Adapting Neural Models with Sequential Monte Carlo Dropout

Pamela Carreno-Medrano **Dana Kulić** **Michael Burke**

Department of Electrical and Computer Systems Engineering
Monash University
Australia
{pamela.carreno, dana.kulic, michael.g.burke}@monash.edu

This appendix is structured as follows: we introduce multi-task model and SMCD ablations results, a detailed description of baseline models, and additional prediction and mask interpretability results for the two-link arm task in Appendix A. We describe data labelling, multi-task model ablations and additional results on mask interpretability and look-ahead prediction for the drone-landing task in Appendix B. In Appendix C, we show how multi-task model training and SMCD adaptation can be used in a model-based Reinforcement Learning setting. Finally, we describe additional properties of the proposed SMCD method in Appendix D.

A Two Link-Arm - Forward Kinematics Task

A.1 Multi-Task Model Ablations

We completed a hyper-parameter sweep of latent dimension, dropout probability, learning rate, batch size, and the number of epochs to find the best multi-task pre-training model for evaluation in prediction and control. Fig. 1 shows the training loss (mean square prediction error) for all hyper-parameters. Results indicate that the best training loss is achieved by models of medium-to-large size (latent dimensions 512 and 1024), with a dropout probability of $p = 0.5$, and learning rate of 1e^{-4}. Furthermore, we observe the batch size has little to no effect, and models converge quickly after approximately 100 epochs.

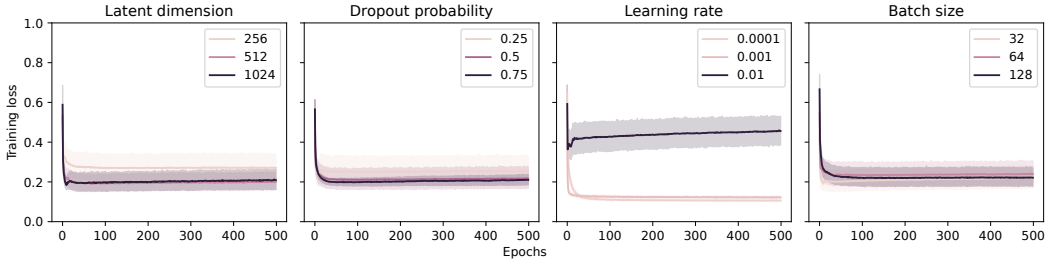


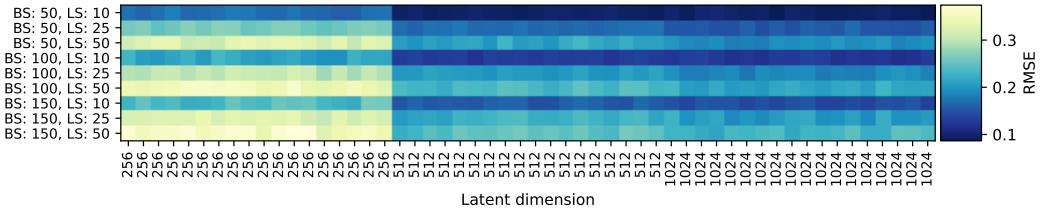
Figure 1: Training loss for all possible hyper-parameters. Each plot shows how the training loss changes when one parameter is fixed (e.g., latent dimension) and the remaining ones take on all possible values.

The same space of parameters was used to find the best Reptile model. Two additional hyper-parameters, the inner learning rate (set to 0.02) and the inner number of epochs (set to 1), were kept constant during the search.

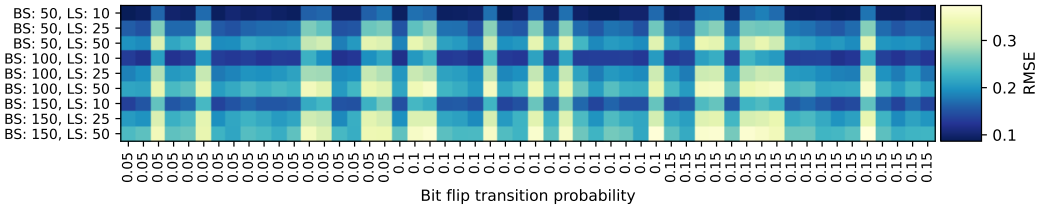
A.2 SMCD Prediction Ablations

A hyper-parameter sweep (bit flip transition probability, measurement noise σ_r, dropout probability, number of particles, and latent dimension) showed that prediction error when using SMCD adaptation is primarily affected by the dropout probability used at training (p=0.5 appeared most effective),

