Published in the proceedings of the Workshop on "Structure & Priors in Reinforcement Learning" at ICLR 2019

PROPRIOCEPTIVE SPATIAL REPRESENTATIONS FOR GENERALIZED LOCOMOTION

Joshua Zhanson, Emilio Parisotto, Ruslan Salakhutdinov
Department of Machine Learning
Carnegie Mellon University
Pittsburgh, PA 15213, USA
`jzhanson@andrew.cmu.edu, {eparisot,rsalakhu}@cs.cmu.edu`

ABSTRACT

In this work, we explore incorporating spatial information into feature and action representations of deep reinforcement learning in the pursuit of a single multi-body locomotion policy. Instead of the traditional feature vectors used in continuous RL, we introduce a body-space state representation that maps sensor readings onto a spatial grid overlay of the robot's body, implicitly encoding relative positional information. Additionally, we introduce a motor-space action representation that projects motor torques out of a similar spatial grid. Models map from input body-space to output motor-space, instead of from the observation space of joint velocities and angles to the action space of joint torques. To demonstrate the multitask and transfer capabilities of models trained with our representations, we introduce an environment based on the Box2D physics simulator that allows creation of robot bodies with arbitrary structure, physical properties, and dimensions and show that models trained using our representations can learn a walking policy transferable across many randomized body configurations. Videos can be found at `https://sites.google.com/view/proprioceptive-representations`.

1 INTRODUCTION

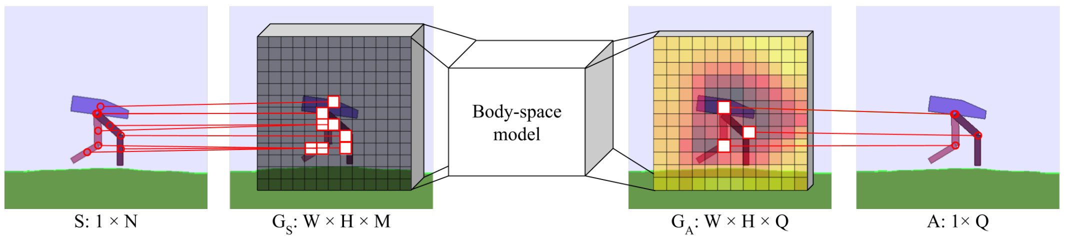

Figure 1: Spatial representations. Note that the policy model now maps from body-space to motor-space, rather than from body and joint feature vector to joint torque vector.

Humans and many animals can actively track the spatial orientation of their bodies without any additional external signals (Proske & Gandevia, 2012) (Riemann & Lephart, 2002) and depend heavily on this proprioceptive sense in order to learn and improve motor function after injury (Aman et al., 2015). Most modern deep reinforcement learning continuous control models use relatively simple feature and action representations in locomotion tasks, often one-dimensional feature vectors containing joint angles or velocities. These representations rarely encode the relative positional information of the agent body, such as the proprioceptive pose of the agent body. Additionally, many modern neural network architectures require a fixed state and action space dimension, which makes transfer to robots with a different number of body parts or joints impossible. Current workarounds range from zero-padding a higher-dimensional state and action space (Chen et al., 2018), learning a representation in a feature space shared across tasks (Gupta et al., 2017), or reusing parts of the model based on a pre-specified semantic parse of the robot morphology (Wang et al., 2018). All of these methods have downsides, either being a potentially unintuitive and inefficient use of model capacity or requiring additional domain expert input.

In this paper, we project joint and sensor information using proprioceptive spatial position onto grid-based representations in order to facilitate learning a general walking policy (see Fig. 1). These new representations only require a small amount of additional information over the standard one-dimensional feature vector. By mapping 1-dimensional feature vectors onto a 3D grid, we can integrate spatial information about the robot in a natural way. With the addition of a similar motor-space representation, where the motor torques are projected out of the appropriate grid cell, we can use models that map from body-space configurations to motor-space torques, rather than from vectors of joint velocities and angles to vectors of joint torques. Such spatial representations enable the use of convolutional models which propagate local information into global representations, and allows the end-to-end discovery of a communication topology between nodes of the agent body structure. Additionally, our spatial representation is invariant to the number of sensors and actuators of a particular robot body, producing policies generalizable to a wide range of robot bodies with differing numbers of sensors or motors.

