Dual Models to Facilitate Learning of Policy Network

Yidong Mei
Department of Automation
Shanghai Jiao Tong University
Shanghai, China
meiyidong@sjtu.edu.cn

Yue Gao*
MoE Key Lab of Artificial Intelligence
Department of Automation
Shanghai Jiao Tong University
Shanghai, China
yuegao@sjtu.edu.cn

Shaoyuan Li
Department of Automation
Shanghai Jiao Tong University
Shanghai, China
syli@sjtu.edu.cn

2020 Chinese Automation Congress (CAC) | 978-1-7281-7687-1/20/$31.00 ©2020 IEEE | DOI: 10.1109/CAC51589.2020.9327288

***Abstract*—Learning to control agents without the prior knowledge of its own kinematic is a challenging problem. Recently neural network architecture models can be utilized for robust nonlinear fitting. However, complex sensor inputs including RGB images and robot joints states bring difficulties to the convergence of the control policy due to large input space and complex network structure. Besides, adding temporal information can bring more perception capabilities to the agent but also increase the number of parameters for policy networks. We present a new method called DualM-Control, which exploits dual models including self model and image model. DualM-Control algorithm can compress high dimensional spatial and temporal sensor inputs into low dimension data, thus making it possible for policy network with a few thousand parameters to evolve with evolution strategy. We test the algorithm on a challenging simulation environment created on gym and the performance exceeds existing approaches.**

***Index Terms*—Evolution Algorithm, Robot control, Image processing, Reinforcement Learning.**

I. Introduction

Human develops a set of complex perception system [1], not only sensing his own self states, but also understanding the environment. We uses these perceptions in a predictive way. In other words, we develop models for our own states and the environment from past experience, hence we can predict the consequences of every action and this guide our decisions [2]. With this mechanics, we are able to accomplish certain tasks even though we never encounter the same situations before. For example, when we hike in a mountain area, we have a brief idea of whether we will be stumbled by the rock and if we are closer to the mountain top. All these judgments include our perception and prediction of not only our own states, but also the environment and task-related information.

Inspired by this mechanism, many algorithm in reinforcement learning (RL) utilizes a learned predictive model [3], [4] to further optimize policy network and it has become an important branch of reinforcement learning, called model-based reinforcement learning. Compared with model-free approaches, model-based RL algorithms are welcomed for its good data efficiency.

This work is sponsored by the National Natural Science Foundation of China (Grant No.61903247 and No.U1613208).

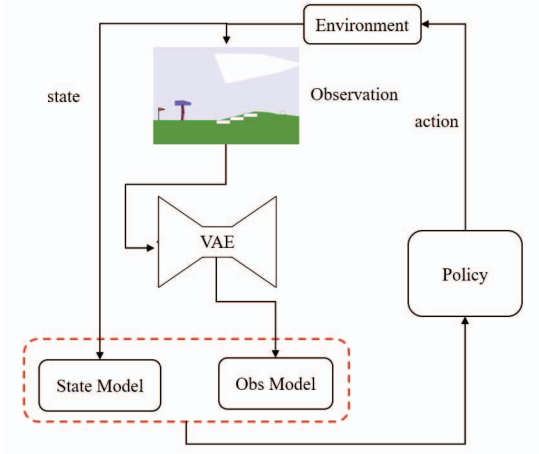


Fig. 1: DualM-Control utilizes states and observations to train state model and observation model first. Policy network is built with inputs of state, observation and LSTM represented models' hidden states.

In agent control tasks, agent learns representations of spatial and temporal information from sensory [5]. For example, in a robot walking task, angles and angular velocities of joints, feet forces or other self states can be directly acquired from sensors reading. In addition, image data from a camera mounted on the robot can provide the robot with environment or task related information. Besides, in order to extract relation and dynamics between data from different time steps, algorithms such as [6]–[8] utilize data obtained from sensors over a period of time. With temporal information, the agents are able to learn a more robust policy. However, a larger input state space brings more challenge for convergence.

In this work, we proposed an algorithm called DualM-Control that exploits recurrent neural networks (RNNs) to fit two separate predictive models of agent's self states from sensors reading and observation images from RGB camera as shown in Fig1. When predicting what the observation will be in next time step, it is impossible to use pure images as input for the reason that a typical 3 channels image will have tens of thousands of pixels and RNNs are not able to process such

high dimensional data. So, DualM-Control uses a Variational Auto-encoder (VAE) [9] to encode images into latent space first. In the experiment done in section IV, we utilize a special form of RNN named Long Short-Term Memory (LSTM) [10] network to construct our dual models for its better performance compared with common RNNs when it comes to exploit inputs information and avoid gradient vanishing.