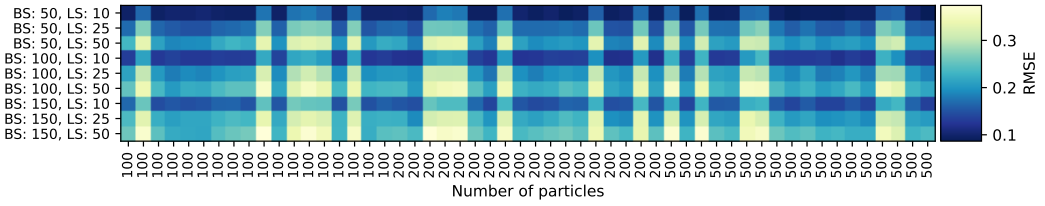
6th Conference on Robot Learning (CoRL 2022), Auckland, New Zealand.



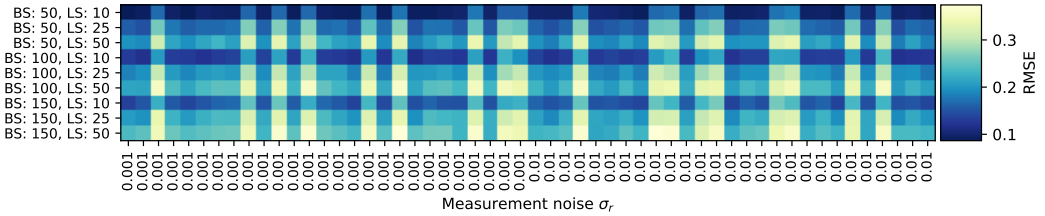
(a) Prediction error sorted by latent dimension.



(b) Prediction error sorted by bit transition probability.



(c) Prediction error sorted by number of particles N.



(d) Prediction error sorted by measurement noise σ_r.

Figure 2: SMCD look-ahead mean squared prediction error for various mask dimensions sorted all parameters considered during hyper-parameter sweep (BS: Burn in adaptation steps, LS: Look-ahead prediction steps.)

followed by the dimensionality of the network layers, indicating that model capacity is important to capture the prediction uncertainty arising from the link length variability. Fig. 2 shows the mean square prediction error for a range of look ahead horizons after a certain number of burn-in adaptation steps sorted by hyper-parameter. It is clear that latent dimension (model capacity) plays the biggest role in adaptation efficacy, and that the filter is relatively robust to changes in transition probability, likelihood noise and mask particle numbers.

A.3 Adaptation Baselines

For the 2-Link forward kinematics prediction task we compared against a Reptile model. The base network (3-layer NN, with ReLU activations) is identical to the model trained using multi-task training and trained on the same dataset. However, instead of sampling data at random, training is sampled by task. We conducted a similar hyper-parameter sweep as in Fig. 1 and selected the best performing model for comparison. We briefly discuss the baselines and draw attention to some noteworthy aspects of their performance below.

- **R+S:** By training the model using Reptile and dropout regularisation, we can apply SMCD for online adaptation. Results show this is relatively effective, and that SMCD can be used alongside meta-learning approaches. We also observe that adapting a model trained using multi-task training appears to be more effective.
- **R+N:** As a meta-learning approach, Reptile seeks to find a representation that can rapidly be adapted to produce low prediction error in new settings with relatively few gradient steps. This means the initial representation may not result in low prediction error. In contrast, the multi-task training approach performs empirical risk minimisation, that is, it seeks to find a model with low prediction error across all settings or task variations. As a result when no adaptation is applied, we find that multi-task training (M+N) is generally more effective.
- **R+G:** Our results show that Reptile performs poorly in a look-ahead prediction setting compared to our proposed approach. A similar trend was observed in an online setting (i.e., adaptation control experiment). Typically, meta-learning strategies like Reptile are adapted to new tasks by collecting and storing large buffers of task specific observations, and then performing a *global* model update using a number of stochastic gradient descent iterations. In a rapidly changing online setting, this is infeasible, and produces additional challenges. Specifically, we can choose to update with a higher learning rate and risk destroying the underlying representation by over-fitting to a local context (this is what appears to happen in R+G), or update very slowly so as to not stray too far from the initial representation (something closer to R+N) resulting in extremely slow convergence. A core benefit of SMCD is that it provides an explicit online update rule, which alleviates these problems.
- **LV+N**: The proposed SMCD method captures task variability and contextual information through a combination of all the weights of the multi-task learned neural model and the inferred dropout masks. Other approaches in the literature do so by explicitly learning a low-dimensional latent embedding (e.g., [1]). To test whether the proposed approach leads to better performance at the cost of a higher-dimensional embedding, we compared against a recurrent variational auto-encoder with a hidden embedding of dimensionality 8. The VRNN model was trained on the same data set as the multi-task model for a total of 200 epochs. Our results indicate that this latent model performs poorly compared to the proposed SMCD model. This can be partially due to the length of the sequences chosen to train the model. Since the multi-task model has no recurrent layer, we limited the sequences fed to the VRNN during training to a length of two time-steps in order to make both models comparable. It is possible that the VRNN model achieves better performance if trained over longer sequences.

