Learning Predictive Models for Efficient Policy Learning

Huazhe Xu

Acknowledgement

First and foremost, I want to express my gratitude to my advisor Prof. Trevor Darrell. Trevor is such a nice and helpful advisor. I can still remember how I started my first project as a Ph.D. student with guidance from Trevor. He always gave me useful advice during meetings and encouraged me to explore methods in a systematic way. Trevor is not only a good academic advisor but also a reliable mentor and friend in my life --- when I met unfortunate events, he helped me go through a difficult time; when I was struggling to find a job, he helped and recommended me. Also, I appreciate the academic freedom he permits, which makes a lot of fun projects happen. I also want to thank my committee member Prof. Pieter Abbeel and Prof. Francesco Borrelli for the support and advice.

Learning Predictive Models for Efficient Policy Learning

by

Huazhe Xu

A dissertation submitted in partial satisfaction of the

requirements for the degree of

Doctor of Philosophy

in

Computer Science

in the

Graduate Division

of the

University of California, Berkeley

Spring 2021

Learning Predictive Models for Efficient Policy Learning

Abstract

Learning Predictive Models for Efficient Policy Learning

by

Huazhe Xu

Doctor of Philosophy in Computer Science

University of California, Berkeley

Professor Trevor Darrell, Chair

For an intelligent agent to interact with the environment efficiently, it must have the ability to predict, plan and generalize. This thesis studies how an intelligent agent can learn to predict future observations and leverage the predictive models for efficient policy learning and generalization. The four instances in this thesis are on high-fidelity video prediction, video prediction that handles multi-modal data distribution, predictive model-based reinforcement learning, and model-based zero-shot policy generalization. In the first case, we use a model that disentangles motion and appearance to predict high-fidelity images. We find this method can alleviate the blurry artifact and shape deformation inherited in previous methods. In the second case, we propose to use an example-guided model in the face of the multi-modal distribution of real-world data. The proposed method can predict diverse, multi-modal data that can also generalize well. In the third instance, we propose a model-based reinforcement learning method with theoretical guarantees. Specifically, we propose a novel value discrepancy loss for predictive model training. We experimentally also prove such framework and loss will significantly improve sample efficiency. Finally, we propose a method that learns both the dynamics model as well as the value of regions for zero-shot policy generalization. We show that this approach can generalize without finetuning to novel tasks. This thesis proposes several methods toward learning and using better predictive models to achieve policies efficiently.

To my parents, Lixin Song and Dongyu Xu

Contents

List of Figures

List of Tables

Acknowledgments

First and foremost, I want to express my gratitude to my advisor Prof. Trevor Darrell. Trevor is such a nice and helpful advisor. I can still remember how I started my first project as a Ph.D. student with guidance from Trevor. He always gave me useful advice during meetings and encouraged me to explore methods in a systematic way. Trevor is not only a good academic advisor but also a reliable mentor and friend in my life — when I met unfortunate events, he helped me go through a difficult time; when I was struggling to find a job, he helped and recommended me. Also, I appreciate the academic freedom he permits, which makes a lot of fun projects happen. I also want to thank my committee member Prof. Pieter Abbeel and Prof. Francesco Borrelli for the support and advice.

During my Ph.D., I am also fortunate enough to work with many mentors who give invaluable advice and help. I want to thank Prof. Tengyu Ma, who is the mentor during my internship at Facebook. Besides many insightful discussions and delicious dinners with Tengyu, I also learn to improve my own efficiency in research from him. I am also grateful to Dr. Yuandong Tian for the help and support during my internship. In my last year of Ph.D. study, I joined the BAIR-FAIR program, in which I met my mentor Dr. Roberto Calandra who also helped me a lot. Roberto is always friendly and helpful. I also want to thank Prof. Sergey Levine and Prof. Anca Dragan for the advice on various projects.

Besides mentors, I am also grateful to my collaborators and friends. I really miss the good old days doing research and going to conferences with Yang Gao. I also enjoyed doing research and gathering with my friends Yi Wu and Xiaolong Wang. I'd like to thank Ronghang Hu for a tremendous amount of help during my Ph.D. I want to thank my dear friend Haozhi Qi for our weekly tennis and my dissertation talk rehearsal. I'd like to thank Yu Sun for playing the piano together, Yubei Chen, Boyuan Chen, Medhini Narasimhan, Haoran Tang for past collaborations, Tianhao Zhang and Mengqiao Yu for being such nice roommates, Qianyi Xie, Yonghuan Ren, Qing Tian, Amy Liu, An Ju, Hongyu Zhang, Shouping Chen for their consistent support, Jingwei Xu, Hang Gao for the push. I also want to thank my friends Ruiyang Song, Ge Zhan, Yu Sun, Xunkai Zhang, Zizhe Xu, Shuhan Mao, Tong Chen, Zhiyu Chen, Xiaoyu Cheng, Xiaohui Zhao, Hao Wang, Jiangrui Liu, Heng Luo, Shizhong Dai for the joy that makes me go through my Ph.D. I also want to thank many labmates: Sayna Ebrahimi, Samaneh Azadi, Deepak Pathak, Evan Shelhamer, Parsa Mahmoudieh, Anna Rohrbach, Shizhan Zhu, Xin Wang, Seth Dong Huk Park, Dequan Wang, Olivia Watkins. To prevent this list from too long, I cannot list all the people that are important to me. But I want to thank you all for spending your time with me.

I would like to thank my parents Dongyu Xu and Lixin Song for the unconditional care and support throughout my life. I want to thank them for teaching me how to learn, to live, to love. I believe whatever I achieve now and in the future also belongs to them. Finally, I want to thank my girlfriend Jiaxin Zhao, who is the source of happiness and the soul mate for my life.

Chapter 1

Introduction

1.1 Motivation

Robots can help people and improve the efficiency of human society. For example, robots in the factory can free up people from dangerous and/or tedious pick and place tasks. Mobile robots can help with searching and rescuing people from catastrophes. While robots can benefit humans in many aspects, most of them are still lack the power to interact with complex real-world scenarios and react to their perception signals. One particular aspect is that humans can usually understand the consequences of their actions even without too many physical trials. This internal model of the world for humans is crucial for accomplishing complex tasks in a sample efficient manner. Hence, it is natural to expect that intelligent robots also learn to obtain a world model and plan on it.

Recent advances in artificial intelligence and deep reinforcement learning (RL) enable robots to learn policies from data and interactions with environments. With model-free RL algorithms, an intelligent agent can have performant policies by interacting with the environments for millions of samples. Hence, this introduces two core problems for using such learning-based robots: 1) the efficiency of policy learning; 2) the generalization power of the policy. More specifically, robots can hardly afford expensive real-world samples as opposed to their simulator and/or game counterparts. Under such constraints, it is more desirable to have robots that can learn efficiently and generalize well to other tasks.

In this thesis, we aim to enable efficient policy learning and advocate the paradigm that a robot learns to predict the future and plan based on the imaginary future, namely model-based approach. In comparison with the model-free method, model-based method usually has better sample efficiency and multi-task generalization with the cost of learning a high-fidelity predictive model. Since we live in an unstructured world with complex visual information, one ideal predictive model can predict in high-dimensional space *i.e.* image space. Beyond learning a good predictive model, how to leverage specific models for training an actionable policy is the following challenge we try to study. Moreover, we propose a new approach to learning generalizable models and corresponding policies for novel tasks.

1.2 Background

Early studies have investigated model-based approaches on robotics before the deep learning era, but they are usually either restricted in specific settings due to limited representation power or requiring a tremendous amount of prior knowledge about the robot and its environment. For example, previous work [31] proposes to use Gaussian process to model the dynamics. Previous work [91] also proposes to use time-varying linear models to do model-based control.

1.3 Thesis Goals and Contributions

This thesis presents a series of studies from learning predictive models in the visual domain to predictive model-based reinforcement learning. Specifically, we propose the following methods: a video predictive model that can disentangle propagation and generation, a video predictive model that can handle multi-modality in real-world data with the guidance of retrieved examples, a model-based reinforcement learning framework that leverages predictive models for locomotion control tasks, and a model-based planning method that can obtain transferable knowledge for zero-shot generalization.

The contributions of this thesis are as follows:

Chapter 2 considers the tasks of modeling the visual dynamics, namely video prediction. Successful prediction of the future in image space might lead to sample efficient policy learning. The task is defined as given past frames to predict future frames. One conventional way to learn such a model is to regress the future frames based on a per-pixel loss function. Although achieving some success, it usually has a blurry artifact in the generated frames. Another line of research uses optical flow to manipulate past frames for predicting future frames. However, shape deformation is not negligible in the predicted frames. We propose a straightforward yet effective method that first uses optical flow to warp the past frame and then uses a context encoder to inpaint the disoccluded areas. This method successfully alleviates the artifacts from previous methods and predicts sharp images without shape deformation. Empirical results on both synthetic and real datasets show that the proposed method is superior to strong baselines.

Chapter 3 continues the study on visual dynamics. In this chapter, we consider another important aspect of video prediction — multi-modality in real-world data. A useful predictive model should be able to predict multiple futures instead of a deterministic one. For example, if a robot needs to avoid an obstacle, it can go around the obstacle from either the left side or the right side. We propose to use retrieved examples from a database as guidance for estimating the multi-modal distribution. However, the new challenge becomes how we can fully utilize visually different samples. Hence, we incorporate a new optimization scheme in the motion feature space. We find that our model can significantly improve the performance in the face of multi-modality challenges. Moreover, this module can be added seamlessly to most existing stochastic predictive models. Since it is guided by example, it naturally generalizes to unseen domains without any finetuning. We believe this can be an important and useful step toward learning useful video predictive models.

Chapter 4 considers not only how to learn a predictive model but also how to design model learning loss function for reinforcement learning so that a robot can achieve good policy in a sample efficient manner. We introduce a novel algorithmic framework for designing and analyzing model-based RL with theoretical guarantees. Concretely, we propose a value discrepancy loss for learning the dynamics model with a neural network that minimizes the difference between values in the real world and the learned model. The instantiated algorithms under this framework can achieve high rewards while only one million or fewer samples are permitted on a range of continuous control benchmark tasks.

Chapter 5 steps toward generalization of model-based agents. We consider that to save samples for robots, a robot agent should not only learn policies in a sample efficient manner but also generalize to unseen environments. In this chapter, we put the robot in the setting where robots are tested in an unseen environment while experience from a related but different environment is provided. In this zero-shot generalization setting, we propose to learn both a visual dynamics model and an observation scoring function on contingency-aware observation to learn transferable knowledge from past experience. Along with the model predictive control (MPC) method, our method can outperform various baselines on both video games and continuous control tasks.

Finally, Chapter 6 is the conclusion and discussion of future directions.

Chapter 2

Disentangling Motion and Generation for Video Prediction

A dynamic scene has two types of elements: those that move fluidly and can be predicted from previous frames, and those which are disoccluded (exposed) and cannot be extrapolated. Prior approaches to video prediction typically learn either to warp or to hallucinate future pixels, but not both. In this chapter, we describe a computational model for high-fidelity video prediction which disentangles motion-specific propagation from motion-agnostic generation. We introduce a confidence-aware warping operator which gates the output of pixel predictions from a flow predictor for non-occluded regions and from a context encoder for occluded regions. Moreover, in contrast to prior works where confidence is jointly learned with flow and appearance using a single network, we compute confidence after a warping step, and employ a separate network to inpaint exposed regions. Empirical results on both synthetic and real datasets show that our disentangling approach provides better occlusion maps and produces both sharper and more realistic predictions compared to strong baselines.

2.1 Background

Video prediction is a challenge due to the many varied factors that combine to generate future appearance.

State-of-the-art approaches to video prediction are often purely *pixel-based* and generate each pixel from scratch (Figure 2.2a) [101, 182, 176, 179, 35, 191, 20]. They rely on 3D or recurrent convolutional networks to encode spatial contents over time, hoping to capture implicit motion representations. Another active line of purely *motion-based* approaches seeks pixel correspondence for explicit motion propagation [134, 100]; in this approach standard networks are augmented with an optical-flow representation and a warping function (Figure 2.2b). Because all pixels are copy-pasted from the history image buffer, a degree of temporal consistency and spatial richness are automatically ensured. However, disocclusions – locations in the target frame to which extrapolated optical flow has no projection – cause

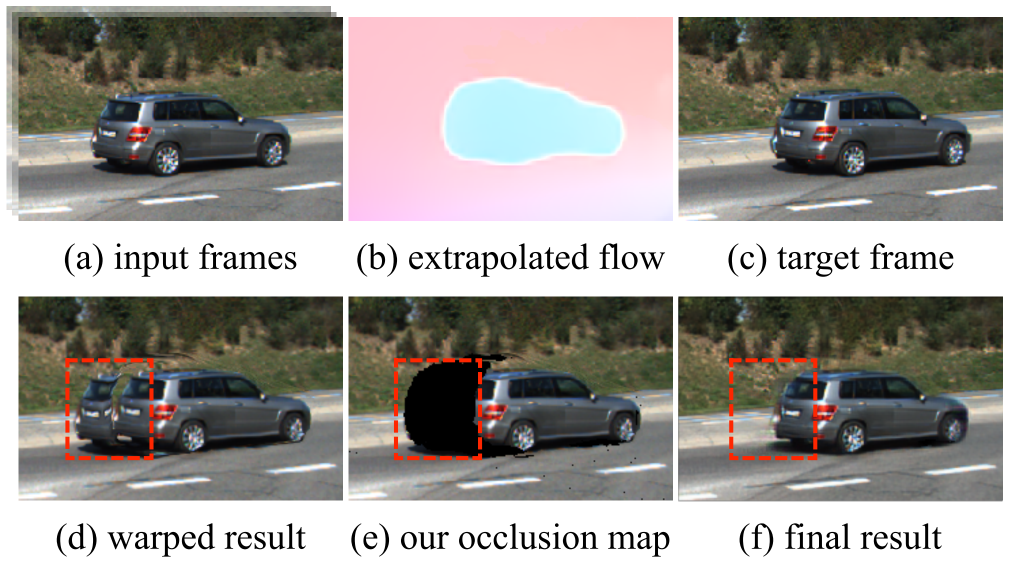


Figure 2.1: Video prediction by extrapolating the motion and hallucinating the pixels to be exposed. Given a video whose last frame shown in **(a)**, our model first extrapolates optic flow **(b)** to its near future target **(c)**. Direct motion propagation introduces ghosting effects **(d)** on pixels to be disoccluded (marked with the bounding box). Our disentangling approach computes a sparse occlusion map **(e)** and uses an contextual encoder to inpaint low-confidence patches, producing the final prediction **(f)**.

severe errors with this family of models (Figure 2.1e).

We observe that humans can easily predict future appearance (e.g., in a scene such as shown in Figure 2.1) by tracking an object's past motion and extrapolating future pixel values, as well as imagine the appearance of disoccluded pixels (the background) based on prior knowledge of similar scenes previously observed. We propose a compositional approach that disentangles spatial and temporal prediction, using a post-warp occlusion map to mediate between flow-based prediction and inpainting-based reconstruction of disoccluded regions.

Disentangling is different from the *linear fusion* approaches [48, 60] that use a single multi-task network to learn *pixel-* and *motion-based* predictions at the same time. To compose their results, the networks may also learn an extra occlusion map for weighted summation over each pixel (Figure 2.2c). Though those approaches are effective in many cases, it is unclear how much pixel and flow predictions can benefit each other. As our experiments indicate on the real dataset, these two tasks do not actually correlate significantly with each other (Figure 2.3).

We propose a compositional framework that explicitly disentangles motion-specific propagation and motion-agnostic generation into two submodules, and thus called DPG, for high-quality video prediction. We factorize flow and pixel predictions into a serialized pipelines. We first employ a flow predictor for extrapolating optical flows and then an occlu-

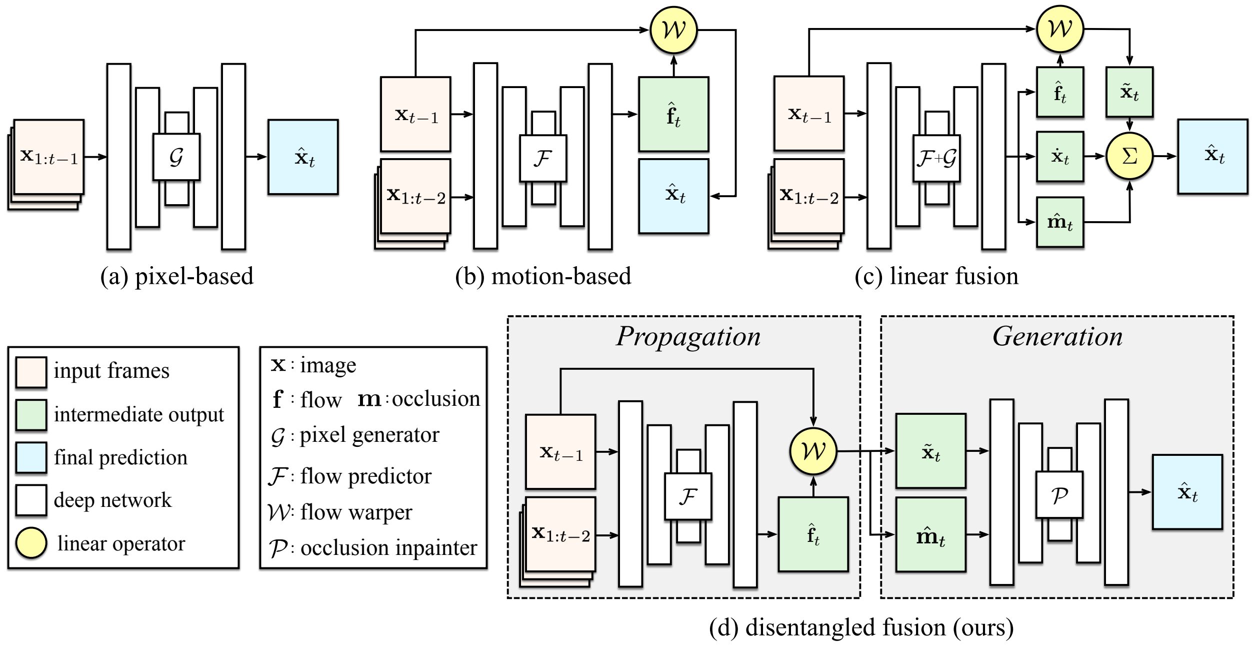


Figure 2.2: Overview of previous works and our proposed frameworks. Given $(t-1)$ frames and to predict one frame, previous video prediction methods typically consist of a flow predictor $\mathcal{F}$, or a pixel generator $\mathcal{G}$, or both: **(a)** *pixel-based* approaches; **(b)** *motion-based* approaches; **(c)** *linear-fusion* models; **(d)** ours.

sion inpainter for context encoding based on images warped by flows. After flow prediction and warping, we compute confidence directly based on predicted flow rather than via joint unsupervised learning. The downstream occlusion inpainter generates pixels on occluded areas based on confidence map.

We evaluate our approach on both standard CalTech Pedestrian dataset [38] and more challenging KITTI Flow dataset [110] with larger motions and occlusions. Our approach achieves new state-of-the-art performance on both datasets with large performance gain on the perceptual realism metrics. To quantitatively evaluate our computed occlusion confidence maps, we perform ablation study on the RoamingImages [68] dataset where occlusion ground-truth masks are available. Our model again compare favorably against previous baselines in occlusion prediction in terms of Intersection over Union (IoU) metrics.

In summary, our main contributions are in two aspects: a fusion pipeline that utilizes both optical flow and image synthesis for video prediction; a post-warp confidence-based occlusion map computation to disentangle pixel propagation and generation.

2.2 Related Work

Photo-realistic Image Synthesis

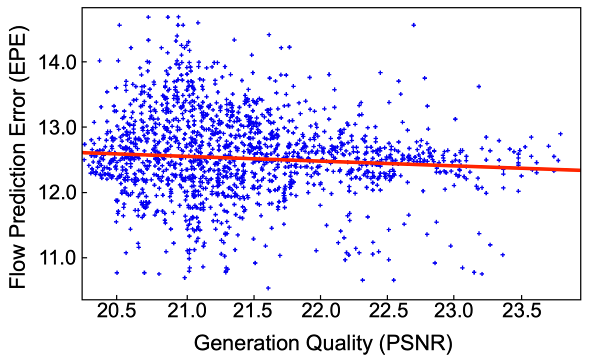
Realism is the constant pursuit of high-quality image synthesis [71, 22, 39, 190, 93, 195]. Recent developments towards photo-realistic image synthesis typically feature Generative Adversarial Networks (GANs) [54]. Conditioned on categorical labels [17], textual descriptions [145] or segmentations [187], high-fidelity images are shown to be able to synthesized. The closest work to ours are Dense Pose Transfer [120] and Transformation-grounded View Synthesis [128], which generally generate images by warping the original image with a learned appearance flow, conditioned on either given view transformation angle or a predefined dense pose, and then inpaint the ambiguous parts. In our case, however, the model needs to *predict* future motion and further synthesize based on both spatial and temporal information.

Spatial Context Encoding. Pixels are not isolated. On the contrary, there are many cases [13, 11, 204] where pixels are referred as context for their nearby neighbors. Spatial context encoders [130] query learned dataset priors with exposed appearance in search for missing patches. To our task, the "mask" is an occlusion map where motion predictions are erroneous. Particularly, we employ partial convolutions [99] in our occlusion inpainter's encoding blocks. Different from prior works on static images, our approach leverages not only the spatial but also temporal context.

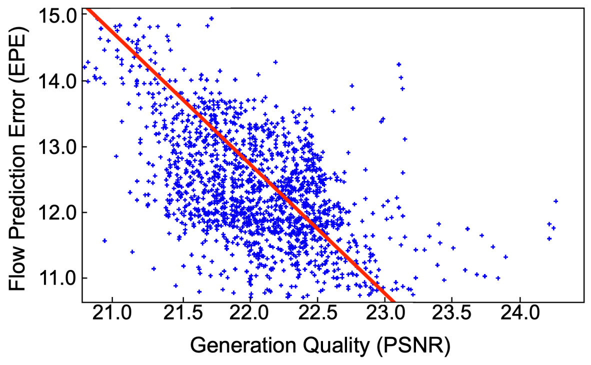
High-fidelity Video Prediction

Different from image synthesis, video prediction not only cares for per-frame visual quality but also cross-frame consistency. Recent approaches [35, 20, 101, 142, 147] are typically *pixel-based*, which generate each pixel from scratch using implicit motion representations. Such methods may suffer from blurry effects, especially in the presence of unseen novel scenes. On the contrary, *motion-based* methods [134, 100] excel in making sharp predictions, yet fail in occlusion areas where motion predictions are erroneous or ill-defined. Meanwhile, Reda *et al.* [144] propose to model moving appearances with both convolutional kernels as in [48] and vectors as optical flow. Our closest prior work is [60] which also composes the pixel- and flow-based predictions through occlusion maps. However, our proposed method can differentiate in three aspects: (1) our pixel and flow prediction tasks are separately trained; (2) we employ an occlusion inpainter for our pixel generation so that more contextual information after warping can be utilized; (3) instead of predicting occlusion as another side task, we directly refer to predicted flows as the proxy for post warping confidence.

Disentangling Motion and Content. Since videos are essentially moving contents whose semantics are agnostic to motions – one person's identity does not change while he's walking – it is natural to disentangle them apart. [176, 179, 35] are the three representative works in such direction. Though sharing the intuition to disentangle motion with other video



(a) linear fusion [60]



(b) disentangled fusion (ours)

Figure 2.3: The correlation between pixel and flow prediction tasks evaluated by different models. We show scatter plots about generation quality *vs.* flow prediction error on KITTI Flow [111] dataset. Pixel and flow prediction tasks do not actually correlate significantly with each other on previous models using multi-task learning. Our disentangled fusion pipeline shows that after factorizing these tasks apart, both of them could be better learned.

property, our approach builds on lower-level vision such as pixel correspondence. We take advantage of optical flow and their embedded occlusion clues to naturally factorize pixel and flow predictions into separate modules. Thanks to the ability of direct copy-pasting from previous frames, our model compares favorably against prior works in this venue, especially when it comes to high-res benchmarks.

2.3 Disentangled Video Prediction Model

Video prediction aims to synthesize future frames given a stack of history frames. For the ease of exposition, we here focus on a $(t-1)$-in 1-out prediction task, which could be later extended into multi-frame prediction by autoregression: given an input video sequence denoted as $\mathbf{x}_{1:t-1}$, the model aims to predict the frame $\mathbf{x}_t$ which should be accurate and visually sharp. As illustrated in Figure 2.2d, our approach is to factorize the per-pixel prediction task into pixel and flow prediction which are learned using two submodules – a flow predictor $\mathcal{F}$ and an occlusion inpainter $\mathcal{P}$.

Given the history frames $\mathbf{x}_{1:t-1}$, the flow predictor $\mathcal{F}$ learns to predict the flow field $\hat{\mathbf{f}}_t$ for the pixel correspondence between the last input frame $\mathbf{x}_{t-1}$ and the target $\mathbf{x}_t$. Utilizing $\hat{\mathbf{f}}_t$, we can propagating the input frame $\mathbf{x}_{t-1}$ into a motion-dependent prediction $\tilde{\mathbf{x}}_t$. Since it neglects the existence of occlusions, we additionally compute an occlusion map $\hat{\mathbf{m}}_t$ during the propagation. Based on $\tilde{\mathbf{x}}_t$ and $\hat{\mathbf{m}}_t$, the inpainting module $\mathcal{P}$ learns to inpaint occluded region from a learned prior within the training set.

Motion Propagation

Our module $\mathcal{F}$ computes the correlation of appearances between each pair of frames from the history inputs and predicts future motion dynamics using optical flow. We choose it over other motion representations, such as frame differences [64] or sparse trajectories [60] because it provides rich information about motion occlusions over pixels.

As illustrated in Figure 2.4a, the module $\mathcal{F}$ is an encoder-decoder network with skip connections. The output of $\mathcal{F}$ is a 2-channel flow field $\hat{\mathbf{f}}_t$ that aims to propagate the last frame $\mathbf{x}_{t-1}$ into the predicted target frame $\tilde{\mathbf{x}}_t$. Formally, let $\{(i, j)\} \in \mathbf{x}_t$ be a Cartesian grid over the target frame, and we have

$$\hat{\mathbf{f}}_t = \{(\Delta_i, \Delta_j)\} = \mathcal{F}(\mathbf{x}_{1:t-1}). \tag{2.3.1}$$

By assuming local linearity, we can sample the future frame from the last given frame as

$$\tilde{\mathbf{x}}_t = \mathcal{S}(\mathbf{x}_{t-1}; \hat{\mathbf{f}}_t), \tag{2.3.2}$$

where $\mathcal{S}(\cdot; \hat{\mathbf{f}}_t)$ is a bilinear sampler that generates the new image by first mapping the regular grid to the transformed grid and then bilinearly interpolating between produced sub-pixels.

For training the module $\mathcal{F}$, we adopt a masked pixel loss $\mathcal{L}_p$ and a smooth loss $\mathcal{L}_{smt}$ which is similar to previous works on unsupervised flow estimation [109, 202]

$$\begin{aligned}\mathcal{L}_p(\tilde{\mathbf{x}}_t, \mathbf{x}_t; \hat{\mathbf{m}}_t) &= \alpha\frac{1 - SSIM(\tilde{\mathbf{x}}_t \odot \hat{\mathbf{m}}_t, \mathbf{x}_t \odot \hat{\mathbf{m}}_t)}{2} \\ &\quad + (1-\alpha)\|\tilde{\mathbf{x}}_t \odot \hat{\mathbf{m}}_t - \mathbf{x}_t \odot \hat{\mathbf{m}}_t\|_1\end{aligned} \tag{2.3.3}$$

$$\mathcal{L}_{smt}(\hat{\mathbf{f}}_t, \mathbf{x}_t) = \sum_{i,j} |\nabla \hat{\mathbf{f}}_t(i,j)| \cdot (e^{-|\nabla \mathbf{x}_t(i,j)|})^T, \tag{2.3.4}$$

where $SSIM(\cdot, \cdot)$ denotes structural similarity index, $\odot$ denotes element-wise product, ∇ is a vector differential operator, T denotes the transpose of image gradient weighting, and α is our trade-off weight between the loss terms, which is fixed at 0.9 through cross-validations.

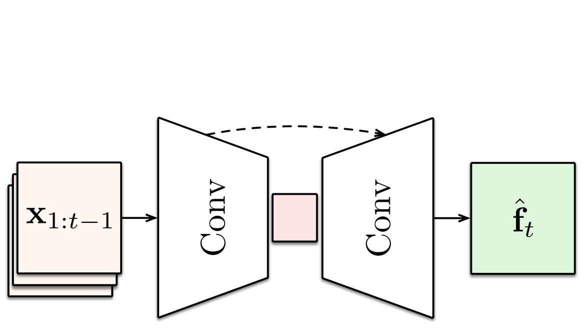
The training loss for module $\mathcal{F}$ is formulated as

$$\mathcal{L}_{\mathcal{F}} = \mathcal{L}_p(\tilde{\mathbf{x}}_t, \mathbf{x}_t; \hat{\mathbf{m}}_t) + \lambda_{smt}\mathcal{L}_{smt}(\hat{\mathbf{f}}_t, \mathbf{x}_t), \tag{2.3.5}$$

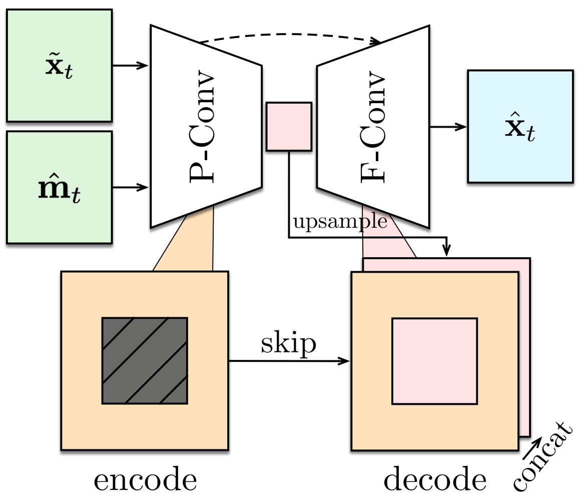
where λ_{smt} is a hyper-parameter that control the training schedule, we use $\lambda_{smt} = 0.1$ through a coarse grid search.

Occlusion Map Computation

However, $\mathcal{F}$ introduces "ghosting" effect in disoccluded regions where extrapolated optical flow has no projection (see Figure 2.1d and Figure 2.5 for detailed explanation). Intuitively, the "ghosting" area should be excluded from the propagated results by an occlusion map. It should be noted that similar ideas have been previously explored in [48, 60]; but in their



(a) Flow predictor $\mathcal{F}$.



(b) Occlusion inpainter $\mathcal{P}$.

Figure 2.4: Conceptual illustrations of our $\mathcal{F}$ and $\mathcal{P}$. **(a)** Our propagation module $\mathcal{F}$ is a standard encoder-decoder fully convolutional network with skip connections, which takes in history frames $\mathbf{x}_{1:t-1}$ and predict the backward flow $\hat{\mathbf{f}}_t$ from $\mathbf{x}_t$ to $\mathbf{x}_{t-1}$. **(b)** Taking in the computed occlusion map $\hat{\mathbf{m}}_t$ and the warped image $\tilde{\mathbf{x}}_t$, our inpainting module $\mathcal{P}$ inpaints for the final prediction $\hat{\mathbf{x}}_t$. It replaces standard convolutional blocks with partial convolutions (P-Conv) and fusion blocks (F-Conv) in its encoder and decoder, respectively.

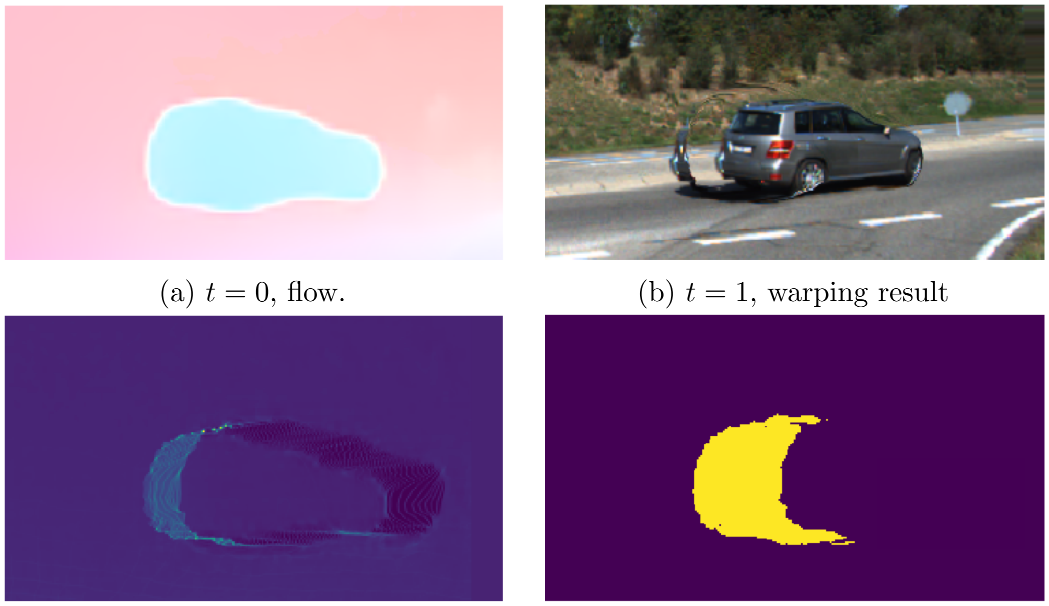
contexts, occlusion maps are learned as a side task to compose pixels from different sources. We argue that these regions can be directly computed based on backward flows, and hence masked out. We here describe the computation of occlusion map from an energy-based perspective. And we will later demonstrate the effectiveness of our computed map over the learned map in Section 2.4.

The pixel density can be viewed as an energy map: initially, it is uniformly distributed over pixel coordinates; later, the motion propagation changes its distribution and make the energy map become dense or sparse on different regions.

We initialize the energy field of the first frame to be a matrix filled with ones, denoted as $\mathbf{E}^1 = \mathbf{1}^{H\times W}$ for an image of $H \times W$ size.

Given a flow field, for each pixel in the first frame, the energy unit on each coordinate will be added into its 4 corresponded coordinates in the second frame bilinearly according to the flow field and we then get a new energy field $\mathbf{E}^2$. We consider two special cases for each coordinate (x, y) in the second frame:

1. If $\mathbf{E}^2_{x,y} = 0$, there is no pixel moving to this coordinate, which indicates it will be *disoccluded*;



(a) $t = 0$, flow. (b) $t = 1$, warping result

(c) $t = 1$, density map. (d) $t = 1$, occlusion map.

