapsto \mathbb{R}$, or Q-value $Q : S \times A \mapsto \mathbb{R}$,

functions depend on the design variable ξ, i.e. $\pi(s_t, \xi) : S \times \Xi \times A \mapsto \mathbb{R}^+$ and $Q(s_t, a_t, \xi) : S \times A \times \Xi \mapsto \mathbb{R}$. For learning a policy π we employ Soft-Actor-Critic algorithm [9] with the double-Q-network approach for learning a probabilistic policy π. Given a set of training experience $\mathcal{D}$, the Q-value function is trained with the altered loss

$$J_Q(\theta) = \mathbb{E}_{(s_t, a_t, s_{t+1}, \xi) \sim \mathcal{D}} \left[\frac{1}{2} \left(Q_\theta(s_t, a_t, \xi) - (r(s_t, a_t, \xi) + \gamma V_{\bar{\theta}}(s_{t+1}, \xi)) \right)^2 \right], \quad (3)$$

where $V_{\bar{\theta}}$ is defined using the target Q-value function $Q_{\bar{\theta}}(s_t, a_t, \xi)$ parametrised by the target network parameters $\bar{\theta}$, given the transition and design (s_t, a_t, s_{t+1}, ξ). Similarly, we use the modified loss

$$J_\pi(\phi) = \mathbb{E}_{(s_t, \xi) \sim \mathcal{D}, \epsilon_t \sim \mathcal{N}} \left[\alpha \log \pi_\phi \left(f_\phi \left(\epsilon_t; s_t, \xi \right) \mid s_t, \xi \right) - Q_\theta \left(s_t, f_\phi \left(\epsilon_t; s_t, \xi \right), \xi \right) \right], \quad (4)$$

for training the parameters of the policy π. Both these equations are modified to include the design parameter ξ.

Furthermore, we train two separate sets of policies and value networks: The population networks (*Pop*), which train on all designs seen thus far, and the individual networks (*Ind*) which are trained only on current design. This is facilitated by using a replay buffer $\mathcal{D}_{\texttt{Pop}}$ containing the collected experience of all designs seen so far, i.e. $(s_t, a_t, s_{t+1}, \xi_s)$ with $\xi_s \in \Xi_{\text{seen}}$. The individual networks have its weights initialised by the *pop* networks each time we start training on a new design and primarily train with experience from the current design to facilitate fast and sample-efficient reinforcement learning. While training on an agent of design ξ_{current}, these individual networks will utilise the replay buffer $\mathcal{D}_{\texttt{Ind}}$ containing only experience of the current agent, i.e. $(s_t, a_t, s_{t+1}, \xi_{\text{current}})$.

Design Optimisation: During the design optimisation stage, we aim to identify a design variable maximising the expected return given the data collected thus far, which is then synthesised and a new round of behavioural learning is executed. While previous approaches have utilised simulations to gauge the potential performance of design candidates, [16] have pointed out that the value function can be utilised as a data-efficient and computationally more efficient alternative. By using the Q-value function as the fitness function, Particle Swarm Optimisation (PSO) [1] is used to quickly find the best next design parameter by optimising the objective

$$\max_{\xi} \mathbb{E}_{(s_0, a_0, s_1, \xi_{\text{orig}}) \sim \mathcal{D}} \left[\mathbb{E}_\pi \left[Q(s_0, a, \xi) \mid a = \pi(s_0, \xi) \right] \right]. \quad (5)$$

The expected return (see Eq. (1)) is estimated by evaluating the Q-function for start states s_0 sampled from a separate replay buffer $\mathcal{D} = Replay_{s_0}$, and replacing the original design variable ξ_{orig} with the design query ξ.

Model Learning: When learning a forward dynamics function in the context of co-adaptation, we have in principle two choices: we either learn a design-specific, individual, dynamics function or learn one multi-dynamics function that captures all possible designs. In the following, we will concentrate on the latter due to its potential to allow for generalisation between designs, and allow for queries of transitions for unseen designs. For learning the multi-dynamics function we will follow the probabilistic

ensemble approach proposed by [3]. The ensemble consists of 3 probabilistic neural networks with outputs parametrising a Gaussian distribution that estimates the next state given the current state, taken action and design, defined as $Pr^i(s_{t+1}|s_t, a_t, \xi) = \mathcal{N}(\mu_{\psi^i}(s_t, a_t, \xi), \Sigma_{\psi^i}(s_t, a_t, \xi))$, where ψ^i corresponds to the i-th network's parameters. The ensemble is trained using the Negative Log Likelihood loss calculated for each network