The relative performance difference between simple multi-task learning and Reptile observed in our experiments agrees with a recently released work directly comparing multi-task training and fine tuning with meta-learning in RL settings, which corroborates our findings [2]. This large scale study across a broad benchmark of meta-RL tasks concludes that multi-task training performs equally as well or better than meta-RL on similar levels of data.

A.4 Online Adaptation during Control

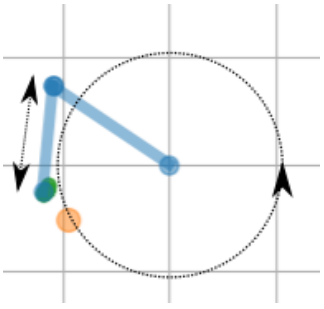


Figure 3: A 2-link arm with sinusoidally varying length l_2 following a moving target.

We also evaluated the performance of SMCD in control (see Fig. 3) using the the two-link arm task. Here, we control the two-link manipulator to follow a moving target $\mathbf{x}_g$ around a circle of radius $r \sim U(0.3, 1.5)$, with starting angle $\phi \sim U(-\pi, \pi)$, measured relative to the base of a randomly initialised manipulator. We use the PD control law $\mathbf{u} = -K_1\mathbf{J}^\dagger(\mathbf{x} - \mathbf{x}_g) - K_2\dot{\boldsymbol{q}}$ to follow the moving target ($K_1 = 20, K_2 = 5$). Here, $\mathbf{J}^\dagger$ is the Moore-Penrose pseudo-inverse of the Jacobian of the network $f_\theta(\boldsymbol{q})$. In order to test online forward kinematics adaptation, we consider a telescoping second link, with sinusoidally varying length $l_2 = 1 + 0.25\sin(\pi k/20)$, with timestep $k \in (0, 100)$. This experiment evaluates convergence speed in response to an initial disturbance, alongside continual online adaptation ability. We benchmark against an oracle model, no adaptation and a model trained using Reptile [3], a first order meta-

learning approach, and investigate adaptation using both gradient descent and SMCD.

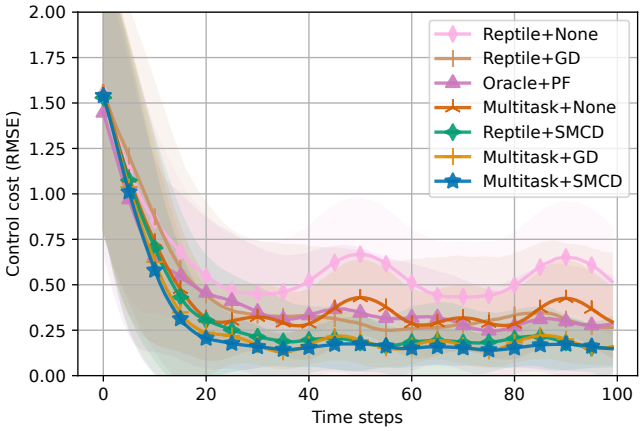


Figure 4: Closed-loop PD control of a 2-link arm following a moving target with a sinusoidally telescoping link (1-sigma shaded traces).

Fig. 4 shows the average control error for the online control task with sinusoidally varying links, across 1000 repetitions. Results are shown for the best performing reptile and multitask trained models for no adaptation (None), online gradient descent adaptation (GD) and online sampling-based adaptation (SMCD). As in the look-ahead prediction case, multitask training with SMCD adaptation is most effective, producing rapid convergence to the moving goal when used for control. Interestingly, particle filtering with a known model (Oracle+PF) struggles to deal with the rapidly changing forward kinematics, but adaptation in mask space or by gradient descent converges much faster.