Figure 2.5: Ghosting effect caused by warping. Consider a foreground object on the background. **(a)** shows a predicted optical flow $\hat{\mathbf{f}}_1$ from target frame to current frame. **(b)** shows ghosting effects on the propagated frame $\tilde{\mathbf{x}}_1$ on locations to which flow has no projection. **(c)** shows our pixel density map computed by procedure described by Section 2.3. **(d)** shows our sparse occlusion map on which occlusion locations are colored as yellow.

2. If $\mathbf{E}^2_{x,y} > 2$, there are at least two pixels in the first frame compete for the same location, which suggests it will be *occluded*.

Given the definition of computation for occlusion and disocclusion, we can then get the occlusion map according to

$$\hat{\mathbf{m}}_{i,j} = \begin{cases} 1 & 0 < \mathbf{E}^2_{i,j} < 2, \\ 0 & \text{Otherwise} \end{cases} \tag{2.3.6}$$

Occlusion Inpainting as Context Encoding

Given the propagated frame $\tilde{\mathbf{x}}_t$ and the computed occlusion map $\hat{\mathbf{m}}_t$, we can now formulate our second modeling stage as context encoding to inpaint the missing pixels that are left out after propagation module. Our inpainting module $\mathcal{P}$ adopts generally the same network architecture as its propagation counterpart with partial convolution blocks [99]. As illustrated in Figure 2.4b, our encoder takes the previous propagated frame $\tilde{\mathbf{x}}_t$ and its occlusion map $\hat{\mathbf{m}}_t$ as inputs, producing a latent feature representation. The decoder then takes this feature representation and synthesizes the missing content.

Specifically in the encoder, the partial convolution operators [99] mask out invalid pixels and re-normalize features within clean receptive fields only. For the extreme cases where all pixels in the receptive field are masked, we will simply return a zero value as the result. One important design choice is that the receptive field of the bottleneck should be bigger than the maximal area of the occlusion masks so that the feature map can be free from mask in our bottleneck.

For the decoder, the kth layer feature maps are upsampled and linked with the kth layer features from encoder counting from the latent layer in reverse order by skip connections. However, this raises a fusion issue that the feature from the encoder is from an image with invalid pixels. Previously in [99], feature maps and masks are concatenated channel-wisely and handled by new partial convolutions in their decoder. We find this could be improved by directly refilling the occluded encoder features by the upsampled decoder features. This decoding fusion is repeated at each layer from the bottleneck up to our final output $\hat{\mathbf{x}}_t$.

To train our inpainting module $\mathcal{P}$, we design all of our losses to be temporal-independent so that the module can focus on the visual quality. In general, our loss terms consists of the following terms.

1. The pixel reconstruction loss

$$\mathcal{L}_{pix} = \mathcal{L}_p(\hat{\mathbf{x}}_t, \mathbf{x}_t; \hat{\mathbf{m}}_t) + \beta\mathcal{L}_p(\hat{\mathbf{x}}_t, \mathbf{x}_t; 1 - \hat{\mathbf{m}}_t),$$

   where $\mathcal{L}_p$ is defined in Equation 2.3.3.

2. The perceptual and style losses in VGG's n-dimensional latent space $\{\mathbf{\Psi}^n\}$ as in [158].

$$\begin{aligned}\mathcal{L}_{prc} &= \frac{1}{n}\sum_n \| [\psi(\tilde{\mathbf{x}}) - \psi(\mathbf{x})] \odot \hat{\mathbf{m}}_t\|_1 \\ &+ \beta\| [\psi(\tilde{\mathbf{x}}) - \psi(\mathbf{x})] \odot (1 - \hat{\mathbf{m}}_t)\|_1, \\ \mathcal{L}_{sty} &= \frac{1}{n}\sum_n \|(\psi(\tilde{\mathbf{x}}) - \psi(\mathbf{x}))^T(\psi(\tilde{\mathbf{x}}) - \psi(\mathbf{x})) \odot \hat{\mathbf{m}}_t\|_1 \\ &+ \beta\|(\psi(\tilde{\mathbf{x}}) - \psi(\mathbf{x}))^T(\psi(\tilde{\mathbf{x}}) - \psi(\mathbf{x})) \odot (1 - \hat{\mathbf{m}}_t)\|_1.\end{aligned}$$

3. The total-variance loss to encourage similar texture in occlusion boundaries

$$\mathcal{L}_{var} = \sum_{i,j} \sqrt{\|\mathbf{x}_{t(i,j+1)} - \mathbf{x}_{t(i,j)}\|_2^2 + \|\mathbf{x}_{t(i+1,j)} - \mathbf{x}_{t(i,j)}\|_2^2}.$$

4. The extra semantic loss to enforce layout consistency between segmentation masks, distilled by a pretrained segmentation network [146] $\mathbf{\Phi} : \mathcal{X} \to \mathcal{Y}$, since unconditional image inpainting tends to remove the foreground objects,

$$\begin{aligned}\mathcal{L}_{seg} &= CrossEnt(\mathbf{\Phi}(\hat{\mathbf{x}}_t) \odot \hat{\mathbf{m}}_t, \mathbf{y}_t \odot \hat{\mathbf{m}}_t) \\ &+ \beta CrossEnt\big(\mathbf{\Phi}(\hat{\mathbf{x}}_t) \odot (1 - \hat{\mathbf{m}}_t), \mathbf{y}_t \odot (1 - \hat{\mathbf{m}}_t)\big).\end{aligned}$$

The total loss for the inpainting module is

$$\begin{aligned}\mathcal{L}_{\mathcal{P}} =& \mathcal{L}_{pix} + \lambda_{prc}\mathcal{L}_{prc} + \lambda_{sty}\mathcal{L}_{sty} \\ &+ \lambda_{var}\mathcal{L}_{var} + \lambda_{seg}\mathcal{L}_{seg}.\end{aligned} \tag{2.3.7}$$

In the above losses, β is our attentive weight for masked regions and all λs are fixed hyper-parameter during training. We use a coarse grid search to set $\lambda_{prc} = 0.05, \lambda_{sty} = 120, \lambda_{var} = 0.1,$ and $\lambda_{seg} = 5$ and we find best experimental performance (see table 2.5) when $\beta = 10$.

Training

The overall training objective is formulated as

$$\min_{\mathcal{F},\mathcal{P}} \big(\mathcal{L}_{\mathcal{F}} + \mathcal{L}_{\mathcal{P}}\big). \tag{2.3.8}$$

Because flow estimation and prediction are hard to learn and sensitive to data biases, we first train our motion propagation module $\mathcal{F}$ and inpainting module $\mathcal{P}$ separately. After gradients become stable, we connect the two components together and fine-tune the whole network in an end-to-end fashion.

2.4 Experiments

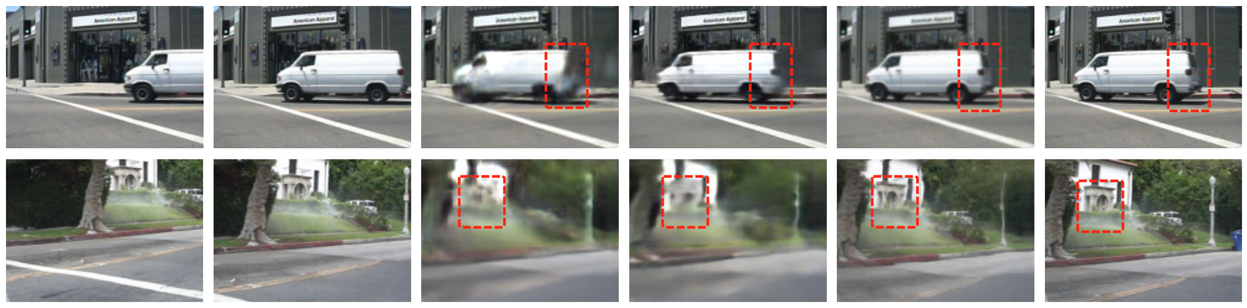
In this section, we will first introduce our evaluation metrics, benchmarks and baselines from previous works. Then we show our model's performance on next-frame and multi-frame prediction task, respectively. We final show ablations on occlusion map for better understanding of how disentangling propagation and generation helps in our framework.

Experiment Setup

Metrics. We adopt the existing Peak Signal-to-Noise Ratio (PSNR) and Structural Similarity Index Measure (SSIM) [190] metrics to measure the pixel/patch-wise accuracy. However, it is well-known that these metrics disagree with the human perception [191, 161, 74, 183] because they tend to encourage blurriness over naturalness. Therefore we also measure the realism of results by Learned Perceptual Image Patch Similarity (LPIPS[1]) proposed by Zhang et al. [205]. Higher PSNR/SSIM scores and lower LPIPS distances suggest better performance.

Dataset. We evaluate our results on the CalTech Pedestrian dataset [38] and the KITTI Flow dataset [110]. The previous setup on the CalTech Pedestrian dataset is to first train a model on the training split of KITTI Raw [111] proposed by Lotter et al. [101] and then

[1]We use LPIPS Version 0.1 available online.



(a) $\mathbf{x}_{t-5}$ (b) $\mathbf{x}_{t-2}$ (c) PredNet (d) ContextVP (e) DPG (Ours) (f) $\mathbf{x}_t$

Figure 2.6: Qualitative comparisons for Next-Frame Prediction on CalTech Pedestrian dataset. Given 10 past frames, all models are required to predict the future frame in the next step. Our DPG produces much sharper results compared to previous state-of-the-art methods on this dataset. Remarkably, it is also capable of inferring semantically sensible appearance for occluded regions. Results are best viewed in color with zoom.

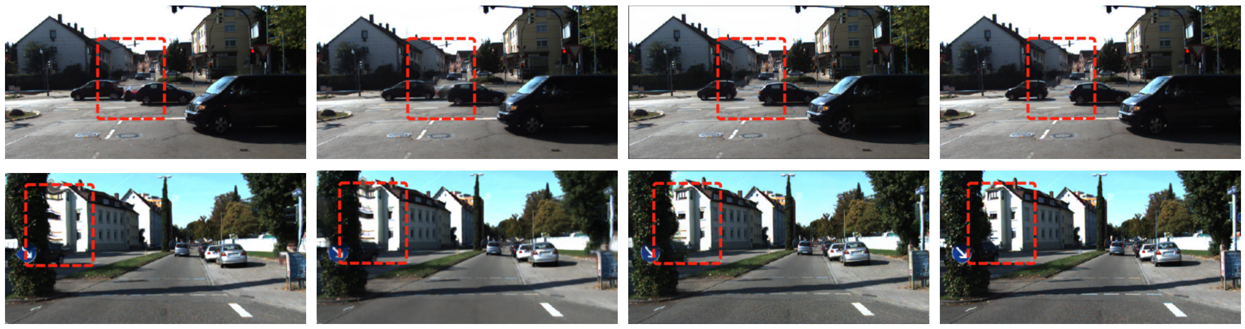
directly test it on the testing set of CalTech Pedestrian. Frames are center-cropped and down-sampled to 128×160 pixels. Every 11 consecutive frames are divided and sampled as a training clip in which the first 10 frames are fed into the model as the input, and the 11th frame is used as prediction target. As the results, the training, validation and testing sets consists of 3738, 14, and 1948 clips.

The KITTI Flow dataset is originally designed as a benchmark for optical-flow evaluation. For it is featured with higher resolution, larger motion and more occlusions, we should mention that it is more challenging compared to the raw KITTI dataset. It contains 3823 examples for training, 378 for validation and 4167 for testing. For all input images, we down-sampled and then center-cropped them into 320×640 pixels. We apply data augmentation techniques such as random cropping and random horizontal flipping for all the models. In addition, we sample video clips of 5 frames (4-in 1-out) from the dataset using a sliding window. This amounts to 3500 clips for training and 4000 clips for testing.

Baselines. We consider representative baselines from three model families for video prediction: (1) *pixel-based* methods, including Beyond MSE [107], PredNet [101], SVP-LP [35], MCNet [179], MoCoGAN [176], and ContextVP [20]; (2) *motion-based* method, including DVF [100]; (3) *linear fusion* methods, including Dual Motion GAN [96] and CtrlGen [60]. Note that we have not included [48] in our experiments since on its proposed RobotPush datasets, PredNet and SVP-LP are stronger baselines. For SVP-LP, we sampled 100 trajectories for each clip as in the original paper. The reported results are the mean performance over all the prediction results.

Method	**Family**	**PSNR**↑	**SSIM**↑	**LPIPS**↓ ($\times\mathbf{10^{-2}}$)
Repeat	–	23.3	0.779	5.11
BeyondMSE [107]	*P*	–	0.881	–
PredNet [101]	*P*	27.6	0.905	7.47
ContextVP [20]	*P*	**28.7**	0.921	6.03
DVF [100]	*M*	26.2	0.897	5.57
Dual Motion GAN [96]	*F*	–	0.899	–
CtrlGen [60]	*F*	26.5	0.900	6.38
DPG (Ours)	*F*	28.2	**0.923**	**5.04**

Table 2.1: Next-Frame Prediction results on CalTech Pedestrian. All models are trained on KITTI Raw dataset. The best results under each metric are marked in bold.



(a) DVF (b) CtrlGen (c) DPG (Ours) (d) $\mathbf{x}_t$

Figure 2.7: Qualitative comparisons for Next-Frame Prediction on the more challenging KITTI Flow dataset. All models are given 4 frames as input and required to predict the next frame. Frames include large motions, scene changes or occlusions. Our method is robust to these cases and consistent in the performance. More results please refer to our supplementary.

Next-Frame Prediction

We begin our evaluations on next-frame prediction. The goal of our task is simple and intuitive: Given the input frames, we need to output the next frame as accurate as possible.

Our result on CalTech Pedestrian dataset is shown on Table 2.1. We compare our model against previous state-of-the-art methods on this dataset. Our model achieves comparable PSNR and SSIM scores with ContextVP [20]. Meanwhile, our method can predict non-stretching textures in those occluded regions, which leads to smaller perceptual dissimilarity measured by LPIPS. As shown in Figure 2.6, our model is robust for both pixel propagation and novel scene inference.

Apart from empirical improvements, we find that, in terms of LPIPS metric, all the evaluated state-of-the-art methods do no better than the most naive baseline — repeating the last input frame as the prediction. This suggests that the CalTech Pedestrian dataset consists of small motions that are not obvious for human perception. This motivates us to

Method	**Family**	**PSNR↑**	**SSIM↑**	**LPIPS↓** ($\times\mathbf{10^{-2}}$)
Repeat	–	16.5	0.489	19.0
PredNet [101]	*P*	17.0	0.527	26.3
SVP-LP [35]	*P*	18.5	0.564	20.2
MCNet [179]	*P*	18.9	0.587	23.7
MoCoGAN [176]	*P*	19.2	0.572	18.6
DVF [100]	*M*	22.1	0.683	16.3
CtrlGen [60]	*F*	21.8	0.678	17.9
DPG(Ours)	*F*	**22.3**	**0.696**	**11.4**

Table 2.2: Next-Frame Prediction results on KITTI Flow. All models are trained to predict next frame given a history buffer of two frames. All evaluation results of the previous methods are obtained by their published codebases. The best results under each metric are marked in bold.

work on KITTI Flow dataset, which is more challenging so that learners can benefit from more inductive biases and thus be more robust.

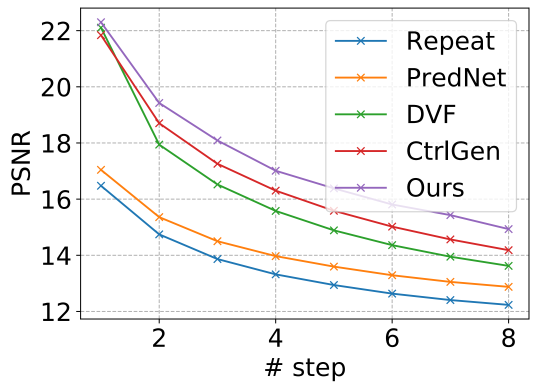
Table 2.2 shows our results on KITTI Flow dataset. As resolution increases, previous pixel-based methods (PredNet, SVP-LP) suffer from a steeper learning curve and more uncertainty in the visual space, resulting in the noticeable drop in their performance. Though achieving the better pixel/patch accuracy, they underperform the weakest repeating baseline in terms of the perceptual similarity. Our DPG achieves the best results in all metrics, especially LPIPS, showing around 30% improvement over the second best result from DVF.

As demonstrated in Figure 2.7, our proposed model again produce more visually appealing predictions than our baselines. In contrast to the pixel-based methods, all demonstrated methods suffer less from blurriness but display the distortion and stretch in shapes due to quick scene changes, which cause inaccurate flow prediction. Our model, instead, can predict better flow so as to alleviate undesirable artifacts in large motion areas. Occluded areas are masked by motion propagation and refilled by in-painting so that they are free of ghosting effects. Our occlusion in-painter learns a scene prior to hallucinate what is missing given the contextual information.

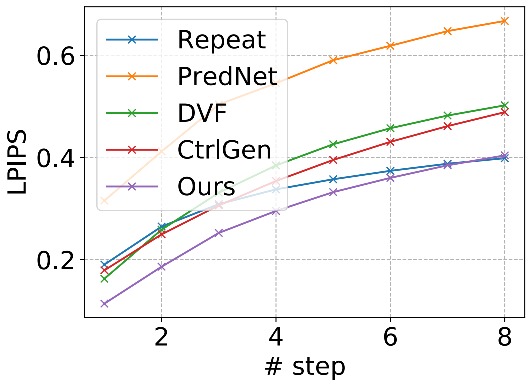
It is interesting to see that our model can achieve better PSNR/SSIM scores with training emphasized on realism terms (perceptual, style, and segmentation). We argue that this could serve as the evidence that our method could effectively learn dataset priors and flow prediction given the same amount of data.

Multi-Frame Prediction

We next move to the more challenging Multi-frame Prediction task. Comparing to Next-Frame Prediction, it requires our models to output a sequence of frames to match the ground-truth future frames. To do a fair comparison between our method and other baselines, we limit all input frames to be 4 frames and requires each model to predict 8 following frames.



(a) PSNR↑



(b) LPIPS↓

Figure 2.8: Quantitative results for Multi-Frame Prediction on KITTI Flow dataset. All the models take in 4 frames as input and recursively predict next 8 frames. Best viewed in color with zoom. For qualitative results and SSIM quantitative result, please refer to our supplementary materials.

Figure 2.8 compares the results on Multi-Frame Prediction of our models with various baselines on KITTI Flow dataset. Our method shows consistent performance gains on all metrics through time. DVF performs similarly to our model for short-term prediction measured by PSNR but quickly decays after 2 steps. This is because that their method is sensitive to propagated error since there are no remedy mechanisms. Our model, however, can mask out undesirable regions and in-paint new pixels instead.

Ablations on Occlusion Map

Does occlusion map really help? We design different baselines to verify our design choice for occlusion map. The first one is to directly warp images without the indicating occlusion at all as in [100] – this forms the result of our motion propagation module. Next, we build another baseline by applying an auto-encoder to directly refine the output of motion propagation module. The third one is to learn a independent learned map to fuse the result of motion propagation module and occlusion in-painter module which is similar to [60]. Our result is shown on Table 2.3. It is interesting that the performance will downgrade if we only refine the motion propagation without any guide. Also, Comparing to warped only and learned occlusion map results, our model shows better performance in all metrics.

Why does occlusion map help? One conjuncture for why our computed map can outperform the learned one is that our computed map is more accurate to indicate the occluded and disoccluded region. To support this argument, we conduct an ablation to investigate the relationship between the ground truth occlusion map and ours. Note that it is hard to get the ground truth occlusion map in the real world. Therefore, we use

Method	**PSNR↑**	**SSIM↑**	**LPIPS↓**
Warped only [100]	21.6	0.684	0.139
No occlusion map (auto-encoder)	21.3	0.674	0.120
Learned occlusion map [60]	21.8	0.679	0.179
Sparse occlusion map (Ours)	**22.3**	**0.696**	**0.114**

Table 2.3: Ablation study on different design choices of occlusion map computation.

Method	Learned occlusion map [60]	Sparse occlusion map (Ours)
IoU↑	0.0411	24.91

Table 2.4: Evaluation for learned occlusion map and our sparse occlusion map against ground-truth occlusion map.

Method	**PSNR↑**	**SSIM↑**	**LPIPS↓**
$\beta = 0$ (unmasked pixel only)	20.7	0.663	0.159
$\beta = 0.1$	20.9	0.671	0.134
$\beta = 1$ (same as unmasked pixel)	21.3	0.674	0.120
$\beta = \infty$ (masked pixel only)	18.4	0.514	0.206
$\beta = 10$ (Ours)	**22.3**	**0.696**	**0.114**

Table 2.5: Ablation study on different masking weight β.

RoamingImages [68] dataset for a quantitative study. It contains $80,000$ examples of random moving a random foreground image in front of a random background image. We compare the IoU between our occlusion map and the ground truth and the results are shown on Table 2.4. It shows that our method can localize occluded region significantly better than multi-task learning approach used in prior work [60].

How does occlusion map help? Given the effectiveness of the our computed map comparing to other design choice, we are also interested in how important the masking weight β is in our training losses for the occlusion inpainter. As shown on Table 2.5, masking weight is a crucial hyperparameter for good performance, as either unreasonably small or large value harms the training. We get the best $\beta = 10$ by searching on the validation set of KITTI Flow for balancing between the masked and the valid regions.

2.5 Discussion

In this work, we present a method for video prediction by disentangling motion-specific propagation and motion-agnostic generation. We propose a disentangled fusion pipeline which gates two tasks into two separate modules. We also directly evaluate pixel density map after warping for sparse occlusion maps. Experiments on synthetic and real datasets show our effectiveness against prior works.

Chapter 3

Video Prediction via Example Guidance

In video prediction tasks, one major challenge is to capture the multi-modal nature of future contents and dynamics. In this work, we propose a simple yet effective framework that can efficiently predict plausible future states. The key insight is that the potential distribution of a sequence could be approximated with analogous ones in a repertoire of training pool, namely, expert examples. By further incorporating a novel optimization scheme into the training procedure, plausible predictions can be sampled efficiently from distribution constructed from the retrieved examples. Meanwhile, our method could be seamlessly integrated with existing stochastic predictive models; significant enhancement is observed with comprehensive experiments in both quantitative and qualitative aspects. We also demonstrate the generalization ability to predict the motion of unseen class, i.e., without access to corresponding data during training phase.

3.1 Background

Video prediction involves accurately generating possible forthcoming frames in a pixel-wise manner given several preceding images as inputs. As a natural routine for understanding the dynamic pattern of real-world motion, it facilitates many promising downstream applications, e.g., robot control, automatous driving and model-based reinforcement learning [86, 119, 131].

Srivastava et al.[162] first proposes to predict simple digit motion with deep neural models. Video frames are synthesized in a deterministic manner [36], which also suffers to achieve long-range and high-quality prediction, even with large model capacity [49]. Babaeizadeh et al.[8] shows that the distribution of frames is a more important aspect that should be modelled. Variational based methods (e.g., SVG [35] and SAVP [89]) are naturally developed to achieve good performance on simple dynamics such as digit moving [162] and robot arm manipulation [49].

However, real-world motion commonly follows multi-modal distributions. With the

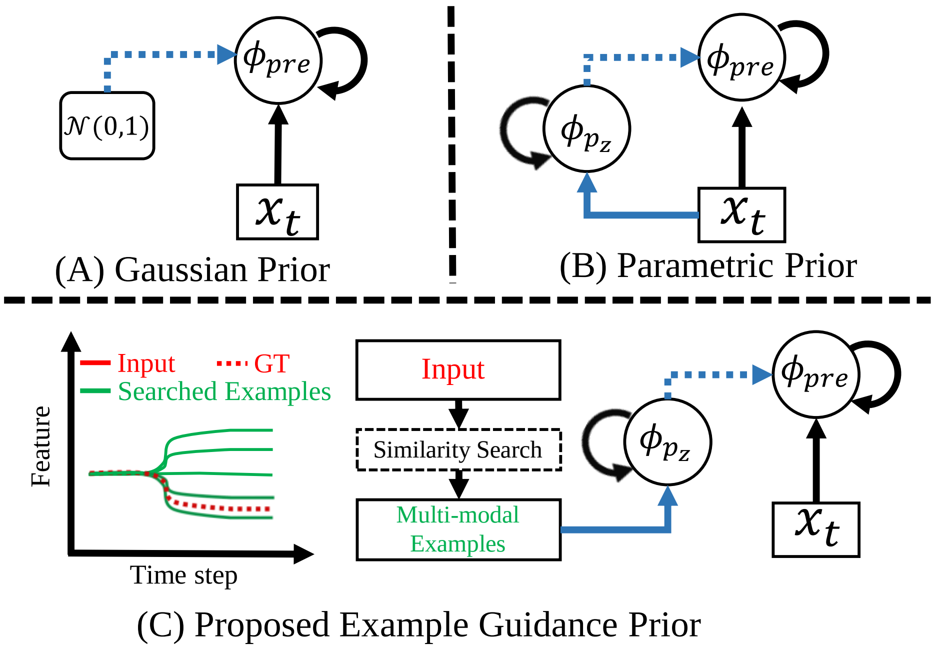


Figure 3.1: Illustration of stochastic prediction with different prior schemes. Rectangle box refers to input. ϕ_{q_z} is for uncertainty modelling and ϕ_{pre} is the prediction model. We omit the output part for simplicity. Blue line corresponds to stochastic modelling and dashed line is the sampling procedure of random variable. (A) Prediction with fixed Gaussian prior, which does not consider the temporal dependency between different time steps. (B) Prediction with parametric prior, which lacks explicit supervision signal for multi-modal future modelling. (C) Proposed prediction scheme with similar examples retrieved in training dataset. These examples are utilized construct an explicit multi-modal distribution target for the training of prediction model.

increase of motion diversity and complexity, variational inference with prior Gaussian distribution is insufficient to cover the wide spectrum of future possibilities. Meanwhile, downstream tasks mentioned in the first paragraph require prediction model with capability to model real-world distribution (i.e., can the multi-modal motion pattern be effectively captured?) and high sampling efficiency (i.e., fewer samples needed to achieve higher prediction accuracy). These are both important factors for stochastic prediction, which are also the focus issues in this chapter. Recent work introduces external information (e.g., object location [201, 181]) to ease the prediction procedure, which is hard to generalize to other scenes.

Predictive models can heavily rely on similarity between past experiences and the new ones, implying that sequences with similar motion might fall into the same modal with a high probability. The key insight of our work, deduced from the above observation, is that the potential distribution of sequence to be predicted can be approximated by analogous ones in a data pool, namely, examples.

In other words, our work (termed as **VPEG**, **V**ideo **P**rediction via **E**xample **G**uidance) bypasses implicit optimization of latent variable relying on variational inference; as shown in

Fig. 3.1C, we introduce an explicit distribution target constructed from analogous examples, which are empirically proved to be critical for distribution modelling. To guarantee output predictions are multi-modal distributed, we further propose a novel optimization scheme which considers the prediction task as a stochastic process for explicit motion distribution modelling. Meanwhile, we incorporate the adversarial training into proposed method to guarantee the plausibility of each predicted sample. It is also worth mentioning that our model is able to integrate with the majority of existing stochastic predictive models. Implementing our method is simply replacing variational method with the proposed optimization framework. We conduct extensive experiments on several widely used datasets, including moving digit [162], robot arm motion [49], and human activity [206]. Considerable enhancement is observed both in quantitative and qualitative aspects. Qualitatively, the high-level semantic structure, e.g., human skeleton topology, could be well preserved during prediction. Quantitatively, our model is able to produce realistic and accurate motion with fewer samples compared to previous methods. Moreover, our model demonstrates generalization ability to predict unseen motion class during testing procedure, which suggests the effectiveness of example guidance.

3.2 Related Work

Distribution Modelling with Stochastic Process. In this filed, one major direction is based on Gaussian process (denoted as GP) [143]. Wang et al.[185] proposes to extend basic GP model with dynamic formation, which demonstrates appealing ability of learning human motion diversity. Another promising branch is determinantal point process (denoted as DPP) [3, 45], which focuses on diversity of modelled distribution by incorporating a penalty term during optimization procedure. Recently, the combination of stochastic process and deep neural network, e.g., neural process [52] leads to a new routine towards applying stochastic process on large-scale data. Neural process [52] combines the best of both worlds between stochastic process (data-driven uncertainty modelling) and deep model (end-to-end training with large-scale data). Our work, which treads on a similar path, focuses on the distribution modelling of real-world motion sequences.

Video Prediction. Video prediction is initially considered as a deterministic task which requires a single output at a time [162]. Hence, many works focus on the architecture optimization of the predictive models. Conv-LSTM based model [155, 49, 189, 199, 102, 21, 188] is then proposed to enhance the spatial-temporal connection within latent feature space to pursue better visual quality. High fidelity prediction could be achieved by larger model and more computation sources [180]. Flow-based prediction model [85] is proposed to increase the interpretability of the predicted results. Disentangled representation learning [36, 51] is proposed to reduce the difficulty of human motion modelling [200] and prediction. Another branch of work [70] attempts to predict the motion with dynamic network, where the deep model is flexibly configured according to inputs, i.e., adaptive prediction. Deterministic model is infeasible to handle multiple possibilities. Stochastic video prediction is then

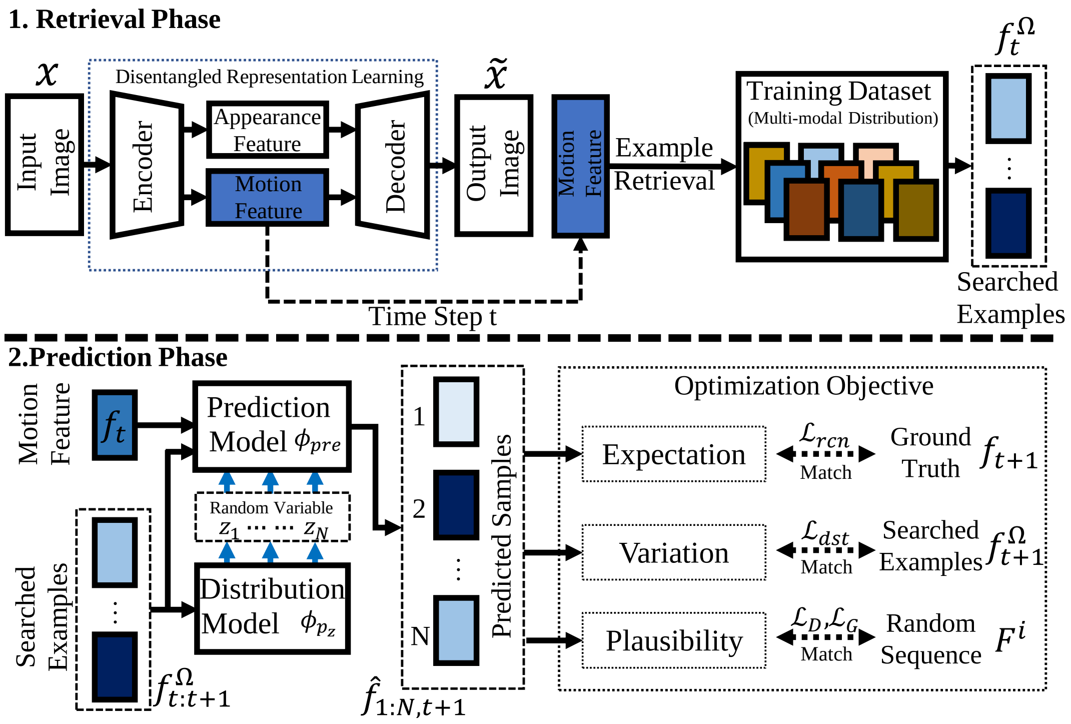


Figure 3.2: Overall framework of proposed video prediction method. The whole procedure is split into two consecutive phases presented at the top and bottom rows respectively. Top row refers to retrieval process of proposed method, while bottom rows is the prediction model with example guidance. It is optimized as a stochastic process to effectively capture the future motion uncertainty.

proposed to address this problem. SV2P [8] is firstly proposed as an stochastic prediction framework incorporated with latent variables and variational inference for distribution modelling. Following a similar inspiration, SAVP [89] demonstrates that the combination of GAN [55] and VAE [82] facilitates better modelling of the future possibilities and significantly boosts the generation quality of predicted frames. Denton & Fergus[35] proposes to model the unknown true distribution in a parametric and learnable manner, i.e., represented by a simple LSTM [63] network. Recently, unsupervised keypoint learning [80] (i.e., human pose [198]) is utilized to ease the modelling difficulty of future frames. Domain knowledge, which helps to reduce the motion ambiguity [201, 172, 103], is proved to be effective in future prediction. In contrast to these works above, we are motivated by one insight that prediction is based on similarity between the current situation and the past experiences. More specifically, we argue that the multi-modal distribution could be effectively approximated with analogous ones (i.e., examples) in training data and real-world motion could be further accurately predicted with high sampling efficiency.

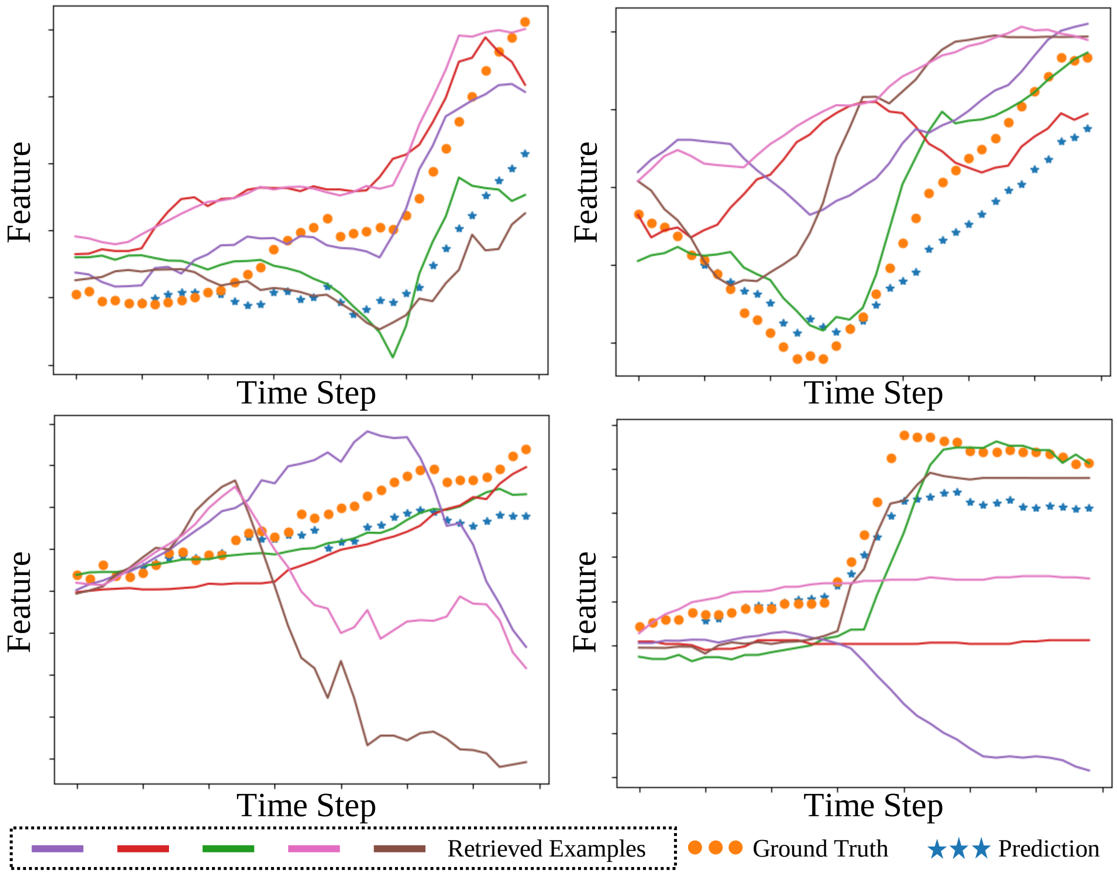


Figure 3.3: Four typical patterns of retrieved examples on PennAction [206] dataset. The five solid lines refer to top-5 examples searched in $\mathcal{D}_s$ and orange-dot line is the ground truth motion sequence. The blue-star line is predicted sequence. The input sequence generally falls into one variation pattern of retrieved examples, which confirms the key insight of our work.