$$J_{\widetilde{p}}(\psi^i) = -\mathbb{E}_{(s_t,a_t,s_{t+1},\xi)\sim Replay_{\texttt{Pop.}}} \log \mathcal{N}(s_{t+1}|\mu_{\psi^i}(s_t, a_t, \xi), \Sigma_{\psi^i}(s_t, a_t, \xi)) = \\ \mathbb{E}_{(s_t,a_t,s_{t+1},\xi)\sim Replay_{\texttt{Pop.}}} [\mu_{\psi^i}(s_t, a_t, \xi) - s_{t+1}]^\top \Sigma^{-1}_{\psi^i}(s_t, a_t, \xi)[\mu_{\psi^i}(s_t, a_t, \xi) - s_{t+1}] \\ + \log\det \Sigma_{\psi^i}(s_t, a_t, \xi), \tag{6}$$

where we use transitions (s_t, a_t, s_{t+1}, ξ) sampled from the replay buffer $Replay_{\texttt{Pop.}}$ containing experience from all designs seen thus far. The learned forward dynamics function $h(s_t, a_t, \xi)$ is then defined using the TS1 propagation method [3], where at each time step the network is uniformly sampled from the ensemble to produce an output. Importantly, we will consider the case where the dynamics functions is learned online, i.e. without pre-training on pre-collected datasets. At the start of the co-adaptation process, the dynamics network will be initialised with random weight initialisation and thereafter trained with the process described above. The number of designs the dynamics networks are trained on will increase with each new design selected (see Algorithm 1).

5 Co-Adaptation By Model-Based Imagination

In this section, we will discuss the changes made to the behavioural learning process necessary to utilise a learned dynamics model in the context of co-adaptation. Specifically, we will propose and investigate the use of a learned forward dynamics model for supporting the model-free reinforcement learning process by supplying artificial, i.e. imagined, transitions on known or unseen designs. Assuming a forward dynamics function $h(s_t, a_t, \xi)$ learned in the manner described above, we re-formulate the model-free loss functions of the value function loss

$$J_Q(\theta) = \mathbb{E}_{(s_t,a_t,s_{t+1},\xi)\sim\mathcal{D}} \left[\frac{1}{2}\left(Q_\theta(s_t, a_t, \xi) - (r(s_t, a_t, \xi) + \gamma V_{\bar{\theta}}(s_{t+1}, \xi))\right)^2\right] \\ + \mathbb{E}_{\substack{(s_t,a_t,s_{t+1},\xi)\sim\mathcal{D} \\ \xi_g\sim G(\xi) \\ a_g\sim\pi(s_t,\xi_g)}} \left[\frac{1}{2}\left(Q_\theta(s_t, a_g, \xi_g) - (r(s_t, a_g, \xi_g) + \gamma V_{\bar{\theta}}(h(s_t, a_g, \xi_g)))\right)^2\right] \tag{7}$$

and for the policy loss as

$$J_\pi(\phi) = \mathbb{E}_{(s_t,\xi)\sim\mathcal{D},\epsilon_t\sim\mathcal{N}} \left[\alpha \log \pi_\phi\left(f_\phi\left(\epsilon_t; s_t, \xi\right) \mid s_t, \xi\right) - Q_\theta\left(s_t, f_\phi\left(\epsilon_t; s_t, \xi\right), \xi\right)\right] \\ + \mathbb{E}_{\substack{(s_t,\xi)\sim\mathcal{D},\epsilon_t\sim\mathcal{N} \\ \xi_g\sim G(\xi)}} \left[\alpha \log \pi_\phi\left(f_\phi\left(\epsilon_t; s_t, \xi_g\right) \mid s_t, \xi_g\right) - Q_\theta\left(s_t, f_\phi\left(\epsilon_t; s_t, \xi_g\right), \xi_g\right)\right], \tag{8}$$

where the design variable ξ_g is sampled from a generator $G(\xi)$. These losses correspond to a one-step imagination-based approach similar to [10]. Many potential choices exist for

the design generator $G(\xi)$. For our experimental evaluation we will consider $G(\xi) = \xi$, $G(\xi) = N(\xi, \sigma^2)$ and $G(\xi) = \text{uniform}(\Xi)$. These correspond to (a) using the identity function, i.e. using the original design from the replay buffer, (b) adding Gaussian noise to ξ, and (c) using a randomly sampled design $\xi \in \Xi$. The action a_g is sampled from the policy with $\pi(s_t, \xi_g)$.

In practice, to increase the computational efficiency during training, we pre-compute a sufficiently large batch of $(s_t, a_g, h(s_t, a_g, \xi_g), \xi_g)$ after updating the forward dynamics model and store them in a separate replay buffer $Replay_{\texttt{Art.}}$ after each episode. During the SAC networks training, the population networks are then trained on data that comes from both real and artificial replay buffers. This is done by taking 80% of the batch from the real experience buffer $Replay_{\texttt{Pop.}}$, and 20% from the artificial experience $Replay_{\texttt{Art.}}$. The final algorithm is shown in Algorithm 1: The algorithm presents a version of the algorithm in which agent designs, and behaviours are updated sequentially, i.e. no population of agents is maintained. This is to emulate the constraints experienced when co-adapting systems in the real world, where mass-parallelization is rarely possible [16]. The algorithm features two learning loops: The first is training and updating the behaviour and neural networks; the second is optimizing the agent design via the neural network surrogates.