A.5 Mask Interpretability

To investigate whether the inferred masks capture information about link lengths, we evaluated top k classification accuracy using the masks, where accuracy is determined by calculating the frequency with which the nearest neighbour in link space (euclidean distance between link lengths) lies within the top k nearest neighbours in mask space (euclidean distance between masks). Fig. 5 shows that the masks do contain information about link lengths, and that the capacity to do so increases with model latent dimensionality ld. Inspection of the distances between masks show that there can be many masks corresponding to a good prediction in a particular angle or link-length configuration. This allows for rapid adaptation, as the mask inference mask space does not need to converge to only one of the 2^d possible masks.

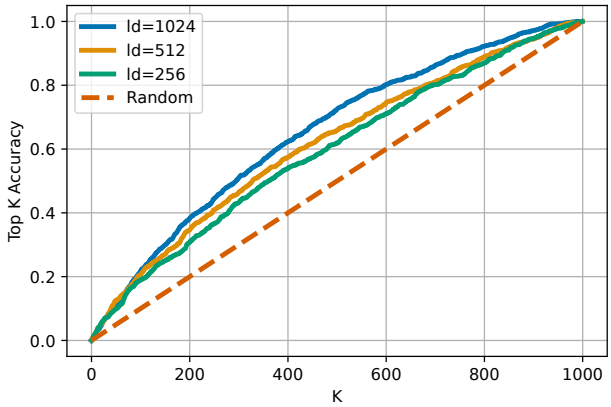


Figure 5: Evaluating link length information captured by inferred masks.

B Drone Landing Task

B.1 Human Data Labelling

A total of four high-level characteristics were tested when evaluating mask interpretability for the drone landing example. These characteristics are landing strategy, operator's skill level, flight phase, and the likelihood of success.

The landing strategy of each trajectory in the dataset was manually assigned after visually inspecting each trajectory. For the flight phase, it was assumed that a tele-operation trajectory is composed of two phases: approach (the drone moves forward toward the target platform) and descent (the drone

comes down to the target platform). An example is shown in Fig. 7. The start and end of each phase were manually defined after visually inspecting each trajectory.

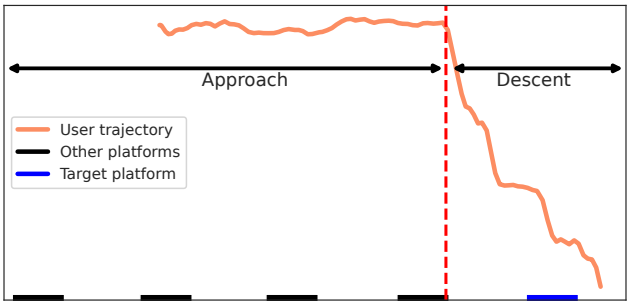


Figure 6: Segmentation of a tele-operation trajectory into two stages: approach (the drone moves forward toward the target platform) and descent (the drone comes down to the target platform).

A trajectory was labelled as successful if and only if the drone's height at the end of the trajectory was inferior to the height of the target platform height and the drone's final position in the XY plane laid within the boundaries platform's top face. To determine an operator's skill level, we computed the number of successful landings for each operator and analysed the operators' performance distribution. We found that performance was normally distributed (see Fig. 7) and proceeded to split users into one of three groups: novices (10% low performers), experts (10% top performers), and intermediate (average performers).

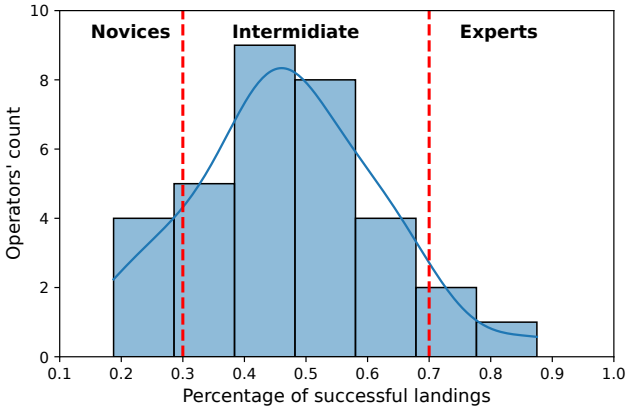


Figure 7: Distribution of operator's performance. Operators in the top and bottom 10% were labelled experts and novices, respectively.