3.3 Example Guided Video Prediction Model

Given M consecutive frames as inputs, we are to predict the future N frames in the pixel-wise manner. Suppose the input (context) frames $\mathbf{X}$ is of length M, i.e., $\mathbf{X} = \{\mathbf{x}_t\}_{t=1}^{M} \in \mathbb{R}^{W\times H\times C\times M}$, where W, H, C are image width, height and channel respectively. Following the notation defined, the prediction output $\mathbf{Y}$ is of length N, i.e., $\mathbf{Y} = \{\mathbf{y}_t\}_{t=1}^{N} \in \mathbb{R}^{W\times H\times C\times N}$. We denote the whole training set as $\mathcal{D}_s$. Fig. 3.2 demonstrates the overall framework of the proposed method. Details are presented in following subsections.

Example Retrieval via Disentangling Model

We conduct the retrieval procedure in training set $\mathcal{D}_s$. To avoid trivial solution, $\mathbf{X}$ is excluded from $\mathcal{D}_s$ if $\mathbf{X}$ is in the training set $\mathcal{D}_s$. Direct search in the image space is infeasible because it generally contains unnecessary information for retrieval, e.g., the appearance of foreground subject and detailed structure of background. Alternatively, a better solution is retrieving in disentangled latent space. Many previous methods [36, 175, 178, 35] have made promising progress in learning to disentangle latent feature. Two competitive methods, i.e., SVG [35] and [80] are adopted as the disentangling model in our work. [80] proposes an unsupervised method to extract keypoints of arbitrary object, whose pretrained model is directly used to extract the pose information as motion feature in our work. Note that the motion feature remains valid when input is only one frame, where the single state is treated as motion feature. SVG [35] unifies the disentangling model and variational inference based prediction into one stage. We remove the prediction part and train the disentangling model as:

$$(\mathbf{b}_t, \mathbf{h}_t) = \phi_{dse}(\mathbf{x}_t), t \in \{i, j\}, \tag{3.3.1}$$

$$\mathcal{L}_{dse} = ||\phi_{dec}(\mathbf{b}_i, \mathbf{h}_j) - \mathbf{x}_j||_2^2, \tag{3.3.2}$$

where i, j are two random time steps sampled from one sequence, ϕ_{dse} and ϕ_{dec} are disentangling model and decoder (for image reconstruction) respectively. $\mathcal{L}_{dse}$ indicates two frames from the same sequence share similar background and should be able to reconstruct each other by exchanging the motion feature. By optimizing this loss function, appearance feature $\mathbf{b}_*$ is expected to be constant while $\mathbf{h}_*$ contains the motion information, which leads the disentanglement model to learn to extract motion feature in a self-supervised way. Both disentanglement models SVG [35] and [80] could be presented in a unified way as shown in Fig. 3.2. Next we focus on the retrieval procedure. Note that all input frames are used in this part. We denote the feature used for retrieval as $\mathbf{F} \in \mathbb{R}^{C_f \times M} = \{\mathbf{f}_t\}_{t=1}^{M}$, where C_f is the number of feature dimension.

Given input sequence $\mathbf{X}$, whose motion feature denoted as $\mathbf{F}$, and training set $\mathcal{D}_s$, we conduct nearest-neighbor search as:

$$\Omega_i = \mathcal{S}(||\mathbf{F}^i - \mathbf{F}||_2^2, K), \tag{3.3.3}$$

where $\mathbf{F}^i$ refers to the extracted feature of $\mathbf{X}^i \in \mathcal{D}_s$. $\mathcal{S}(\bullet, K)$ refers to top K selection from a set in the ascending order. Ω_i is the retrieved index set corresponding to ith sample. Note that the subscript i is omitted for simplicity in following contexts. K is treated as a hyper-parameter in our experiments, whose influence is validated through ablation study in Sec. 3.4. We perform first-order difference along the temporal axis to focus on state difference during the retrieval procedure if multiple frames are available. We plot retrieved examples (solid line, $K = 5$) together with the input sequence (orange-dot line) in Fig. 3.3. Here we have two main observations: (1) The input sequence generally falls into one motion pattern of retrieved examples, which confirms the key insight of our work. (2) The examples have non-Gaussian distribution, which implies the difficulty on the optimization side by a variational inference method.

Discussion of Retrieval Efficiency. Note that the retrieval module is introduced in this work, which is an additional step compared to the majority of previous methods. One concern would thus be the retrieval time, which is highly correlated with efficiency of the whole model.

We would like to clarify that the retrieval step is highly efficient: (1) It is executed in the low dimensional feature space (i.e., $\mathbf{f} \in \mathbb{R}^{C_f}$) rather than in the image space, which requires less computation; (2) It is implemented with the efficient quick-sort algorithm. The averaged retrieval complexity is O(NlogN), where N is the number of video sequences. For example, on the PennAction dataset [206] (containing 1172 sequences in total) the whole running (including retrieval) time of predicting 32 frames is 354ms, while the retrieval time only takes 80ms.

Example Guided Multi-modal Prediction

Stochastic Video Prediction Revisited. The majority works [36, 201, 35, 89] of stochastic video prediction are based on variational inference. We first briefly review previous works in this field and then analyze the inferiority of stochastic prediction based on variational inference.

These methods use a latent variable (denoted as $\mathbf{z}$) to model the future uncertainty. The distribution of $\mathbf{z}$ (denoted as p_z) is trained to match with a (possibly fixed) prior distribution (denoted as q_z) as follows,

$$\mathcal{L} = ||\phi_{pre}(\mathbf{F}_{1:t-1}, \mathbf{z}_t) - \mathbf{f}_t||_2^2 + \mathcal{L}_{KL}(p_z||q_z), \tag{3.3.4}$$

where ϕ_{pre}, q_z, $\mathcal{L}_{KL}()$ are the prediction model, target distribution and Kullback-Leibler divergence function [84] respectively. p_z is generally modelled with deep neural network (e.g., $\phi_{p_z}(\mathbf{f}_{t-1})$). q_z is fixed, e.g., $\mathcal{N}(\mathbf{0}, \mathbf{I})$. This implies that the predicted image $\mathbf{X}_t$ is controlled by $\mathcal{N}(\mathbf{0}, \mathbf{I})$, not real-world motion distribution. [35] proposes to model the potential distribution with $\phi_{q_z}(\mathbf{f}^t)$, which still lacks an explicit supervision signal on the distribution of motion feature.

The essence of the modelling difficulty is from the optimization target of the $\mathcal{L}_{KL}$ term. Under the framework of variational inference, the form of q_z is generally restricted to a normal distribution for tractability. However, this is in conflict with the multi-modal distribution nature of real-world motion. We need a more explicit and reliable target and thus propose to construct it with similar examples $\mathbf{f}^{\Omega}$ whose retrieval procedure is described in Sec. 3.3.

Prediction with Examples. Given retrieved examples $\mathbf{f}^{\Omega}$, we first construct a new distribution target and then learn to approximate it. The most straightforward way is directly replacing the prior distribution q_z with the new one. More specifically, at time step t the distribution model ϕ_{p_z} is trained as:

$$\hat{\mu}_t, \hat{\sigma}_t = \phi_{p_z}(\hat{\mathbf{F}}_{1:t-1}), \mathbf{z}_t \sim \mathcal{N}(\hat{\mu}_t, \hat{\sigma}_t), \tag{3.3.5}$$

$$\mu_t, \sigma_t = \phi_{q_z}(\mathbf{f}_t^{\Omega}), \mathcal{L}_{KL} = \log(\frac{\sigma_t}{\hat{\sigma}_t}) + \frac{\hat{\sigma}_t + (\mu_t - \hat{\mu}_t)^2}{2\sigma_t}, \tag{3.3.6}$$

where $\mathbf{z}_t$ models the possibility of future state and μ_t, σ_t are commonly supervised with $\mathcal{L}_{KL}$.

Algorithm 1 Example Guided Video Prediction

```
Input: Training Set $\mathcal{D}_s$, disentangling model $\phi_{dse}$, predictor $\phi_{pre}$ and discriminator $\phi_{dcm}$.
#Example retrieval phase
for Input sequence $\mathbf{X}$ in $\mathcal{D}_s$ do
    Get motion feature $\mathbf{F} = \phi_{dse}(\mathbf{X})$,
    Example retrieval to obtain $\mathbf{F}^{\Omega}$ as Eqn. 3.3.3.
end for
#Prediction phase
repeat
    Get a random batch of $(\mathbf{F}, \mathbf{F}^{\Omega})$ pairs.
    #Optimization as a stochastic process
    for $i = 1$ to $N$ do
        Sample noise $\mathbf{z}_{i,t+1}$ as Eqn. 3.3.8,
        Predict next state $\hat{\mathbf{f}}_{i,t+1}$ as Eqn. 3.3.9,
    end for
    Optimize w.r.t. Eqn. 3.3.10, 3.3.11, 3.3.12 and 3.3.13.
until the training objective $\mathcal{L}_{fin}$ (Eqn. 3.3.14) converged.
```

However, it is difficult to obtain promising results with the above method which simply replaces $\mathbf{f}_t$ with $\mathbf{f}_t^{\Omega}$. The reason mainly lies in two aspects: Firstly, the diversity of predicted motion feature at time step t (denoted as $\hat{\mathbf{f}}_t$) lacks an explicit supervision signal. Secondly, the distribution of latent variable $\mathbf{z}_t$ (i.e., $\mathcal{N}(\mu_t, \sigma_t)$) is infeasible to accurately represent the motion diversity of $\mathbf{f}_t^{\Omega}$, because no dedicated training objective is designed for this target.

Optimization as Stochastic Process. Motivated by the above two issues, we consider the prediction task as a stochastic process targeting at explicit distribution modelling. The whole prediction procedure is conducted in motion feature space. The inputs of prediction model ϕ_{pre} include $\mathbf{f}_t^{\Omega}$ and $\mathbf{f}_t$. We calculate the mean and variance of example feature $\mathbf{f}_t^{\Omega}$, i.e., $\mathcal{E}(\mathbf{f}_t^{\Omega})$ and $\mathcal{V}(\mathbf{f}_t)$, for the subsequent random sampling in motion space. The prediction procedure at time step t is conducted as follows,

$$(\mu_t, \sigma_t) = \phi_{q_z}(\mathcal{E}(\mathbf{f}_t^{\Omega}), \mathcal{V}(\mathbf{f}_t^{\Omega})), \tag{3.3.7}$$

$$(\mathbf{z}_{1,t}, ..., \mathbf{z}_{N,t}) \overset{\text{i.i.d.}}{\sim} \mathcal{N}(\mu_t, \sigma_t), \tag{3.3.8}$$

$$\hat{\mathbf{f}}_{i,t+1} = \phi_{pre}(\hat{\mathbf{f}}_{i,t}, \mathbf{z}_{i,t}, \mathbf{f}_t^{\Omega}), i = 1, ..., N, \tag{3.3.9}$$

where $(\mathbf{z}_{1,t}, ..., \mathbf{z}_{N,t})$ is a group of independent and identically sampled values and $\mathbf{z}_{i,t} \in \mathbb{R}^h$. The subscript i, t refers to ith sample at time step t. Predicted state $\hat{\mathbf{f}}_{i,t}$ is not fed into ϕ_{q_z}, where we empirically get sub-optimal results. Because at initial training stage $\hat{\mathbf{f}}_{i,t}$ is noisy and non-informative, which in turn acts as a distractor for training ϕ_{q_z}. The prediction model is trained as follows,

$$\mathcal{L}_{rcn} = ||\hat{\mathbf{f}}_{j,t+1} - \mathbf{f}_{t+1}||_2^2, \tag{3.3.10}$$

$$\mathcal{L}_{dst} = ||\mathcal{V}(\{\hat{\mathbf{f}}_{i,t+1}\}_{i=1}^N) - \mathcal{V}(\mathbf{f}_{t+1}^{\Omega})||_2^2, \tag{3.3.11}$$

where $j = \min_i ||\{\hat{\mathbf{f}}_{i,t+1}\}_{i=1}^N - \mathbf{f}_{t+1}||_2^2$. $\mathcal{L}_{rcn}$ indicates that the best matched one is used for training [197]. Empirically, it is proved to be useful for stabilizing prediction when having multiple outputs.

$\mathcal{L}_{dst}$ aims to restrict the variety of N predicted features to match with $\mathbf{f}_{t+1}^{\Omega}$. In this way, the motion information of examples is effectively utilized and the distribution of predicted sequences is explicitly supervised. Meanwhile, to guarantee the plausibility of each predicted sequence, we incorporate the adversarial training into our method. More specifically, a motion discriminator ϕ_{dcm} is utilized to facilitate realistic prediction.

$$\mathcal{L}_D = \frac{1}{2}(\phi_{dcm}(1 - \mathbf{F}) + \phi_{dcm}(1 + \hat{\mathbf{F}}_i)), \tag{3.3.12}$$

$$\mathcal{L}_G = -\phi_{dcm}(1 - \hat{\mathbf{F}}_i), \tag{3.3.13}$$

where $i \in [1, N]$, $\mathcal{L}_D, \mathcal{L}_G$ are adversarial losses for ϕ_{dcm} and ϕ_{pre} [55]. Adversarial training effectively guarantees the predicted sequence not drifting far away from the real-wold motion examples. For clarity we present the whole prediction procedure in Alg. 1.

Improvement upon existing models. Our work mainly focuses on multi-modal distribution modelling and sampling efficiency, which is adaptive to multiple neural models. In Sec. 3.4, we demonstrate extensive results by combining proposed framework with two baselines, i.e., SVG [35] and [80]. The final objective is shown below:

$$\mathcal{L}_{fin} = \lambda_1 \mathcal{L}_{rcn} + \lambda_2 \mathcal{L}_{dst} + \lambda_3 \mathcal{L}_D + \lambda_4 \mathcal{L}_G. \tag{3.3.14}$$

3.4 Experiments

Datasets and Evaluation Metrics

We evaluate our model with three widely used video prediction datasets: (1) MovingMnist [162], (2) Bair RobotPush [44] and (3) PennAction [206]. Following the evaluation practice of SVG [8] and [80], we calculate the per-step prediction accuracy in terms of PSNR and SSIM. The overall prediction quality of video frames is evaluated with Fréchet Video Distance (FVD) [177]. To ensure fair evaluation, we compare with models whose source code is publicly available. Specifically, on MovingMnist [162] dataset we compare with SVG [35] and DFN [155]; On RobotPush [44] dataset SVG [35], SV2P [8] and CDNA [49] are treated as baselines; On PennAction [206] dataset the works of [80, 95, 192, 181] are used for comparison. Note that to follow the best practice of the baseline model [80], the

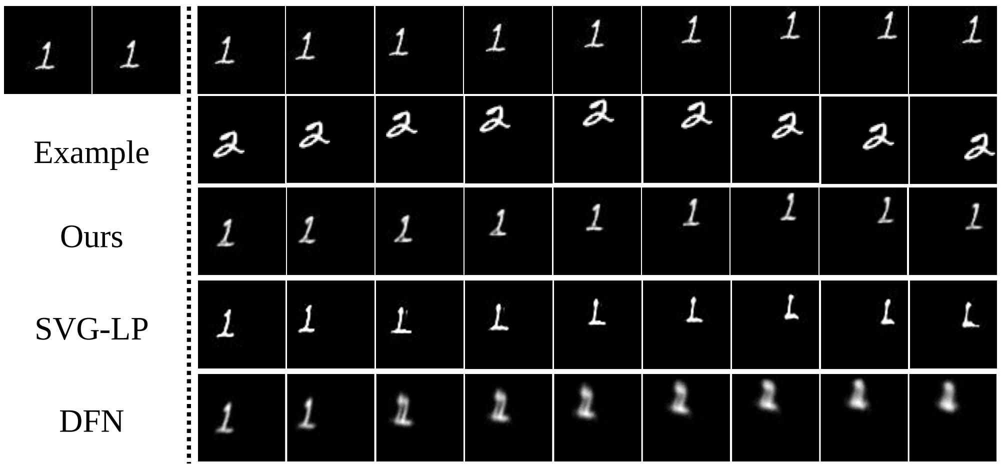


Figure 3.4: Visualization of prediction results on MovingMnist [162] dataset under stochastic setting. First row refers to ground truth. Following three rows correspond to example, predicted sequences of proposed model, SVG [192] and DFN [155] respectively.

Mode	**Model**	T=1	T=3	T=5	T=7	T=9	T=11	T=13	T=15	T=17
D	DFN	25.3	23.8	22.9	22.0	21.2	20.1	19.5	19.1	18.9
	SVG-LP	24.7	22.8	21.3	19.5	18.8	18.2	17.9	17.7	17.4
	Ours	25.6	23.2	22.5	21.7	20.8	20.3	19.8	19.5	19.3
S	DFN	25.1	22.1	18.9	16.5	16.2	15.7	15.2	14.9	14.3
	SVG-LP	25.4	23.9	22.9	19.5	19.0	18.7	18.7	18.2	17.6
	Ours	26.0	24.8	23.1	22.1	21.0	20.5	19.7	19.5	19.2

Table 3.1: Prediction accuracy on MovinMnist dataset [162] in terms of PSNR. Mode refers to experiment setting, i.e., stochastic (S) or deterministic (D). We compare our model with SVG-LP [35] and DFN [70].

prediction procedure on the PennAction Dataset [206] is a implementation-wise variant of Eqn. 3.3.7-3.3.9. More specifically, the random noise is sampled only at the first time stamp. Please refer to prediction procedure of the baseline model [80] for more details. In all experiments we empirically set $N = K = 5$.

Motivating Experiments: Moving Digit Prediction

For MovingMnist [162] dataset, inputs/outputs are of length 5 and 10 respectively during training. Note that this dataset is configured with two different settings, i.e., to be deterministic or stochastic. The deterministic version implies that the motion is determined by initial direction and velocity, while for the stochastic one, a new direction and velocity are applied after the digit hitting the boundary. The prediction model should be able to accurately estimate motion patterns under both settings.

Deterministic Motion Prediction. Tab. 3.1 shows prediction accuracy (in terms of PSNR)

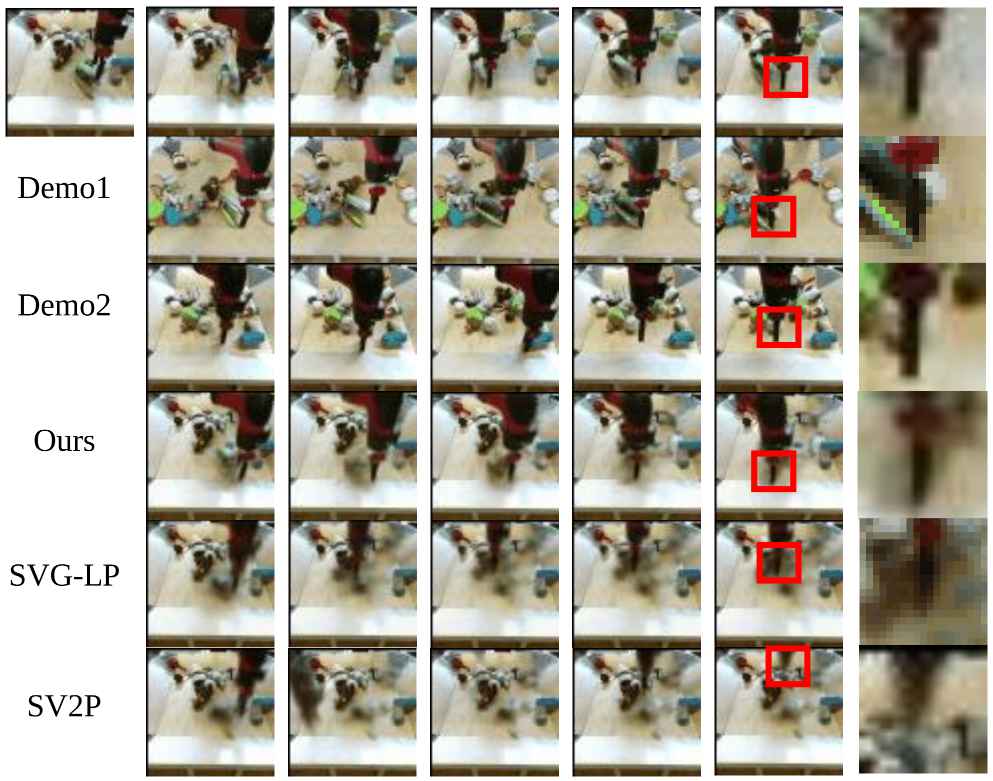


Figure 3.5: Comparison of the predicted sequences on RobotPush [44] dataset. Rows from top to bottom: ground truth, two retrieved examples, predicted results of our model, SVG [35] and SV2P [8].

from T=1 to T=17. One can observe that our model outperforms SVG-LP [35] by a large margin and is comparable to DFN [70]. Under deterministic setting the retrieved examples provide exact motion information to facilitate prediction procedure.

Stochastic Motion Prediction. Under stochastic setting, the best PSNR value of 20 random samples is reported (bottom three rows of Tab. 3.1). Considerable improvement over SVG-LP [35] could be observed from Tab. 3.1. Despite the retrieved example sequence not perfectly matching with ground truth (Fig. 3.4, first two columns refer to input.), informative motion pattern is provided, i.e., bouncing back after reaching the boundary. The deterministic model (DFN [70]), which only produces a single output, is infeasible to properly handle stochastic motion. For example, the blur effect (last row in Fig. 3.4) is observed after hitting the boundary.

Deterministic and stochastic datasets possess different motion patterns and distributions. Non-stochastic method (e.g., DFN [70]) is insufficient to capture motion uncertainty, while SVG-LP [35], empirically restricted by the stochastic prior nature in variational inference, is not capable of accurately predicting the trajectory under the deterministic condition. Under

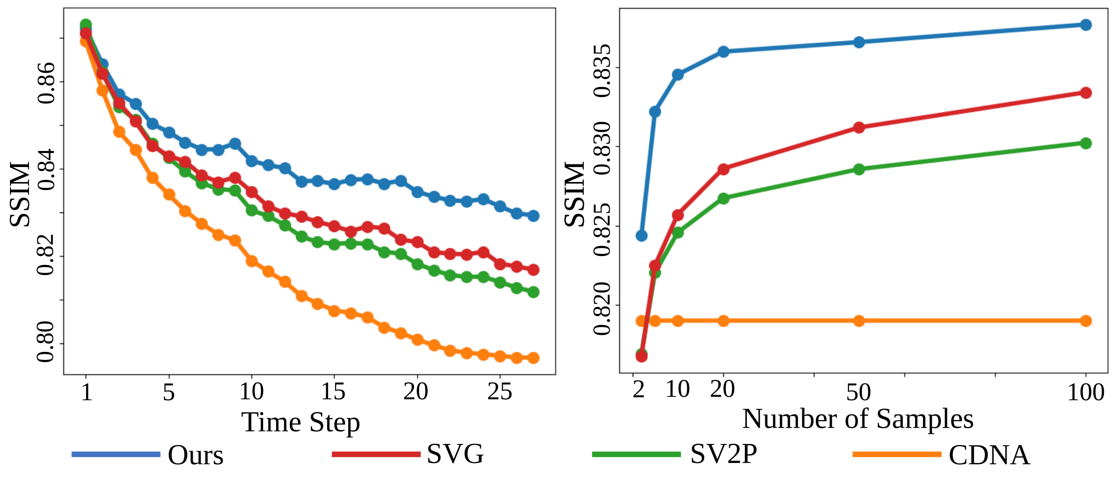


Figure 3.6: Evaluation in terms of SSIM on RobotPush [44] dataset. Left figure: X-axis is the time step and Y-axis is SSIM. Right figure: X-axis refers to the number of random samples during evaluation and Y-axis is averaged SSIM over a whole predicted sequence.

deterministic setting, retrieved examples generally follow similar trajectory, whose variance is low. For the stochastic version, searched sequences are highly diverse but follow the same motion pattern, i.e., bouncing back when hitting the boundary. Guided by examples from these experiences, our model is able to reliably capture the motion pattern under both settings. It implies that compared to fixed/learned prior, the motion variety could be better represented by similar examples.

Robot Arm Motion Prediction

Experiments on RobotPush [44] dataset take 5 frames as inputs and predict the following 10 frames during training. As illustrated in Fig. 3.6 (first column refers to input), we present quantitative evaluation in terms of SSIM. For the stochastic method, the best value of 20 random samples is presented. Fig. 3.6 implies that our method outperforms all previous methods by a large margin. We find CDNA [49] (deterministic method) is inferior to stochastic ones. We attribute this to the high uncertainty of robot motion in this dataset. Our model, facilitated by example guidance, is capable of capturing the real motion dynamics in a more efficient manner. *To comprehensively evaluate the distribution modelling ability and sampling efficiency of the proposed method,* we calculate the mean accuracy w.r.t. the number of samples (denoted as P): Fig. 3.6 shows the accuracy improvement for all stochastic methods along with the increase of P, which tends to be saturated when P is large. It is worth mentioning that our model still outperforms SVG [35] by a large margin when P is sufficiently large, e.g., 100. This clearly indicates a higher upper bound of accuracy achieved by our model (with guidance of retrieved examples) compared to variational inference based method, i.e., superior capability to capture real-world motion pattern.

Predicted sequences are shown in Fig. 3.5. For row arrangement please refer to the caption.

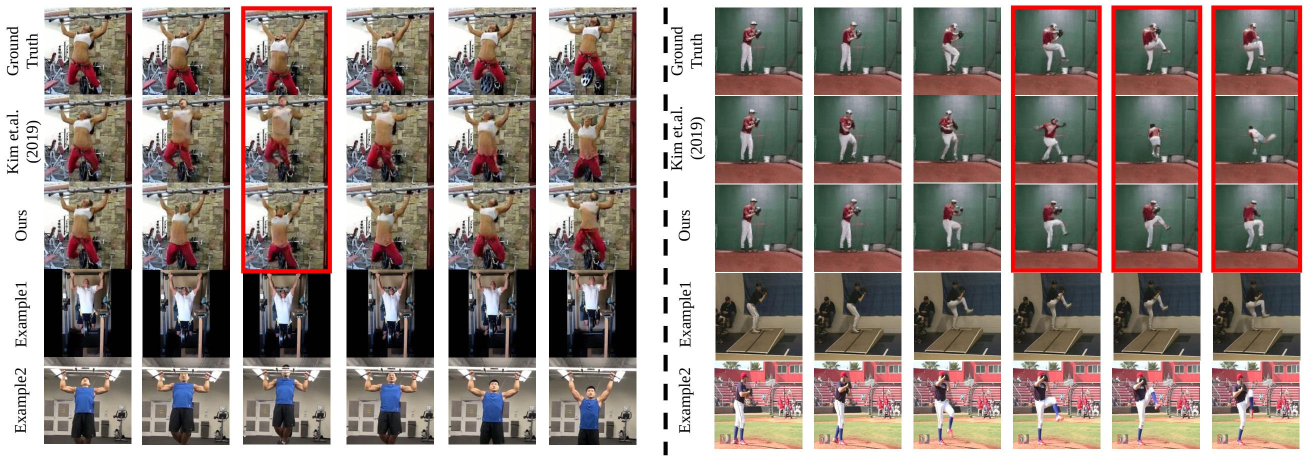


Figure 3.7: Qualitative evaluation of human motion prediction on PennAction [206] dataset. We present ground truth, results of [80], predicted sequence of our model and two searched examples. The left part refers to a pull-up action with multi-modal futures based on the current input and searched examples are capable of matching with possibilities. The right part aims to show that our model is capable of preserving the general structure during prediction. Red-boxes highlight the corresponding evidences on both sides.

The key region (highlighted with red boxes) of predicted frames is zoomed in for better visualization of details (last column). Compared to stochastic baselines, our model achieves higher image quality of predicted sequences, i.e., object edges and general structure are better preserved. Meanwhile, the overall trajectory is more accurately predicted by our model if compared to two stochastic baselines, which is mainly facilitated by the effective guidance of retrieved examples.

Human Motion Prediction

We report the experimental results on a human daily activity dataset, i.e., PennAction [206]. We follow the setting of [80], which is also a strong baseline for comparison. More specifically, the class label and first frame are fed as inputs. Note that under this situation we retrieve the examples according to the first frame in sequences with an identical action label.

Metric	[1]	[2]	[3]	[4]	Ours
Action Acc↑	15.89	40.00	47.14	68.89	**73.23**
FVD↓	4083.3	3324,9	2187.5	1509.0	**1283.5**

Table 3.2: Quantitative evaluation of predicted sequences in terms of Fréchet Video Distance (FVD) [177] (lower is better) and action recognition accuracy (higher is better). Previous works [1]-[4] refer to [95, 192, 181] respectively. Experiment is conducted on PennAction dataset [206].

Metric	K=2	K=3	K=4	K=5	K=6	K=7
PSNR	17.81	18.19	18.28	18.35	18.31	18.25
SSIM	0.78	0.82	0.83	0.84	0.84	0.83

Table 3.3: Influence of the example number K evaluated in terms of PSNR (first row) and SSIM (second row) on RobotPush [44] dataset. Note that each number reported in this table is averaged over the whole predicted sequence.

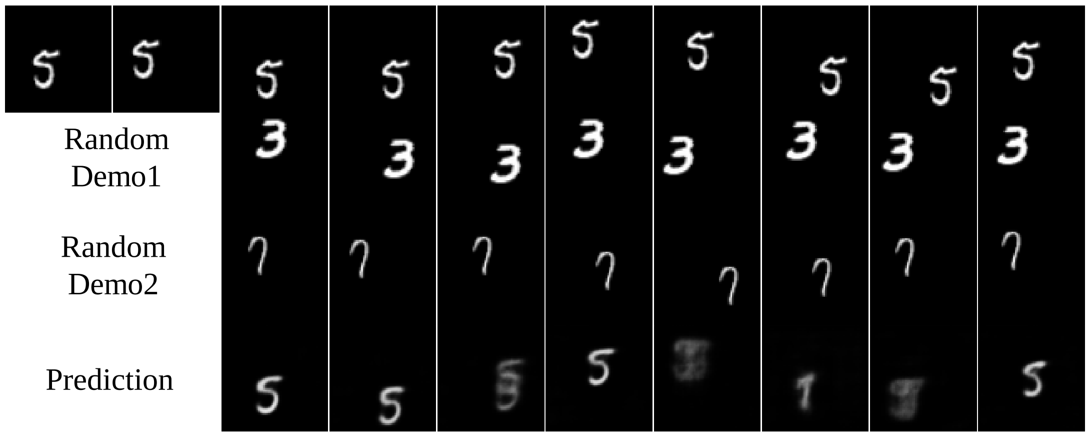


Figure 3.8: Prediction results with random example guidance on MovingMnist [162] dataset. The top and bottom rows correspond to ground truth and predicted sequence, while the middle two rows are randomly selected examples in this dataset. Unnatural motion is observed during prediction (last row).

To evaluate the multi-modal distribution modelling capability, Fig. 3.7A presents the best prediction sequence of 20 random samples in terms of PSNR and please refer to the caption for row arrangement. First column refers to input. The pull-up action generally possesses two motion modalities, i.e., up and down. We can observe that [80] fails to predict corresponding motion precisely even with 20 samples (third time step highlighted with red-boxes). Our model, guided by similar examples (last two rows in Fig. 3.7A), is capable of synthesizing the correct motion pattern compared to the groundtruth sequence. Meanwhile, from Fig. 3.7B we can notice that [80] fails to preserve the general structure during prediction. The human topology is severely distorted especially at the late stage of prediction (last 3 time steps highlighted with red-boxes). As comparison the structure of subject is well maintained predicted by our model, which is visually more natural than the results of [80]. This implies reliably capturing the motion distribution facilitates better visual quality of final predicted image sequences.

For quantitative evaluation, we follow [80] to calculate the action recognition accuracy and FVD [177] score. As shown in Tab. 3.2, our model outperforms all previous methods in terms of both action recognition accuracy and FVD score by a large margin. This mainly benefits from the retrieved examples, which provides effective guidance for future prediction.

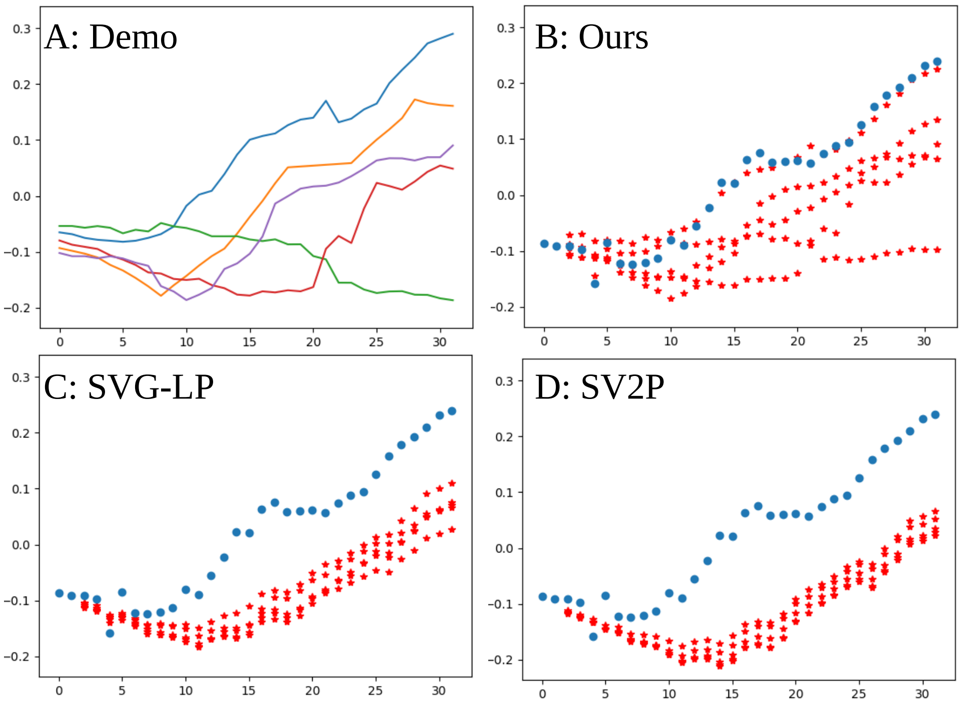


Figure 3.9: Visualization of retrieved examples and randomly sampled sequences on RobotPush [44] dataset. Top left refers to searched examples, while the other three figures correspond to sampled sequences by proposed model, SVG [192] and SV2P [8] respectively.