The key contributions of this paper are summarized as follows:

- DualM-Control algorithm builds its policy network with both observation images and self states, which makes it possible to accomplish more complex tasks.
- The use of VAE has greatly reduced the dimensions of images and further reduced the complexity of the policy network, easing the difficulty of policy network convergence.
- Two pretrained LSTM represented models compress sequential temporal information into hidden states of LSTM, decreasing the dimensions of policy's inputs.

II. Background

A. Model-based control with neural network

Models in control refer to the dynamics of agents. To be more specific, given current states and actions or given states and actions over a period of time, *model* is able to predict what next states will be. On the other hand, control with neural network have arouse increasing attentions in academic community and combination of models with NN controller has led to many novel and interesting progress in recent years.

When we mention model-based control with neural networks, an important field is model-based RL. Among all these model-based algorithms, policy based Guided Policy Search (GPS) utilizes a Gaussian mixture model (GMM) [11] to fit dynamic model of agent and optimal control algorithm is then used to guide policy network to global optimum [12], [13]. Works based on GPS have further broaden the application scenarios to tasks that use raw image as input to policy network [14]. Another branch of model-based RL algorithms is value based method like Value Iteration networks (VIN) which exploit structure of convolution network to perform classical value iteration algorithm [15], [16]. In VIN, transition kernels are learned to encode probability of where the agent will be given current position and action, i.e., *transition probability model* of the agent. Besides, Algorithms like probabilistic inference for learning control (PILCO) [17], [18] utilize a learned model to help estimate long term rewards and thus directing update of the policy network.

Apart from model-based RL, Evolution strategy (ES) also plays an important role in optimizing model-based NN controllers. Concept like *World model* [19] proposed by David Ha has aroused the interest of many researchers [20]. It learns a representative model from pure image input and builds policy network upon it, then ES is served to update its parameters. However, it failed to take self states information into consideration, which limits its applications in more challenging scenarios.

B. Variational Auto Encoders

A Variational Auto Encoder [9], [21] is consist of three parts including encoder, sampling and decoder. The encoder network learns the features of the input image x and output mean μ_x and variance σ_x of a Gaussian distribution, which is latent space of input image. Then it samples on the Gaussian distribution and input the samples into the decoder network. Decoder of VAE restores latent parameters to the same size of the input image, denoted as $\hat{x}$. Loss function of VAE is set as in:

$$loss = \|x - \hat{x}\|_2 + KL[\mathcal{N}(\mu_x, \sigma_x), \mathcal{N}(0, 1)] \tag{1}$$

First part in (1) is to ensure output of the VAE is as consistent as possible with the input. Second part is to prevent the network from merely storing image features and make sure it retain certain "generating" characteristics. After training VAE via unsupervised learning, DualM-Control algorithm utilizes the encoder of VAE to extract features and reduce dimensions of observation image.

III. Learning to control with dual models

Core ideas of our algorithms are came up with based on three intentions: (1) We want to use information from both states and observations over a period of time, i.e., sequential temporal inputs with both sensors reading and images. (2) In order to use images as inputs for the policy network, DualM-Control utilize VAE to reduce dimensions of image, otherwise the policy network may be too complex to converge. (3) Pretrained LSTM-based dual models are able to compress high-dimensional temporal inputs into low-dimensional hidden states of LSTM.

In this section, we intent to introduce details and theories of our proposed algorithms. The contents are mainly on why DualM-Control builds its algorithm this way and how to train the policy network.

A. Dual Models

When human beings or any other intelligent creatures trying to complete certain tasks, they tend to build more than one mental model in the brain. Generally speaking, self state model and observation model are the most basic models. A self model will enable us to have an overall understanding of our abilities and is essential for safety protection of ourselves. Whereas vision is the most important and useful part of human perception systems and a observation model will help us a lot when accomplishing a variety of different tasks in almost all scenarios.