6 Experiments

Using the developed data augmentation techniques to imagine transitions of unseen behaviour and designs, we will now investigate whether these techniques can improve the performance of model-free co-adaptation. To this end, we will train a multi-dynamics model on three continuous control tasks: Half-Cheetah, Hopper and Walker. In these environments, the co-adaptation algorithm can change design variables such as the limb lengths every hundred episodes, and the goal is to maximize the performance given a reward function. The trained models are then used to augment the training data with imagined trajectories from (a) previously seen designs, (b) previous designs with noise and (c) randomly selected designs. We will focus in our evaluation specifically on the data efficiency of the developed algorithms in a scenario where each agent is sequentially updated in behaviour and design, as it would be the case in real-world co-adaptation. We will conclude this section by discussing open challenges and the apparent non-linearities in the dynamics predictions when varying the design variables.

6.1 Experimental Setup

During co-adaptation, each experiment starts on the same initial five agent designs that were selected randomly but are kept constant for all experiments, then the algorithm runs for 50 more design iterations which are optimised (exploitation) or randomly chosen (exploration) in alternating fashion. For each of the initial designs, 300 episodes are being executed, and thereafter 100 episodes. For all environments, we execute 1000 steps per episode. During design optimisation, we use a batch size of 36 initial states to estimate the fitness. SAC parameters are: a discount of 0.99, a tau of 0.005, 3E-4 for policy and Q function learning rates, $\alpha = 0.01$, and 3 hidden layers with 200 neurons each for policy

Algorithm 1 CoIm: Co-Imagination

Initialise replay buffers: $Replay_{\texttt{Pop.}}$, $Replay_{\texttt{Ind.}}$, $Replay_{s_0}$, $Replay_{\texttt{Art.}}$
Initialise first design ξ
for $i \in (1, 2, ..., M)$ **do**
  $\pi_{\texttt{Ind.}} = \pi_{\texttt{Pop.}}$
  $Q_{\texttt{Ind.}} = Q_{\texttt{Pop.}}$
  Initialise an empty $Replay_{\texttt{Ind.}}$
  Initialise an empty $Replay_{\texttt{Art.}}$
  Fill $Replay_{\texttt{Art.}}$ with random batches from $Replay_{\texttt{Pop.}}$ based on design strategy
  while *Not finished optimising local policy* **do**
    Collect experience $(s_0, a_0, r_1, s_1, ..., s_T, r_T)$ for current design ξ with policy $\pi_{\texttt{Ind.}}$
    Add quintuples $(s_t, a_t, r_{t+1}, s_{t+1})$ to $Replay_{\texttt{Ind.}}$
    Add quintuples $(s_t, a_t, r_{t+1}, s_{t+1}, \xi)$ to $Replay_{\texttt{Pop.}}$
    After each episode refresh $Replay_{\texttt{Art.}}$ experience using $\widetilde{p}$ and $\pi_{\texttt{Pop.}}$.
    Add start state s_0 to $Replay_{s_0}$
    Train networks $\pi_{\texttt{Ind.}}$ and $Q_{\texttt{Ind.}}$ with random batches from $Replay_{\texttt{Ind.}}$ and Eq. (7-8)
    Train $\pi_{\texttt{Pop.}}$ and $Q_{\texttt{Pop.}}$ with batches from $Replay_{\texttt{Pop.}}$ and $Replay_{\texttt{Art.}}$ via Eq. (7-8)
    Train Gaussian Dynamics Ensemble model $\widetilde{p}$ with batches from $Replay_{\texttt{Pop.}}$ with Eq. (6)
  end
  if i *is even* **then**
    Sample batch of start states $s_{\texttt{b}} = (s_0^1, s_0^2, ..., s_0^n)$ from $Replay_{s_0}$
    Find optimal design ξ with objective function $\max_{\xi} \frac{1}{n} \sum_{s \in s_{\texttt{b}}} Q_{\texttt{Pop.}}(s, \pi_{\texttt{Pop.}}(s, \xi), \xi)$
  else
    Sample design ξ with exploration strategy
  end
end

and q-value networks. Individual networks have 1000 updates per episode, population networks have 250, and the batch size is 256. The probabilistic dynamics ensemble [3] with 3 networks has 3 hidden layers with 500 neurons, the propagation method is the random model, the learning rate is 3E-4 and the weight decay is 1E-5.