Ablation Study

Does example guidance really help? To evaluate the effectiveness of retrieved examples, we replace the retrieval procedure described in Sec. 3.3 with the random selection, i.e., the examples have no motion similarity with inputs. We conduct this experiment on MovingMnist [162] dataset. Results are presented in Fig. 3.8 and please refer to caption for detailed row arrangement. Due to the lack of motion similarity between examples and the input sequence, the predicted sequence demonstrates unnatural motion. The double-image effect of digit 5 (last row in Fig. 3.8), resulting from the misleading information of motion trajectory provided by random examples, implies the critical value of retrieval procedure proposed in Sec. 3.3.

Does the proposed model really capture multi-modal distribution? We present the sampled motion features (Fig. 3.9) in RobotPush [44] dataset to evaluate the capability of distribution modelling. For row arrangement please refer to the caption. For sub-figures from B to D, red-dot lines refer to predicted sequences and blue ones are ground truth. We can observe that the sampled states of SVG [35] and SV2P [8] are not multi-modal distributed. Guided by retrieved examples whose multi-modality distribution generally cover the ground truth motion, our model is able to predict the future motion in a more efficient way. Meanwhile, we present more visualization results to show that predicted sequences are not simply copied from examples and they are highly diverse.

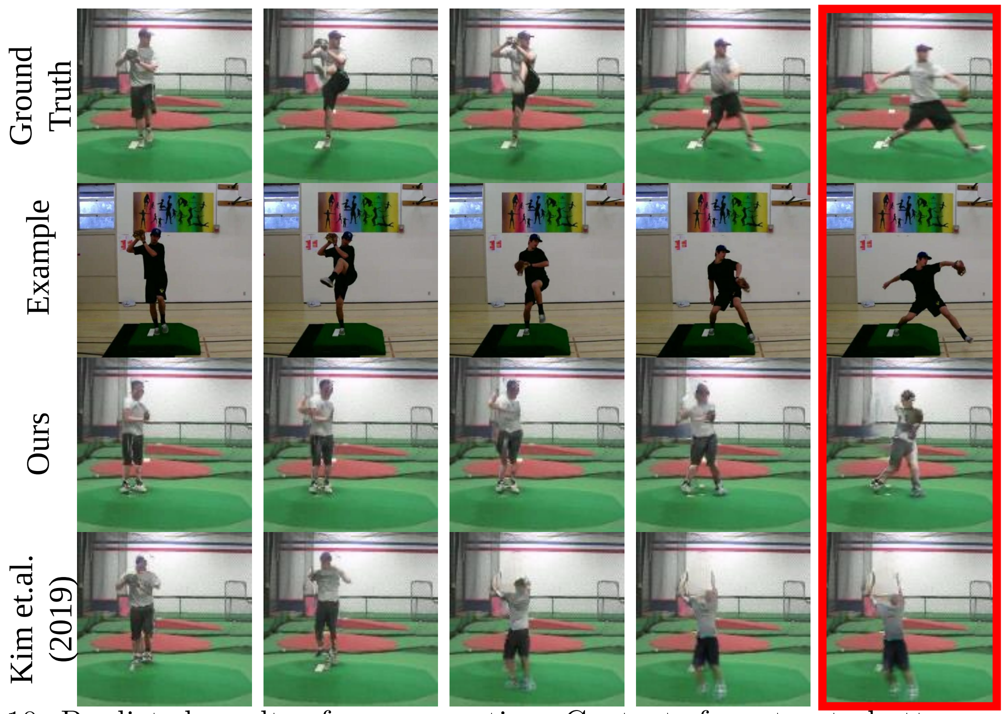


Figure 3.10: Predicted results of unseen motion. Contents from top to bottom are ground truth sequence, retrieved example, prediction of our model and the results of [80] respectively.

Influence of Example Number K. As illustrated in Tab. 3.3, we conduct corresponding ablation study about K on RobotPush [44] dataset. Performance under two metrics, i.e., PSNR and SSIM, is reported. PSNR and SSIM are averaged over the whole sequence and the best of 20 random sequences is reported. K ranges from 2 to 7. We can see that both PSNR and SSIM keep increase when K is no larger than 5 and then decrease. It indicates that multi-modal examples facilitate better modelling the target distribution, but noise information (or irrelevant motion pattern) might be introduced when K is too large.

Motion Prediction Beyond Seen Class

To further evaluate the generalization ability of the proposed model, we are motivated to predict the motion sequence on unseen class. The majority of video prediction methods are merely able to forecast the motion pattern accessible during training, which are hardly generalizable to novel motion. We conduct experiments on PennAction [206] dataset. We choose three actions, i.e., golf_swing, pull_ups and tennis_serve as known action during training and baseball_pitch as the unseen motion used during testing. Our model as well as that of [80] is retrained without label class. During testing, the examples for guidance are retrieved baseball_pitch sequences. Fig. 3.10 demonstrates the predicted results. For row arrangement please refer to the caption. We can observe that [80] fails to give rational prediction regarding the input, where there should be a baseball_pitch motion but visually resemble tennis_serve. Facilitated by the guidance of examples, our model produces a visually natural tennis_serve sequence, which clearly demonstrates the generalization capability of proposed model. We argue that the majority of previous works are (implicitly) forced to memorize motion categories in the training set. In contrast to the paradigm, our work is relieved from such burden because the retrieved examples contain the category information in assistance of prediction. We thus focus only on intra-class diversity. If given examples with unseen motion categories, our model is still able to give reasonable predictions, thanks to the example guidance.

3.5 Discussion

In this work, we present a simple yet effective framework for multi-modal video prediction, which mainly focuses on the capability of multi-modal distribution modelling. We first retrieve similar examples in the training set and then use these searched sequences to explicitly construct a distribution target. With proposed optimization method based on stochastic process, our model achieves promising performance on both prediction accuracy and visual quality.

Chapter 4

Model-based Deep Reinforcement Learning

Model-based reinforcement learning (RL) is considered to be a promising approach to reduce the sample complexity that hinders model-free RL. However, the theoretical understanding of such methods has been rather limited. This paper introduces a novel algorithmic framework for designing and analyzing model-based RL algorithms with theoretical guarantees. We design a meta-algorithm with a theoretical guarantee of monotone improvement to a local maximum of the expected reward. The meta-algorithm iteratively builds a lower bound of the expected reward based on the estimated dynamical model and sample trajectories, and then maximizes the lower bound jointly over the policy and the model. The framework extends the optimism-in-face-of-uncertainty principle to non-linear dynamical models in a way that requires *no explicit* uncertainty quantification. Instantiating our framework with simplification gives a variant of model-based RL algorithms Stochastic Lower Bounds Optimization (SLBO). Experiments demonstrate that SLBO achieves state-of-the-art performance when only one million or fewer samples are permitted on a range of continuous control benchmark tasks.

4.1 Background

In recent years deep reinforcement learning has achieved strong empirical success, including super-human performances on Atari games and Go [113, 156] and learning locomotion and manipulation skills in robotics [92, 151, 97]. Many of these results are achieved by model-free RL algorithms that often require a massive number of samples, and therefore their applications are mostly limited to simulated environments. Model-based deep reinforcement learning, in contrast, exploits the information from state observations explicitly — by planning with an estimated dynamical model — and is considered to be a promising approach to reduce the sample complexity. Indeed, empirical results [32, 33, 92, 118, 87, 139] have shown strong improvements in sample efficiency.

Despite promising empirical findings, many of *theoretical* properties of model-based deep reinforcement learning are not well-understood. For example, how does the error of the estimated model affect the estimation of the value function and the planning? Can model-based RL algorithms be guaranteed to improve the policy monotonically and converge to a local maximum of the value function? How do we quantify the uncertainty in the dynamical models?

It's challenging to address these questions theoretically in the context of deep RL with continuous state and action space and non-linear dynamical models. Due to the high-dimensionality, learning models from observations in one part of the state space and extrapolating to another part sometimes involves a leap of faith. The uncertainty quantification of the non-linear parameterized dynamical models is difficult — even without the RL components, it is an active but widely-open research area. Prior work in model-based RL mostly quantifies uncertainty with either heuristics or simpler models [115, 194, 31].

Previous theoretical work on model-based RL mostly focuses on either the finite-state MDPs [67, 12, 50, 88, 62, 136, 135], or the linear parametrization of the dynamics, policy, or value function [1, 157, 29, 167, 170], but not much on non-linear models. Even with an oracle prediction intervals[1] or posterior estimation, to the best of our knowledge, there was no previous algorithm with convergence guarantees for model-based deep RL.

Towards addressing these challenges, the main contribution of this paper is to propose a novel algorithmic framework for model-based deep RL with theoretical guarantees. Our meta-algorithm (Algorithm 2) extends the optimism-in-face-of-uncertainty principle to non-linear dynamical models in a way that requires *no explicit* uncertainty quantification of the dynamical models.

Let V^π be the value function V^π of a policy π on the true environment, and let $\widehat{V}^\pi$ be the value function of the policy π on the estimated model $\widehat{M}$. We design provable upper bounds, denoted by $D^{\pi,\widehat{M}}$, on how much the error can compound and divert the expected value $\widehat{V}^\pi$ of the imaginary rollouts from their real value V^π, in a neighborhood of some reference policy. Such upper bounds capture the intrinsic difference between the estimated and real dynamical model with respect to the particular reward function under consideration.

The discrepancy bounds $D^{\pi,\widehat{M}}$ naturally leads to a lower bound for the true value function:

$$V^\pi \geq \widehat{V}^\pi - D^{\pi,\widehat{M}}\,. \tag{4.1.1}$$

Our algorithm iteratively collects batches of samples from the interactions with environments, builds the lower bound above, and then maximizes it over both the dynamical model $\widehat{M}$ and the policy π. We can use any RL algorithms to optimize the lower bounds, because it will be designed to only depend on the sample trajectories from a fixed reference policy (as opposed to requiring new interactions with the policy iterate.)

We show that the performance of the policy is guaranteed to monotonically increase, assuming the optimization within each iteration succeeds (see Theorem 4.3.1.) To the

[1]We note that the confidence interval of parameters are likely meaningless for over-parameterized neural networks models.

best of our knowledge, this is the first theoretical guarantee of monotone improvement for model-based deep RL.

Readers may have realized that optimizing a robust lower bound is reminiscent of robust control and robust optimization. The distinction is that we optimistically and iteratively maximize the RHS of (4.1.1) jointly over the model and the policy. The iterative approach allows the algorithms to collect higher quality trajectory adaptively, and the optimism in model optimization encourages explorations of the parts of space that are not covered by the current discrepancy bounds.

To instantiate the meta-algorithm, we design a few valid discrepancy bounds in Section 4.4. In Section 4.4, we recover the norm-based model loss by imposing the additional assumption of a Lipschitz value function. The result suggests a norm is preferred compared to the square of the norm. Indeed in Section 4.5, we show that experimentally learning with ℓ_2 loss significantly outperforms the mean-squared error loss (ℓ_2^2).

In Section 4.4, we design a discrepancy bound that is *invariant* to the representation of the state space. Here we measure the loss of the model by the difference between the value of the predicted next state and the value of the true next state. Such a loss function is shown to be invariant to one-to-one transformation of the state space. Thus we argue that the loss is an intrinsic measure for the model error without any information beyond observing the rewards. We also refine our bounds in Section 4.6 by utilizing some mathematical tools of measuring the difference between policies in χ^2-divergence (instead of KL divergence or TV distance).

Our analysis also sheds light on the comparison between model-based RL and on-policy model-free RL algorithms such as policy gradient or TRPO [153]. The RHS of equation (4.1.1) is likely to be a good approximator of V^π in a larger neighborhood than the linear approximation of V^π used in policy gradient is (see Remark 4.4.5.)

Finally, inspired by our framework and analysis, we design a variant of model-based RL algorithms Stochastic Lower Bounds Optimization (SLBO). Experiments demonstrate that SLBO achieves state-of-the-art performance when only 1M samples are permitted on a range of continuous control benchmark tasks.

4.2 Related Work

Notations and Preliminaries

We denote the state space by $\mathcal{S}$, the action space by $\mathcal{A}$. A policy $\pi(\cdot|s)$ specifies the conditional distribution over the action space given a state s. A dynamical model $M(\cdot|s,a)$ specifies the conditional distribution of the next state given the current state s and action a. We will use $M^\star$ globally to denote the unknown true dynamical model. Our target applications are problems with the continuous state and action space, although the results apply to discrete state or action space as well. When the model is deterministic, $M(\cdot|s,a)$ is a dirac measure. In this case, we use $M(s,a)$ to denote the unique value of s' and view M as a function from

$\mathcal{S} \times \mathcal{A}$ to $\mathcal{S}$. Let $\mathcal{M}$ denote a (parameterized) family of models that we are interested in, and Π denote a (parameterized) family of policies.

Unless otherwise stated, for random variable X, we will use p_X to denote its density function.

Let S_0 be the random variable for the initial state. Let $S_t^{\pi,M}$ to denote the random variable of the states at steps t when we execute the policy π on the dynamic model M stating with S_0. Note that $S_0^{\pi,M} = S_0$ unless otherwise stated. We will omit the subscript when it's clear from the context. We use A_t to denote the actions at step t similarly. We often use τ to denote the random variable for the trajectory $(S_0, A_1, \dots, S_t, A_t, \dots)$. Let $R(s,a)$ be the reward function at each step. We assume R is *known* throughout the paper, although R can be also considered as part of the model if unknown. Let γ be the discount factor.

Let $V^{\pi,M}$ be the value function on the model M and policy π defined as:

$$V^{\pi,M}(s) = \mathop{\mathbb{E}}_{\substack{\forall t \geq 0, A_t \sim \pi(\cdot|S_t) \\ S_{t+1} \sim M(\cdot|S_t, A_t)}} \left[\sum_{t=0}^{\infty} \gamma^t R(S_t, A_t) \mid S_0 = s \right] \tag{4.2.1}$$

We define $V^{\pi,M} = \mathbb{E}\left[V^{\pi,M}(S_0)\right]$ as the expected reward-to-go at Step 0 (averaged over the random initial states). Our goal is to maximize the reward-to-go on the true dynamical model, that is, $V^{\pi,M^\star}$, over the policy π. For simplicity, throughout the paper, we set $\kappa = \gamma(1-\gamma)^{-1}$ since it occurs frequently in our equations. Every policy π induces a distribution of states visited by policy π:

Definition 4.2.1. *For a policy π, define $\rho^{\pi,M}$ as the discounted distribution of the states visited by π on M. Let ρ^π be a shorthand for $\rho^{\pi,M^\star}$ and we omit the superscript $M^\star$ throughout the paper. Concretely,we have $\rho^\pi = (1-\gamma)\sum_{t=0}^{\infty} \gamma^t \cdot p_{S_t^\pi}$*

Additional Related work

Model-based reinforcement learning is expected to require fewer samples than model-free algorithms [33] and has been successfully applied to robotics in both simulation and in the real world [32, 117, 34] using dynamical models ranging from Gaussian process [32, 83], time-varying linear models [91, 98, 90, 203], mixture of Gaussians [79], to neural networks [66, 118, 87, 173, 148, 129]. In particular, the work of [87] uses an ensemble of neural networks to learn the dynamical model, and significantly reduces the sample complexity compared to model-free approaches. The work of [25] makes further improvement by using a probabilistic model ensemble. Clavera et al. [26] extended this method with meta-policy optimization and improve the robustness to model error. In contrast, we focus on theoretical understanding of model-based RL and the design of new algorithms, and our experiments use a *single* neural network to estimate the dynamical model.

Our discrepancy bound in Section 4.4 is closely related to the work [46] on the value-aware model loss. Our approach differs from it in three details: a) we use the absolute value of the value difference instead of the squared difference; b) we use the imaginary value function from the estimated dynamical model to define the loss, which makes the loss purely a function of the estimated model and the policy; c) we show that the iterative algorithm, using the loss function as a building block, can converge to a local maximum, partly by cause of the particular choices made in a) and b). [7] also study the discrepancy bounds under Lipschitz condition of the MDP.

Prior work explores a variety of ways of combining model-free and model-based ideas to achieve the best of the two methods [165, 166, 141, 116, 164]. For example, estimated models [91, 57, 75] are used to enrich the replay buffer in the model-free off-policy RL. [140] proposes goal-conditioned value functions trained by model-free algorithms and uses it for model-based controls. [47, 18] use dynamical models to improve the estimation of the value functions in the model-free algorithms.

On the control theory side, [30, 29] provide strong finite sample complexity bounds for solving linear quadratic regulator using model-based approach. [14] provide finite-data guarantees for the "coarse-ID control" pipeline, which is composed of a system identification step followed by a robust controller synthesis procedure. Our method is inspired by the general idea of maximizing a low bound of the reward in [29]. By contrast, our work applies to non-linear dynamical systems. Our algorithms also estimate the models iteratively based on trajectory samples from the learned policies.

Strong model-based and model-free sample complexity bounds have been achieved in the tabular case (finite state space). We refer the readers to [72, 28, 169, 77, 67, 4] and the reference therein. Our work focus on continuous and high-dimensional state space (though the results also apply to tabular case).

Another line of work of model-based reinforcement learning is to learn a dynamic model in a hidden representation space, which is especially necessary for pixel state spaces [72, 28, 169, 77, 67]. [160] shows the possibility to learn an abstract transition model to imitate expert policy. [125] learns the hidden state of a dynamical model to predict the value of the future states and applies RL or planning on top of it. [154, 58] learns a bottleneck representation of the states. Our framework can be potentially combined with this line of research.

4.3 Algorithmic Framework

As mentioned in the introduction, towards optimizing $V^{\pi,M^\star}$,[2] our plan is to build a lower bound for $V^{\pi,M^\star}$ of the following type and optimize it iteratively:

$$V^{\pi,M^\star} \geq V^{\pi,\widehat{M}} - \mathcal{D}(\widehat{M},\pi) \tag{4.3.1}$$

where $\mathcal{D}(\widehat{M},\pi) \in \mathbb{R}_{\geq 0}$ bounds from above the discrepancy between $V^{\pi,\widehat{M}}$ and $V^{\pi,M^\star}$. Building such an optimizable discrepancy bound globally that holds for all $\widehat{M}$ and π turns out to be

[2]Note that in the introduction we used V^π for simplicity, and in the rest of the paper we will make the dependency on $M^\star$ explicit.

rather difficult, if not impossible. Instead, we shoot for establishing such a bound over the neighborhood of a reference policy π_{ref}.

$$V^{\pi,M^\star} \geq V^{\pi,\widehat{M}} - \mathcal{D}_{\pi_{\text{ref}},\delta}(\widehat{M},\pi), \qquad \forall \pi \text{ s.t. } d(\pi,\pi_{\text{ref}}) \leq \delta \tag{R1}$$

Here $d(\cdot,\cdot)$ is a function that measures the closeness of two policies, which will be chosen later in alignment with the choice of $\mathcal{D}$. We will mostly omit the subscript δ in $\mathcal{D}$ for simplicity in the rest of the paper. We will require our discrepancy bound to vanish when $\widehat{M}$ is an accurate model:

$$\widehat{M} = M^\star \Longrightarrow \mathcal{D}_{\pi_{\text{ref}}}(\widehat{M},\pi) = 0, \quad \forall \pi, \pi_{\text{ref}} \tag{R2}$$

The third requirement for the discrepancy bound $\mathcal{D}$ is that it can be estimated and optimized in the sense that

$$\mathcal{D}_{\pi_{\text{ref}}}(\widehat{M},\pi) \text{ is of the form } \mathop{\mathbb{E}}_{\tau\sim\pi_{\text{ref}},M^\star}[f(\widehat{M},\pi,\tau)] \tag{R3}$$

where f is a known differentiable function. We can estimate such discrepancy bounds for *every* π in the neighborhood of π_{ref} by sampling empirical trajectories $\tau^{(1)},\dots,\tau^{(n)}$ from executing policy π_{ref} on the real environment $M^\star$ and compute the average of $f(\widehat{M},\pi,\tau^{(i)})$'s. We would have to insist that the expectation cannot be over the randomness of trajectories from π on $M^\star$, because then we would have to re-sample trajectories for every possible π encountered.

For example, assuming the dynamical models are all deterministic, one of the valid discrepancy bounds (under some strong assumptions) that will prove in Section 4.4 is a multiple of the error of the prediction of $\widehat{M}$ on the trajectories from π_{ref}:

$$\mathcal{D}_{\pi_{\text{ref}}}(\widehat{M},\pi) = L \cdot \mathop{\mathbb{E}}_{S_0,\dots,S_t,\sim\pi_{\text{ref}},M^\star}\left[\|\widehat{M}(S_t) - S_{t+1}\|\right] \tag{4.3.2}$$

Suppose we can establish such an discrepancy bound $\mathcal{D}$ (and the distance function d) with properties (R1), (R2), and (R3), — which will be the main focus of Section 4.4 —, then we can devise the following meta-algorithm (Algorithm 2). We iteratively optimize the lower bound over the policy π_{k+1} and the model M_{k+1}, subject to the constraint that the policy is not very far from the reference policy π_k obtained in the previous iteration. For simplicity, we only state the population version with the exact computation of $\mathcal{D}_{\pi_{\text{ref}}}(\widehat{M},\pi)$, though empirically it is estimated by sampling trajectories.

We first remark that the discrepancy bound $\mathcal{D}_{\pi_k}(M,\pi)$ in the objective plays the role of learning the dynamical model by ensuring the model to fit to the sampled trajectories. For example, using the discrepancy bound in the form of equation (4.3.2), we roughly recover the standard objective for model learning, with the caveat that we only have the norm instead of the square of the norm in MSE. Such distinction turns out to be empirically important for better performance (see Section 4.5).

Algorithm 2 Meta-Algorithm for Model-based RL

```
Inputs: Initial policy $\pi_0$. Discrepancy bound $D$ and distance function $d$ that satisfy equation (R1) and (R2).
For $k = 0$ to $T$:
```

$$\pi_{k+1}, M_{k+1} = \arg \max_{\pi \in \Pi,\ M \in \mathcal{M}} \quad V^{\pi,M} - \mathcal{D}_{\pi_k,\delta}(M, \pi) \tag{4.3.3}$$

$$\text{s.t. } \ d(\pi, \pi_k) \leq \delta \tag{4.3.4}$$

Second, our algorithm can be viewed as an extension of the optimism-in-face-of-uncertainty (OFU) principle to non-linear parameterized setting: jointly optimizing M and π encourages the algorithm to choose the most optimistic model among those that can be used to accurately estimate the value function. (See [67, 12, 50, 88, 136, 135] and references therein for the OFU principle in finite-state MDPs.) The main novelty here is to optimize the lower bound directly, without explicitly building any confidence intervals, which turns out to be challenging in deep learning. In other words, the uncertainty is measured straightforwardly by how the error would affect the estimation of the value function.

Thirdly, the maximization of $V^{\pi,M}$, when M is fixed, can be solved by any model-free RL algorithms with M as the environment without querying any real samples. Optimizing $V^{\pi,M}$ jointly over π, M can be also viewed as another RL problem with an extended actions space using the known "extended MDP technique". See [67, section 3.1] for details.

Our main theorem shows formally that the policy performance in the real environment is non-decreasing under the assumption that the real dynamics belongs to our parameterized family $\mathcal{M}$.[3]

Theorem 4.3.1. *Suppose that $M^\star \in \mathcal{M}$, that $\mathcal{D}$ and d satisfy equation* (R1) *and* (R2)*, and the optimization problem in equation* (4.3.3) *is solvable at each iteration. Then, Algorithm 2 produces a sequence of policies $\pi_0, \ldots, \pi_T$ with monotonically increasing values:*

$$V^{\pi_0,M^\star} \leq V^{\pi_1,M^\star} \leq \cdots \leq V^{\pi_T,M^\star} \tag{4.3.5}$$

Moreover, as $k \to \infty$, the value $V^{\pi_k,M^\star}$ converges to some $V^{\bar{\pi},M^\star}$, where $\bar{\pi}$ is a local maximum of $V^{\pi,M^\star}$ in domain Π.

The theorem above can also be extended to a finite sample complexity result with standard concentration inequalities. We show in Theorem 4.6.17 that we can obtain an approximate local maximum in $O(1/\epsilon)$ iterations with sample complexity (in the number of trajectories) that is polynomial in dimension and accuracy ϵ and is logarithmic in certain smoothness parameters.

[3]We note that such an assumption, though restricted, may not be very far from reality: optimistically speaking, we only need to approximate the dynamical model accurately on the trajectories of the optimal policy. This might be much easier than approximating the dynamical model globally.

Proof of Theorem 4.3.1. Since $\mathcal{D}$ and d satisfy equation (R1), we have that

$$V^{\pi_{k+1},M^\star} \geq V^{\pi_{k+1},M_{k+1}} - \mathcal{D}_{\pi_k}(M_{k+1},\pi_{k+1})$$

By the definition that π_{k+1} and M_{k+1} are the optimizers of equation (4.3.3), we have that

$$V^{\pi_{k+1},M_{k+1}} - \mathcal{D}_{\pi_k}(M_{k+1},\pi_{k+1}) \geq V^{\pi_k,M^\star} - \mathcal{D}_{\pi_k}(M^\star,\pi_k) = V^{\pi_k,M^\star} \qquad \text{(by equation R2)}$$

Combing the two equations above we complete the proof of equation (4.3.5).

For the second part of the theorem, by compactness, we have that a subsequence of π_k converges to some $\bar{\pi}$. By the monotonicity we have $V^{\pi_k,M^\star} \leq V^{\bar{\pi},M^\star}$ for every $k \geq 0$. For the sake of contradiction, we assume $\bar{\pi}$ is a not a local maximum, then in the neighborhood of $\bar{\pi}$ there exists π' such that $V^{\pi',M^\star} > V^{\bar{\pi},M^\star}$ and $d(\bar{\pi},\pi') < \delta/2$. Let t be such that π_t is in the $\delta/2$-neighborhood of $\bar{\pi}$. Then we see that $(\pi', M^\star)$ is a better solution than (π_{t+1}, M_{t+1}) for the optimization problem (4.3.3) in iteration t because $V^{\pi',M^\star} > V^{\bar{\pi},M^\star} \geq V^{\pi_{t+1},M^\star} \geq V^{\pi_{t+1},M_{t+1}} - \mathcal{D}_{\pi_t}(M_{t+1},\pi_{t+1})$. (Here the last inequality uses equation (R1) with π_t as π_{ref}.) The fact $(\pi', M^\star)$ is a strictly better solution than (π_{t+1}, M_{t+1}) contradicts the fact that (π_{t+1}, M_{t+1}) is defined to be the optimal solution of (4.3.3) . Therefore $\bar{\pi}$ is a local maximum and we complete the proof.

$\square$

Particularly, the condition (R2) and (R3) are at odds with each other—(R3) essentially says that the discrepancy bound D can only carry information about $M^\star$ through the sampled data, and the non-trivial uncertainty about $M^\star$ from observing only sampled data implies that the discrepancy bound $\mathcal{D}_{\pi_{\text{ref}}}(\widehat{M},\pi)$ can never be zero (regardless of $M^\star = \widehat{M}$ or not). We can formally see this from the linear bandit setting (which corresponds to the horizon $H = 1$ case.)

In the linear bandit setting, with a bit abuse of notation, we still use $M \in \mathbb{R}^d$ for the model parameters and use π for the action in $\mathbb{R}^d$. We assume that the reward is linear: $V^{\pi,M} = \langle \pi, M\rangle$. In the sequel, we will show that (R3) and (R1) imply the "$\Leftarrow$" direction in (R2), and therefore we have,

$$\widehat{M} = M^* \iff \forall \pi, \pi_{\text{ref}}, D_{\pi_{\text{ref}}}(\widehat{M},\pi) = 0, \tag{4.3.6}$$

Note that (4.3.6) is statistically impossible because it allows us to find the true model M^* directly through checking if D is zero for all policies π, π_{ref}, which is not possible given finite data (at least not before we have sufficient data).

Now we prove the reverse direction of (R2). For the sake of contradiction, we assume that there exists $\widehat{M} \neq M^*$ such that for all policies π, π_{ref}, we have $d(\pi,\pi_{\text{ref}}) \leq \delta$ and $D_{\pi_{\text{ref}}}(\widehat{M},\pi) = 0$. There exists a policy π such that $V^{\pi,\widehat{M}} = \langle \pi, \widehat{M}\rangle > \langle \pi, M^\star\rangle = V^{\pi,M^*}$ because we can take a policy π that correlates with $\widehat{M}$ more than $M^\star$ (and this is the only place that we use linearity.) By (R1), we have $0 \geq D_{\pi_{\text{ref}}}(\widehat{M},\pi) \geq V^{\pi,\widehat{M}} - V^{\pi,M^*}$, which is a contradiction.

4.4 Discrepancy Bounds Design

In this section, we design discrepancy bounds that can provably satisfy the requirements (R1), (R2), and (R3).We design increasingly stronger discrepancy bounds from Section 4.4 to Section 4.6.

Norm-based prediction error bounds

In this subsection, we assume the dynamical model $M^\star$ is deterministic and we also learn with a deterministic model $\widehat{M}$. Under assumptions defined below, we derive a discrepancy bound $\mathcal{D}$ of the form $\|\widehat{M}(S,A) - M^\star(S,A)\|$ averaged over the observed state-action pair (S,A) on the dynamical model $\widehat{M}$. This suggests that the norm is a better metric than the mean-squared error for learning the model, which is empirically shown in Section 4.5. Through the derivation, we will also introduce a telescoping lemma, which serves as the main building block towards other finer discrepancy bounds.

We make the (strong) assumption that the value function $V^{\pi,\widehat{M}}$ on the estimated dynamical model is L-Lipschitz w.r.t to some norm $\|\cdot\|$ in the sense that

$$\forall s, s' \in \mathcal{S}, \left|V^{\pi,\widehat{M}}(s) - V^{\pi,\widehat{M}}(s')\right| \le L \cdot \|s - s'\| \tag{4.4.1}$$

In other words, nearby starting points should give reward-to-go under the same policy π. We note that not every real environment $M^\star$ has this property, let alone the estimated dynamical models. However, once the real dynamical model induces a Lipschitz value function, we may penalize the Lipschitz-ness of the value function of the estimated model during the training.

We start off with a lemma showing that the expected prediction error is an upper bound of the discrepancy between the real and imaginary values.

Lemma 4.4.1. *Suppose $V^{\pi,\widehat{M}}$ is L-Lipschitz (in the sense of* (4.4.1)*). Recall $\kappa = \gamma(1-\gamma)^{-1}$.*

$$\left|V^{\pi,\widehat{M}} - V^{\pi,M^\star}\right| \le \kappa L \mathop{\mathbb{E}}_{\substack{S\sim\rho^\pi \\ A\sim\pi(\cdot|S)}} \left[\|\widehat{M}(S,A) - M^\star(S,A)\|\right] \tag{4.4.2}$$

However, in RHS in equation 4.4.2 cannot serve as a discrepancy bound because it does not satisfy the requirement (R3) — to optimize it over π we need to collect samples from ρ^π for every iterate π — the state distribution of the policy π on the *real* model $M^\star$. The main proposition of this subsection stated next shows that for every π in the neighborhood of a reference policy π_{ref}, we can replace the distribution ρ^π be a fixed distribution $\rho^{\pi_{\text{ref}}}$ with incurring only a higher order approximation. We use the expected KL divergence between two π and π_{ref} to define the neighborhood:

$$d^{\text{KL}}(\pi, \pi_{\text{ref}}) = \mathop{\mathbb{E}}_{S\sim\rho^\pi} \left[KL(\pi(\cdot|S), \pi_{\text{ref}}(\cdot|S))^{1/2}\right] \tag{4.4.3}$$

Proposition 4.4.2. *In the same setting of Lemma 4.4.1, assume in addition that π is close to a reference policy π_{ref} in the sense that $d^{\text{KL}}(\pi, \pi_{\text{ref}}) \leq \delta$, and that the states in $\mathcal{S}$ are uniformly bounded in the sense that $\|s\| \leq B, \forall s \in \mathcal{S}$. Then,*

$$\left|V^{\pi,\widehat{M}} - V^{\pi,M^\star}\right| \leq \kappa L \mathop{\mathbb{E}}_{\substack{S\sim\rho^{\pi_{\text{ref}}}\\ A\sim\pi(\cdot|S)}} \left[\|\widehat{M}(S,A) - M^\star(S,A)\|\right] + 2\kappa^2\delta B \tag{4.4.4}$$

In a benign scenario, the second term in the RHS of equation (4.4.4) should be dominated by the first term when the neighborhood size δ is sufficiently small. Moreover, the term B can also be replaced by $\max_{S,A}\|\widehat{M}(S,A) - M^\star(S,A)\|$ (see the proof that is deferred to Section 4.6.). The dependency on κ may not be tight for real-life instances, but we note that most analysis of similar nature loses the additional κ factor [153, 2], and it's inevitable in the worst-case.

A telescoping lemma. Towards proving Propositions 4.4.2 and deriving stronger discrepancy bound, we define the following quantity that captures the discrepancy between $\widehat{M}$ and $M^\star$ on a single state-action pair (s, a).

$$G^{\pi,\widehat{M}}(s,a) = \mathop{\mathbb{E}}_{\hat{s}'\sim\widehat{M}(\cdot|s,a)} V^{\pi,\widehat{M}}(\hat{s}') - \mathop{\mathbb{E}}_{s'\sim M^\star(\cdot|s,a)} V^{\pi,\widehat{M}}(s') \tag{4.4.5}$$

Note that if $M, \widehat{M}$ are deterministic, then $G^{\pi,\widehat{M}}(s,a) = V^{\pi,\widehat{M}}(\widehat{M}(s,a)) - V^{\pi,\widehat{M}}(M^\star(s,a))$. We give a telescoping lemma that decompose the discrepancy between $V^{\pi,M}$ and $V^{\pi,M^\star}$ into the expected single-step discrepancy G.