State model is focus on agent's self state and is expected to help extract information referred to as affecting its basic motions, such as walking and jumping. Whereas observation model is regarded as decoding abstract representations of environment or task-related information like terrains or targets. They both serve as inputs of the policy network and working together they enable policy network of DualM-Control with more comprehensions of the tasks. With this two models, agents can estimate relative long terms of both state related

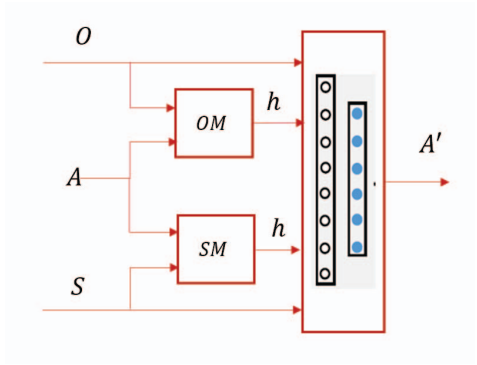


Fig. 2: Policy network takes states, observation and LSTM represented models' hidden states as inputs and a simple MLP outputs action in next step.

and task related rewards or consequences of every actions, making it possible for the agent to get more acquainted with both condition of itself and the environment.

Models in our implement is define as an estimator or predictor of what next states will be given states and actions over a period of time. Classically, a simple multi-layer perceptron (MLP) is used to fit models of the agent, but one obvious flaw is that it fails when the information continuously input to the model network over time. That is to say, MLP cannot handle temporal inputs. To tackle this problem, DualM-Control exploits RNNs to represent both state model and observation model. RNN represented models have the ability to compress temporal inputs into RNN' hidden states. Hidden states together with current inputs (usually state and action) can predict next state of the agent. Therefore, a relatively simple Neural Network, called policy network, is built to exploit the representations of memories of past observations and states by taking hidden states of RNN models as input. In experiment, we use a special type of RNN called LSTM as introduced as dual models.

But before building our LSTM implemented observation model, we should know that a typical image is consisted of a matrix of more than tens of thousands of pixels and it is unrealistic to predict on such high dimensions of input using LSTM. Hence, we use the encoder part of VAE to perform dimension reduction on original image data and predict with image latent parameters.

Let s_t, o_t and a_t denote state, observation and action at time t. z_t is defined as latent parameters of o_t. We define $P^{\theta}_{sm}(s_{t+1}|s_t, a_t, h^{sm}_t)$, $P^{\psi}_{om}(z_{t+1}|z_t, a_t, h^{om}_t)$ state model and observation model of DualM-Control where h^{sm}_t and h^{om}_t are hidden states of models. Learning objectives of this two models are to minimize the difference between estimation of the future states and observation with real ones:

$$\min_{\theta} \sum_{t=0}^{N-1} \left\| P^{\theta}_{sm}(s_t, a_t) - s_{t+1}) \right\|_2 \tag{2}$$

$$\min_{\psi} \sum_{t=0}^{N-1} \left\| P^{\psi}_{om}(z_t, a_t) - z_{t+1}) \right\|_2 \tag{3}$$

Algorithm 1 DualM-Control Algorithm

1. Collect actions, agent states, images(Observations) in a set T generated from interaction with environment with a random policy π (parameters ϕ)
2. Train VAE with the image subset of T (Denote encoder of VAE as $f(\cdot)$) via unsupervised learning.
3. Train dual model network, Observation model (parameters ψ) and State model (parameters θ) using T via supervised learning.
4. Evolve policy network with CMA-ES to maximize the expected cumulative reward of each episode.

B. Policy network

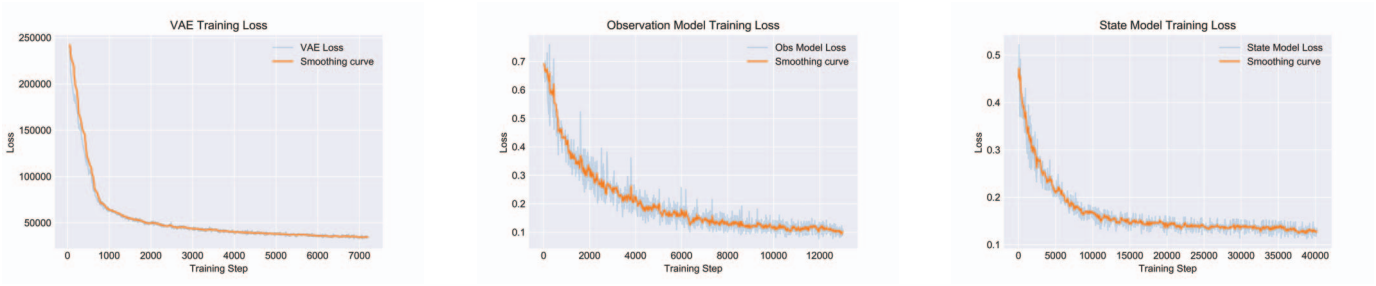
Policy network of DualM-Control is designed to take multiple inputs and output an action at each time step t. Let N_s, N_a be positive integer constants and represent the size of state and action of agents. The images from a camera mounted on agent has N_{oc} channels (Usually images from a RGB camera have three channels representing red, green and blue information whereas RGB-D cameras have one more depth channel) and length and width of the image are N_{ol}, N_{ow}. Besides, we denote the size of the LSTM hidden states in state model and observation model N_{hs} and N_{ho}. Let N_z denote the size of latent space of image after porcessed by decoder of VAE.