6.2 Environments

Experiments are performed both in PyBullets' [4, 16] Half-Cheetah and MuJoCos' [27] Walker and Hopper continuous control environments:

Half Cheetah: Half Cheetah has a 17-dimensional state space that contain joint positions and velocities, angular velocity, the horizontal and vertical speed of the centre of mass. Morphological parameters of Half Cheetah are described by continuous design parameters $\xi \in \mathbb{R}^6$, where ξ is used to calculate the scaled version of the original Half-Cheetah leg lengths as $(0.29 \times \xi_1, 0.3 \times \xi_2, 0.188 \times \xi_3, 0.29 \times \xi_4, 0.3 \times \xi_5, 0.188 \times \xi_6)$, where each $\xi_i \in [0.8, 2.0]$. Design bounds are defined as 0.8 for lower and 2 for upper bound

on every ξ_i. Action $a \in \mathbb{R}^6$ defines the joint acceleration and the reward function r is given by $r(s) = max(\frac{\Delta x}{10}, 10)$, where Δx is the horizontal speed defining the forward motion of the agent.

Walker: Similarly, Walker's state space is 18-dimensional, with an action space of $a \in \mathbb{R}^6$ and reward function of $r(s) = \frac{1}{10}((h_{torso} > 0.8) \times (max(\Delta x, 0)+1) - ||y_{rot}||_2 \times 0.1)$, where h_{torso} is the height of the torso and y_{rot} is the vertical orientation of the torso. The morphology parameters of Walker scale the agent's leg and foot lengths calculated as $(0.4 \times \xi_1, 0.45 \times \xi_2, 0.6 \times \xi_3, 0.2 \times \xi_4, 0.45 \times \xi_5, 0.6 \times \xi_6, 0.2 \times \xi_7)$.

Hopper: Hopper's state space is 11-dimensional, action space is 3-dimensional and reward function is similarly to Walker given by $r(s) = (max(h_{torso} > 0.5, 0.1) \times (max(\Delta x, 0) + 1) - ||y_{rot}||_2 \times 0.1)$. The morphology of Hopper scales the original leg segments and is defined as $(0.4 \times \xi_1, 0.45 \times \xi_2, 0.5 \times \xi_3, 0.39 \times \xi_4)$. Design bounds for both Walker and Hopper are defined as 0.5 for lower and 2 for upper bound on every ξ_i.

6.3 Co-Imagination of unaltered Designs ($G(\xi) = \xi$)

First, we study how our method performs when the artificial experience is created without disturbance to experienced design parameters, using the design generator $G(\xi) = \xi$. This way the learned dynamics model imagines the transitions only for designs seen in the past or the current design. Fig. 1 shows the maximum cumulative episodic reward achieved for each design, with the performance of the model-free baseline shown in blue and our proposed model-based approach shown in red. We are able to show that using model-based imagination leads to at least a similar performance in Half-Cheetah (Fig. 1(a)), but leads to an increase in data efficiency for Walker and Hopper (Fig. 1(b-c)). When it comes to the Walker task in 1(c), which is more difficult compared to Half-Cheetah, the outcome of using our artificial experience is more significant with a smaller variance indicating a more stable and reliable increase in performance. This shows that model-free co-adaptation can benefit from model-based data augmentation, or at least reach similar performance in the case of Half-Cheetah. To better compare both methods and investigate the performance of designs uncovered for Walker task, we train SAC from scratch on the final designs found by model-free and model-based co-adaptation. Fig. 2 shows the maximum cumulative rewards achieved by the top 50% performing designs for both model-free and model-based co-adaptation, demonstrating the ability of the proposed method to outperform model-free co-adaptation (Welch's t-test: $p = 0.024$) and uncover better performing designs.

6.4 Co-Imagination of unknown Designs ($G(\xi) \sim N(\xi, \sigma^2)$ or uniform(Ξ))

Given the previous result, we hypothesise that adding even more diverse imagined experience should further improve our co-adaptation method's performance. We use the design generator $G(\xi) \sim N(\xi, \sigma^2)$, with $\sigma = 0.1$, and $G(\xi) \sim$ uniform(Ξ), meaning adding noise or selecting random designs respectively, to create the imagined rollouts with our dynamics model. Figure 3 shows the results of imagining transitions for unknown designs on the co-adaptation performance with the random selection strategy in black and noisy