The main contributions of our work are summarized as follows:

- An observation- and action-space representation for continuous control that enables straight-forward multi-body transfer.
- An extension to the OpenAI Gym BipedalWalker-v2 environment to support a variety of pre-defined robot bodies with arbitrary shape and structure.
- Results demonstrating that the grid-based representations match or exceed baselines on randomized multi-body locomotion tasks.

2 Related Work

A large body of work surrounds transfer in reinforcement learning domains (Taylor & Stone, 2009). One approach involves learning embeddings in a shared feature space to facilitate transfer between morphologically-different robots (Gupta et al., 2017). Another approach focuses on meta-learning combined with a dynamics model of the desired task to ease online adaptation to unseen perturbations of the environment (Clavera et al., 2019). Modular neural network architectures have been shown to improve transfer across tasks and robots (Devin et al., 2017). Evolving the task (Wang et al., 2019a) and the agent morphology (Ha, 2018) can result in complex learned policies, and jointly optimizing the two can yield high-performing agents for a wide range of tasks (Schaff et al., 2018).

In this paper, we examine models with a basic level of transferrability across seen and unseen bodies without any explicit transfer techniques in order to exhibit the applicability of our spatial state representations to general robotic control. Previous work in learning policies transferable across different robot morphologies has examined the issue from several perspectives. The one most aligned with our own is to develop state and and action representations that are generalizable across bodies. This is typically achieved by parameterizing the policy (Wang et al., 2018; 2019b; Pathak et al., 2019) or a model (Sanchez-Gonzalez et al., 2018) with a graph neural network. Wang et al. (2018) employ message-passing graph neural networks on a hand-specified graph mimicking the physical connectivity structure of the robot body. Using this graph connectivity, they are able to achieve transfer of periodic walk-cycle locomotion policies to bodies with an unseen number of body parts or joints. More recent work extended this model to have an evolutionary process drive the design of the robot's physical connectivity in tandem with its graph-based policy (Wang et al., 2019b). Similarly, Sanchez-Gonzalez et al. (2018) and Pathak et al. (2019) show that the propagation of information in GNNs to different body parts plays an important role in robot locomotion. An alternative to representation design is to condition a policy on an embedding computed from characteristics of the robot body, such as was done in Chen et al. (2018), where vector embeddings computed from structural information about the robot body were used as side-information to modulate a policy. Results demonstrated success of this conditioning mechanism on both zero-shot transfer and transfer after fine-tuning. Our work differs from these previous approaches in that we forgo explicitly providing the (graph) communication topology of the robot, and instead aim to self-discover an optimal communication between nodes in the body through the end-to-end learning of a convolutional network.

3 Spatial Representations

In this section, we describe our body-space and motor-space representations. Our body-space encoding consists of a grid superimposed over the agent body. Sensor information is projected into the grid cells occupied by the sensors. Action outputs, i.e. motor torques, can be likewise projected out of the companion motor-space grid output of our models. See Fig. 1 for an overview of the process. We describe our spatial representation in two dimensions for clarity and consistency with our two-dimensional test environment. However, the generalization to three-dimensional control is straightforward.

3.1 Body-Space and Motor-Space Encodings

In standard continuous control, we are given a N-dimensional feature vector $f = (f_1, \ldots, f_N)$ consisting of information obtained from sensors about the robot's state. Each feature has one of M sensor types $\{S_1, \ldots, S_M\}$. We assume that each of the elements of the feature vector were obtained from sensors which have a physical location on the robot's body. Let $p(f_i)$ denote the physical sensor location of the ith feature and let the P-dimensional action vector $a = (a_1, \ldots, a_P)$ be the desired actuator control outputs at any given time step. Each actuator has a physical location on the robot body, denoted by $p(a_i)$, and has one of Q actuator types $\{A_1, \ldots, A_Q\}$.