Size of image is $N_{ol} \times N_{ow} \times N_c$ and VAE takes it as input and output size of the encoder is N_z. On the LSTM represented state model, we input states and action, size of $N_s + N_a$, in every time step and ensure the output is as similar as possible to the state in next time step through the learning objective in (2). Similar to state model, observation model has an input size of $N_z + N_a$, i.e., it takes latent parameters of image and action as input.

Policy network can be very simple, implement of policy network in section IV is constructed by two fully connected layers with a nonlinear activation function after each layer. Inputs of the policy include states of the agent (size of N_s), latent parameters (size of N_z) of the image after VAE's decoder, hidden states of the state model (size of N_{hs}) and hidden states of the observation model (size of N_{ho}). To sum up, inputs of policy network are variables of size of $N_s + N_z + N_{hs} + N_{ho}$ and it outputs action of the agent, the size of N_a, as shown in Fig. 2.

With this policy, dualM-Control can achieve the intention of obtaining both spatial and temporal information, both states and observations with a simple MLP in a few thousand parameters, which greatly reduce the difficulty of convergence of policy network compared with tradition complex policy network structure.

To optimize the parameters of our policy, we use the Covariance-Matrix Evolution Strategy (CMA-ES) [22], [23] to train the policy network. CMA-ES improves the likelihood of successful update step size compared with traditional ES and is widely used in RLs. This algorithm evolves the parameters of the policy network on multiple CPU cores with multiple rollouts of the experiments running in simulation.



(a) VAE Training Loss (b) Observation Model Training Loss (c) State Model Training Loss

Fig. 3: Network convergence curves of VAE, Observation model and State model.

C. Learn to control

Two models extract different information and serve as inputs of DualM-Control's policy, helping solving more complex tasks. But network structure of VAE and two LSTM models are not simple, if we train our policy with VAE and LSTM models end-to-end, in other words, we train VAE, LSTM and policy at same time with state, image as input and action as output, the advantages of the simplified network structure we proposed will no longer exist. In fact, VAE and LSTM models are pre-trained before actual policy network is interacting with environment.

First, random actions are generated to interaction with the environment and collect the necessary data including states, observation, action. Although the action is generated randomly, we hope that the agent will experience as many states as possible and get as many kinds of observations as possible. Then, we divide the generated dataset into different subsets use them to pretrain VAE and LSTM represented models. Specifically, we use images to train VAE via unsupervised learning, then we take out the encoder of VAE separately and encode images into latent parameters. After that, observation model is trained with sequential actions and latent parameters of images via supervised learning. Similarly, state model is trained with pre-processed temporal sequential states and actions, as shown in Algorithm 1.

IV. Experiments

When optimizing parameters of policy network using CMA-ES, VAE and LSTM models will no longer change their parameters anymore. After policy network outputs an action and agent carrying it out in the environment, agent will get next step state and observation along with reward, then we process observation into latent parameters with decoder of VAE. Input latent parameters, state and action into observation model and state model and we will get hidden states of these two model, then through policy network we obtain action in next step. This process runs in circles until current episode ends. Using cumulative reward, i.e., return of episodes we can then evolve our policy.

The experiments to test DualM-Control algorithmIII are explained in this section. The test environment is a bipedal walker in gym [24]. By comparing with state of the art RL algorithms, we demonstrate that DualM-Control can achieve higher average return for each episode.