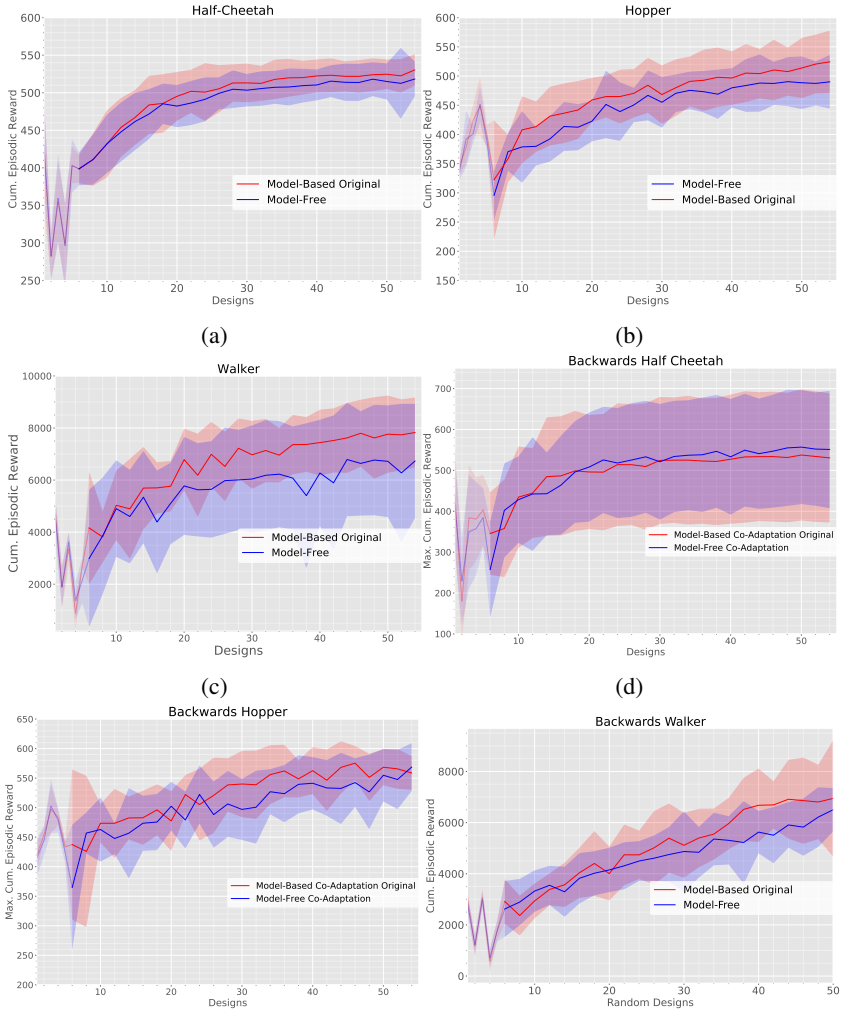

Fig. 1: Cumulative episodic rewards collected by our method with known designs in red and the model-free co-adaptation in blue. The first row shows forward walking and second row showing backwards walking task. Half-Cheetah includes 30 seeds, Hopper 10 and Walker 15. For each design the best performance is reported after 100 episodes.

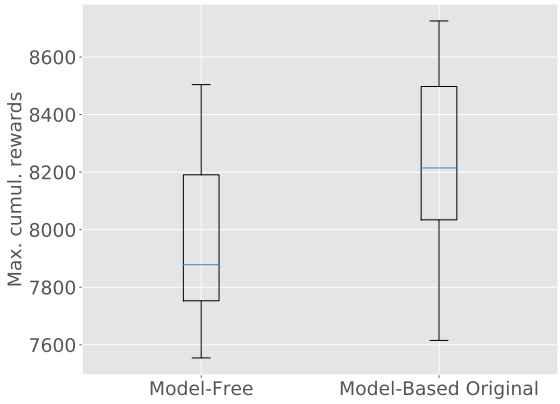

Fig. 2: Comparison between model-free and model-based co-adaptation on the best 50% of designs uncovered (18 samples).

in yellow. When it comes to Half-Cheetah in 3(a) which is easier to learn than others, the noisy strategy is able to continuously learn in it, however, is still not performing as well as the original strategy where the design parameter is unchanged. However, in the more difficult Hopper in 3(b) and Walker in 3(c) where falling over is possible, we can see that noisy and sometimes also random strategy quickly leads to divergence. This divergence can be explained by the limitations of the learned multi-dynamics function, the complex dynamics in some areas of design space or the insufficient training data.

6.5 Discussion

While we see that, as expected, the proposed process of co-imagination shows an increase in performance when hallucinating transitions for known designs, we found that this is not the case when hallucinating transitions for unseen designs. Surprisingly, we found that imagining transitions with learned dynamics functions for previously not seen designs selected either randomly or utilising Gaussian noise can lead to a drastic, even catastrophic, loss in performance. To investigate this further, we evaluated the modelling error of our learned dynamics functions in the Half-Cheetah task. Fig. 4(a) shows the mean square error (MSE) of the model on trajectories from seen designs in red and on trajectories from unseen designs in blue. We can see that the model continues to improve its prediction error throughout the learning process. However, there is a noticeable gap between both errors. We find that the rate at which the prediction error slows decreasing for designs not yet experienced is much higher. This seems to hint at the limited capability of the dynamics model to generalise across the design space. Further analysis of the dynamics of the environment provides additional insight: Given a randomly selected state-action pair, Fig. 4(b) shows the magnitude of change in the predicted next state when changing the design variables only, i.e. $\parallel h(s_t, a_t, \xi_{\text{original}}) - h(s_t, a_t, \xi_{\text{new}}) \parallel^2$ with the original design ξ_{original} and the adapted design ξ_{new}. The magenta circle shows where the original state s_t lies in the design space. Due to the high dimensionality of the design variable, the plot shows the resulting change across a two-dimensional manifold in the