We define a body-space representation by projecting the sensor values onto a discrete uniform 3D grid G_S with dimensions $W \times H \times M$, with a channel for each of M sensor types. The resolution $W \times H$ of the grid to be superimposed over the agent body is chosen as a hyperparameter. Each sensor has its information written into the superimposed grid cell containing its physical sensor location $p(f_i)$, in the channel corresponding to its sensor type. The remaining cells are filled with zeros. Note that we can encode the implicit relative positional information explicitly by writing a 1 instead of the sensor information into each grid cell containing a sensor and appending this indicator grid to the existing feature vector.

We likewise define an action-space representation with Q different actuator types as a $W \times H \times Q$ grid, superimposed over the agent's body. The action for each actuator is projected out of the grid cell containing the actuator location $p(a_i)$, in the channel corresponding to its actuator type. Such an action space representation enables a straightforward fully-convolutional mapping from the body-space representation and allows learning of a more general policy independent of specific motor index orderings, as well as of feature vector and action vector dimensions.

3.2 Models

Given a body-space representation, we can utilize several models to obtain a policy. Convolutional neural networks are capable of capturing spatial dependencies in our encodings and are therefore a natural choice for learning policies that can take advantage of both the proprioceptive information encoded in the relative positions of the sensors and joints as well as the readings of those sensors and joints.

We tested three different convolutional architectures: a standard deep convolutional network (CNN), a Residual-Network (ResNet) architecture and a convolutional-LSTM (ConvLSTM) architecture. The ConvLSTM is the most expressive of these architectures, propagating information not only locally through space but also locally through time, and was the most successful in training out of the three body-space models. We compare our models to a generic multi-layer perceptron (MLP) baseline with a late-fusion LSTM, optimized for the BipedalWalker-v2 environment. We also study the effect of adding positional information, consisting of a flattened indicator grid (with ones in cells containing sensors), to the MLP baseline, which we call FlatMLP. These models are described further in Appendix B.

4 Experiments

In the following section, we examine the body-space representation's role in learning a general walking policy transferable over a range of similar robot body configurations. We introduce an extension of the OpenAI Gym (Brockman et al., 2016) BipedalWalker-v2 environment called JSONWalker,

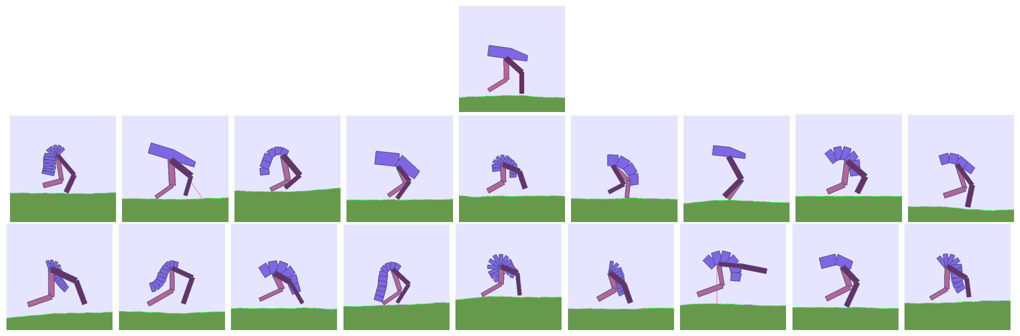

Figure 2: Canonical Bipedal Walker (top) and sampling of randomized variations.

which allows creation of arbitrary robot bodies via an input JSON file. The goal of the environment remains to travel as far as possible without falling, i.e. without the predesignated "hull" segment touching the ground. Reward is given based on distance traveled, with a small penalty for applied motor torques and a larger penalty for the hull hitting the ground. We call the base robot body for each environment (i.e. the original BipedalWalker) the *Canonical* body while variations of this canonical body are referred to as *Randomized* bodies. In order to maintain consistency among randomized bodies during model comparison, we generate a dataset of random walker bodies and perform a train/valid/test split. At training time, a body is randomly sampled from the training set at the start of each episode. Early stopping / model selection is performed by running every model checkpoint for a set of 100 episodes on every body in the validation set and averaging the results. We then use the test set after model selection to determine zero-shot policy transfer to new walker morphologies.