A. Experiment setup

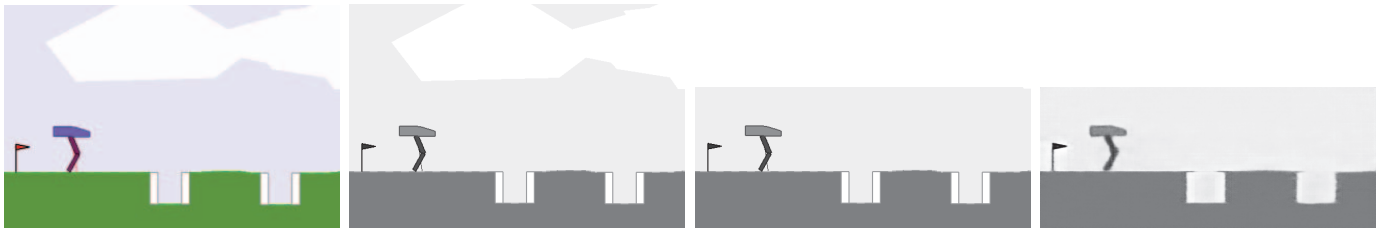
Gym is a popular toolkit developed by openAI for developing and comparing reinforcement learning algorithms. Among all the environments in gym, we choose to re-implement *BipedalWalkerHardcore* and modify the 16-line Lidar states that providing distance to terrains into a image observation from the third person perspective.

More specifically, in our customized Bipedal Walker environment, we provide image of size of $400 \times 600 \times 3$ to obtain terrain information. States of the agent comprise velocity of the walker, angle and angular velocity of the torso of the agent, angle and angular velocity of the four joints of the agent, and two indicators that show whether the two feet of the robot touch the ground or not, in total 14 different data. Actions of the agent is consist of motor torques of four leg joints, and we normalize it to interval $[0, 1]$.

Terrains in this environment include grass, upward stairs, down stairs and pits in the ground. We set the reward to the simplest way as that robot gets rewards for moving forward and if the robot falls, it gets -10 penalty. We expect the agent to walk as far as possible without falling down to the ground.

For VAE, we set the latent variables N_z to the size of 16, which means that preprocessed $250 \times 600 \times 1$ image input is reduced to 16 latent parameters. Size of hidden states of LSTM represented models are designed to be twice the size of states or observation latent parameters, that means, for example, dimension of input of our observation models are 20 (observation latent parameters and actions, $N_z + N_a$) and hidden state of this model is 28 ($2 \times N_z$).

We compare DualM-Control with some state-of-the-art reinforcement learning algorithms like Proximal Policy Optimization (PPO) [25] and Twin Delayed DDPG (TD3) [26], [27]. Policy network of these algorithms are set to be be consistent with our proposed algorithm to remove irrelevant disturbance factors. To be more specific, image first passes through layers of convolution networks, structure of this part are the same as encoder in VAE. And then we concatenate output of CNNs with state and input them into two layer of fully connected network which are identical with our policy network.



(a) Origin image (b) Grey image (c) Cut image (d) Restored image

Fig. 4: (a), (b) and (c): Original image will be turned into gray image first and then useless part of the gray image will be cut off. (d): Restored image from VAE. The restored image is basically the same with the original image.

Fig. 5: Observation model prediction result: Original observation sequence in first line is taken from a random episode, second line are images of VAE output with original observation images as input. The third lines are images of restored latent parameters from Observation Model prediction.

B. Pretrain results

From section III, we know that pretraining of both VAE and two models takes a very important part of our proposed DualM-Control and results of them are worth discussing. The convergence curves of these three networks are shown in Fig. 3.

1) VAE: Before input images into VAE, we first turn color images into gray images and then cut out the useful part below the images and save them. After this, a $400 \times 600 \times 3$ image will now have only $250 \times 600 \times 1$ pixels and we take them as inputs of VAE, as demonstrate in Fig. 4.

Three parts of VAE is encoder that is implemented by a series of convolution networks, sampler that sampling over a Gaussian distribution and decoder, a series of deconvolution networks. Encoder of VAE serves as a down-sampler over original image and in this experiment, we encode original $250 \times 600 \times 1$ image input to 16 latent parameters. And the result shows that 16 variable can restore most of the image details, as shown in Fig. 4.