B.2 Multi-Task Model Ablations

A hyper-parameter sweep (latent dimension, dropout probability, learning rate, batch size, and the number of epochs) was done to find the best multi-task pre-training model for the drone landing task. Only a subset of the training dataset was used in the search (192 trajectories). We limited the number of epochs to 100 based on the fast convergence observed for the forward kinematics example. Fig. 8 shows the training loss (mean square prediction error) for all hyper-parameters. Results seem to indicate that models of medium-to-large size (latent dimensions 512 and 1024), with a dropout probability of $p = 0.5$, learning rates of $1e^{-4}$, and batch size of 64 samples performed the best during traning.

The Reptile model used in the evaluation look-ahead prediction error was set to have the same hyper-parameters as the best multi-task pre-training model found during the search. The two additional hyper-parameters required by the reptile model (the inner learning rate and the inner number of epochs) were fixed to the same values used for the forward kinematics tasks.

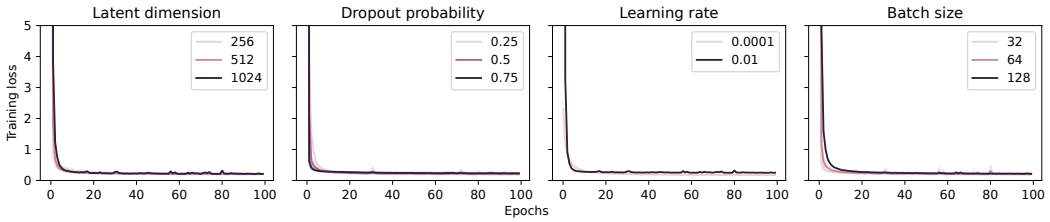


Figure 8: Training loss for all possible hyper-parameters. Each plot shows how the training loss changes when one parameter is fixed (e.g., latent dimension) and the remaining ones take on all possible values.

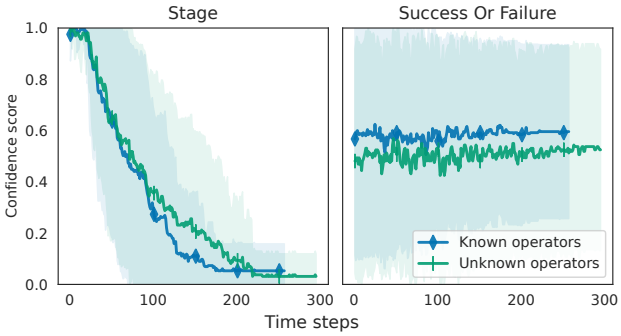


Figure 9: Average confidence score for the correct assignment a trajectory' stage and likelihood of success for all testing trajectories (1-sigma shaded traces). Results are presented for the set of 5-nearest masks.

B.3 Mask Interpretability - Additional Results

Fig. 9 shows the average confidence scores obtained for the current stage of a given trajectory (approach or descent) and whether this trajectory will end in a successful or failed landing. Results are shown for all trajectories in our testing sets.

Overall, although we can identify with high confidence (approx. 1.0) the approach stage of a trajectory (left plot), the masks inferred later on fail to encode enough information to allow for the correct identification of the descent stage. Similarly, the confidence scores for the prediction of success or failure are slightly above the chance level (~ 0.55) for the trajectories of both known and unknown operators. These results indicate that although the inferred masks capture some contextual information about trajectories and operators, not all of the high-level characteristics we have identified as important can be predicted through a simple mask comparison process.

B.4 Look-Ahead Prediction Error

We further assessed the performance of the proposed approach in look-ahead prediction using the drone landing task. Specifically, we evaluated prediction error on trajectories obtained from known (some trajectories were seen during training) and unknown (all trajectories excluded during training) operators.

Fig. 10 shows the RMSE for held-out tele-operation trajectories. We consider two test sets: known (left) and unknown (right) operators for varied burn in and prediction horizon ratios. We use ratios instead of steps since trajectories are of varying lengths. As with the two-link example, we compare the best performing models trained using multi-task pre-training and Reptile meta-learning, with and without adaptation. Note that multi-task pre-training without adaptation corresponds to a behaviour cloning approach to action prediction.