Lemma 4.4.3. *[Telescoping Lemma] Recall that $\kappa := \gamma(1-\gamma)^{-1}$. For any policy π and dynamical models $M, \widehat{M}$, we have that*

$$V^{\pi,\widehat{M}} - V^{\pi,M} = \kappa \mathop{\mathbb{E}}_{\substack{S\sim\rho^{\pi,M}\\ A\sim\pi(\cdot|S)}} \left[G^{\pi,\widehat{M}}(S,A)\right] \tag{4.4.6}$$

The proof is reminiscent of the telescoping expansion in [73] (c.f. [153]) for characterizing the value difference of two policies, but we apply it to deal with the discrepancy between models. With the telescoping Lemma 4.4.3, Proposition 4.4.1 follows straightforwardly from Lipschitzness of the imaginary value function. Proposition 4.4.2 follows from that ρ^π and $\rho^{\pi_{\text{ref}}}$ are close. We defer the proof to Section 4.6.

Representation-invariant Discrepancy Bounds

The main limitation of the norm-based discrepancy bounds in previous subsection is that it depends on the state representation. Let $\mathcal{T}$ be a one-to-one map from the state space $\mathcal{S}$ to some other space $\mathcal{S}'$, and for simplicity of this discussion let's assume a model M is deterministic. Then if we represent every state s by its transformed representation $\mathcal{T}s$, then the transformed model $M^{\mathcal{T}}$ defined as $M^{\mathcal{T}}(s,a) \triangleq \mathcal{T}M(\mathcal{T}^{-1}s,a)$ together with the transformed reward $R^{\mathcal{T}}(s,a) \triangleq R(\mathcal{T}^{-1}s,a)$ and transformed policy $\pi^{\mathcal{T}}(s) \triangleq \pi(\mathcal{T}^{-1}s)$ is equivalent to the original set of the model, reward, and policy in terms of the performance(Lemma 4.6.3). Thus such transformation $\mathcal{T}$ is not identifiable from only observing the reward. However, the norm in the state space is a notion that depends on the hidden choice of the transformation $\mathcal{T}$. [4]

Another limitation is that the loss for the model learning should also depend on the state itself instead of only on the difference $\widehat{M}(S,A) - M^\star(S,A)$. It is possible that when S is at a critical position, the prediction error needs to be highly accurate so that the model $\widehat{M}$ can be useful for planning. On the other hand, at other states, the dynamical model is allowed to make bigger mistakes because they are not essential to the reward.

We propose the following discrepancy bound towards addressing the limitations above. Recall the definition of $G^{\pi,\widehat{M}}(s,a) = V^{\pi,\widehat{M}}(\widehat{M}(s,a)) - V^{\pi,\widehat{M}}(M^\star(s,a))$ which measures the difference between $\widehat{M}(s,a))$ and $M^\star(s,a)$ according to their imaginary rewards. We construct a discrepancy bound using the absolute value of G. Let's define ϵ_1 and $\epsilon_{\max}$ as the average of $|G^{\pi,\hat{M}}|$ and its maximum: $\epsilon_1 = \mathbb{E}_{S\sim\rho^{\pi_{\text{ref}}}}\left[\left|G^{\pi,\widehat{M}}(S,A)\right|\right]$ and $\epsilon_{\max} = \max_S \left|G^{\pi,\widehat{M}}(S)\right|$ where $G^{\pi,\widehat{M}}(S) = \mathbb{E}_{A\sim\pi}\left[G^{\pi,\widehat{M}}(S,A)\right]$. We will show that the following discrepancy bound $\mathcal{D}^G_{\pi_{\text{ref}}}(\widehat{M},\pi)$ satisfies the property (R1), (R2).

$$\mathcal{D}^G_{\pi_{\text{ref}}}(\widehat{M},\pi) = \kappa \cdot \epsilon_1 + \kappa^2\delta\epsilon_{\max} \tag{4.4.7}$$

Proposition 4.4.4. *Let d^{KL} and $\mathcal{D}^G$ be defined as in equation* (4.4.3) *and* (4.4.7)*. Then the choice $d = d^{\text{KL}}$ and $\mathcal{D} = \mathcal{D}^G$ satisfies the basic requirements (equation* (R1) *and* (R2)*). Moreover, G is invariant w.r.t any one-to-one transformation of the state space (in the sense of equation 4.6.4 in the proof).*

The proof follows from the telescoping lemma (Lemma 4.4.3) and is deferred to Section 4.6. We remark that the first term $\kappa\epsilon_1$ can in principle be estimated and optimized approximately: the expectation be replaced by empirical samples from $\rho^{\pi_{\text{ref}}}$, and $G^{\pi,\hat{M}}$ is an analytical function of π and $\widehat{M}$ when they are both deterministic, and therefore can be optimized by back-propagation through time (BPTT). (When π and $\widehat{M}$ and are stochastic with a re-parameterizable noise such as Gaussian distribution [81], we can also use back-propagation to estimate the gradient.) The second term in equation (4.4.7) is difficult to optimize because it involves the maximum. However, it can be in theory considered as a second-order term because δ can be chosen to be a fairly small number. (In the refined bound in Section 4.6, the dependency on δ is even milder.)

Remark 4.4.5. *Proposition 4.4.4 intuitively suggests a technical reason of why model-based approach can be more sample-efficient than policy gradient based algorithms such as TRPO or PPO [153, 152]. The approximation error of $V^{\pi,\widehat{M}}$ in model-based approach decreases as the*

[4]That said, in many cases the reward function itself is known, and the states have physical meanings, and therefore we may be able to use the domain knowledge to figure out the best norm.

model error $\epsilon_1, \epsilon_{\max}$ *decrease or the neighborhood size* δ *decreases, whereas the approximation error in policy gradient only linearly depends on the the neighborhood size [153]. In other words, model-based algorithms can trade model accuracy for a larger neighborhood size, and therefore the convergence can be faster (in terms of outer iterations.) This is consistent with our empirical observation that the model can be accurate in a descent neighborhood of the current policy so that the constraint* (4.3.4) *can be empirically dropped. We also refine our bonds in Section 4.6, where the discrepancy bounds is proved to decay faster in* δ.

4.5 Experiments

Practical implementation

We design with simplification of our framework a variant of model-based RL algorithms, Stochastic Lower Bound Optimization (SLBO). First, we removed the constraints (4.3.4). Second, we stop the gradient w.r.t M (but not π) from the occurrence of M in $V^{\pi,M}$ in equation (4.3.3) (and thus our practical implementation is not optimism-driven.)

Extending the discrepancy bound in Section 4.4, we use a multi-step prediction loss for learning the models with ℓ_2 norm. For a state s_t and action sequence $a_{t:t+h}$, we define the h-step prediction $\hat{s}_{t+h}$ as $\hat{s}_t = s_t,$ and for $h \geq 0, \hat{s}_{t+h+1} = \widehat{M}_\phi(\hat{s}_{t+h}, a_{t+h})$, The *H-step loss* is then defined as

$$\mathcal{L}_\phi^{(H)}((s_{t:t+h}, a_{t:t+h}); \phi) = \frac{1}{H}\sum_{i=1}^{H} \|(\hat{s}_{t+i} - \hat{s}_{t+i-1}) - (s_{t+i} - s_{t+i-1})\|_2. \tag{4.5.1}$$

A similar loss is also used in [118] for validation. We note that motivation by the theory in Section 4.4, we use ℓ_2-norm instead of the square of ℓ_2 norm. The loss function we attempt to optimize at iteration k is thus[5]

$$\max_{\phi,\theta} \quad V^{\pi_\theta, \mathrm{sg}(\widehat{M}_\phi)} - \lambda \underset{(s_{t:t+h}, a_{t:t+h}) \sim \pi_k, M^\star}{\mathbb{E}} \left[\mathcal{L}_\phi^{(H)}((s_{t:t+h}, a_{t:t+h}); \phi)\right] \tag{4.5.2}$$

where λ is a tunable parameter and sg denotes the stop gradient operation.

We note that the term $V^{\pi_\theta, \mathrm{sg}(\widehat{M}_\phi)}$ depends on both the parameter θ and the parameter ϕ but there is no gradient passed through ϕ, whereas $\mathcal{L}_\phi^{(H)}$ only depends on the ϕ. We optimize equation (4.5.2) by alternatively maximizing $V^{\pi_\theta, \mathrm{sg}(\widehat{M}_\phi)}$ and minimizing $\mathcal{L}_\phi^{(H)}$: for the former, we use TRPO with samples from the estimated dynamical model $\widehat{M}_\phi$ (by treating $\widehat{M}_\phi$ as a fixed simulator), and for the latter we use standard stochastic gradient methods. Algorithm 3 gives a pseudo-code for the algorithm. The n_{model} and n_{policy} iterations are used to balance the number of steps of TRPO and Adam updates within the loop indexed by n_{inner}.[6]

[5]This is technically not a well-defined mathematical objective. The sg operation means identity when the function is evaluated, whereas when computing the update, $\mathrm{sg}(M_\phi)$ is considered fixed.

[6]In principle, to balance the number of steps, it suffices to take one of n_{model} and n_{policy} to be 1. However, empirically we found the optimal balance is achieved with larger n_{model} and n_{policy}, possibly due to complicated interactions between the two optimization problem.

Algorithm 3 Stochastic Lower Bound Optimization (SLBO)

```
1: Initialize model network parameters $\phi$ and policy network parameters $\theta$
2: Initialize dataset $\mathcal{D} \leftarrow \emptyset$
3: for $n_{\text{outer}}$ iterations do
4:   $\mathcal{D} \leftarrow \mathcal{D} \cup \{$ collect $n_{\text{collect}}$ samples from real environment using $\pi_\theta$ with noises $\}$
5:   for $n_{\text{inner}}$ iterations do
6:     for $n_{\text{model}}$ iterations do
7:       optimize (4.5.1) over $\phi$ with sampled data from $\mathcal{D}$ by one step of Adam
8:     end for
9:     for $n_{\text{policy}}$ iterations do
10:      $\mathcal{D}' \leftarrow \{$ collect $n_{\text{trpo}}$ samples using $\widehat{M}_\phi$ as dynamics $\}$
11:      optimize $\pi_\theta$ by running TRPO on $\mathcal{D}'$
12:    end for
13:  end for
14: end for
```

Power of stochasticity and connection to standard MB RL: We identify the main advantage of our algorithms over standard model-based RL algorithms is that we alternate the updates of the model and the policy within an outer iteration. By contrast, most of the existing model-based RL methods only optimize the models once (for a lot of steps) after collecting a batch of samples (see Algorithm 4 for an example). The stochasticity introduced from the alternation with stochastic samples seems to dramatically reduce the overfitting (of the policy to the estimated dynamical model) in a way similar to that SGD regularizes ordinary supervised training. [7] Another way to view the algorithm is that the model obtained from line 7 of Algorithm 3 at different inner iteration serves as an ensemble of models. We do believe that a cleaner and easier instantiation of our framework (with optimism) exists, and the current version, though performing very well, is not necessarily the best implementation.

Entropy regularization: An additional component we apply to SLBO is the commonly-adopted entropy regularization in policy gradient method [193, 112], which was found to significantly boost the performance in our experiments (ablation study in Section 4.6). Specifically, an additional entropy term is added to the objective function in TRPO. We hypothesize that entropy bonus helps exploration, diversifies the collected data, and thus prevents overfitting.

[7]Similar stochasticity can potentially be obtained by an extreme hyperparameter choice of the standard MB RL algorithm: in each outer iteration of Algorithm 4, we only sample a very small number of trajectories and take a few model updates and policy updates. We argue our interpretation of stochastic optimization of the lower bound (4.5.2) is more natural in that it reveals the regularization from stochastic optimization.

Experimental Results

We evaluate our algorithm SLBO (Algorithm 3) on five continuous control tasks from rllab [42], including Swimmer, Half Cheetah, Humanoid, Ant, Walker. All environments that we test have a maximum horizon of 500, which is longer than most of the existing model-based RL work [118, 87]. (Environments with longer horizons are commonly harder to train.) More details can be found in Section 4.6.

Baselines. We compare our algorithm with 3 other algorithms including: (1) Soft Actor-Critic (SAC) [59], the state-of-the-art model-free off-policy algorithm in sample efficiency; (2) Trust-Region Policy Optimization (TRPO) [153], a policy-gradient based algorithm; and (3) Model-Based TRPO, a standard model-based algorithm described in Algorithm 4. Details of these algorithms can be found in Section 4.6.[8]

The result is shown in Figure 4.1. In Fig 4.1, our algorithm shows superior convergence rate (in number of samples) than all the baseline algorithms while achieving better final performance with 1M samples. Specifically, we mark model-free TRPO performance after 8 million steps by the dotted line in Fig 4.1 and find out that our algorithm can achieve comparable or better final performance in one million steps. For ablation study, we also add the performance of SLBO-MSE, which corresponds to running SLBO with squared ℓ_2 model loss instead of ℓ_2. SLBO-MSE performs significantly worse than SLBO on four environments, which is consistent with our derived model loss in Section 4.4. We also study the performance of SLBO and baselines with 4 million training samples in 4.6. Ablation study of multi-step model training can be found in Section 4.6.

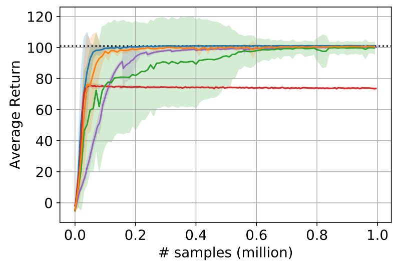
4.6 Additional Details

Refined bounds

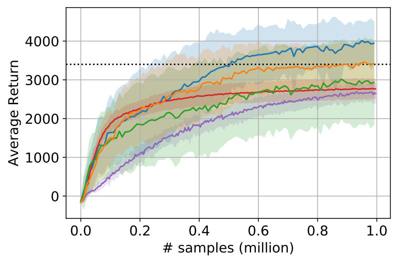
The theoretical limitation of the discrepancy bound $\mathcal{D}^G(\widehat{M}, \pi)$ is that the second term involving $\epsilon_{\max}$ is not rigorously optimizable by stochastic samples. In the worst case, there seem to exist situations where such infinity norm of $G^{\pi,\widehat{M}}$ is inevitable. In this section we tighten the discrepancy bounds with a different closeness measure d, χ^2-divergence, in the policy space, and the dependency on the $\epsilon_{\max}$ is smaller (though not entirely removed.) We note that χ^2-divergence has the same second order approximation as KL-divergence around the local neighborhood the reference policy and thus locally affects the optimization much.

We start by defining a re-weighted version β^π of the distribution ρ^π where examples in later step are slightly weighted up. We can effectively sample from β^π by importance sampling from ρ^π.

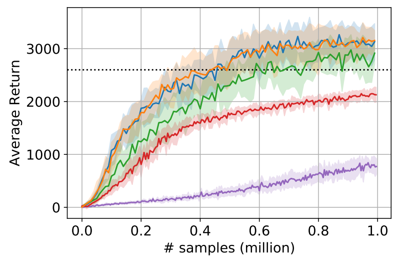
[8]We did not have the chance to implement the competitive random search algorithms in [104] yet, although our test performance with 500 episode length is higher than theirs with 1000 episode on Half Cheetach (3950 by ours vs 2345 by theirs) and Walker (3650 by ours vs 894 by theirs).



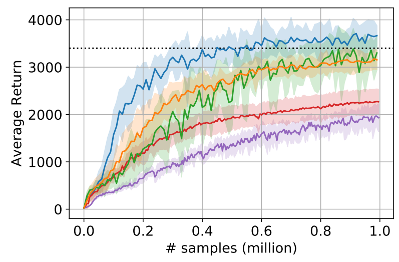
(a) Swimmer



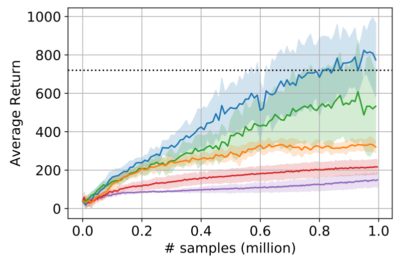
(b) Half Cheetah



(c) Ant



(d) Walker



(e) Humanoid

Figure 4.1: Comparison between SLBO (ours), SLBO with squared ℓ^2 model loss (SLBO-MSE), vanilla model-based TRPO (MB-TRPO), model-free TRPO (MF-TRPO), and Soft Actor-Critic (SAC). We average the results over 10 different random seeds, where the solid lines indicate the mean and shaded areas indicate one standard deviation. The dotted reference lines are the total rewards of MF-TRPO after 8 million steps.

Definition 4.6.1. *For a policy π, define β^π as the re-weighted version of discounted distribution of the states visited by π on $M^\star$. Recall that $p_{S_t^\pi}$ is the distribution of the state at step t, we define $\beta^\pi = (1-\gamma)^2 \sum_{t=1}^{\infty} t\gamma^{t-1} p_{S_t^\pi}$.*

Then we are ready to state our discrepancy bound. Let

$$d^{\chi^2}(\pi, \pi_{\text{ref}}) = \max\{\mathop{\mathbb{E}}_{S\sim\rho^{\pi_{\text{ref}}}}\left[\chi^2(\pi(\cdot|S), \pi_{\text{ref}}(\cdot|S))\right], \mathop{\mathbb{E}}_{S\sim\beta^{\pi_{\text{ref}}}}\left[\chi^2(\pi(\cdot|S), \pi_{\text{ref}}(\cdot|S))\right]\} \tag{4.6.1}$$

$$D^{\chi^2}_{\pi_{\text{ref}}}(\widehat{M}, \pi) = (1-\gamma)^{-1}\epsilon_1 + (1-\gamma)^{-2}\delta\epsilon_2 + (1-\gamma)^{-5/2}\delta^{3/2}\epsilon_{\max} \tag{4.6.2}$$

where $\epsilon_2 = \mathbb{E}_{S\sim\beta^{\pi_{\text{ref}}}}\left[G^{\pi,\widehat{M}}(S,A)^2\right]$ and $\epsilon_1, \epsilon_{\max}$ are defined in equation (4.4).

Proposition 4.6.2. *The discrepancy bound D^{χ^2} and closeness measure d^{χ^2} satisfies requirements* (R1) *and* (R2)*.*

We defer the proof to Section 4.6 so that we can group relevant proofs with similar tools together. Some of these tools may be of independent interests and used for better analysis of model-free reinforcement learning algorithms such as TRPO [153], PPO [152] and CPO [2].

Proof of Lemma 4.4.3

Proof of Lemma 4.4.3. Let W_j be the cumulative reward when we use dynamical model M for j steps and then $\widehat{M}$ for the rest of the steps, that is,

$$W_j = \mathop{\mathbb{E}}_{\substack{\forall t \geq 0, A_t \sim \pi(\cdot|S_t) \\ \forall j > t \geq 0, S_{t+1} \sim M(\cdot|S_t, A_t) \\ \forall t \geq j, S_{t+1} \sim \widehat{M}(\cdot|S_t, A_t)}} \left[\sum_{t=0}^{\infty} \gamma^t R(S_t, A_t) \mid S_0 = s \right]$$

By definition, we have that $W_\infty = V^{\pi,M}(s)$ and $W_0 = V^{\pi,\widehat{M}}(s)$. Then, we decompose the target into a telescoping sum,

$$V^{\pi,M}(s) - V^{\pi,\widehat{M}}(s) = \sum_{j=0}^{\infty} (W_{j+1} - W_j) \tag{4.6.3}$$

Now we re-write each of the summands $W_{j+1} - W_j$. Comparing the trajectory distribution in the definition of W_{j+1} and W_j, we see that they only differ in the dynamical model applied in j-th step. Concretely, W_j and W_{j+1} can be rewritten as $W_j = R + \mathbb{E}_{S_j, A_j \sim \pi, M} \left[\mathbb{E}_{\hat{S}_{j+1} \sim \widehat{M}(\cdot|S_j, A_j)} \left[\gamma^{j+1} V^{\pi,\widehat{M}}(\hat{S}_{j+1}) \right] \right]$ and $W_{j+1} = R + \mathbb{E}_{S_j, A_j \sim \pi, M^\star} \left[\mathbb{E}_{S_{j+1} \sim M(\cdot|S_j, A_j)} \left[\gamma^{j+1} V^{\pi,\widehat{M}}(S_{j+1}\right.\right.$ where R denotes the reward from the first j steps from policy π and model $M^\star$. Canceling the shared term in the two equations above, we get

$$W_{j+1} - W_j = \gamma^{j+1} \mathop{\mathbb{E}}_{S_j, A_j \sim \pi, M} \left[\mathop{\mathbb{E}}_{\substack{\hat{S}_{j+1} \sim \widehat{M}(\cdot|S_j, A_j) \\ S_{j+1} \sim M(\cdot|S_j, A_j)}} \left[V^{\pi,\widehat{M}}(S_{j+1}) - V^{\pi,\widehat{M}}(\widehat{S}_{j+1}) \right] \right]$$

Combining the equation above with equation (4.6.3) concludes that

$$V^{\pi,M} - V^{\pi,\widehat{M}} = \frac{\gamma}{1-\gamma} \mathop{\mathbb{E}}_{S \sim \rho^\pi, A \sim \pi(S)} \left[\mathop{\mathbb{E}}_{S' \sim M^\star(\cdot|S,A)} V^{\pi,\widehat{M}}(S') - \mathop{\mathbb{E}}_{\hat{S}' \sim \widehat{M}(\cdot|S,A)} V^{\pi,\widehat{M}}(\hat{S}') \right]$$

□

Missing Proofs in Section 4.4

Proof of Proposition 4.4.4. Towards proving the second part of Proposition 4.4.4 regarding the invariance, we state the following lemma:

Lemma 4.6.3. *Suppose for simplicity the model and the policy are both deterministic. For any one-to-one transformation from $\mathcal{S}$ to $\mathcal{S}'$, let $M^{\mathcal{T}}(s, a) \triangleq \mathcal{T} M(\mathcal{T}^{-1}s, a)$, $R^{\mathcal{T}}(s, a) \triangleq R(\mathcal{T}^{-1}s, a)$,* *and* $\pi^{\mathcal{T}}(s) \triangleq \pi(\mathcal{T}^{-1}s)$ *be a set of transformed model, reward and policy. Then we have that* (M, π, R) *is equivalent to* $(M^{\mathcal{T}}, \pi^{\mathcal{T}}, R^{\mathcal{T}})$ *in the sense that*

$$V^{\pi^{\mathcal{T}}, M^{\mathcal{T}}}(\mathcal{T}s) = V^{\pi,M}(s)$$

where the value function $V^{\pi^{\mathcal{T}}, M^{\mathcal{T}}}$ *is defined with respect to* $R^{\mathcal{T}}$*.*

Proof. Let $s_0^{\mathcal{T}} = \mathcal{T}s, \dots,$ be the sequence of states visited by policy $\pi^{\mathcal{T}}$ on model $M^{\mathcal{T}}$ starting from s. We have that $s_0^{\mathcal{T}} = \mathcal{T}s = \mathcal{T}s_0$. We prove by induction that $s_t^{\mathcal{T}} = \mathcal{T}s_t$. Assume this is true for some value t, then we prove that $s_{t+1}\mathcal{T} = \mathcal{T}s_{t+1}$ holds:

$$\begin{aligned} s_{t+1}^{\mathcal{T}} &= M^{\mathcal{T}}(s_t^{\mathcal{T}}, \pi^{\mathcal{T}}(s_t^{\mathcal{T}})) = M^{\mathcal{T}}(\mathcal{T}s_t, \pi^{\mathcal{T}}(\mathcal{T}s_t)) && \text{(by inductive hypothesis)} \\ &= \mathcal{T}M(s_t, \pi(s_t)) && \text{(by defintion of } M^{\mathcal{T}}, \pi^{\mathcal{T}}) \\ &= \mathcal{T}s_{t+1} \end{aligned}$$

Thus we have $R^{\mathcal{T}}(s_t^{\mathcal{T}}, a_t^{\mathcal{T}}) = R(s_t, a_t)$. Therefore $V^{\pi^{\mathcal{T}}, M^{\mathcal{T}}}(\mathcal{T}s) = V^{\pi,M}(s)$. □

Proof of Proposition 4.4.4. We first show the invariant of G under deterministic models and policies. The same result applies to stochastic policies with slight modification. Let $s^{\mathcal{T}} = \mathcal{T}s$. We consider the transformation applied to M and $M^\star$ and the resulting G function

$$G^{\mathcal{T}}(s^{\mathcal{T}}, a) \triangleq |V^{\pi^{\mathcal{T}}, M^{\mathcal{T}}}(M^{\mathcal{T}}(s^{\mathcal{T}}, a)) - V^{\pi^{\mathcal{T}}, M^{\mathcal{T}}}(M^{\star,\mathcal{T}}(s^{\mathcal{T}}, a))|$$

Note that by Lemma 4.6.3, we have that $V^{\pi^{\mathcal{T}}, M^{\mathcal{T}}}(M^{\mathcal{T}}(s^{\mathcal{T}}, a)) = V^{\pi,M}(\mathcal{T}^{-1}M^{\mathcal{T}}(s^{\mathcal{T}}, a)) = V^{\pi,M}(M(s,a))$. Similarly, $V^{\pi^{\mathcal{T}}, M^{\mathcal{T}}}(M^{\star,\mathcal{T}}(s^{\mathcal{T}}, a)) = V^{\pi,M}(M^\star(s,a))$. Therefore we obtain that

$$G^{\mathcal{T}}(s^{\mathcal{T}}, a) = G(s,a) \tag{4.6.4}$$

By Lemma 4.4.3 and triangle inequality, we have that

$$\begin{aligned} \frac{1-\gamma}{\gamma}\left|V^{\pi,M} - V^{\pi,\widehat{M}}\right| &\le \mathop{\mathbb{E}}_{S\sim\rho^\pi}\left[\left|G^{\pi,\widehat{M}}(S)\right|\right] && \text{(triangle inequality)} \\ &\le \mathop{\mathbb{E}}_{S\sim\rho^{\pi_{\text{ref}}}}\left[\left|G^{\pi,\widehat{M}}(S)\right|\right] + |\rho^\pi - \rho^{\pi_{\text{ref}}}|_1 \cdot \max_S \left|G^{\pi,\widehat{M}}(S)\right| \\ & && \text{(Holder inequality)} \end{aligned}$$

By Corollary 4.6.15 we have that $|\rho^\pi - \rho^{\pi_{\text{ref}}}|_1 \le \frac{\gamma}{1-\gamma}\mathbb{E}_{S\sim\rho^{\pi_{\text{ref}}}}\left[KL(\pi(S), \pi_{\text{ref}}(S))^{1/2}|S\right] = \frac{\delta\gamma}{1-\gamma}$. Combining this with the equation above, we complete the proof. □

Proof of Proposition 4.6.2.

Proof of Proposition 4.6.2. Let μ be the distribution of the initial state S_0, and let P' and P be the state-to-state transition kernel under policy π and π_{ref}. Let $\bar{G} = (1-\gamma)\sum_{k=0}^{\infty}\gamma^k P^k$ and

$\bar{G}' = (1-\gamma)\sum_{k=0}^{\infty}\gamma^k P'^k$. Under these notations, we can re-write $\rho^{\pi_{\text{ref}}} = \bar{G}\mu$ and $\rho^{\pi} = \bar{G}'\mu$. Moreover, we observe that $\beta^{\pi_{\text{ref}}} = \bar{G}P\bar{G}\mu$.

Let $\delta_1 = (1-\gamma)^{-1}\chi^2_{\bar{G}\mu}(P', P)^{1/2}$ and $\delta_2 = (1-\gamma)^{-1}\chi^2_{\bar{G}P\bar{G}\mu}(P', P)^{1/2}$ by the χ^2 divergence between P' and P, measured with respect to distributions $\bar{G}\mu = \rho^{\pi_{\text{ref}}}$ and $\bar{G}P\bar{G}\mu = \beta^{\pi_{\text{ref}}}$. By Lemma 4.6.4, we have that the χ^2-divergence between the states can be bounded by the χ^2-divergence between the actions in the sense that:

$$\chi^2_{\bar{G}\mu}(P', P)^{1/2} = \chi^2_{\rho^{\pi_{\text{ref}}}}(P', P)^{1/2} \le \mathop{\mathbb{E}}_{S\sim\rho^{\pi_{\text{ref}}}}\left[\chi^2(\pi(\cdot|S), \pi_{\text{ref}}(\cdot|S))\right]^{1/2}$$
$$\chi^2_{\bar{G}P\bar{G}\mu}(P', P)^{1/2} = \chi^2_{\beta^{\pi_{\text{ref}}}}(P', P)^{1/2} \le \mathop{\mathbb{E}}_{S\sim\beta^{\pi_{\text{ref}}}}\left[\chi^2(\pi(\cdot|S), \pi_{\text{ref}}(\cdot|S))\right]^{1/2}$$

Therefore we obtain that $\delta_1 \le (1-\gamma)^{-1}\delta$, $\delta_2 \le (1-\gamma)^{-1}\delta$. Let $f(s) = G^{\pi,\widehat{M}}(s)$. By Lemma 4.6.7, we can control the difference between $\langle\rho^{\pi_{\text{ref}}}, f\rangle$ and $\langle\rho^{\pi}, f\rangle$ by

$$\begin{aligned}\left|\mathop{\mathbb{E}}_{S\sim\rho^{\pi_{\text{ref}}}}\left[G^{\pi,\widehat{M}}(S)\right] - \mathop{\mathbb{E}}_{S\sim\rho^{\pi}}\left[G^{\pi,\widehat{M}}(S)\right]\right| &= |\langle\rho^{\pi_{\text{ref}}}, f\rangle - \langle\rho^{\pi}, f\rangle| \\ &\le \delta_1\langle\bar{G}P\bar{G}\mu, f^2\rangle^{1/2} + \delta_1\delta_2^{1/2}\|f\|_\infty \\ &\le \delta_1\epsilon_2 + \delta_1\delta_2^{1/2}\epsilon_{\max}\end{aligned}$$

It follows that

$$\begin{aligned}\left|V^{\pi,\widehat{M}} - V^{\pi,M}\right| &\le \gamma(1-\gamma)^{-1}\left|\mathop{\mathbb{E}}_{S\sim\rho^{\pi}}\left[G^{\pi,\widehat{M}}(S)\right]\right| \qquad \text{(by Lemma 4.4.3)} \\ &\le \gamma(1-\gamma)^{-1}\left(\left|\mathop{\mathbb{E}}_{S\sim\rho^{\pi_{\text{ref}}}}\left[G^{\pi,\widehat{M}}(S)\right]\right| + \left|\mathop{\mathbb{E}}_{S\sim\rho^{\pi_{\text{ref}}}}\left[G^{\pi,\widehat{M}}(S) - \mathop{\mathbb{E}}_{S\sim\rho^{\pi}}\left[G^{\pi,\widehat{M}}(S)\right]\right]\right|\right) \\ &\le \gamma(1-\gamma)^{-1}\left|\mathop{\mathbb{E}}_{S\sim\rho^{\pi_{\text{ref}}}}\left[G^{\pi,\widehat{M}}(S)\right]\right| + \gamma(1-\gamma)^{-1}\delta_1\epsilon_2 + \gamma(1-\gamma)^{-1}\delta_1\delta_2^{1/2}\epsilon_{\max} \\ &\le (1-\gamma)^{-1}\epsilon_1 + (1-\gamma)^{-2}\delta\epsilon_2 + (1-\gamma)^{-5/2}\delta^{3/2}\epsilon_{\max}\end{aligned}$$

□

Proof of Proposition 4.4.1 and 4.4.2.

Proof of Proposition 4.4.1 and 4.4.2 . By definition of G and the Lipschitzness of $V^{\pi,\widehat{M}}$, we have that $|G^{\pi,\widehat{M}}(s,a)| \le L|\widehat{M}(s,a) - M^\star(s,a)|$. Then, by Lemma 4.4.3 and triangle inequality, we have that

$$\begin{aligned}\left|V^{\pi,\widehat{M}} - V^{\pi,M^\star}\right| &= \kappa\cdot\Big|\mathop{\mathbb{E}}_{\substack{S\sim\rho^{\pi,M}\\A\sim\pi(\cdot|S)}}\left[G^{\pi,\widehat{M}}(S,A)\right]\Big| \le \kappa\mathop{\mathbb{E}}_{\substack{S\sim\rho^{\pi,M}\\A\sim\pi(\cdot|S)}}\left[\left|G^{\pi,\widehat{M}}(S,A)\right|\right] \\ &\le \kappa\mathop{\mathbb{E}}_{\substack{S\sim\rho^{\pi}\\A\sim\pi(\cdot|S)}}\left[\|\widehat{M}(S,A) - M^\star(S,A)\|\right]. \qquad (4.6.5)\end{aligned}$$

Next we prove the main part of the proposition. Thus we proved Proposition 4.4.1. Note that for any distribution ρ and ρ' and function f, we have $\mathbb{E}_{S\sim\rho} f(S) = \mathbb{E}_{S\sim\rho'} f(S) + \langle \rho - \rho', f\rangle \le \mathbb{E}_{S\sim\rho'} f(S) + \|\rho - \rho'\|_1 \|f\|_\infty$. Thus applying this inequality with $f(S) = \mathbb{E}_{A\sim\pi(\cdot|S)}\left[\|\widehat{M}(S,A) - M^\star(S,A)\|\right]$, we obtain that

$$\begin{aligned}\mathop{\mathbb{E}}_{\substack{S\sim\rho^\pi\\A\sim\pi(\cdot|S)}}\left[\|\widehat{M}(S,A) - M^\star(S,A)\|\right] &\le \mathop{\mathbb{E}}_{\substack{S\sim\rho^{\pi_{\text{ref}}}\\A\sim\pi(\cdot|S)}}\left[\|\widehat{M}(S,A) - M^\star(S,A)\|\right]\\ &\quad + \|\rho^{\pi_{\text{ref}}} - \rho\|_1 \max_S \mathop{\mathbb{E}}_{A\sim\pi(\cdot|S)}\left[\|\widehat{M}(S,A) - M^\star(S,A)\|\right]\\ &\le \mathop{\mathbb{E}}_{\substack{S\sim\rho^{\pi_{\text{ref}}}\\A\sim\pi(\cdot|S)}}\left[\|\widehat{M}(S,A) - M^\star(S,A)\|\right] + 2\delta\kappa B \qquad (4.6.6)\end{aligned}$$

where the last inequality uses the inequalities (see Corollary 4.6.15) that $\|\rho^\pi - \rho^{\pi_{\text{ref}}}\|_1 \le \frac{\gamma}{1-\gamma}\mathbb{E}_{S\sim\rho^{\pi_{\text{ref}}}}\left[KL(\pi(S), \pi_{\text{ref}}(S))^{1/2}|S\right] = \delta\kappa$ and that $\|\widehat{M}(S,A) - M^\star(S,A)\| \le 2B$. Combining (4.6.6) and (4.6.5) we complete the proof of Proposition 4.4.2. □

χ^2-Divergence Based Inequalities

Lemma 4.6.4. *Let S be a random variable over the domain $\mathcal{S}$. Let π and π' be two policies and and $A \sim \pi(\cdot \mid S)$ and $A' \sim \pi'(\cdot \mid S)$. Let $Y \sim M(\cdot \mid S, A)$ and $Y' \sim M(\cdot \mid S, A')$ be the random variables for the next states under two policies. Then,*

$$\mathbb{E}\left[\chi^2(Y|S, Y'|S)\right] \le \mathbb{E}\left[\chi^2(A|S, A'|S)\right]$$

Proof. By definition, we have that $Y|S = s, A = a$ has the same density as $Y'|S = s, A' = a$ for any a and s. Therefore by Theorem 4.6.12 (setting X, X', Y, Y' in Theorem 4.6.12 by $A|S = s$, $A'|S = s$, $Y|S = s$, $Y'|S = s$ respectively), we have

$$\chi^2(Y|S = s, Y'|S = s) \le \chi^2(A|S = s, A'|S = s)$$

Taking expectation over the randomness of S we complete the proof.