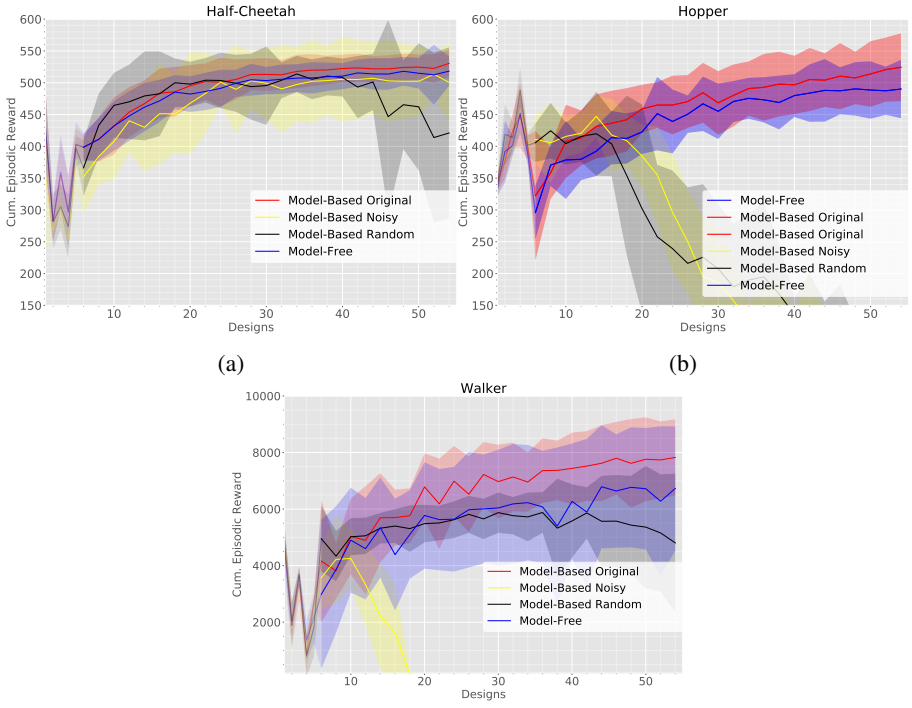

Fig. 3: Cumulative episodic rewards collected by our model-based co-adaptation method with known designs in red, noisy designs in yellow and random designs in black. The model-free co-adaptation baseline [16] in blue. Three agents from left are Half-Cheetah, Hopper and Walker with forward walking task. For each design the best performance is reported after 100 episodes.

design space with two principal components. We can see that the change in the predicted next state is highly non-linear and may explain the inability of the dynamics function to accurately predict transitions for designs not yet experienced. This effect can also be found in much simpler dynamical systems, such as the cartpole system, where changes to the pole length and mass impact the x acceleration (Fig. 4(c)), showing non-linear changes to acceleration along the x-axis in certain states.

7 Conclusion

The problem of co-adapting agents' behaviour and morphology has been studied primarily using model-free learning frameworks or algorithms utilising known dynamics models and simulators. We investigated and proposed using a model-based learning approach to improve the data-efficiency of model-free co-adaptation by co-imagining transitions for unseen behaviour and designs. We showed a performance improvement