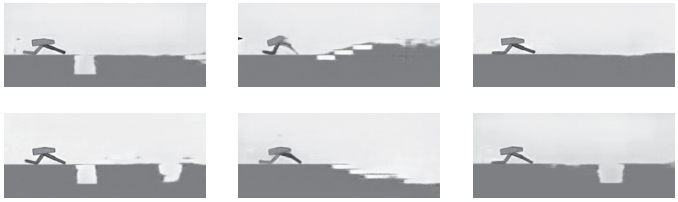


Fig. 6: Generating unseen terrains

In addition, as VAE has trained on a dataset that contain every kind of terrains that could be seen on this environment, it should have the ability to not only encode but also decode every terrain. Hence, we randomized a set of latent parameters and restore it with decoder of VAE, results in Fig. 6 show that random latent parameters generate some unseen terrain combinations but these terrains resemble real images from environment.

2) Predictive observation model: LSTM represented Models take sequential temporal information and compress them into low-dimension hidden states to help predict what the next observation will be. Hence, we visualize the output of our observation model and review if it estimate the future observation well. The good prediction will prove that the LSTM compress all useful historical information from another angle. We compare the result from our observation model with original image and image generated from VAE processed original image, which can be seen in Fig. 5. Results demonstrate that observation model predicts the terrain features well.

C. Results

In this subsection, we show how DualM-Control exceeds current RL algorithms in a bipedal walker task. Our evaluations are based on the average return (cumulative rewards within an episode), which stands for the length the robot move forward in our test environment.

In order to discuss the value of adding dual models to policy network, we design an extra policy network that takes states and image latent parameters as input and compare the average return of it with DualM-Control policy network

1) Control with latent parameters and self states: Training the robot to walk in complex terrains is not a easy task especially when we use observation images to represent terrains in environment. Typically, feature extraction of the terrain image needs complex network with multiple convolution layers and train policy with image feature extraction network via

TABLE I: Average episode return of DualM-Control algorithm compared with SOTA algorithms

Algorithm	1500 Episodes	After Convergence
Proximal Policy Optimization (PPO)	-4	-3
Twin Delayed DDPG (TD3)	-3	-3
Control with S and O	35	39
DualM-Control	25	48

reinforcement learning may stuck into local optimal. In our implement, as shown in TABLE I, Policy networks trained with PPO and TD3 algorithms failed to guide the robot to walk forward even after thousands of training episodes. However, policy with latent parameters and self states trained via CMA-ES enables the robot to move forwards but it still has some difficulties in stepping over some of the designed terrains in our environment.

2) DualM-Control: From the TABLE I, we find that the convergence speed of policy with just states and image latent parameters is slightly faster than our DualM-control algorithm. After 1500 training episodes, average return of it is 36 whereas policy of DualM-control is just 25. That is because the first policy is more simple with a relative small size of input. However, DualM-control exceeds all policies in average episode return. It have a higher possibility of stepping over rough terrains and thus it has a longer walking length compared with all other algorithms. The reason is that inputs of DualM-Control policy network further include temporal information from pretrained dual models compared with policy of control with just latent parameters and self states. Hence, we can conclude that our DualM-Control algorithms exceeds existing approaches in performance in our test environment.

V. Conclusion

We present DualM-Control, an algorithm that utilizes both spatial and temporal information, both self state and observation with a simple policy network. VAE compress images into latent parameters and LSTM models compress temporal information into hidden states, and all three networks are pretrained via unsupervised and supervised learning. Therefore, training of the policy network can be fast and easy to converge. DualM-Control outperforms previous methods in task completion and we test it in a robot control task in gym. Future researches can focus on further using the pretrained models to infer future states and observations, which may speed up the convergence time of the policy network.

References

[1] L. Chang and D. Y. Tsao, "The code for facial identity in the primate brain," *Cell*, vol. 169, no. 6, pp. 1013–1028, 2017.

[2] N. Nortmann, S. Rekauzke, S. Onat, P. König, and D. Jancke, "Primary visual cortex represents the difference between past and present," *Cerebral Cortex*, vol. 25, no. 6, pp. 1427–1440, 2015.

[3] D. Hafner, T. Lillicrap, J. Ba, and M. Norouzi, "Dream to control: Learning behaviors by latent imagination," *arXiv preprint arXiv:1912.01603*, 2019.

[4] D. Pathak, P. Agrawal, A. A. Efros, and T. Darrell, "Curiosity-driven exploration by self-supervised prediction," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2017, pp. 16–17.