□

Properties of Markov Processes. In this subsection, we consider bounded the difference of the distributions induced by two markov process starting from the same initial distributions μ. Let P, P' be two transition kernels. Let $G = \sum_{k=0}^\infty \gamma^k P^k$ and $\bar{G} = (1-\gamma)G$. Define G' and $\bar{G}'$ similarly. Therefore we have that $\bar{G}\mu$ is the discounted distribution of states visited by the markov process starting from distribution μ. In other words, if μ is the distribution of S_0, and P is the transition kernel induced by some policy π, then $\bar{G}\mu = \rho^\pi$.

First of all, let $\Delta = \gamma(P' - P)$ and we note that with simple algebraic manipulation,

$$\bar{G}' - \bar{G} = (1-\gamma)^{-1}\bar{G}'\Delta\bar{G} \qquad (4.6.7)$$

Let f be some function. We will mostly interested in the difference between $\mathbb{E}_{S\sim\bar{G}\mu}[f]$ and $\mathbb{E}_{S\sim\bar{G}'\mu}[f]$, which can be rewritten as $\langle(\bar{G}'-G)\mu, f\rangle$. We will bound this quantity from above by some divergence measure between P' and P.

We start off with a simple lemma that controls the form $\langle p-q, f\rangle$ by the χ^2 divergence between p and q. With this lemma we can reduce our problem of bounding $\langle(\bar{G}'-G)\mu, f\rangle$ to characterizing the χ^2 divergence between $\bar{G}'\mu$ and $\bar{G}\mu$.

Lemma 4.6.5. *Let p and q be probability distributions. Then we have*

$$\langle q-p, f\rangle^2 \le \chi^2(q,p)\cdot\langle p, f^2\rangle$$

Proof. By Cauchy-Schwartz inequality, we have

$$\langle q-p, f\rangle^2 \le \left(\int \frac{(q(x)-p(x))^2}{p(x)}dx\right)\left(\int p(x)f(x)^2\right) = \chi^2(q,p)\cdot\langle p, f^2\rangle$$

□

The following Lemma is a refinement of the lemma above. It deals with the distributions p and q with the special structure $p = WP'\mu$ and $q = WP\mu$.

Lemma 4.6.6. *Let W, P', P be transition kernels and μ be a distribution. Then,*

$$\langle W(P'-P)\mu, f\rangle^2 \le \chi^2_\mu(P', P)\langle WP\mu, f^2\rangle$$

where $\chi^2_\mu(P', P)$ is a divergence between transitions defined in Definition 4.6.11.

Proof. By Lemma 4.6.5 with $p = WP\mu$ and $q = WP'\mu$, we conclude that

$$\langle W(P'-P)\mu, f\rangle^2 \le \chi^2(q,p)\cdot\langle p, f^2\rangle \le \chi^2(WP'\mu, WP\mu)\langle WP\mu, f^2\rangle$$

By Theorem 4.6.12 and Theorem 4.6.13 we have that $\chi^2(WP'\mu, WP\mu) \le \chi^2(P'\mu, P\mu) \le \chi^2_\mu(P', P)$, plugging this into the equation above we complete the proof. □

Now we are ready to state the main result of this subsection.

Lemma 4.6.7. *Let $\bar{G}, \bar{G}', P', P, f$ as defined in the beginning of this section. Let $\delta_1 = (1-\gamma)^{-1}\chi^2_{\bar{G}\mu}(P', P)^{1/2}$ and $\delta_2 = (1-\gamma)^{-1}\chi^2_{\bar{G}P\bar{G}\mu}(P', P)^{1/2}$. Then,*

$$\left|\langle\bar{G}'\mu, f\rangle - \langle\bar{G}\mu, f\rangle\right| \le \delta_1\|f\|_\infty \tag{4.6.8}$$
$$\left|\langle\bar{G}'\mu, f\rangle - \langle\bar{G}\mu, f\rangle\right| \le \delta_1\langle\bar{G}P\bar{G}\mu, f^2\rangle^{1/2} + \delta_1\delta_2^{1/2}\|f\|_\infty$$

Proof. Recall by equation (4.6.7), we have

$$\langle(\bar{G}'-\bar{G})\mu, f\rangle = (1-\gamma)^{-1}\langle\bar{G}'\Delta\bar{G}\mu, f\rangle \tag{4.6.9}$$

By Lemma 4.6.6,

$$\langle \bar{G}'\Delta\bar{G}\mu, f\rangle \le \chi^2_{\bar{G}\mu}(P', P)^{1/2}\langle \bar{G}'P\bar{G}\mu, f^2\rangle^{1/2} \tag{4.6.10}$$

By Holder inequality and the fact that $\|\bar{G}\|_{1\to1} = 1$, $\|\bar{G}'\|_{1\to1} = 1$ and $\|P\|_{1\to1} = 1$, we have

$$\begin{aligned}\langle \bar{G}'\Delta\bar{G}\mu, f\rangle &\le \chi^2_{\bar{G}\mu}(P', P)^{1/2}\langle \bar{G}'P\bar{G}\mu, f^2\rangle^{1/2}\\ &\le \chi^2_{\bar{G}\mu}(P', P)^{1/2}\|\bar{G}'P\bar{G}\mu\|_1^{1/2}\|f^2\|_\infty^{1/2}\\ &\le \chi^2_{\bar{G}\mu}(P', P)^{1/2}\|f\|_\infty \quad (\text{by } \|\bar{G}'P\bar{G}\mu\|_1 \le \|\bar{G}'\|_{1\to1}\|P\|_{1\to1}\|\bar{G}\|_{1\to1}\|\mu\|_1 \le 1)\\ &\le (1-\gamma)\delta_1\|f\|_\infty \end{aligned} \tag{4.6.11}$$

Combining equation (4.6.9) and (4.6.11) we complete the proof of equation (4.6.8).

Next we bound $\langle \bar{G}'P\bar{G}\mu, f^2\rangle^{1/2}$ in a more refined manner. By equation (4.6.7), we have

$$\begin{aligned}\langle \bar{G}'P\bar{G}\mu, f^2\rangle^{1/2} &= \left(\langle \bar{G}P\bar{G}\mu, f^2\rangle + \frac{1}{1-\gamma}\langle \bar{G}'\Delta\bar{G}P\bar{G}\mu, f^2\rangle\right)^{1/2}\\ &\le \langle \bar{G}P\bar{G}\mu, f^2\rangle^{1/2} + (1-\gamma)^{-1/2}\langle \bar{G}'\Delta\bar{G}P\bar{G}\mu, f^2\rangle^{1/2}\end{aligned} \tag{4.6.12}$$

By Lemma 4.6.6 again, we have that

$$\langle \bar{G}'\Delta\bar{G}P\bar{G}, f^2\rangle^2 \le \chi^2_{\bar{G}P\bar{G}\mu}(P', P)\langle \bar{G}'P\bar{G}P\bar{G}\mu, f^4\rangle \tag{4.6.13}$$

By Holder inequality and the fact that $\|\bar{G}\|_{1\to1} = 1$, $\|\bar{G}'\|_{1\to1} = 1$ and $\|P\|_{1\to1} = 1$, we have

$$\langle \bar{G}'P\bar{G}P\bar{G}\mu, f^4\rangle \le \|\bar{G}'P\bar{G}P\bar{G}\mu\|_1\|f^4\|_\infty \le \|f\|_\infty^4 \tag{4.6.14}$$

Combining equation (4.6.12), (4.6.14) gives

$$(1-\gamma)^{-1/2}\langle \bar{G}'\Delta\bar{G}P\bar{G}, f^2\rangle^{1/2} \le ((1-\gamma)^{-1}\chi^2_{\bar{G}P\bar{G}\mu}(P', P)^{1/2})^{1/2}\|f\|_\infty = \delta_2^{1/2}\|f\|_\infty \tag{4.6.15}$$

Then, combining equation (4.6.9), (4.6.10), (4.6.15), we have

$$\begin{aligned}\langle (\bar{G}'-\bar{G})\mu, f\rangle &= (1-\gamma)^{-1}\chi^2_{\bar{G}\mu}(P', P)^{1/2}\langle \bar{G}'P\bar{G}\mu, f^2\rangle^{1/2} \quad (\text{by equation (4.6.9), (4.6.10)})\\ &= \delta_1\langle \bar{G}'P\bar{G}\mu, f^2\rangle^{1/2}\\ &\le \delta_1\langle \bar{G}P\bar{G}\mu, f^2\rangle^{1/2} + \delta_1(1-\gamma)^{-1/2}\langle \bar{G}'\Delta\bar{G}P\bar{G}\mu, f^2\rangle^{1/2}\\ &\qquad (\text{by equation (4.6.12)})\\ &\le \delta_1\langle \bar{G}P\bar{G}\mu, f^2\rangle^{1/2} + \delta_1\delta_2^{1/2}\|f\|_\infty \qquad (\text{by equation (4.6.15)})\end{aligned}$$

□

The following Lemma is a stronger extension of Lemma 4.6.7, which can be used to future improve Proposition 4.6.2, and may be of other potential independent interests. We state it for completeness.

Lemma 4.6.8. *Let $\bar{G}, \bar{G}', P', P, f$ as defined in the beginning of this section. Let $d_k = (\bar{G}P)^k\bar{G}\mu$ and $\delta_k = (1-\gamma)^{-1}\chi^2_{d_{k-1}}(P', P)^{1/2}$, then we have that for any K,*

$$\begin{aligned}\left|\langle \bar{G}'\mu, f\rangle - \langle \bar{G}\mu, f\rangle\right| \le \delta_1\langle d_1, f^2\rangle^{-1/2} + \delta_1\delta_2^{1/2}\langle d_2, f^4\rangle^{-1/4} \\ + \delta_1\ldots\delta_K^{2^{-K+1}}\langle d_k, f^{2^K}\rangle^{2^{-K}} + \delta_1\ldots\delta_K^{2^{-K+1}}\|f\|_\infty\end{aligned}$$

Proof. We first use induction to prove that:

$$\begin{aligned}|\langle(\bar{G}' - \bar{G})\mu, f\rangle| \le \sum_{k=1}^{K}\left(\prod_{0\le s\le k-1}\delta_{s+1}^{2^{-s}}\right)\langle d_k, f^{2^k}\rangle^{2^{-k}} \\ + \left(\prod_{0\le s\le K-1}\delta_{s+1}^{2^{-s}}\right)\langle \bar{G}'\Delta(\bar{G}P)^K\bar{G}\mu, f^{2^K}\rangle^{2^{-K}} \quad (4.6.16)\end{aligned}$$

By the first equation of Lemma 4.6.7, we got the case for $K = 1$. Assuming we have proved the case for K, then applying

$$\begin{aligned}\langle \bar{G}'\Delta(\bar{G}P)^K\bar{G}\mu, f^{2^K}\rangle &= \langle \bar{G}\Delta(\bar{G}P)^K\bar{G}\mu, f^{2^K}\rangle + (1-\gamma)^{-1}\langle \bar{G}\Delta(\bar{G}P)^{K+1}\bar{G}\mu, f^{2^K}\rangle \quad (4.6.17)\\ &\le \langle \bar{G}\Delta(\bar{G}P)^K\bar{G}\mu, f^{2^K}\rangle \\ &\quad + (1-\gamma)^{-1}\chi^2_{d_K}(P', P)^{1/2}\langle \bar{G}'\Delta(\bar{G}P)^{K+1}\bar{G}\mu, f^{2^{K+1}}\rangle^{1/2} \\ &\le \langle \bar{G}\Delta(\bar{G}P)^K\bar{G}\mu, f^{2^K}\rangle + \delta_{K+1}\langle \bar{G}'\Delta(\bar{G}P)^{K+1}\bar{G}\mu, f^{2^{K+1}}\rangle^{1/2}\end{aligned}$$

By Cauchy-Schwartz inequality, we obtain that

$$\begin{aligned}\langle \bar{G}'\Delta(\bar{G}P)^K\bar{G}\mu, f^{2^K}\rangle^{2^{-K}} \le \langle \bar{G}\Delta(\bar{G}P)^K\bar{G}\mu, f^{2^K}\rangle^{2^{-K}} + \\ \delta_{K+1}\langle \bar{G}'\Delta(\bar{G}P)^{K+1}\bar{G}\mu, f^{2^{K+1}}\rangle^{2^{-K-1}}\end{aligned}$$

Plugging the equation above into equation (4.6.16), we provide the induction hypothesis for the case with $K + 1$.

Now applying $\langle \bar{G}'\Delta(\bar{G}P)^K\bar{G}\mu, f^{2^K}\rangle^{2^{-K}} \le \|f\|_\infty$ with equation (4.6.16) we complete the proof.

□

Toolbox

Definition 4.6.9 (χ^2 distance, c.f. [124, 27])**.** *The Neyman χ^2 distance between two distributions p and q is defined as*

$$\chi^2(p, q) \triangleq \int \frac{(p(x) - q(x))^2}{q(x)}dx = \int \frac{p(x)^2}{q(x)}dx - 1$$

For notational simplicity, suppose two random variables X and Y has distributions p_X and p_Y, we often write $\chi^2(X, Y)$ as a simplification for $\chi^2(p_X, p_Y)$.

Theorem 4.6.10 ([149])**.** *The Kullback-Leibler (KL) divergence between two distributions p, q is bounded from above by the χ^2 distance:*

$$KL(p, q) \leq \chi^2(p, q)$$

Proof. Since log is a concave function, by Jensen inequality we have

$$\begin{aligned} KL(p, q) &= \int p(x) \log \frac{p(x)}{q(x)} dx \leq \log \int p(x) \cdot \frac{p(x)}{q(x)} dx \\ &= \log(\chi^2(p, q) + 1) \leq \chi^2(p, q) \end{aligned}$$

□

Definition 4.6.11 (χ^2 distance between transitions)**.** *Given two transition kernels P, P'. For any distribution μ, we define $\chi^2_\mu(P', P)$ as:*

$$\chi^2_\mu(P', P) \triangleq \int \mu(x) \chi^2(P'(\cdot|X = x), P(\cdot|X = x)) dx$$

Theorem 4.6.12. *Suppose random variables (X, Y) and (X', Y') satisfy that $p_{Y|X} = p_{Y'|X'}$. Then*

$$\chi^2(Y, Y') \leq \chi^2(X, X')$$

Or equivalently, for any transition kernel P and distribution μ, μ', we have

$$\chi^2(P\mu, P\mu') \leq \chi^2(\mu, \mu')$$

Proof. Denote $p_{Y|X}(y \mid x) = p_{Y'|X'}(y \mid x)$ by $p(y \mid x)$, and we rewrite p_X as p and $p_{X'}$ as p'. By Cauchy-Schwarz inequality, we have:

$$\begin{aligned} p_Y(y)^2 = \left(\int p(y|x) p(x) dx \right)^2 &\leq \left(\int p(y|x) p'(x) dx \right) \left(\int p(y|x) \frac{p(x)^2}{p'(x)} dx \right) \\ &= p_{Y'}(y) \left(\int p(y|x) \frac{p(x)^2}{p'(x)} dx \right) \end{aligned} \tag{4.6.18}$$

It follows that

$$\chi^2(Y, Y') = \int \frac{p_Y(y)^2}{p_{Y'}(y)} dy - 1 \leq \int dy \int p(y|x) \frac{p(x)^2}{p'(x)} dx - 1 = \chi^2(X, X')$$

□

Theorem 4.6.13. *Let X, Y, Y' are three random variables. Then,*

$$\chi^2(Y, Y') \le \mathbb{E}\left[\chi^2(Y|X, Y'|X)\right]$$

We note that the expectation on the right hand side is over the randomness of X.[9] *As a direct corollary, we have for transition kernel P' and P and distribution μ,*

$$\chi^2(P'\mu, P\mu) \le \chi^2_\mu(P', P)$$

Proof. We denote $p_{Y'|X}(y|x)$ by $p'(y \mid x)$ and $p_{Y|X}(y|x)$ by $p(y|x)$, and let $p(x)$ be a simplification for $p_X(x)$. We have by Cauchy-Schwarz,

$$\frac{p_Y(y)^2}{p_{Y'}(y)} = \frac{\left(\int p(y|x)p(x)dx\right)^2}{\int p'(y \mid x)p(x)dx} \le \int \frac{p(y|x)^2}{p'(y|x)} p(x)dx$$

It follows that

$$\chi^2(Y, Y') = \int \frac{p_Y(y)^2}{p_{Y'}(y)} dy - 1 \le \int \frac{p(y|x)^2}{p'(y|x)} p(x)dxdy - 1 = \mathbb{E}\left[\chi^2(Y|X, Y'|X) \mid X\right]$$

□

Claim 4.6.14. *Let μ be a distribution over the state space $\mathcal{S}$. Let P and P' be two transition kernels. $G = \sum_{k=0}^{\infty}(\gamma P)^k = (\mathrm{Id} - \gamma P)^{-1}$ and $G' = \sum_{k=0}^{\infty}(\gamma P')^k = (\mathrm{Id} - \gamma P')^{-1}$. Let $d = (1-\gamma)G\mu$ and $d' = (1-\gamma)G'\mu$ be the discounted distribution starting from μ induced by the transition kernels G and G'. Then,*

$$|d - d'|_1 \le \frac{1}{1-\gamma}|\Delta d|_1$$

Moreover, let $\gamma(P' - P) = \Delta$. Then, we have

$$G' - G = \sum_{k=1}^{\infty}(G\Delta)^k G$$

Proof. With algebraic manipulation, we obtain,

$$\begin{aligned} G' - G &= (\mathrm{Id} - \gamma P')^{-1}((\mathrm{Id} - \gamma P) - (\mathrm{Id} - \gamma P')(\mathrm{Id} - \gamma P)^{-1} \\ &= G'\Delta G \end{aligned} \tag{4.6.19}$$

It follows that

$$\begin{aligned} |d - d'|_1 &= (1-\gamma)|G'\Delta G\mu|_1 \le |\Delta G\mu|_1 && (\text{since } (1-\gamma)|G'|_{1\to 1} \le 1) \\ &= \frac{1}{1-\gamma}|\Delta d|_1 \end{aligned}$$

[9] Observe $\chi^2(Y|X, Y'|X)$ deterministically depends on X.

Replacing G' in the RHS of the equation (4.6.19) by $G' = G + G'\Delta G$, and doing this recursively gives

$$G' - G = \sum_{k=1}^{\infty} (G\Delta)^k G$$

□

Corollary 4.6.15. *Let π and π' be two policies and let ρ^π be defined as in Definition 4.2.1. Then,*

$$|\rho^\pi - \rho'|_1 \le \frac{\gamma}{1-\gamma} \mathop{\mathbb{E}}_{S\sim\rho^\pi} \left[KL(\pi(S), \pi'(S))^{1/2} \mid S \right]$$

Proof. Let P and P' be the state-state transition matrix under policy π and π' and $\Delta = \gamma(P' - P)$ By Claim 4.6.14, we have that

$$\begin{aligned} |\rho^\pi - \rho^{\pi'}|_1 \le \frac{1}{1-\gamma} |\Delta\rho^\pi|_1 &= \frac{\gamma}{1-\gamma} \mathop{\mathbb{E}}_{S\sim\rho^\pi} \left[|p_{M^\star(S,\pi(S))|S} - p_{M^\star(S,\pi'(S))|S}|_1 \right] \\ &\le \frac{\gamma}{1-\gamma} \mathop{\mathbb{E}}_{S\sim\rho^\pi} \left[|p_{\pi(S)|S} - p_{\pi'(S)|S}|_1 \right] \\ &\le \frac{\gamma}{1-\gamma} \mathop{\mathbb{E}}_{S\sim\rho^\pi} \left[KL(\pi(S), \pi'(S))^{1/2} \mid S \right] \quad \text{(by Pinkser's inequality)} \end{aligned}$$

□

Implementation Details

Environment Setup. We benchmark our algorithm on six tasks based on physics simulator Mujoco [174]. We use rllab's implementation [42] [10] to interact with Mujoco. All the environments we use have a maximum horizon of 500 steps. We remove all contact information from observation. To compute reward from states, we put the velocity of center of mass into the states.

Network Architecture and Model Learning. We use the same reward function as in rllab, except that all the coefficients C_{contact} in front of the contact force s are set to 0 in our case. We refer the readers to [42] Supp Material 1.2 for more details. All actions are projected to the action space by clipping. We normalize all observations by $s' = \frac{s-\mu}{\sigma}$ where $\mu, \sigma \in \mathbb{R}^{d_{\text{observation}}}$ are computed from all observations we collect from the real environment. Note that μ, σ may change as we collect new data. Our policy will always produce an action a in $[-1, 1]^{d_{\text{action}}}$ and the action a', which is fed into the environment, is scaled linearly by $a' = \frac{1-a}{2} a_{\min} + \frac{1+a}{2} a_{\max}$, where $a_{\min}, a_{\max}$ are the min or max values allowed at each entry.

[10] commit b3a2899 in `https://github.com/rll/rllab/`

Table 4.1: TRPO Hyperparameters.

Hyperparameters	Values
batch size	4000
max KL divergence	0.01
discount γ	0.99
GAE λ	0.95
CG iterations	10
CG damping	0.1

SLBO Details. The dynamical model is represented by a feed-forward neural network with two hidden layers, each of which contains 500 hidden units. The activation function at each layer is ReLU. We use Adam to optimize the loss function with learning rate 10^{-3} and L_2 regularization 10^{-5}. The network does not predict the next state directly; instead, it predicts the normalized difference of $s_{t+1} - s_t$. The normalization scheme and statistics are the same as those of observations: We maintain μ, σ from collected data in the real environment and may change them as we collect more, and the normalized difference is $\frac{s_{t+1}-s_t-\sigma}{\mu}$.

The policy network is a feed-forward network with two hidden layers, each of which contains 32 hidden units. The policy network uses tanh as activation function and outputs a Gaussian distribution $\mathcal{N}(\mu(s), \sigma^2)$ where σ a state-independent trainable vector.

During our evaluation, we use $H = 2$ for multi-step model training and the batch size is given by $\frac{256}{H} = 128$, i.e., we enforce the model to see 256 transitions at each batch.

We run our algorithm $n_{\text{outer}} = 100$ iterations. We collect $n_{\text{train}} = 10000$ steps of real samples from the environment at the start of each iteration using current policy with Ornstein-Uhlunbeck noise (with parameter $\theta = 0.15, \sigma = 0.3$) for better exploration. At each iteration, we optimize dynamics model and policy alternatively for $n_{\text{inner}} = 20$ times. At each iteration, we optimize dynamics model for $n_{\text{model}} = 100$ times and optimize policy for $n_{\text{policy}} = 40$ times.

Baselines: TRPO. TRPO hyperparameters are listed at Table 4.1, which are the same as OpenAI Baselines' implementation. These hyperparameters are fixed for all experiments where TRPO is used, including ours, MB-TRPO and MF-TRPO. We do not tune these hyperparameters. We also normalize observations as our algorithm and OpenAI Baselines do.

We use a neural network as the value function to reduce variance, which has 2 hidden layers of units 64 and uses tanh as activation functions. We use Generalized Advantage Estimator (GAE) [151] to estimate advantages. Both TRPO used in our algorithm and that in model-free algorithm share the same set of hyperparameters.

SAC. For fair comparison, we do not use a large policy network (2 hidden layers, one of which has 256 hidden units) as the authors suggest, but use exactly the same policy network as ours. All other hyperparameters are kept the same as the authors'. Note that Q network and value network have 256 hidden units at each hidden layers, which is more than TRPO's. We refer the readers to [59] Section D for more details.

MB-TRPO. Model-Based TRPO (MB-TRPO) is similar to our algorithm SLBO but does not optimize model and policy alternatively during one iteration. We do not tune the hyperparameter n_{model} since any number beyond a certain threshold would bring similar results. For n_{policy} we try $\{100, 200, 400, 800\}$ on Ant and find $n_{\text{policy}} = 200$ works best in Ant so we use it for all other environments. Note that when Algo 3 uses 800 Adam updates (per outer iteration), it has the same amount of updates (per outer iteration) as in Algo 4. As suggested by Section 4.5, we use 0.005 as the coefficient of entropy bonus for all experiments.

Algorithm 4 Model-Based Trust Region Policy Optimization (MB-TRPO)

```
1: initialize model network parameters $\phi$ and policy network parameters $\theta$
2: initialize dataset $\mathcal{D} \leftarrow \emptyset$
3: for $n_{\text{outer}}$ iterations do
4:     $\mathcal{D} \leftarrow \mathcal{D} \cup \{$ collect $n_{\text{collect}}$ samples from real environment using $\pi_\theta$ with noises $\}$
5:     for $n_{\text{model}}$ iterations do
6:         optimize (4.5.1) over $\phi$ with sampled data from $\mathcal{D}$ by one step of Adam
7:     end for
8:     for $n_{\text{policy}}$ iterations do
9:         $\mathcal{D}' \leftarrow \{$ collect $n_{\text{trpo}}$ samples using $\widehat{M}_\phi$ as dynamics $\}$
10:        optimize $\pi_\theta$ by running TRPO on $\mathcal{D}'$
11:    end for
12: end for
```

SLBO. We tune multi-step model training parameter $H \in \{1, 2, 4, 8\}$, entropy regularization coefficient $\lambda \in \{0, 0.001, 0.003, 0.005\}$ and $n_{\text{policy}} \in \{10, 20, 40\}$ on Ant and find $H = 2, \lambda = 0.005, n_{\text{policy}} = 40$ work best, then we fix them in all environments, though environment-specific hyperparameters may work better. The other hyperparameters, including $n_{\text{inner}}, n_{\text{model}}$ and network architecture, are never tuned. We observe that at the first several iterations, the policy overfits to the learnt model so a reduction of n_{policy} at the beginning can further speed up convergence but we omit this for simplicity.

The most important hyperparameters we found are n_{policy} and the coefficient in front of the entropy regularizer λ. It seems that once n_{model} is large enough we don't see any significant changes. We did have a held-out set for model prediction (with the same distribution as the training set) and found out the model doesn't overfit much. As mentioned in 4.6, we also found out normalizing the state helped a lot since the raw entries in the state have different magnitudes; if we don't normalize them, the loss will be dominated by the loss of some large entries.

Ablation Study: Multi-step model training. We compare multi-step model training with single-step model training and the results are shown on Figure 4.2. Note that $H = 1$ means we use single-step model training. We observe that small H (e.g., 2 or 4) can be beneficial, but larger H (e.g., 8) can hurt. We hypothesize that smaller H can help the model learn the uncertainty in the input and address the error-propagation issue to some extent. [133] uses an auto-regressive recurrent model to predict a multi-step loss on a trajectory, which is closely related to ours. However, theirs differs from ours in the sense that they do not use the predicted output x_{t+1} as the input for the prediction of x_{t+2}, and so on and so forth.

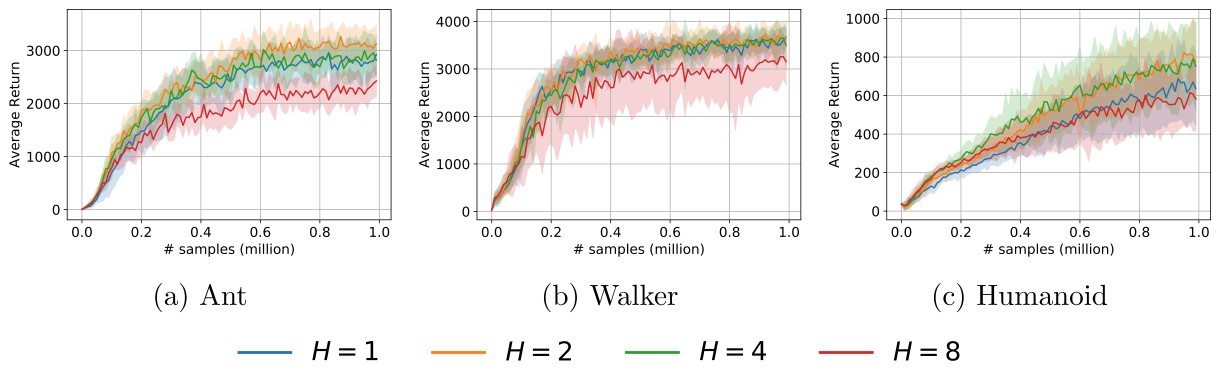
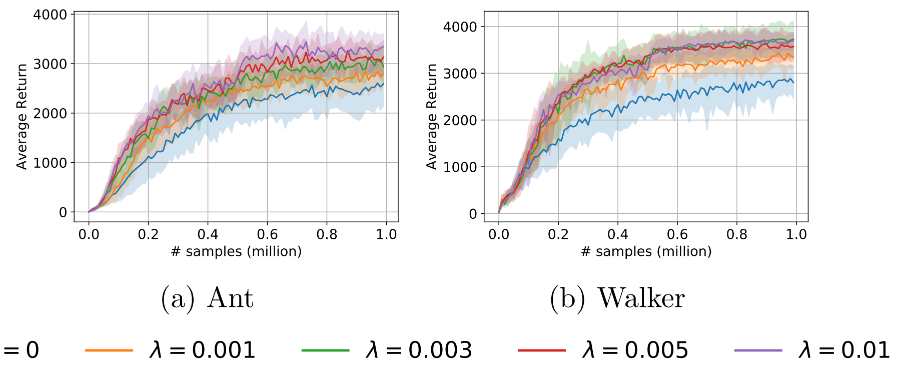


Figure 4.2: Ablation study on multi-step model training. All the experiments are average over 10 random seeds. The x-axis shows the total amount of real samples from the environment. The y-axis shows the averaged return from execution of our learned policy. The solid line is the mean of the total rewards from each seed. The shaded area is one-standard deviation.

Entropy regularization. Figure 4.3 shows that entropy reguarization can improve both sample efficiency and final performance. More entropy regularization leads to better sample efficiency and higher total rewards. We observe that in the late iterations of training, entropy regularization may hurt the performance thus we stop using entropy regularization in the second half of training.

SLBO with 4M training steps. Figure 4.4 shows that SLBO is superior to SAC and MF-TRPO in Swimmer, Half Cheetah, Walker and Humanoid when 4 million samples or fewer samples are allowed. For Ant environment , although SLBO with less than one million samples reaches the performance of MF-TRPO with 8 million samples, SAC's performance surpasses SLBO after 2 million steps of training. Since model-free TRPO almost stops improving after 8M steps and our algorithms uses TRPO for optimizing the estimated environment, we don't expect SLBO can significantly outperform the reward of TRPO at 8M steps. The result shows that SLBO is also satisfactory in terms of asymptotic convergence (compared to TRPO.) It also indicates a better planner or policy optimizer instead of TRPO might be necessary to further improve the performance.



(a) Ant (b) Walker

Figure 4.3: Ablation study on entropy regularization. λ is the coefficient of entropy regularization in the TRPO's objective. All the experiments are averaged over 10 random seeds. The x-axis shows the total amount of real samples from the environment. The y-axis shows the averaged return from execution of our learned policy. The solid line is the mean of the total rewards from each seed. The shaded area is one-standard deviation.

Sample Complexity Bounds

In this section, we extend Theorem 4.3.1 to a final sample complexity result. For simplicity, let $L^{\pi,M}_{\pi_{\text{ref}},\delta} = V^{\pi,M} - \mathcal{D}_{\pi_k,\delta}(M,\pi)$ be the lower bound of $V^{\pi,M^\star}$. We omit the subscript δ when it's clear from contexts. When $\mathcal{D}$ satisfies (R1), we have that,

$$V^{\pi,M^\star} \geq L^{\pi,M}_{\pi_{\text{ref}},\delta} \qquad \forall \pi \text{ s.t. } d(\pi,\pi_{\text{ref}}) \leq \delta \tag{4.6.20}$$

When $\mathcal{D}$ satisfies (R3), we use $\widehat{L}^{\pi,M}_{\pi_{\text{ref}},\delta}$ to denotes its empirical estimates. Namely, we replace the expectation in equation (R3) by empirical samples $\tau^{(1)},\dots,\tau^{(n)}$. In other words, we optimize

$$\pi_{k+1}, M_{k+1} = \arg\max_{\pi\in\Pi,\ M\in\mathcal{M}} \widehat{L}^{\pi,M}_{\pi_{\text{ref}},\delta} = \quad V^{\pi,M} - \frac{1}{n}\sum_{i=1}^{n} f(\widehat{M},\pi,\tau^{(i)}) \tag{4.6.21}$$

instead of equation (4.3.3).

Let p be the total number of parameters in the policy and model parameterization. We assume that we have a discrepancy bound $\mathcal{D}_{\pi_{\text{ref}}}(\pi, M)$ satisfying (R3) with a function f that is bounded with $[-B_f, B_f]$ and that is L_f-Lipschitz in the parameters of π and M. That is, suppose π is parameterized by θ and M is parameterized by ϕ, then we require $|f(M_\phi,\phi_\theta,\tau) - f(M_{\phi'},\phi_{\theta'},\tau)| \leq L_f(\|\phi-\phi'\|_2^2 + \|\theta-\theta'\|^2)$ for all τ, $\theta,\theta',\phi,\phi'$. We note that L_f is likely to be exponential in dimension due to the recursive nature of the problem, but our bounds only depends on its logarithm. We also restrict our attention to parameters in an Euclidean ball $\{\theta : \|\theta\|_2 \leq B\}$ and $\{\phi : \|\phi\|_2 \leq B\}$. Our bounds will be logarithmic in B.