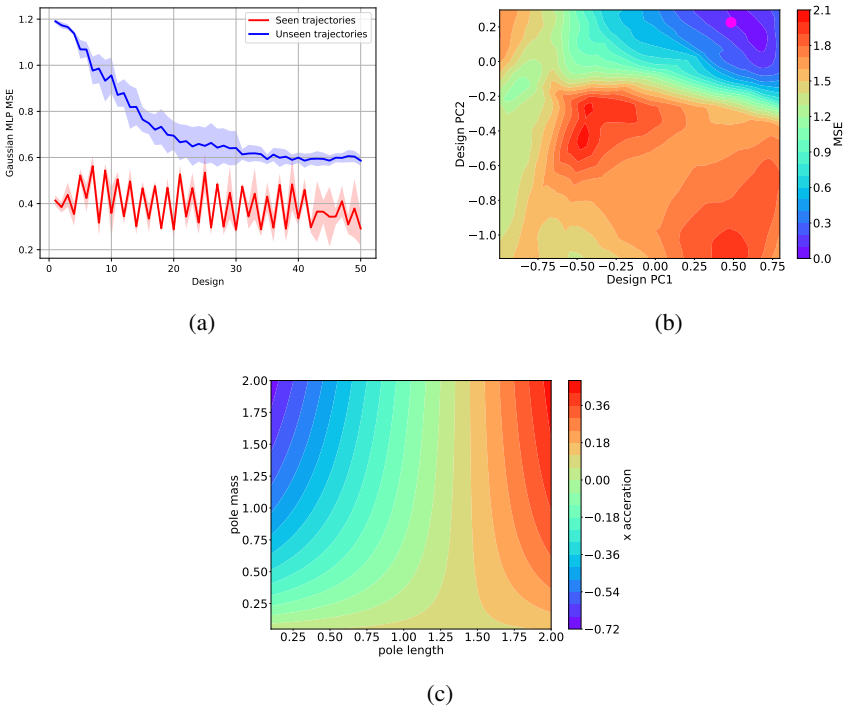

Fig. 4: (a) Mean square error of dynamics model during co-adaptation in Half-Cheetah on designs seen (red) vs an unknown design (blue). (b) Difference in dynamics model over changes of design in Half-Cheetah. Axes are the principal components of the design space. Colours indicate the magnitude of change when predicting the next state, using one design (purple) as reference point. (c) Difference in dynamics when changing pole length and mass of the classic CartPole task.

when co-adapting the design and behaviour in two out of three continuous control tasks. However, we also found limitations of using model-learning approaches in the context of co-adaptation: Using learned models to predict transitions of not yet experienced agent designs and augmenting the training process with the same leads to a severe and sometimes catastrophic deterioration in learning performance, impacting both the policy learning and design optimization process. This uncovered limitation of current approaches provides an interesting avenue for future research into robust and generalizable model-learning approaches, suitable for their use in co-adaptation problems, where we face an infinite number of design variations.

Acknowledgments

This work was supported by the Research Council of Finland Flagship programme: Finnish Center for Artificial Intelligence FCAI and by the German Research Foundation (DFG, Deutsche Forschungsgemeinschaft) as part of Germany's Excellence Strategy – EXC 2050/1 – Project ID 390696704 – Cluster of Excellence "Centre for Tactile Internet with Human-in-the-Loop" (CeTI) of Technische Universität Dresden.
The authors wish to acknowledge the generous computational resources provided by the Aalto Science-IT project and the CSC – IT Center for Science, Finland.
We thank the reviewers for their insightful comments and help for improving the manuscript.

References

1. Bonyadi, M.R., Michalewicz, Z.: Particle swarm optimization for single objective continuous space problems: A review. Evolutionary Computation **25**(1), 1–54 (2017). https://doi.org/10.1162/EVCO_r_00180
2. Chen, T., He, Z., Ciocarlie, M.: Hardware as Policy: Mechanical and Computational Co-Optimization using Deep Reinforcement Learning (CoRL) (2020), http://arxiv.org/abs/2008.04460
3. Chua, K., Calandra, R., McAllister, R., Levine, S.: Deep Reinforcement Learning in a Handful of Trials using Probabilistic Dynamics Models. Advances in Neural Information Processing Systems **2018-Decem**(Nips), 4754–4765 (2018)
4. Coumans, E., Bai, Y.: Pybullet, a python module for physics simulation for games, robotics and machine learning. http://pybullet.org (2016–2021)
5. Dinev, T., Mastalli, C., Ivan, V., Tonneau, S., Vijayakumar, S.: Co-designing robots by differentiating motion solvers. arXiv preprint arXiv:2103.04660 (2021)
6. Gupta, A., Savarese, S., Ganguli, S., Fei-Fei, L.: Embodied intelligence via learning and evolution. Nature Communications **12**(1) (2021). https://doi.org/10.1038/s41467-021-25874-z, http://dx.doi.org/10.1038/s41467-021-25874-z
7. Ha, D.: Reinforcement Learning for Improving Agent Design. Artificial Life **25**(4), 352–365 (11 2019). https://doi.org/10.1162/artl_a_00301, https://doi.org/10.1162/artl_a_00301
8. Ha, S., Coros, S., Alspach, A., Kim, J., Yamane, K.: Computational co-optimization of design parameters and motion trajectories for robotic systems. The International Journal of Robotics Research **37**(13-14), 1521–1536 (2018)
9. Haarnoja, T., Zhou, A., Hartikainen, K., Tucker, G., Ha, S., Tan, J., Kumar, V., Zhu, H., Gupta, A., Abbeel, P., Levine, S.: Soft actor-critic algorithms and applications (2018). https://doi.org/10.48550/ARXIV.1812.05905, https://arxiv.org/abs/1812.05905
10. Hafner, D., Lillicrap, T., Ba, J., Norouzi, M.: Dream to Control: Learning Behaviors by Latent Imagination pp. 1–20 (2019), http://arxiv.org/abs/1912.01603
11. Harpak, A., Garud, N., Rosenberg, N.A., Petrov, D.A., Combs, M., Pennings, P.S., Munshi-South, J.: Genetic Adaptation in New York City Rats. Genome biology and evolution **13**(1) (2021). https://doi.org/10.1093/gbe/evaa247
12. Jackson, L., Walters, C., Eckersley, S., Senior, P., Hadfield, S.: Orchid: Optimisation of robotic control and hardware in design using reinforcement learning. In: 2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). pp. 4911–4917 (2021). https://doi.org/10.1109/IROS51168.2021.9635865