[5] M. A. Lee, Y. Zhu, K. Srinivasan, P. Shah, S. Savarese, L. Fei-Fei, A. Garg, and J. Bohg, "Making sense of vision and touch: Self-supervised learning of multimodal representations for contact-rich tasks," in *2019 International Conference on Robotics and Automation (ICRA)*. IEEE, 2019, pp. 8943–8950.

[6] M. Hausknecht and P. Stone, "Deep recurrent q-learning for partially observable mdps," *arXiv preprint arXiv:1507.06527*, 2015.

[7] N. Haber, D. Mrowca, S. Wang, L. F. Fei-Fei, and D. L. Yamins, "Learning to play with intrinsically-motivated, self-aware agents," in *Advances in Neural Information Processing Systems*, 2018, pp. 8388–8399.

[8] J. Oh, V. Chockalingam, H. Lee *et al.*, "Control of memory, active perception, and action in minecraft," in *International Conference on Machine Learning*, 2016, pp. 2790–2799.

[9] I. Higgins, L. Matthey, A. Pal, C. Burgess, X. Glorot, M. Botvinick, S. Mohamed, and A. Lerchner, "beta-vae: Learning basic visual concepts with a constrained variational framework," 2016.

[10] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[11] D. A. Reynolds, "Gaussian mixture models." *Encyclopedia of biometrics*, vol. 741, 2009.

[12] S. Levine and P. Abbeel, "Learning neural network policies with guided policy search under unknown dynamics," in *Advances in Neural Information Processing Systems*, 2014, pp. 1071–1079.

[13] S. Levine and V. Koltun, "Guided policy search," in *International Conference on Machine Learning*, 2013, pp. 1–9.

[14] S. Levine, C. Finn, T. Darrell, and P. Abbeel, "End-to-end training of deep visuomotor policies," *The Journal of Machine Learning Research*, vol. 17, no. 1, pp. 1334–1373, 2016.

[15] A. Tamar, Y. Wu, G. Thomas, S. Levine, and P. Abbeel, "Value iteration networks," in *Advances in Neural Information Processing Systems*, 2016, pp. 2154–2162.

[16] S. Niu, S. Chen, H. Guo, C. Targonski, M. C. Smith, and J. Kovacevic, "Generalized value iteration networks: Life beyond lattices," in *32nd AAAI Conference on Artificial Intelligence, AAAI 2018*. AAAI press, 2018, pp. 6246–6253.

[17] M. Deisenroth and C. E. Rasmussen, "Pilco: A model-based and data-efficient approach to policy search," in *Proceedings of the 28th International Conference on machine learning (ICML-11)*, 2011, pp. 465–472.

[18] Y. Gal, R. McAllister, and C. E. Rasmussen, "Improving pilco with bayesian neural network dynamics models," in *Data-Efficient Machine Learning workshop, ICML*, vol. 4, 2016, p. 34.

[19] D. Ha and J. Schmidhuber, "Recurrent world models facilitate policy evolution," in *Advances in Neural Information Processing Systems*, 2018, pp. 2450–2462.

[20] Ł. Kaiser, M. Babaeizadeh, P. Miłos, B. Osiński, R. H. Campbell, K. Czechowski, D. Erhan, C. Finn, P. Kozakowski, S. Levine *et al.*, "Model based reinforcement learning for atari," in *International Conference on Learning Representations*, 2019.

[21] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," *stat*, vol. 1050, p. 1, 2014.

[22] N. Hansen, S. D. Müller, and P. Koumoutsakos, "Reducing the time complexity of the derandomized evolution strategy with covariance matrix adaptation (cma-es)," *Evolutionary computation*, vol. 11, no. 1, pp. 1–18, 2003.

[23] N. Hansen and A. Ostermeier, "Completely derandomized self-adaptation in evolution strategies," *Evolutionary computation*, vol. 9, no. 2, pp. 159–195, 2001.

[24] G. Brockman, V. Cheung, L. Pettersson, J. Schneider, J. Schulman, J. Tang, and W. Zaremba, "Openai gym," *arXiv preprint arXiv:1606.01540*, 2016.

[25] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," *arXiv preprint arXiv:1707.06347*, 2017.

[26] S. Fujimoto, H. Van Hoof, and D. Meger, "Addressing function approximation error in actor-critic methods," *arXiv preprint arXiv:1802.09477*, 2018.

[27] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra, "Continuous control with deep reinforcement learning," *arXiv preprint arXiv:1509.02971*, 2015.