4.1 Bipedal Walker

We validate our body-space representations and corresponding models on a randomized dataset based on the BipedalWalker body, which consists of a walker with 2-joint legs (hip + knee) without any feet and a hull connecting the two legs together. In addition to randomizing the dimensions of each body part of the canonical BipedalWalker-v2 up to a 25% tolerance, we add variation by splitting the hull into a number of flexible segments with equal length. Our datasets have an equal number of bodies with each number of segments—e.g. there are 10 bodies with 1 hull segment, 10 bodies with 2 hull segments, etc. However, we randomly choose which hull segment to attach the legs to, resulting in bodies that range from approximately balanced to severely front- or back-heavy (see Figure 2).

Although the hull vertebrate are deformable and actuated, we do not allow the agent to control the motors between hull segments. Thus, the difficulty in this environment lies in the often-unbalanced nature of the multi-segment bodies, which requires a single policy to learn different gaits and adapt on-the-fly to the particular body configuration and balance. We do not enable sensor reporting of the segment bodies or joints, which requires the models to deduce the balance of the body from the behavior of the legs and center hull segment alone. Additionally, the hanging hull vertebrae add some swinging force, requiring the bodies to have some level of robustness as they journey across the uneven terrain of the environment. For details on the construction and composition of our dataset, see Appendix A.

In addition, for the experiments presented in this paper, we resolve potential issues that arise when sensors and actuators overlap in 2D position by introducing a simple 2-channel "depth" to our mapping. This extends the setting described in Section 3.1 from 2-dimensional pixel features to 3-dimensional voxels. Since there are only 2 depth values, we do not require a 3D convolution and instead just stack the two depth channels and, for simplicity, use the same models we would in a 2-dimensional setting.

Figure 3 shows training and validation results of the baseline MLP model ("MLP"), our convolutional-recurrent body-space model ("ConvLSTM"), the MLP augmented with grid information ("FlatMLP"), and, as a sanity check, an MLP baseline trained only on the canonical body ("Canonical MLP"). Table 1 gives the proportion of bodies in each dataset solved, where "solved" is defined as a threshold of mean 200 episode return. To test performance with bodies more "out-of-distribution", the validation

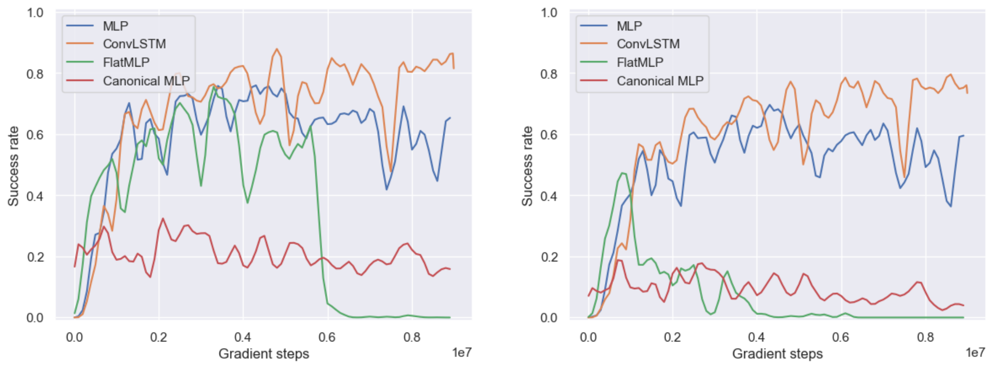

Figure 3: Smoothed training (left) and validation (right) curves of Randomized BipedalWalker with off-center, segmented hull.

Model	Multitask	Zero-Shot Transfer
MLP	84.6%	76.1%
ConvLSTM	**89.9%**	**83.2%**
FlatMLP	51.9%	21.8%
Canonical MLP	39.3%	24.6%

Table 1: Proportion of bodies solved by each model on each dataset. A body is solved if 100 evaluation episodes achieve return greater than 200.

and testing datasets contain bodies with unseen numbers of hull segments—see Appendix A for more information.