13. Leong, M., Bertone, M.A., Savage, A.M., Bayless, K.M., Dunn, R.R., Trautwein, M.D.: The Habitats Humans Provide: Factors affecting the diversity and composition of arthropods in houses. Scientific Reports **7**(1), 15347 (2017). https://doi.org/10.1038/s41598-017-15584-2, https://doi.org/10.1038/s41598-017-15584-2
14. Liao, T., Wang, G., Yang, B., Lee, R., Pister, K., Levine, S., Calandra, R.: Data-efficient learning of morphology and controller for a microrobot. In: IEEE International Conference on Robotics and Automation (ICRA). pp. 2488–2494 (2019). https://doi.org/10.1109/ICRA.2019.8793802
15. Lipson, H., Pollack, J.B.: Automatic design and manufacture of robotic lifeforms. Nature **406**(6799), 974–978 (2000). https://doi.org/10.1038/35023115, https://doi.org/10.1038/35023115
16. Luck, K.S., Amor, H.B., Calandra, R.: Data-efficient co-adaptation of morphology and behaviour with deep reinforcement learning. In: Kaelbling, L.P., Kragic, D., Sugiura, K. (eds.) Proceedings of the Conference on Robot Learning. Proceedings of Machine Learning Research, vol. 100, pp. 854–869. PMLR (30 Oct–01 Nov 2020), https://proceedings.mlr.press/v100/luck20a.html
17. Mitteroecker, P.: How human bodies are evolving in modern societies. Nature Ecology & Evolution **3**(3), 324–326 (2019). https://doi.org/10.1038/s41559-018-0773-2, https://doi.org/10.1038/s41559-018-0773-2
18. Parks, S.E., Johnson, M., Nowacek, D., Tyack, P.L.: Individual right whales call louder in increased environmental noise. Biology letters **7**(1), 33–35 (2011)
19. Potts, R.: Evolution and environmental change in early human prehistory. Annual Review of Anthropology **41**(1), 151–167 (2012). https://doi.org/10.1146/annurev-anthro-092611-145754, https://doi.org/10.1146/annurev-anthro-092611-145754
20. Racanière, S., Weber, T., Reichert, D., Buesing, L., Guez, A., Jimenez Rezende, D., Puigdomènech Badia, A., Vinyals, O., Heess, N., Li, Y., et al.: Imagination-augmented agents for deep reinforcement learning. Advances in neural information processing systems **30** (2017)
21. Rajani, C., Arndt, K., Blanco-Mulero, D., Luck, K.S., Kyrki, V.: Co-imitation: Learning design and behaviour by imitation. Proceedings of the AAAI Conference on Artificial Intelligence **37**(5), 6200–6208 (Jun 2023). https://doi.org/10.1609/aaai.v37i5.25764, https://ojs.aaai.org/index.php/AAAI/article/view/25764
22. Reil, T., Husbands, P.: Evolution of central pattern generators for bipedal walking in a real-time physics environment. IEEE Transactions on Evolutionary Computation **6**(2), 159–168 (2002). https://doi.org/10.1109/4235.996015
23. Rosser, K., Kok, J., Chahl, J., Bongard, J.: Sim2real gap is non-monotonic with robot complexity for morphology-in-the-loop flapping wing design. In: 2020 IEEE International Conference on Robotics and Automation (ICRA). pp. 7001–7007. IEEE (2020)
24. Schaff, C., Yunis, D., Chakrabarti, A., Walter, M.R.: Jointly learning to construct and control agents using deep reinforcement learning. Proceedings - IEEE International Conference on Robotics and Automation **2019-May**, 9798–9805 (2019). https://doi.org/10.1109/ICRA.2019.8793537
25. Sims, K.: Evolving 3D Morphology and Behavior by Competition. Artificial Life **1**(4), 353–372 (1994). https://doi.org/10.1162/artl.1994.1.4.353
26. Stanley, K.O., Miikkulainen, R.: Competitive coevolution through evolutionary complexification. Journal of artificial intelligence research **21**, 63–100 (2004)
27. Todorov, E., Erez, T., Tassa, Y.: Mujoco: A physics engine for model-based control. In: 2012 IEEE/RSJ International Conference on Intelligent Robots and Systems. pp. 5026–5033 (2012). https://doi.org/10.1109/IROS.2012.6386109