We need the following definition of approximate local maximum since with sampling error we cannot hope to converge to the exact local maximum.

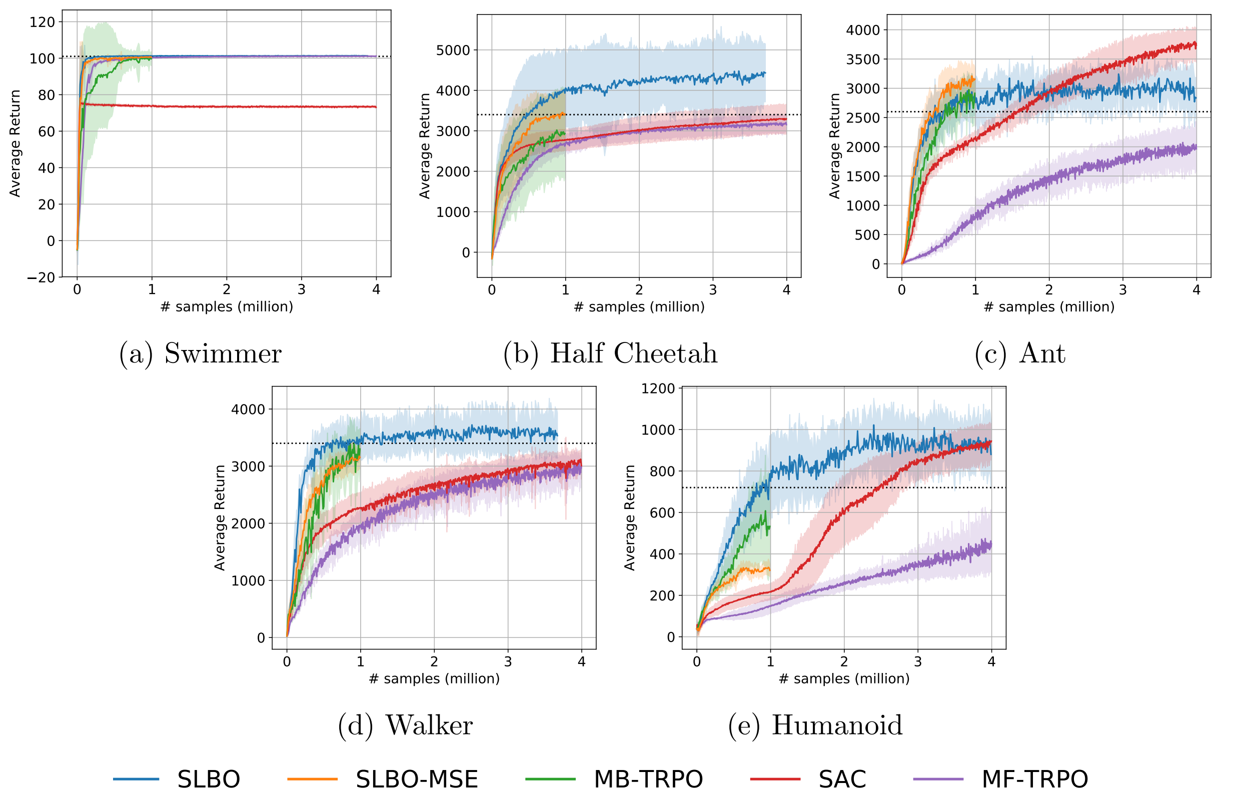


Figure 4.4: Comparison among SLBO (ours), SLBO with squared ℓ^2 model loss (SLBO-MSE), vanilla model-based TRPO (MB-TRPO), model-free TRPO (MF-TRPO), and Soft Actor-Critic (SAC) with more samples than in Figure 4.1. SLBO, SAC, MF-TRPO are trained with 4 million real samples. We average the results over 10 different random seeds, where the solid lines indicate the mean and shaded areas indicate one standard deviation. The dotted reference lines are the total rewards of MF-TRPO after 8 million steps.

Definition 4.6.16. *We say π is a (δ, ϵ)-local maximum of $V^{\pi,M^\star}$ with respect to the constraint set Π and metric d, if for any $\pi' \in \Pi$ with $d(\pi, \pi') \le \delta$, we have $V^{\pi,M^\star} \ge V^{\pi',M^\star} - \epsilon$.*

We show a sample complexity bounds that scales linearly in p and logarithmically in L_f, B and B_f.

Theorem 4.6.17. *Let $\epsilon > 0$. In the setting of Theorem 4.3.1, under the additional assumptions above, suppose we use $n = O(B_f p \log(BL_f/\epsilon)/\epsilon^2)$ trajectories to estimate the discrepancy bound in Algorithm 2. Then, for any t, if π_t is not a (δ, ϵ)-local maximum, then the total reward will increase in the next step: with high probability,*

$$V^{\pi_{t+1},M^\star} \ge V^{\pi_t,M^\star} + \epsilon/2 \tag{4.6.22}$$

As a direct consequence, suppose the maximum possible total reward is B_R and the initial total reward is 0, then for some $T = O(B_R/\epsilon)$, we have that π_T is a (δ, ϵ)-local maximum of the $V^{\pi,M^\star}$.

Proof. By Hoeffiding's inequality, we have for fix π and $\widehat{M}$, with probability $1 - n^{O(1)}$ over the randomness of $\tau^{(1)}, \dots, \tau^{(n)}$,

$$\left| \frac{1}{n} \sum_{i=1}^{n} f(\widehat{M}, \pi, \tau^{(i)}) - \mathop{\mathbb{E}}_{\tau \sim \pi_{\mathrm{ref}}, M^\star} [f(\widehat{M}, \pi, \tau)] \right| \leq 4\sqrt{\frac{B_f \log n}{n}}. \tag{4.6.23}$$

In more succinct notations, we have $|\widehat{\mathcal{D}}_{\pi_k,\delta}(M, \pi) - \mathcal{D}_{\pi_k,\delta}(M, \pi)| \leq 4\sqrt{\frac{B_f \log n}{n}}$, and therefore

$$|\widehat{L}^{\pi,M} - L^{\pi,M}| \leq 4\sqrt{\frac{B_f \log n}{n}}. \tag{4.6.24}$$

By a standard ϵ-cover + union bound argument, we can prove the uniform convergence: with high probability (at least $1 - n^{O(1)}$) over the choice of $\tau^{(1)}, \dots, \tau^{(n)}$, for all policy and model, for all policy π and dynamics M,

$$|\widehat{L}^{\pi,M} - L^{\pi,M}| \leq 4\sqrt{\frac{B_f p \log(nBL_f)}{n}} = \epsilon/4. \tag{4.6.25}$$

Suppose at iteration t, we are at policy π_t which is not a (δ, ϵ)-local maximum of $V^{\pi,M^\star}$. Then, there exists π' such that $d(\pi', \pi_t) \leq \delta$ and

$$V^{\pi',M^\star} \geq V^{\pi_t,M^\star} + \epsilon. \tag{4.6.26}$$

Then, we have that

$$\begin{aligned}
V^{\pi_{t+1},M^\star} &\geq L_{\pi_t}^{\pi_{t+1},M_{t+1}} && \text{(by equation (4.6.20))} \\
&\geq \widehat{L}_{\pi_t}^{\pi_{t+1},M_{t+1}} - \epsilon/4 && \text{(by uniform convergence, equation (4.6.25))} \\
&\geq \widehat{L}_{\pi_t}^{\pi',M^\star} - \epsilon/4 && \text{(by the definition of } \pi_{t+1}, M_{t+1}) \\
&\geq L_{\pi_t}^{\pi',M^\star} - \epsilon/2 && \text{(by uniform convergence, equation (4.6.25))} \\
&= V^{\pi',M^\star} - \epsilon/2 && \text{(by equation (R2))} \\
&= V^{\pi_t,M^\star} + \epsilon/2 && \text{(by equation (4.6.26))}
\end{aligned}$$

Note that the total reward can only improve by $\epsilon/2$ for at most $O(B_R/\epsilon)$ steps. Therefore, in the first $O(B_R/\epsilon)$ iterations, we must have hit a solution that is a (δ, ϵ)-local maximum. This completes the proof.

□

4.7 Discussion

We devise a novel algorithmic framework for designing and analyzing model-based RL algorithms with the guarantee to convergence monotonically to a local maximum of the reward. Experimental results show that our proposed algorithm (SLBO) achieves new state-of-the-art performance on several mujoco benchmark tasks when one million or fewer samples are permitted.

A compelling (but obvious) empirical open question then given rise to is whether model-based RL can achieve near-optimal reward on other more complicated tasks or real-world robotic tasks with fewer samples. We believe that understanding the trade-off between optimism and robustness is essential to design more sample-efficient algorithms. Currently, we observed empirically that the optimism-driven part of our proposed meta-algorithm (optimizing $V^{\pi,\widehat{M}}$ over $\widehat{M}$) may lead to instability in the optimization, and therefore don't in general help the performance. It's left for future work to find practical implementation of the optimism-driven approach.

Chapter 5

Model-based Zero-shot Policy Learning

It is a long-standing challenge to enable an intelligent agent to learn in one environment and generalize to an unseen environment without further data collection and finetuning. In this chapter, we consider a zero shot generalization problem setup that complies with biological intelligent agents' learning and generalization processes. The agent is first presented with previous experiences in the training environment, along with task description in the form of trajectory-level sparse rewards. Later when it is placed in the new testing environment, it is asked to perform the task without any interaction with the testing environment. We find this setting natural for biological creatures and at the same time, challenging for previous methods. Behavior cloning, state-of-art RL along with other zero-shot learning methods perform poorly on this benchmark. Given a set of experiences in the training environment, our method learns a neural function that decomposes the sparse reward into particular regions in a contingency-aware observation as a per step reward. Based on such decomposed rewards, we further learn a dynamics model and use Model Predictive Control (MPC) to obtain a policy. Since the rewards are decomposed to finer-granularity observations, they are naturally generalizable to new environments that are composed of similar basic elements. We demonstrate our method on a wide range of environments, including a classic video game – Super Mario Bros, as well as a robotic continuous control task.

5.1 Background

While deep Reinforcement Learning (RL) methods have shown impressive performance on video games [114] and robotics tasks [151, 97], they solve each problem *tabula rasa*. Hence, it will be hard for them to generalize to new tasks without re-training even due to small changes. However, humans can quickly adapt their skills to a new task that requires similar priors *e.g.* physics, semantics and affordances to past experience. The priors can be learned from a spectrum of examples ranging from perfect demonstrative ones that accomplish certain tasks

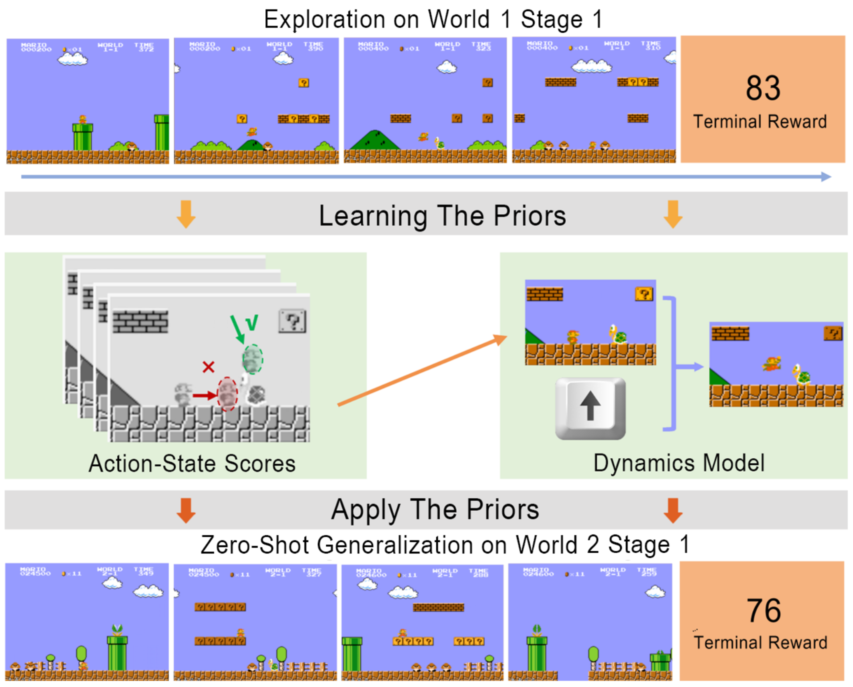


Figure 5.1: Illustrative figure: An agent is learning priors from exploration data from World 1 Stage 1 in Nintendo Super Mario Bros game. In this chapter, the agent focuses on learning two types of priors: learning an action-state preference score for contingency-aware observation and a dynamics model. The action-state scores on the middle left learns that approaching the "Koopa" from the left is undesirable while from the top is desirable. On the middle right, a dynamics model can be learned to predict a future state based on the current state and action. The agent can apply the priors to a new task World 2 Stage 1 to achieve reasonable policy with zero shot.

to aimless exploration.

A parameterized intelligent agent "Mario" who learns to reach the destination in the upper level in Figure 5.1 would fail to do the same in the lower new level because of the change of configurations and background, *e.g.* different placement of blocks, new monsters. When an inexperienced human player is controlling the Mario to move it to the right in the upper level, it might take many trials for him/her to realize the falling to a pit and approaching the "koopa"(turtle) from the left are harmful while standing on the top of the "koopa"(turtle) is not. However, once learned, one can infer similar mechanisms in the lower level in Figure 5.1 without additional trials because human have a variety of priors including the concept of object, similarity, semantics, affordance, etc [53, 43]. In this chapter, we teach machine agents to realize and utilize useful priors from exploration data in the form of decomposed rewards to generalize to new tasks without fine-tuning.

To achieve such zero-shot generalizable policy, a learning agent should have the ability to understand finer-granularity of the observation space e.g. to understand the value of a "koopa" in various configuration and background. However, these quintessentially human abilities are particularly hard for learning agents because of the lack of temporally and spatially fine-grained supervision signal and contemporary deep learning architectures are not designed for compositional properties of scenes. Many recent works rely heavily on the generalization ability of neural networks learning algorithms without capturing the compositional nature of scenes.

In this work, we propose a method, which leverages imperfect exploration data that only have terminal sparse rewards to learn decomposed rewards on specific regions of an observation and further enable zero-shot generalization with a model predictive control (MPC) method. Specifically, given a batch of trajectories with terminal sparse rewards, we use a neural network to assign a reward for the contingency-aware observation o_t at timestep t so that the aggregation of the reward from each contingency-aware observation o_t can be an equivalence of the original sparse reward. We adopt the contingency-aware observations [24] that enables an agent to be aware of its own location. Further, we divide the contingency-aware observation into K sub-regions to obtain more compositional information. To further enable actionable agents to utilize the decomposed score, a neural dynamics model can be learned using self-supervision. We show that how an agent can take advantage of the decomposed rewards and the learned dynamics model with planning algorithms [108]. Our method is called SAP where "S" refers to scoring networked used to decompose rewards,"A" refers to the aggregation of per step rewards for fitting the terminal rewards, and "P" refers to the planning part.

The proposed scoring function, beyond being a reward function for planning, can also be treated as an indicator of the existence of objects that affect the evaluation of a trajectory. We empirically evaluate the decomposed rewards for objects extracted in the context of human priors and hence find the potential of using our method as an unsupervised method for object discovery.

In this chapter, we have two major contributions. First, we demonstrate the importance of decomposing sparse rewards into temporally and spatially smaller observation for obtaining zero-shot generalizable policy. Second, we develop a novel instance that uses our learning-based decomposition function and neural dynamics model that have strong performance on various challenging tasks.

5.2 Related Work

Zero-Shot Generalization and Priors. To generalize in a new environment in a zero-shot manner, the agent needs to learn priors from its previous experiences, including priors on physics, semantics and affordances. Recently, researchers have shown the importance of priors in playing video games [43]. More works have also been done to utilize visual priors in many other domains such as robotics for generalization, etc. [184, 69, 37, 207, 41]. [78, 94, 61] explicitly extended RL to handle object level learning. While our method does not explicitly model objects, we have shown that meaningful scores are learned for objects enabling SAP to generalize to new tasks in zero-shot manner. Recent works [159, 126] try to learn compositional skills for zero-shot transfer, which is complementary to the proposed method.

Inverse Reinforcement Learning. The seminal work [123] proposed inverse reinforcement learning (IRL). IRL aims to learn a reward function from a set of expert demonstrations. IRL and SAP fundamentally study different problem — IRL learns a reward function from *expert* demonstrations, while our method learns from exploratory data that is not necessarily related to any tasks. There are some works dealing with violation of the assumptions of IRL, such as inaccurate perception of the state [15, 186, 16, 23], or incomplete dynamics model [168, 15, 90, 9, 121]; however, IRL does not study the case when the dynamics model is purely learned and the demonstrations are suboptimal. Recent work [196] proposed to leverage failed demonstrations with model-free IRL to perform grasping tasks; though sharing some intuition, our work is different because of the model-based nature.

RL with Sparse Reward. When only sparse rewards are provided, an RL agent suffers a harder exploration problem. Previous work [122] studied the problem of reward shaping, *i.e.* how to change the form of the reward without affecting the optimal policy. The scoring-aggregating part can be thought as a novel form of learning-based reward shaping. A corpus of literature [106, 56, 105] try to learn the reward shaping automatically. However, the methods do not apply to the high-dimensional input such as image. One recent work RUDDER [6] utilizes an LSTM to decompose rewards into per-step rewards. This method can be thought of an scoring function of the full state in our framework.

There are more categories of methods to deal with this problem: (1) Unsupervised exploration strategies, such as curiosity-driven exploration [132, 150], or count-based exploration [171, 163] (2) In goal-conditioned tasks one can use Hindsight Experience Replay [5] to learn from experiences with different goals. (3) defining auxiliary tasks to learn a meaningful intermediate representations [40]. In contrast to previous methods, we effectively convert the single terminal reward to a set of rich intermediate representations, on top of which we can apply planning algorithms. Although model-based RL has been extensively studied, none of the previous work has explored the use of reward decomposition for zero-shot transfer.

5.3 Scoring-Aggregating-Planning Model

Problem Formulation

To be able to generalize better in an unseen environment, an intelligent agent should understand the consequence of its behavior both spatially and temporally. In the RL terminology, we propose to learn rewards that correspond to observations spatially and temporally from its past experiences. To facilitate such goals, we formulate the problem as follows.

Two environments $\mathcal{E}_1, \mathcal{E}_2$ are sampled from the same task distribution. $\mathcal{E}_1$ and $\mathcal{E}_2$ share the same physics and goals but different configurations. e.g. different placement of objects, different terrains.

The agent is first presented with a bank of exploratory trajectories $\{\tau_i\}, i = 1, 2 \cdots N$ collected in training environment $\mathcal{E}_1 = (\mathcal{S}, \mathcal{A}, p)$. Each τ_i is a trajectory and $\tau_i = \{(\mathbf{s}_t, \mathbf{a}_t)\}, t = 1, 2 \cdots K_i$. These trajectories are random explorations in the environment. Note that the agent only learns from the bank of trajectories without further environment interactions, which mimics human utilizing only prior experiences to perform a new task. We provide a scalar terminal evaluation $r(\tau)$ of the entire trajectory when a task $\mathcal{T}$ is specified.

At test time, we evaluate task $\mathcal{T}$ using zero extra interaction in the new environment, $\mathcal{E}_2 = (\mathcal{S}', \mathcal{A}, p')$. We assume identical action space. There is no reward provided at test time. In this chapter, we focus on locomotion tasks with object interactions, such as Super Mario running with other objects in presence, or a Reacher robot acting with obstacles around.

Spatial Temporal Reward Decomposition

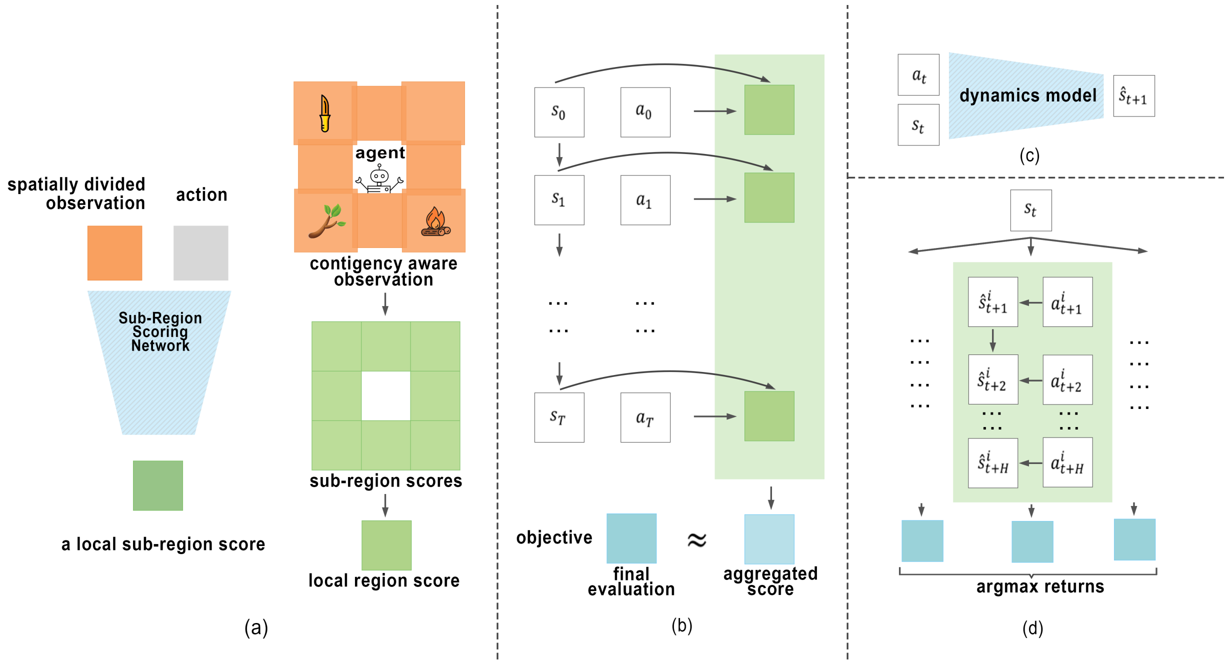


Figure 5.2: An overview of the SAP method. (a) For each time step, a scoring network scores contigent sub-regions conditioned on action. (b) we aggregate the prediction over all time steps to fit terminal reward (c)&(d) describe the dynamics learning and planning in mpc in the policy stage.

In this section, we introduce the method to decompose the terminal sparse reward into specific time step and spatial location. First, we introduce the temporal decomposition and then discuss the spatial decomposition.

Temporal Reward Decomposition. The temporal reward decomposition can be described as $\mathrm{S}_\theta(\mathrm{W}(\mathbf{s}_t), \mathbf{a}_t)$. Here θ denotes parameters in a neural network, W is a function that extracts contingency-aware observations from states. Here, contingency-aware means a subset of spatial global observation around the agent, such as pixels surrounding a game character or voxels around end-effector of a manipulator. We note that the neural network's parameters are shared for every contingency-aware observation. Intuitively, this function measures how well an action $\mathbf{a}_t$ performs on this particular state, and we refer to S_θ as the *scoring function*.

To train this network, we aggregate the decomposed rewards $\mathrm{S}_\theta(\mathrm{W}(\mathbf{s}_t), \mathbf{a}_t)$ for each step into a single aggregated reward J, by an aggregating function G:

$$J_\theta(\tau) = \mathrm{G}_{(\mathbf{s}_t, \mathbf{a}_t) \in \tau}(S_\theta(W(s_t), a_t))$$

The aggregated reward J are then fitted to the sparse terminal reward. In practice, G is chosen based on the form of the sparse terminal reward, *e.g.* a max or a sum function. In the learning process, the S_θ function is learned by back-propagating errors between the terminal sparse reward and the predicted J through the aggregation function. In this chapter, we use ℓ_2 loss that is:

$$\min_\theta \frac{1}{2}(J_\theta(\tau) - r(\tau))^2$$

Spatial Reward Decomposition. An environment usually contains multiple objects. Those objects might re-appear at various spatial locations over time. To further assist the learned knowledge to be transferrable to the new environment, we take advantage of the compositionality of the environment by also decomposing the reward function spatially. More specifically, we divide the contingency-aware observation into smaller sub-regions. For example, in a Cartesian coordinate system, we can divide each coordinate independently and uniformly. With the sub-regions, we re-parametrize the scoring function as $\sum_{l \in \mathcal{L}} \mathrm{S}_\theta(\mathrm{W}_l(\mathbf{s}_t), \mathbf{a}_t)$, where l is the index of the sub-regions and we retarget S_θ for the sub-region instead of the whole contingency-aware observation. The intuition is that the smaller sub-regions contains objects or other unit elements that are also building blocks of unseen environments. Such sub-regions become crucial later to generalize to the new environment.

Policy Stage

To solve the novel task with zero interaction, we propose to use planning algorithms to find optimal actions based on the learned scoring function and a learned dynamics model. As shown in the part (c) of Figure 5.2, we learn a forward dynamics model $\mathcal{M}_\phi$ based on the exploratory data with a supervised loss function. Specifically, we train a neural network that takes in the action a_t, state s_t and output $\hat{\mathrm{s}}_{t+1}$, which is an estimate of s_{t+1}. We use an ℓ_2 loss as the objective:

$$\min_\phi \frac{1}{2}(\mathcal{M}_\phi(\mathrm{s}_t, \mathrm{a}_t) - \mathrm{s}_{t+1})^2$$

With the learned dynamics model and the scoring function, we solve an optimization problem using the Model Predictive Control (MPC) algorithm to find the best trajectory for a task $\mathcal{T}$ in environment $\mathcal{E}_2$. The objective of the optimization problem is to minimize $-J_\theta(\tau')$. Here we randomly sample multiple action sequences up to length H, unroll the states based on the current state and the learned dynamics model while computing the cost with the scoring function. We select the action sequence with the minimal cost, and execute the first action in the selected sequence in the environment.

Discussion on zero-shot generalization

Although neural networks have some generalization capabilities, it is still easy to overfit to the training domain. Previous works [127] notice that neural network "memorizes" the optimal action throughout the training process. One can not expect an agent that only remembers optimal actions to generalize well when placed in a new environment. Our method does not suffer from the memorization issue as much as the neural network policy because it can come up with novel solutions, that are not necessarily close to the training examples, since our method learns generalizable model and scores that makes planning possible. The generalization power comes mainly from the smaller building blocks shared across environments as well as the universal dynamics model. This avoids the SAP method to replay the action of the nearest neighbour state in the training data.

Section 5.4, Figure 5.3a 5.3b compares our method and a neural network RL policy. It shows that in the training environment, RL policy is only slightly worse than our method, however, the RL policy performs much worse than ours in the zero shot generalization case.

5.4 Experiments

In this section, we study how well the proposed method performs compare to baselines, and the roles of the proposed temporal and spatial reward decompositions. We conduct experiments on two domains: a famous video game "Super Mario Bros" [76] and a robotics blocked reacher task [65].

Experiment on Super Mario Bros

To evaluate our proposed algorithm in a challenging environment, we run our method and baseline methods in the Super Mario Bros environment. This environment features high-dimensional visual observations, which is challenging since we have a large hypothesis space. The original game has 240×256 image input and discrete action space with 5 choices. We wrap the environment following [114]. Finally, we obtain a 84×84 size 4-frame gray-scale stacked observation. The goal for an agent is to survive and go toward the right as far as possible. We don't have access to the dense reward for each step but the environment returns how far the agent moves towards target at the end of a trajectory as the delayed terminal sparse reward.

Baselines. We compare our method with various types of baselines, including state-of-art zero-shot RL algorithms, generic policy learning algorithms as well as oracles with much more environment interactions.

Exploration Data Exploration Data is the data from which we learn the scores, dynamics model and imitate. The data is collected from noisy sub-optimal version of policy trained using [132]. The average reward on this dataset is a baseline for all other methods.

Behavioral Cloning [138, 10] Behavioral Cloning (BC) learns a mapping from a state to an action on the exploration data using supervised learning. We use cross-entropy loss for predicting the actions.

Model Based with Human Prior Model Based with Human Priors method (MBHP) incorporates model predictive control with predefined human priors, which is +1 score if the agent tries to move or jump toward the right and 0 otherwise.

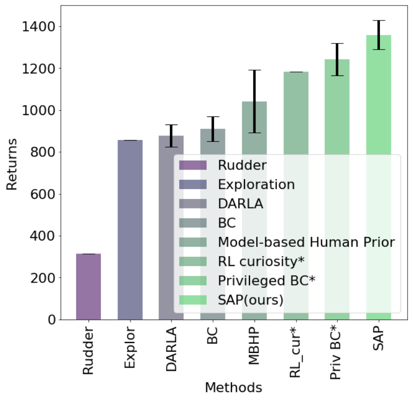
DARLA [61] DARLA relies on learning a latent state representation that can be transferred from the training environments to the testing environment. It achieves this goal by obtaining a disentangled representation of the environment's generative factors before learning to act. We use the latent representation as the observations for a behavioral cloning agent.

RUDDER [6] RUDDER proposes to use LSTM to decompose the delayed sparse reward to dense rewards. It first trains a function $f(\tau_{1:T})$ predict the terminal reward of a trajectory $\tau_{1:T}$. It then use $f(\tau_{1:t}) - f(\tau_{1:t-1})$ as dense reward at step t to train RL policies. We change policy training to MPC so this reward can be used in zero-shot setting.

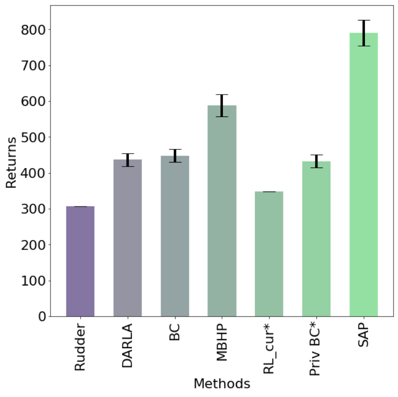
Behavior Clone with Privilege Data Instead of using exploratory trajectories from the environment, we collect a set of near optimal trajectories in the training environment and train a behavior clone agent from it. Note that this is not a fair comparison with other methods, since this method uses better performing training data.

RL curiosity We use a PPO [152] agent that is trained with curiosity driven reward [19] and the final sparse reward in the training environment. This also violates the setting we have as it interacts with the environment. We conduct this experiment to test the generalization ability of an RL agent.

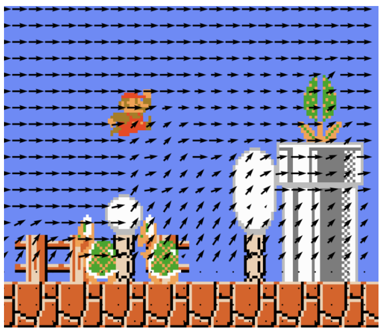
Analysis. Fig. 5.3 a and Fig. 5.3 b show how the above methods performs in the Super Mario Bros. The Explore baseline shows the average training trajectory performance. We found that RUDDER fails to match the demonstration performance. This can attributed to the LSTM in RUDDER not having sufficient supervision signals from the long horizon (2k steps) delayed rewards in the Mario environment. DARLA slightly outperforms the demonstration data in training. Its performance is limited by the unsupervised visual disentangled learning step, which is a hard problem in the complex image domain. Behavior Cloning is slightly better than the exploration data, but much worse than our method. MBHP is a model-based approach where human defines the per step reward function. However, it is prohibitive to



(a) Mario train world returns



(b) Mario test world returns



(c) Visualized greedy actions from the learned scores in a new level(W5S1)

Figure 5.3: (a) & (b) The total returns of different methods on the Super Mario Bros. Methods with $*$ have privilege access to optimal data instead of sub-optimal data. Error bars are shown as 95% confidence interval. See Section 5.4 for details.

Table 5.1: Ablation of the temporal and spatial reward decomposition.

	W1S1(train)	W2S1(test)
SAP	**1359.3**	**790.1**
SAP w/o spatial	1258.0	737.0
SAP w/o spatial temporal	1041.3	587.9

obtain detailed manual rewards every step. In this task, it is inferior to SAP's learned priors. We further compare to two methods that unfairly utilize additional privileged information. The Privileged BC method trains on near optimal data, and performs quite well during training; however, it performs much worse during the zero shot testing. The similar trend happens with RL Curiosity, which has the privileged access to online interactions in the training environment.

To conclude, we found generic model free algorithms, including both RL (RL Curiosity), and behavior cloning (Privilege BC) perform well on training data but suffer from severe generalization issue. The zero-shot algorithms (DARLA), BC from exploratory data (BC) and moded-based method (MBHP) suffer less from degraded generalization, but they all under-perform our proposed algorithm.

Ablative Studies. In order to have a more fine-grained understanding of the effect of our newly proposed temporal and spatial reward decomposition, we further conduct ablation studies on those two components. We run two other versions of our algorithm, removing the spatial reward decomposition and the temporal reward decomposition one at a time. More specifically, the SAP w/o spatial method does not divide the observation into sub-regions, but simply use a convolution network to approximate the scoring function. SAP w/o spatial temporal further removes the temporal reward learning part, and replace the scoring function with a human-designed prior. I.e. it is the same as the MBHP baseline.

Table 5.1 shows the result. We found that both the temporal and the spatial reward learning component contributes to the final superior performance. Without temporal reward decomposition, the agent either have to deal with sparse reward or use a manually specified rewards which might be tedious or impossible to collect. Without the spatial decomposition, the agent might not find correctly which specific region or object is important to the task and hence fail to generalize.

Visualization of learned scores. In this section, we qualitatively study the induced action by greedily maximizing one-step score. I.e. for any location on the image, we assumes the Super Mario agent were on that location and find the action a that maximize the learned scoring function $S_\theta(W(s), a)$. We visualize the computed actions on World 5 Stage 1 (Fig. 5.3 c) which is visually different from previous tasks. More visualization can be found in [1] In this testing case, we see that the actions are reasonable, such as avoiding obstacles and monsters by jumping over them, even in the face of previously unseen configurations and different backgrounds. However, the "Piranha Plants" are not recognized because all the prior scores are learned from W1S1 where it never appears. More visualization of action maps are available in our videos. Those qualitative studies further demonstrate that the SAP method can assign meaningful scores for different objects in an unsupervised manner. It also produces good actions even in a new environment.

SAP on the 3-D robotics task

In this section, we further study the SAP method to understand its property with a higher dimensional observation space. We conduct experiments in a 3-D robotics environment, BlockedReacher-v0. In this environment, a robot hand is initialized at the left side of a table and tries to reach the right. Between the robot hand and the goal, there are a few blocks standing as obstacles. The task is moving the robot hand to reach a point on $y = 1.0$ as fast as possible. To test the generalization capability, we create four different configurations of the obstacles, as shown in Figure 5.4. Figure 5.4 A is the environment where we collect exploration data from and Figure 5.4 B, C, D are the testing environments. Note that the exploration data has varying quality, where many of the trajectories are blocked by the obstacles. We introduce more details about this experiment.