As expected, the look-ahead prediction error obtained for trajectories executed by known operators is lower than the error rates obtained for the unknown operator trajectories. However, as initially observed in the two-link arm example, this difference in performance grows smaller as the duration

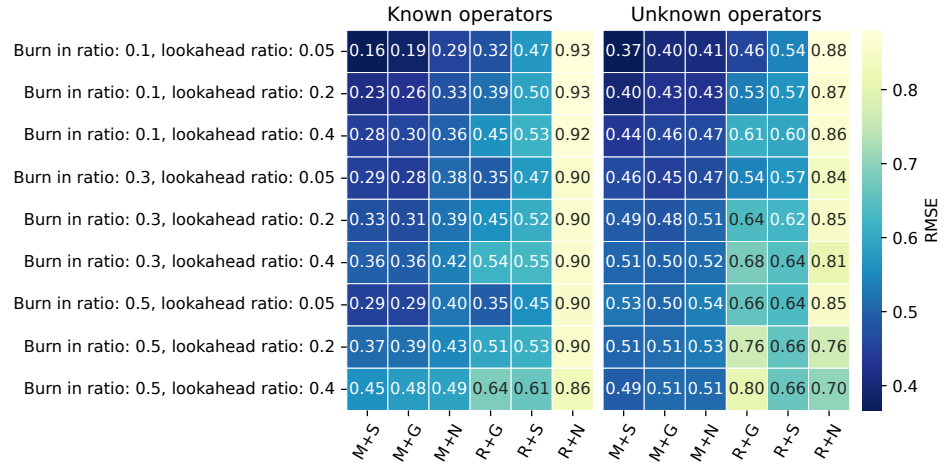


Figure 10: Look-ahead prediction errors for human data, ordered by performance. (M: multi-task, R: Reptile, G: Gradient descent, S: SMCD, N: No adaptation).

of the adaptation and look-ahead prediction horizons increases. This suggests that rather than globally adapting to a particular task instance or operator's tele-operation style, the proposed approach is better at providing accurate models in specific regions of the state space. In terms of overall model performance, we observe that multi-task training with SMCD is most effective, followed by multi-task training with Gradient Descent adaption. Once again, the model trained using Reptile performs poorly even when trained for substantially longer (1000 epochs) than the other models (400 epochs).

C SMCD Application to Reinforcement Learning Settings

C.1 Quadrupedal Robot (Ant)

We also consider a continuous control task from the OpenAI Gym suite. In this task, the goal is to make a four-legged robot to move forward as quickly as possible. We employ the task implementation first introduced in [4] in which the robot's observed state o_t[1] and action u_t spaces are high-dimensional, i.e., $o_t \in \mathcal{R}^{41}$ and $u_t \in \mathcal{R}^{8}$. We use multi-task training with dropout to approximate the robot dynamics and compare different combinations of model-based controllers (Cross Entropy Method - CEM [5], Model Predictive Path Integral - MPPI [6], and Random Shooting - RS [7]) and adaptation strategies (No adaptation, SMCD, and gradient descent) during an online control task. During evaluation, we randomly sample a leg to cripple on this quadrupedal robot. This task measures the ability of the proposed approach to work in high-dimensional spaces and adapt to an unexpected and drastic change to the underlying dynamics. We note that the approximate dynamic model is trained using samples in which all legs are functional.

C.2 Online Control Results

Fig. 11 shows the average reward obtained for a range of model-based RL control strategies, along with gradient and SMCD model adaptation. We used a 3-layer, 1024 dimensional fully connected network with ReLU activations to learn the approximate dynamics model of the quadrupedal robot. The model was trained with 1000000 randomly sampled state-action pairs obtained in an environment where all legs could be actuated. The average rewards reported in Fig. 11 were obtained after completing 110 episode rollouts of 1000 steps each for both the normal and randomly crippled-leg environment. The same setup was applied for each controller and adaptation strategy combination.

Both adaptation strategies perform similarly here, in and out of distribution. Importantly model adaptation does improve control performance, particularly when drastic changes in the underlying dynamics are introduced (right-plot). These results suggest that the proposed approach is also

[1]This observation representation includes the torso position and orientation, joint angles, the torso linear and angular velocity, joint velocities, and the Cartesian orientation and centre of mass.

suitable for high-dimensional task spaces. However, it should be noted that there is significant confounding arising from the controller performance (no model-specific parameter tuning was performed for each controller) and the sampling strategy used to gather training data (all samples were obtained using randomly chosen actions), which may not adequately reflect the performance of the underlying adaptation strategies across all regions of the state space to the extent that the Panda control experiments test. It is likely that improved RL control and exploration strategies are needed to fully reap the benefits of SMCD.