The ConvLSTM model is able to obtain the highest success rates across all three datasets. The addition of positional information to the baseline MLP policy (FlatMLP) does not result in a higher quality policy, which is likely due to the spatial variance of the lower feedforward layers. Because there could exist physically larger models in the test set, grid positions further away from the model center which were not visited during training have effectively random weights. These random weights could thus explain the catastrophic performance of FlatMLP when transfering to new bodies. The results of the canonical MLP serve as a demonstration that training on a single body (the original BipedalWalker-v2) is not sufficient to perform well in this multi-body setting.

5 Conclusion and Future Work

In this work, we addressed the issue of learning a general walking policy that can be transferred between robot bodies requiring different gaits. We introduce body-space and motor-space spatial representations in order to facilitate learning such policies, and propose a randomized locomotion task to evaluate models trained with and without these representations. We demonstrated that our spatial representations enabled superior multitask and transfer learning within these domains when compared to baselines that did not take spatial information into account.

Future directions we are looking at is to validate our representations on a larger variety of robot configurations, including the Raptor and Dog bodies of Peng et al. (2016), deformable soft-body robots, and popular MuJoCo control benchmarks including Hopper, HalfCheetah, and Ant. We are also interested in determining whether our method can be extended to work on more complex locomotion scenarios (e.g. a randomized BipedalWalkerHardcore). In addition, we plan to release the JSONWalker environment as a potential rapid test-bed for research in multi-robot transfer. The JSONWalker enviornment enables the rapid construction and automatic randomization of a large potential variety of robot morphologies, and unifies them within a simple gym-based API.

REFERENCES

Joshua E Aman, Naveen Elangovan, I-Ling Yeh, and Jürgen Konczak. The effectiveness of proprioceptive training for improving motor function: a systematic review. *Frontiers in human neuroscience*, 8:1075, 2015.

Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. Openai gym. *CoRR*, abs/1606.01540, 2016.

Tao Chen, Adithyavairavan Murali, and Abhinav Gupta. Hardware conditioned policies for multi-robot transfer learning. In *Advances in Neural Information Processing Systems 31*, pp. 9333–9344, 2018.

Ignasi Clavera, Anusha Nagabandi, Simin Liu, Ronald S. Fearing, Pieter Abbeel, Sergey Levine, and Chelsea Finn. Learning to adapt in dynamic, real-world environments through meta-reinforcement learning. In *International Conference on Learning Representations*, 2019.

Coline Devin, Abhishek Gupta, Trevor Darrell, Pieter Abbeel, and Sergey Levine. Learning modular neural network policies for multi-task and multi-robot transfer. In *International Conference on Robotics and Automation*, 2017.

Abhishek Gupta, Coline Devin, Yuxuan Liu, Pieter Abbeel, and Sergey Levine. Learning invariant feature spaces to transfer skills with reinforcement learning. In *International Conference on Learning Representations*, 2017.

David Ha. Reinforcement learning for improving agent design. *CoRR*, abs/1810.03779, 2018.

Deepak Pathak, Chris Lu, Trevor Darrell, Phillip Isola, and Alexei A. Efros. Learning to Control Self-Assembling Morphologies: A Study of Generalization via Modularity. *CoRR*, abs/1902.05546, 2019.

Xue Bin Peng, Glen Berseth, and Michiel van de Panne. Terrain-adaptive locomotion skills using deep reinforcement learning. *ACM Trans. Graph.*, 35(4):81:1–81:12, 2016.

Uwe Proske and Simon C. Gandevia. The proprioceptive senses: Their roles in signaling body shape, body position and movement, and muscle force. *Physiological Reviews*, 92(4):1651–1697, 2012.

Bryan L Riemann and Scott M Lephart. The sensorimotor system, part ii: The role of proprioception in motor control and functional joint stability. *Journal of athletic training*, 37(1):80–84, 2002.