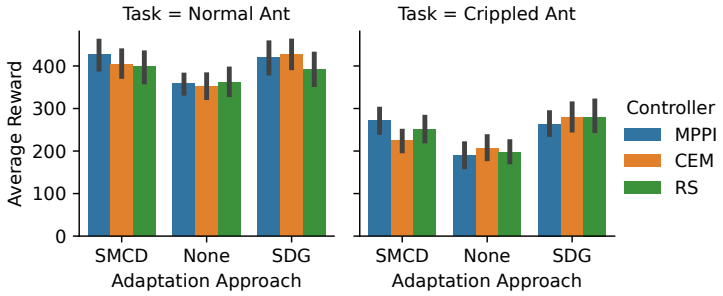


Figure 11: Average reward obtained for the quadrupedal robot online control task. Results are shown for rollouts sampled from both inside (i.e., all legs are functional (left plot)) and outside (i.e., one leg is randomly crippled (right plot)) of the training distribution.

D Additional Properties of SMCD

D.1 Minimum Mean Square Estimator

For control oriented tasks we use a minimum mean square estimator (MMSE) to find a single mask for prediction. This is a good choice of estimator if the distribution over masks is uni-modal. Our primary motivation for choosing this estimator was speed - a MAP estimator would require some level of density estimation and maximisation. Our control experiments did not show problems with estimates arising from multi-modality, and the update rule considers the full distribution. This suggests that the MMSE is not vulnerable to issues arising from this. We suspect that severe multi-modality will result in an average mask with values close to 0.5, which in turn results in a prediction close to the empirical risk minimising mean we'd receive after multi-task training with dropout.

D.2 Computational Complexity

SMCD relies on N forward passes through the network, where N is the number of masks. Although these can be easily batched, this does have an increased memory footprint when compared to gradient based approaches, which require approximately twice the number of model parameters. This greater memory footprint allows for a faster update scheme and more rapid adaptation.

Importantly, SMCD relies on a standard bootstrap particle filter, which breaks the curse of dimensionality – convergence is independent of the state dimension (mask size) and the rate of convergence is in $\frac{1}{N}$ [8]. There is extensive theory on choosing N for filters of this form [9], and the formulation allows for variable batch sizes if desired, to allow even more efficient adaptation.

As a result,we believe the proposed approach is highly scalable, particularly given the typical VRAM associated with modern GPUs, where batch sizes of 100 (see ablations above for particles) are entirely feasible for models of the size typically used in robot control[2].

[2]Our largest model with 5000 particles used 3GB of GPU VRAM

References

[1] S. Sæmundsson, K. Hofmann, and M. P. Deisenroth. Meta reinforcement learning with latent variable gaussian processes. *arXiv preprint arXiv:1803.07551*, 2018.

[2] Z. Mandi, P. Abbeel, and S. James. On the effectiveness of fine-tuning versus meta-reinforcement learning. *arXiv preprint arXiv:2206.03271*, 2022.

[3] A. Nichol, J. Achiam, and J. Schulman. On first-order meta-learning algorithms. *arXiv preprint arXiv:1803.02999*, 2018.

[4] I. Clavera, A. Nagabandi, S. Liu, et al. Learning to adapt in dynamic, real-world environments through meta-reinforcement learning. In *ICLR*, 2019.

[5] M. Kobilarov. Cross-entropy motion planning. *IJRR*, 31(7):855–871, 2012.

[6] G. Williams, A. Aldrich, and E. A. Theodorou. Model predictive path integral control: From theory to parallel computation. *Journal of Guidance, Control, and Dynamics*, 40(2):344–357, 2017.

[7] A. Nagabandi, G. Kahn, R. S. Fearing, and S. Levine. Neural network dynamics for model-based deep reinforcement learning with model-free fine-tuning. In *ICRA*, page 7559–7566, 2018.

[8] D. Crisan and A. Doucet. A survey of convergence results on particle filtering methods for practitioners. *IEEE TSP*, 50(3):736–746, 2002. doi:10.1109/78.984773.

[9] A. Kong, J. S. Liu, and W. H. Wong. Sequential imputations and bayesian missing data problems. *JASA*, 89(425):278–288, 1994.