Alvaro Sanchez-Gonzalez, Nicolas Heess, Jost Tobias Springenberg, Josh Merel, Martin A. Riedmiller, Raia Hadsell, and Peter Battaglia. Graph networks as learnable physics engines for inference and control. *CoRR*, abs/1806.01242, 2018.

Charles B. Schaff, David Yunis, Ayan Chakrabarti, and Matthew R. Walter. Jointly learning to construct and control agents using deep reinforcement learning. *CoRR*, abs/1801.01432, 2018.

Matthew E. Taylor and Peter Stone. Transfer learning for reinforcement learning domains: A survey. *Journal of Machine Learning Research*, 10:1633–1685, 2009.

Rui Wang, Joel Lehman, Jeff Clune, and Kenneth O. Stanley. Paired open-ended trailblazer (POET): endlessly generating increasingly complex and diverse learning environments and their solutions. *CoRR*, abs/1901.01753, 2019a.

Tingwu Wang, Renjie Liao, Jimmy Ba, and Sanja Fidler. Nervenet: Learning structured policy with graph neural networks. In *International Conference on Learning Representations*, 2018.

Tingwu Wang, Yuhao Zhou, Sanja Fidler, and Jimmy Ba. Neural graph evolution: Towards efficient automatic robot design. In *International Conference on Learning Representations*, 2019b.

Ronald J. Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning*, 8(3):229–256, 1992.

Attribute	Canonical value $\pm$ tolerance
Hull height	16.0 $\pm$ 4.0
Hull width	64.0 $\pm$ 16.0
Upper leg height	34.0 $\pm$ 8.5
Upper leg width	8.0 $\pm$ 2.0
Lower leg height	34.0 $\pm$ 8.5
Lower leg width	8.0 $\pm$ 2.0

Table 2: Parameters altered to produce the random BipedalWalker bodies.

A Environment Details and Randomization Parameters

We randomized the parameters listed in Table 2 by $\pm 25\%$ of their canonical values, resulting in walker bodies that are different from the canonical body but could still share a fixed-size body-space and motor-space grid. Since changes to sizes of body parts can compound, we found that 50% randomization was too aggressive and would require different grid sizes and resolutions for the largest and smallest of the generated bodies. We assign 6 configurations of hull segments $i \in [1, \ldots, 12]$ to be training configurations, 3 to be validation configurations, and 3 to be testing configurations. Then, for each configuration, we generate 10 randomized bodies, resulting in a total of 120 body configurations—a training set of 60 bodies, a validation set of 30 bodies, and a testing set of 30 bodies. The hull segment configurations assigned to the training dataset were 1, 2, 4, 6, 10, and 12, the configurations assigned to the validation dataset were 3, 7, and 8, and the configurations assigned to the testing dataset were 5, 9, and 11.

B Model Architectures and Hyperparameters

Hyperparameter	Values
Adam learning rate	[0.0001, 0.0005, 0.00025, 0.001, 0.00005, 0.000025, 0.00001]
Adam betas	(0.9, 0.999)
Adam epsilon	0.001
Discount factor γ	0.99
GAE parameter τ	1.00
Timesteps per rollout	20
Frame stack	1
Max episode length	10000
Body-space grid dimensions $W \times H \times M$	$24 \times 24 \times 20$
Motor-space grid dimensions $W \times H \times Q$	$23 \times 23 \times 2$
Length of single grid cell in world space	[5.0, 10.88]

Table 3: Hyperparameters tested.

Gaussian policies with mean and variance μ and σ are a natural fit for continuous control environments (Williams, 1992). Therefore, our models map from the state-space of the task to the parameter space of our policy distribution $\pi \sim \mathcal{N}(\mu, \sigma)$, from which we sample an action. See Figure 4 for a breakdown of the layers used in each network architecture.

For clarity, we also show the state encoder and action decoders, as well as the body-space representation and action-space representation when appropriate. These modules simply encode the feature vector into the appropriate body-space representation and decode the action-space representation output of our models into an action vector of motor torques. The value baseline of body-space models is an action-space output, as described in the previous section. A value baseline scalar is decoded from this action-space output by taking the mean of all cells that contain an actuator, which we found to be a simple and intuitive method of obtaining a value baseline from our action-space representation that does not add any additional parameters or layers.

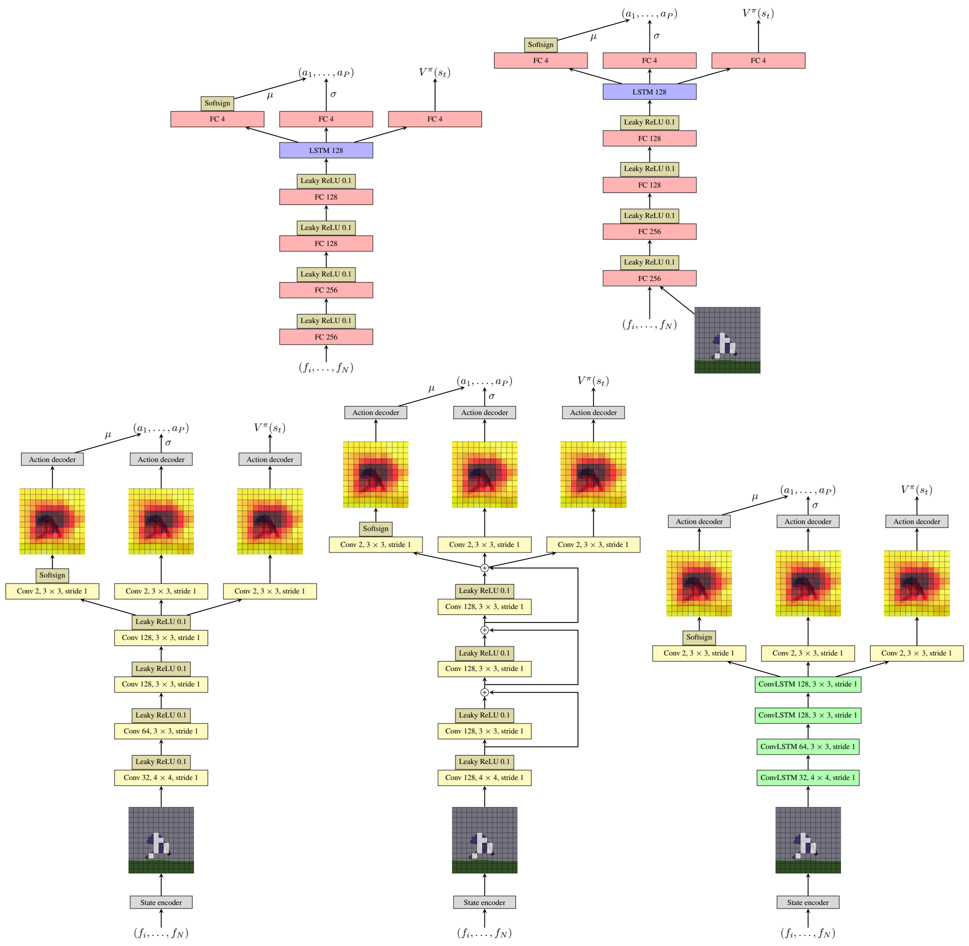

Figure 4: Model architectures. **Top left:** MLP baseline with late-fusion LSTM (MLP). Tensor dimensions are for canonical BipedalWalker body. **Top right:** MLP baseline with added grid positional information (FlatMLP). **Bottom left:** Body-space convolutional model (CNN). **Bottom center:** Body-space convolutional model with residual skip connections (ResNet). **Bottom right:** Body-space convolutional LSTM model (ConvLSTM).

Using the AMSGrad variant on the Adam optimizer, we performed a relatively simple grid search where we only varied the learning rate. Table 3 summarizes the hyperparameters tested. Due to the relatively high cost of running a comprehensive battery of evaluations on all randomized bodies in a dataset, we chose high-performing models by running a test episode on a random body every 10K gradient steps and choosing the body that exhibited the strongest training curves before the process of model selection by validation data.