Environment. In the Blocked Reach environment, we use a 7-DoF robotics arm to reach a specific point. For more details, we refer the readers to [137]. We discretize the robot world into a $200 \times 200 \times 200$ voxel cube. For the action space, we discretize the actions into two choices for each dimension which are moving 0.5 or -0.5. Hence, in total there are 8 actions. We design four configurations for evaluating different methods as shown in Figure 5.4. For each configurations, there are three objects are placed in the middle as obstacles.

[1] https://sites.google.com/view/sapnew/home

Applying SAP on the 3D robotic task. We apply the SAP framework as follows. The original observation is a 25-dimensional continuous state and the action space is a 3-dimensional continuous control. They are discretized into voxels and 8 discrete actions. The scoring function is a fully-connected neural network that takes in a flattened voxel sub-region. We also train a 3D convolutional neural net as the dynamics model. The dynamics model takes in the contingency-aware observation as well as an action, and outputs the next robot hand location. With the learned scores and the dynamics model, we plan using the MPC method with a horizon of 8 steps.

Architectures for score function and dynamics model. For the score function, we train a 1 hidden layer fully-connected neural networks with 128 units. We use ReLu functions as activation except for the last layer. Note that the input 5 by 5 by 5 voxels are flattened before put into the scoring neural network.

For the dynamics model, we train a 3-D convolution neural network that takes in the contingency-aware observation (voxels), action and last three position changes. The voxels contingent to end effector are encoded using three 3d convolution with kernel size 3 and stride 2. Channels of these 3d conv layers are 16, 32, 64, respectively. A 64-unit FC layer is connected to the flattened features after convolution. The action is encoded with one-hot vector connected to a 64-unit FC layer. The last three δ positions are also encoded with a 64-unit FC layer. The three encoded features are concatenated and go through a 128-unit hidden FC layer and output predicted change in position. All intermediate layers use ReLu as activation.

Table 5.2: Evaluation of SAP, MBHP and Rudder on the 3D Reacher environment. Numbers are the avg. steps to reach the goal. The lower the better. Numbers in the brackets are the 95% confidence interval. "L" denotes the learned dynamics, and "P" denotes the perfect dynamics.

	Config A	Config B	Config C	Config D
SAP(L)	**97.53**[2.5]	**86.53**[1.9]	**113.3**[3.0]	**109.2**[3.0]
MBHP(L)	124.2[4.5]	102.0[3.0]	160.7[7.9]	155.4[7.2]
Rudder(L)	1852[198]	1901[132]	1933[148]	2000[0.0]
SAP(P)	**97.60**[2.6]	**85.38**[1.8]	**112.2**[3.1]	**114.1**[3.2]
MBHP(P)	125.3[4.3]	102.4[3.0]	153.4[5.3]	144.9[4.8]
Rudder(P)	213.6[4.2]	194.2[7.0]	208.2[5.6]	201.5[6.3]

Results. We evaluate similar baselines as in the previous section that is detailed in 5.4. In Table 5.2, we compare our method with the MBHP and RUDDER on the 3D robot reaching task. We found that our method needs significantly fewer steps than the two baselines, in both training environment and testing ones. We find that SAP significantly moves faster

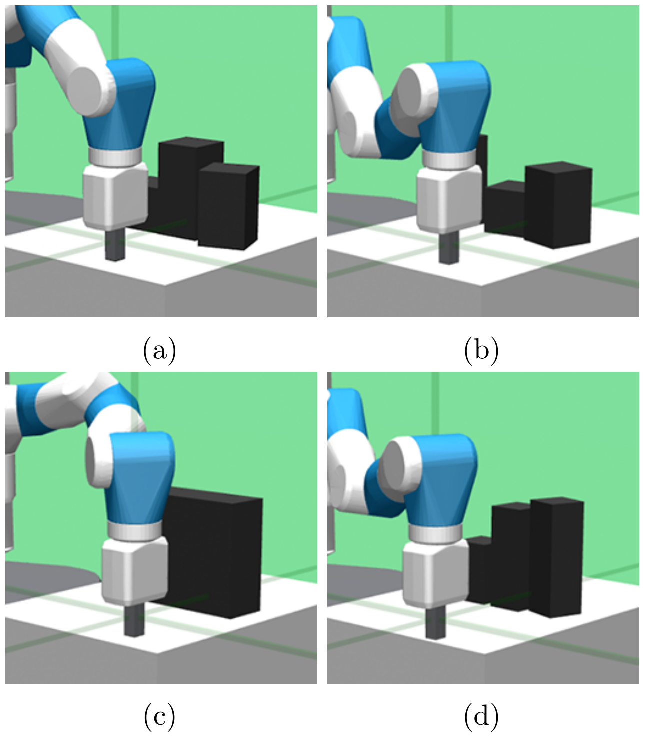


Figure 5.4: Four variants of the 3D robot reacher environments. See Section 5.4 for details.

to the right because it learns a negative score for crashing into the obstacles. However, the MBHP method, which has +1 positive for each 1 meter moved to the right, would be stuck by the obstacles for a longer duration. When training with RUDDER, the arm also frequently waste time by getting stuck at obstacle. We found that our SAP model is relatively insensitive to the errors in the learned dynamics, such that the performance using the learned dynamics is close to that of perfect dynamics. These experiments show that our method can be applied to robotics environment that can be hard for some algorithms due to the 3-D nature.

5.5 Discussion

In this chapter, we introduced a new method called SAP that aims to generalize in the new environment without any further interactions by learning the temporally and spatially decomposed rewards. We empirically demonstrate that the newly proposed algorithm outperform previous zero-shot RL method by a large margin, on two challenging environments, i.e. the Super Mario Bros and the 3D Robot Reach. The proposed algorithm along with a wide range of baselines provide a comprehensive understanding of the important aspect zero-shot RL problems.

Chapter 6

Conclusion

This thesis presents a series of methods toward learning predictive models and using them for efficient policy learning. These studies are on high-fidelity video prediction, multi-modal video prediction, model-based reinforcement learning, and model-based policy zero-shot generalization. In Chapter 2, we introduce a method that disentangles propagation and generation that alleviates blurry artifact and shape deformation. In Chapter 3, we continue to learn a video predictive method that can leverage retrieved examples to model multi-modal data distribution. In Chapter 4, we further seek to learn a predictive model for policy learning with deep reinforcement learning algorithms. We propose a new model-based framework and a value discrepancy loss that can achieve sample-efficient policy learning, In Chapter 5, we explore modeling methods that can make policy generalize to unseen tasks in a zero-shot manner.

Admittedly, predictive models can also be applied to abstracted states space or spaces with more semantic meaning. For example, computer vision techniques can be used to extract objects from images as structured representations. We believe it is still important to learn a predictive model in the image space when limited prior information is given. On the one hand, video prediction can be one of the most general objectives in the face of unknown types of tasks. On the other hand, the techniques from video prediction can also be transferred to relevant domains if necessary. In the future, we can explore learning predictive models on more structured states on specific real-world tasks. We will also keep pushing the boundary of video prediction in order to improve sample efficient robotics tasks.

In model-based reinforcement learning, it is usually hard to learn a predictive model because of the narrow data distribution and the over generalization of neural networks. In Chapter 4, we introduce an iterative training scheme to avoid the policy over-fitting issue. In this direction, we can explore stabilizing model-based RL by either learning a conservative dynamics model or ensemble models. Another challenge is that model-based RL usually requires an agent to collect data from the environment multiple times which is time-consuming. Hence, it is promising to leverage some curated offline datasets for model-based RL. Toward better generalization with the model-based approach, we propose in Chapter 5 to learn both visual dynamics and a value prior for contingency-aware observation. In the future, we want to learn better predictive models that can handle more complicated tasks that involve robotics manipulation.

One interesting future direction on learning predictive is to learn models that are useful for long-horizon tasks. In previous model-based RL and video prediction works, most models are per-step ones that predict future observations step by step. However, humans usually only predict key events in the temporal axis rather than predict future states for every time step. It is still unclear that how intelligent agents can learn temporally abstracted models. For example, one can choose to learn hierarchical dynamics models from the execution of pre-defined primitives and/or from the entropy of states. Toward this direction, we can also try to learn better representations that model abstract temporal information. If we can learn such models, compositionally generalizable task-level planning would be possible in complex real-world scenarios for long-horizon tasks. While this method might potentially solve many well-defined tasks, it might suffer from some tasks that are hard to define. Hence, it might be possible to use model-free algorithms for "low-level" tasks while using model-based approaches for "high-level" planning. To smartly combine model-free algorithms with model-based approaches is another interesting unanswered question for future research.

Bibliography

[1] Yasin Abbasi-Yadkori and Csaba Szepesvári. “Regret bounds for the adaptive control of linear quadratic systems”. In: *Proceedings of the 24th Annual Conference on Learning Theory*. 2011, pp. 1–26.

[2] Joshua Achiam et al. “Constrained policy optimization”. In: *arXiv preprint arXiv:1705.10528* (2017).

[3] Raja Hafiz Affandi et al. “Learning the Parameters of Determinantal Point Process Kernels”. In: *ICML*. 2014.

[4] Shipra Agrawal and Randy Jia. “Optimistic posterior sampling for reinforcement learning: worst-case regret bounds”. In: *Advances in Neural Information Processing Systems*. 2017, pp. 1184–1194.

[5] Marcin Andrychowicz et al. “Hindsight experience replay”. In: *Neurips*. 2017, pp. 5048–5058.

[6] Jose A Arjona-Medina et al. “Rudder: Return decomposition for delayed rewards”. In: *arXiv preprint arXiv:1806.07857* (2018).

[7] Kavosh Asadi, Dipendra Misra, and Michael L Littman. “Lipschitz continuity in model-based reinforcement learning”. In: *arXiv preprint arXiv:1804.07193* (2018).

[8] Mohammad Babaeizadeh et al. “Stochastic Variational Video Prediction”. In: *ICLR*. 2018.

[9] JA Bagnell et al. “Boosting structured prediction for imitation learning”. In: *Advances in Neural Information Processing Systems*. 2007, p. 1153.

[10] Michael Bain and Claude Sammut. “A Framework for Behavioural Cloning.” In: *Machine Intelligence 15*. 1995, pp. 103–129.

[11] Connelly Barnes et al. “PatchMatch: A randomized correspondence algorithm for structural image editing”. In: *ACM Transactions on Graphics (ToG)* (2009).

[12] Peter L Bartlett and Ambuj Tewari. “REGAL: A regularization based algorithm for reinforcement learning in weakly communicating MDPs”. In: *Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence*. AUAI Press. 2009, pp. 35–42.

[13] Marcelo Bertalmio et al. “Image inpainting”. In: *Proceedings of the 27th annual conference on Computer graphics and interactive techniques*. 2000.

[14] Ross Boczar, Nikolai Matni, and Benjamin Recht. “Finite-Data Performance Guarantees for the Output-Feedback Control of an Unknown System”. In: *arXiv preprint arXiv:1803.09186* (2018).

[15] Kenneth Bogert and Prashant Doshi. “Multi-robot inverse reinforcement learning under occlusion with state transition estimation”. In: *AAMAS*. 2015, pp. 1837–1838.

[16] Kenneth Bogert et al. “Expectation-maximization for inverse reinforcement learning with hidden data”. In: *AAMAS*. 2016, pp. 1034–1042.

[17] Andrew Brock, Jeff Donahue, and Karen Simonyan. “Large scale gan training for high fidelity natural image synthesis”. In: *arXiv preprint arXiv:1809.11096* (2018).

[18] J. Buckman et al. “Sample-Efficient Reinforcement Learning with Stochastic Ensemble Value Expansion”. In: *ArXive-prints* (July 2018). arXiv: `1807.01675`.

[19] Yuri Burda et al. “Exploration by random network distillation”. In: *arXiv preprint arXiv:1810.12894* (2018).

[20] Wonmin Byeon et al. “ContextVP: Fully Context-Aware Video Prediction”. In: *ECCV*. 2018.

[21] Wonmin Byeon et al. “ContextVP: Fully Context-Aware Video Prediction”. In: *ECCV*. 2018.

[22] Liang-Chieh Chen et al. “Semantic image segmentation with deep convolutional nets and fully connected crfs”. In: *arXiv preprint arXiv:1412.7062* (2014).

[23] Jaedeug Choi and Kee-Eung Kim. “Inverse reinforcement learning in partially observable environments”. In: *Journal of Machine Learning Research* 12.Mar (2011), pp. 691–730.

[24] Jongwook Choi et al. “Contingency-aware exploration in reinforcement learning”. In: *arXiv preprint arXiv:1811.01483* (2018).

[25] Kurtland Chua et al. “Deep Reinforcement Learning in a Handful of Trials using Probabilistic Dynamics Models”. In: *arXiv preprint arXiv:1805.12114* (2018).

[26] Ignasi Clavera et al. “Model-Based Reinforcement Learning via Meta-Policy Optimization”. In: *arXiv preprint arXiv:1809.05214* (2018).

[27] Thomas M Cover and Joy A Thomas. *Elements of information theory*. John Wiley & Sons, 2012.

[28] Christoph Dann, Tor Lattimore, and Emma Brunskill. “Unifying PAC and regret: Uniform PAC bounds for episodic reinforcement learning”. In: *Advances in Neural Information Processing Systems*. 2017, pp. 5713–5723.

[29] Sarah Dean et al. “On the sample complexity of the linear quadratic regulator”. In: *arXiv preprint arXiv:1710.01688* (2017).

[30] Sarah Dean et al. "Regret Bounds for Robust Adaptive Control of the Linear Quadratic Regulator". In: *arXiv preprint arXiv:1805.09388* (2018).

[31] Marc Deisenroth and Carl 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.

[32] Marc Deisenroth and Carl 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.

[33] Marc Peter Deisenroth, Gerhard Neumann, Jan Peters, et al. "A survey on policy search for robotics". In: *Foundations and Trends® in Robotics* 2.1–2 (2013), pp. 1–142.

[34] Marc Peter Deisenroth, Carl Edward Rasmussen, and Dieter Fox. "Learning to control a low-cost manipulator using data-efficient reinforcement learning". In: (2011).

[35] Emily Denton and Rob Fergus. "Stochastic Video Generation with a Learned Prior". In: *ICML*. 2018.

[36] Emily L. Denton and Vighnesh Birodkar. "Unsupervised Learning of Disentangled Representations from Video". In: *NeurIPS*. 2017.

[37] Coline Devin et al. "Deep object-centric representations for generalizable robot learning". In: *ICRA*. IEEE. 2018, pp. 7111–7118.

[38] Piotr Dollar et al. "Pedestrian detection: An evaluation of the state of the art". In: *IEEE transactions on pattern analysis and machine intelligence* (2012).

[39] Alexey Dosovitskiy and Thomas Brox. "Generating images with perceptual similarity metrics based on deep networks". In: *NIPS*. 2016.

[40] Alexey Dosovitskiy and Vladlen Koltun. "Learning to act by predicting the future". In: *arXiv preprint arXiv:1611.01779* (2016).

[41] Yilun Du and Karthik Narasimhan. "Task-Agnostic Dynamics Priors for Deep Reinforcement Learning". In: *arXiv preprint arXiv:1905.04819* (2019).

[42] Yan Duan et al. "Benchmarking deep reinforcement learning for continuous control". In: *International Conference on Machine Learning*. 2016, pp. 1329–1338.

[43] Rachit Dubey et al. "Investigating human priors for playing video games". In: *arXiv preprint arXiv:1802.10217* (2018).

[44] Frederik Ebert et al. "Self-Supervised Visual Planning with Temporal Skip Connections". In: *CoRL*. 2017.

[45] Mohamed Elfeki et al. "GDPP: Learning Diverse Generations using Determinantal Point Processes". In: *ICML*. 2019.

[46] Amir-massoud Farahmand, Andre Barreto, and Daniel Nikovski. "Value-Aware Loss Function for Model-based Reinforcement Learning". In: *Artificial Intelligence and Statistics*. 2017, pp. 1486–1494.

[47] Vladimir Feinberg et al. “Model-Based Value Estimation for Efficient Model-Free Reinforcement Learning”. In: *arXiv preprint arXiv:1803.00101* (2018).

[48] Chelsea Finn, Ian Goodfellow, and Sergey Levine. “Unsupervised learning for physical interaction through video prediction”. In: *NIPS*. 2016.

[49] Chelsea Finn, Ian J. Goodfellow, and Sergey Levine. “Unsupervised Learning for Physical Interaction through Video Prediction”. In: *NeurIPS*. 2016.

[50] Ronan Fruit et al. “Efficient Bias-Span-Constrained Exploration-Exploitation in Reinforcement Learning”. In: *arXiv preprint arXiv:1802.04020* (2018).

[51] Hang Gao et al. “Disentangling Propagation and Generation for Video Prediction”. In: *CoRR* (2018).

[52] Marta Garnelo et al. “Conditional Neural Processes”. In: *ICML*. 2018.

[53] James Gibson. *The ecological approach to visual perception*. Psychology Press, 2014.

[54] Ian Goodfellow et al. “Generative adversarial nets”. In: *NIPS*. 2014.

[55] Ian J. Goodfellow et al. “Generative Adversarial Nets”. In: *NeurIPS*. 2014.

[56] Marek Grześ and Daniel Kudenko. “Online learning of shaping rewards in reinforcement learning”. In: *Neural Networks* 23.4 (2010), pp. 541–550.

[57] Shixiang Gu et al. “Continuous deep q-learning with model-based acceleration”. In: *International Conference on Machine Learning*. 2016, pp. 2829–2838.

[58] David Ha and Jürgen Schmidhuber. “World Models”. In: *arXiv preprint arXiv:1803.10122* (2018).

[59] Tuomas Haarnoja et al. “Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor”. In: *arXiv preprint arXiv:1801.01290* (2018).

[60] Zekun Hao, Xun Huang, and Serge Belongie. “Controllable Video Generation with Sparse Trajectories”. In: *CVPR*. 2018.

[61] Irina Higgins, Arka Pal, et al. “Darla: Improving zero-shot transfer in reinforcement learning”. In: *ICML*. JMLR. org. 2017, pp. 1480–1490.

[62] Karl Hinderer. “Lipschitz continuity of value functions in Markovian decision processes”. In: *Mathematical Methods of Operations Research* 62.1 (2005), pp. 3–22.

[63] Sepp Hochreiter and Jürgen Schmidhuber. “Long Short-Term Memory”. In: *Neural Computation* (1997).

[64] De-An Huang et al. “What Makes a Video a Video: Analyzing Temporal Information in Video Understanding Models and Datasets”. In: *CVPR*. 2018.

[65] Zhiao Huang, Fangchen Liu, and Hao Su. “Mapping State Space using Landmarks for Universal Goal Reaching”. In: *arXiv preprint arXiv:1908.05451* (2019).

[66] K Jetal Hunt et al. “Neural networks for control systems—a survey”. In: *Automatica* 28.6 (1992), pp. 1083–1112.

[67] Thomas Jaksch, Ronald Ortner, and Peter Auer. “Near-optimal regret bounds for reinforcement learning”. In: *Journal of Machine Learning Research* 11.Apr (2010), pp. 1563–1600.

[68] Joel Janai et al. “Unsupervised Learning of Multi-Frame Optical Flow with Occlusions”. In: *ECCV*. 2018.

[69] Eric Jang et al. “Grasp2vec: Learning object representations from self-supervised grasping”. In: *arXiv preprint arXiv:1811.06964* (2018).

[70] Xu Jia et al. “Dynamic Filter Networks”. In: *NIPS*. 2016.

[71] Justin Johnson, Alexandre Alahi, and Li Fei-Fei. “Perceptual losses for real-time style transfer and super-resolution”. In: *ECCV*. 2016.

[72] S. Kakade, M. Wang, and L. F. Yang. “Variance Reduction Methods for Sublinear Reinforcement Learning”. In: *ArXiv e-prints* (Feb. 2018). arXiv: `1802.09184 [cs.AI]`.

[73] Sham Kakade and John Langford. “Approximately optimal approximate reinforcement learning”. In: *ICML*. Vol. 2. 2002, pp. 267–274.

[74] Nal Kalchbrenner et al. “Video pixel networks”. In: *arXiv preprint arXiv:1610.00527* (2016).

[75] Gabriel Kalweit and Joschka Boedecker. “Uncertainty-driven Imagination for Continuous Deep Reinforcement Learning”. In: *Conference on Robot Learning*. 2017, pp. 195–206.

[76] Christian Kauten. *Super Mario Bros for OpenAI Gym*. GitHub. 2018. URL: `https://github.com/Kautenja/gym-super-mario-bros`.

[77] Michael Kearns and Satinder Singh. “Near-optimal reinforcement learning in polynomial time”. In: *Machine learning* 49.2-3 (2002), pp. 209–232.

[78] Ramtin Keramati et al. “Strategic object oriented reinforcement learning”. In: *arXiv preprint arXiv:1806.00175* (2018).

[79] S Mohammad Khansari-Zadeh and Aude Billard. “Learning stable nonlinear dynamical systems with gaussian mixture models”. In: *IEEE Transactions on Robotics* 27.5 (2011), pp. 943–957.

[80] Yunji Kim et al. “Unsupervised Keypoint Learning for Guiding Class-Conditional Video Prediction”. In: *NeurIPS*. 2019.

[81] Diederik P Kingma and Max Welling. “Auto-encoding variational bayes”. In: *arXiv preprint arXiv:1312.6114* (2013).

[82] Diederik P. Kingma and Max Welling. “Auto-Encoding Variational Bayes”. In: *ICLR*. 2014.

[83] Jonathan Ko and Dieter Fox. “GP-BayesFilters: Bayesian filtering using Gaussian process prediction and observation models”. In: *Autonomous Robots* 27.1 (2009), pp. 75–90.

[84] S. Kullback and R. A. Leibler. In: *Ann. Math. Statist.* (1951).

[85] Manoj Kumar et al. “VideoFlow: A Conditional Flow-Based Model for Stochastic Video Generation”. In: *ICLR*. 2020.

[86] Thanard Kurutach et al. “Learning Plannable Representations with Causal InfoGAN”. In: *NeurIPS*. 2018.

[87] Thanard Kurutach et al. “Model-ensemble trust-region policy optimization”. In: *arXiv preprint arXiv:1802.10592* (2018).

[88] Kailasam Lakshmanan, Ronald Ortner, and Daniil Ryabko. “Improved regret bounds for undiscounted continuous reinforcement learning”. In: *International Conference on Machine Learning*. 2015, pp. 524–532.

[89] Alex X. Lee et al. “Stochastic Adversarial Video Prediction”. In: *CoRR* (2018).

[90] Sergey Levine and Pieter Abbeel. “Learning neural network policies with guided policy search under unknown dynamics”. In: *Advances in Neural Information Processing Systems*. 2014, pp. 1071–1079.

[91] Sergey Levine and Vladlen Koltun. “Guided policy search”. In: *International Conference on Machine Learning*. 2013, pp. 1–9.

[92] Sergey Levine et al. “End-to-end training of deep visuomotor policies”. In: *The Journal of Machine Learning Research* 17.1 (2016), pp. 1334–1373.

[93] Chuan Li and Michael Wand. “Combining markov random fields and convolutional neural networks for image synthesis”. In: *CVPR*. 2016.

[94] Minne Li, Pranav Nashikkar, and Jun Wang. “Optimizing Object-based Perception and Control by Free-Energy Principle”. In: *CoRR* abs/1903.01385 (2019). arXiv: `1903.01385`. URL: `http://arxiv.org/abs/1903.01385`.

[95] Yijun Li et al. “Flow-Grounded Spatial-Temporal Video Prediction from Still Images”. In: *ECCV*. 2018.

[96] Xiaodan Liang et al. “Dual motion GAN for future-flow embedded video prediction”. In: *ICCV*. 2017.

[97] Timothy P Lillicrap et al. “Continuous control with deep reinforcement learning”. In: *arXiv preprint arXiv:1509.02971* (2015).

[98] Rudolf Lioutikov et al. “Sample-based informationl-theoretic stochastic optimal control”. In: *Robotics and Automation (ICRA), 2014 IEEE International Conference on*. IEEE. 2014, pp. 3896–3902.

[99] Guilin Liu et al. “Image inpainting for irregular holes using partial convolutions”. In: *arXiv preprint arXiv:1804.07723* (2018).

[100] Ziwei Liu et al. "Video Frame Synthesis Using Deep Voxel Flow." In: *ICCV*. 2017.

[101] William Lotter, Gabriel Kreiman, and David Cox. "Deep predictive coding networks for video prediction and unsupervised learning". In: *ICLR*. 2017.

[102] William Lotter, Gabriel Kreiman, and David D. Cox. "Deep Predictive Coding Networks for Video Prediction and Unsupervised Learning". In: *ICLR*. 2017.

[103] Pauline Luc et al. "Predicting Deeper into the Future of Semantic Segmentation". In: *ICCV*. 2017.

[104] Horia Mania, Aurelia Guy, and Benjamin Recht. "Simple random search provides a competitive approach to reinforcement learning". In: *arXiv preprint arXiv:1803.07055* (2018).

[105] Maryam Marashi, Alireza Khalilian, and Mohammad Ebrahim Shiri. "Automatic reward shaping in Reinforcement Learning using graph analysis". In: *ICCKE*. IEEE. 2012, pp. 111–116.

[106] Bhaskara Marthi. "Automatic shaping and decomposition of reward functions". In: *Proceedings of the 24th International Conference on Machine learning*. ACM. 2007, pp. 601–608.

[107] Michael Mathieu, Camille Couprie, and Yann LeCun. "Deep multi-scale video prediction beyond mean square error". In: *arXiv preprint arXiv:1511.05440* (2015).

[108] David Q Mayne et al. "Constrained model predictive control: Stability and optimality". In: *Automatica* 36.6in (2000), pp. 789–814.

[109] Simon Meister, Junhwa Hur, and Stefan Roth. "UnFlow: Unsupervised learning of optical flow with a bidirectional census loss". In: *arXiv preprint arXiv:1711.07837* (2017).

[110] Moritz Menze and Andreas Geiger. "Object scene flow for autonomous vehicles". In: *CVPR*. 2015.

[111] Moritz Menze, Christian Heipke, and Andreas Geiger. "Joint 3D Estimation of Vehicles and Scene Flow". In: *ISPRS Workshop on Image Sequence Analysis (ISA)*. 2015.

[112] Volodymyr Mnih et al. "Asynchronous methods for deep reinforcement learning". In: *International conference on machine learning*. 2016, pp. 1928–1937.

[113] Volodymyr Mnih et al. "Human-level control through deep reinforcement learning". In: *Nature* 518.7540 (2015), p. 529.

[114] Volodymyr Mnih et al. "Human-level control through deep reinforcement learning". In: *Nature* 518.7540 (2015), p. 529.

[115] Teodor Mihai Moldovan et al. "Optimism-driven exploration for nonlinear systems". In: *Robotics and Automation (ICRA), 2015 IEEE International Conference on*. IEEE. 2015, pp. 3239–3246.

[116] Igor Mordatch et al. "Combining model-based policy search with online model learning for control of physical humanoids". In: *ICRA*. IEEE. 2016, pp. 242–248.

[117] Jun Morimoto and Christopher G Atkeson. "Minimax differential dynamic programming: An application to robust biped walking". In: *Advances in neural information processing systems*. 2003, pp. 1563–1570.

[118] Anusha Nagabandi et al. "Neural network dynamics for model-based deep reinforcement learning with model-free fine-tuning". In: *arXiv preprint arXiv:1708.02596* (2017).

[119] Ashvin Nair et al. "Visual Reinforcement Learning with Imagined Goals". In: *NeurIPS*. 2018.

[120] Natalia Neverova, Rıza Alp Güler, and Iasonas Kokkinos. "Dense Pose Transfer". In: *arXiv preprint arXiv:1809.01995* (2018).

[121] Andrew Y Ng. "Feature selection, L 1 vs. L 2 regularization, and rotational invariance". In: *Proceedings of the twenty-first international conference on Machine learning*. ACM. 2004, p. 78.

[122] Andrew Y Ng, Daishi Harada, and Stuart Russell. "Policy invariance under reward transformations: Theory and application to reward shaping". In: *ICML*. Vol. 99. 1999, pp. 278–287.

[123] Andrew Y Ng, Stuart J Russell, et al. "Algorithms for inverse reinforcement learning." In: *ICML*. 2000.

[124] Frank Nielsen and Richard Nock. "On the chi square and higher-order chi distances for approximating f-divergences". In: *IEEE Signal Processing Letters* 21.1 (2014), pp. 10–13.

[125] Junhyuk Oh, Satinder Singh, and Honglak Lee. "Value prediction network". In: *Advances in NeuralInformationProcessingSystems*. 2017, pp. 6118–6128.

[126] Junhyuk Oh et al. "Zero-shot task generalization with multi-task deep reinforcement learning". In: *ICML*. JMLR. org. 2017, pp. 2661–2670.

[127] Charles Packer et al. "Assessing generalization in deep reinforcement learning". In: *arXiv preprint arXiv:1810.12282* (2018).

[128] Eunbyung Park et al. "Transformation-grounded image generation network for novel 3d view synthesis". In: *CVPR*. 2017.

[129] Razvan Pascanu et al. "Learning model-based planning from scratch". In: *arXiv preprint arXiv:1707.06170* (2017).

[130] Deepak Pathak et al. "Context encoders: Feature learning by inpainting". In: *CVPR*. 2016.

[131] Deepak Pathak et al. "Curiosity-driven Exploration by Self-supervised Prediction". In: *ICML*. 2017.

[132] Deepak Pathak et al. “Curiosity-driven exploration by self-supervised prediction”. In: *CVPR workshops*. 2017, pp. 16–17.

[133] Deepak Pathak et al. “Zero-shot visual imitation”. In: *International Conference on Learning Representations*. 2018.

[134] Silvia L Pintea, Jan C van Gemert, and Arnold WM Smeulders. “Déja vu”. In: *ECCV*. 2014.

[135] Matteo Pirotta, Marcello Restelli, and Luca Bascetta. “Adaptive step-size for policy gradient methods”. In: *Advances in Neural Information Processing Systems*. 2013, pp. 1394–1402.

[136] Matteo Pirotta, Marcello Restelli, and Luca Bascetta. “Policy gradient in lipschitz markov decision processes”. In: *Machine Learning* 100.2-3 (2015), pp. 255–283.

[137] Matthias Plappert, Marcin Andrychowicz, et al. “Multi-goal reinforcement learning: Challenging robotics environments and request for research”. In: *arXiv preprint arXiv:1802.09464* (2018).

[138] Dean A Pomerleau. “Alvinn: An autonomous land vehicle in a neural network”. In: *Advances in neural information processing systems*. 1989, pp. 305–313.

[139] Vitchyr Pong et al. “Temporal Difference Models: Model-Free Deep RL for Model-Based Control”. In: *arXiv preprint arXiv:1802.09081* (2018).

[140] Vitchyr Pong et al. “Temporal Difference Models: Model-Free Deep RL for Model-Based Control”. In: *International Conference on Learning Representations* (2018).

[141] Sébastien Racanière et al. “Imagination-augmented agents for deep reinforcement learning”. In: *AdvancesinNeuralInformationProcessingSystems*. 2017, pp. 5690–5701.

[142] MarcAurelio Ranzato et al. “Video (language) modeling: a baseline for generative models of natural videos”. In: *arXiv preprint arXiv:1412.6604* (2014).

[143] Carl Edward Rasmussen and Christopher K. I. Williams. *Gaussian processes for machine learning*. 2006.

[144] Fitsum A Reda et al. “SDC-Net: Video prediction using spatially-displaced convolution”. In: *arXiv preprint arXiv:1811.00684* (2018).

[145] Scott Reed et al. “Generative adversarial text to image synthesis”. In: *arXiv preprint arXiv:1605.05396* (2016).

[146] Samuel Rota Bulò, Lorenzo Porzi, and Peter Kontschieder. “In-Place Activated BatchNorm for Memory-Optimized Training of DNNs”. In: *CVPR*. 2018.

[147] Masaki Saito, Eiichi Matsumoto, and Shunta Saito. “Temporal generative adversarial nets with singular value clipping”. In: *ICCV*. 2017.

[148] Alvaro Sanchez-Gonzalez et al. “Graph networks as learnable physics engines for inference and control”. In: *arXiv preprint arXiv:1806.01242* (2018).

[149] Igal Sason and Sergio Verdú. "f-divergence Inequalities". In: *IEEE Transactions on Information Theory* 62.11 (2016), pp. 5973–6006.

[150] Jürgen Schmidhuber. "A possibility for implementing curiosity and boredom in model-building neural controllers". In: *Proc. of the international conference on simulation of adaptive behavior: From animals to animats*. 1991, pp. 222–227.

[151] John Schulman et al. "High-dimensional continuous control using generalized advantage estimation". In: *arXiv preprint arXiv:1506.02438* (2015).

[152] John Schulman et al. "Proximal policy optimization algorithms". In: *arXiv preprint arXiv:1707.06347* (2017).

[153] John Schulman et al. "Trust region policy optimization". In: *International Conference on Machine Learning*. 2015, pp. 1889–1897.

[154] Iulian Vlad Serban et al. "The Bottleneck Simulator: A Model-based Deep Reinforcement Learning Approach". In: *arXiv preprint arXiv:1807.04723* (2018).

[155] Xingjian Shi et al. "Convolutional LSTM Network: A Machine Learning Approach for Precipitation Nowcasting". In: *NeurIPS*. 2015.

[156] David Silver et al. "Mastering the game of go without human knowledge". In: *Nature* 550.7676 (2017), p. 354.

[157] Max Simchowitz et al. "Learning Without Mixing: Towards A Sharp Analysis of Linear System Identification". In: *arXiv preprint arXiv:1802.08334* (2018).

[158] Karen Simonyan and Andrew Zisserman. "Very deep convolutional networks for large-scale image recognition". In: *arXiv preprint arXiv:1409.1556* (2014).

[159] Sungryull Sohn, Junhyuk Oh, and Honglak Lee. "Hierarchical reinforcement learning for zero-shot generalization with subtask dependencies". In: *Neurips*. 2018, pp. 7156–7166.

[160] Aravind Srinivas et al. "Universal Planning Networks". In: *arXiv preprint arXiv:1804.00645* (2018).

[161] Nitish Srivastava, Elman Mansimov, and Ruslan Salakhudinov. "Unsupervised learning of video representations using lstms". In: *International conference on machine learning*. 2015.

[162] Nitish Srivastava, Elman Mansimov, and Ruslan Salakhutdinov. "Unsupervised Learning of Video Representations using LSTMs". In: *ICML*. 2015.

[163] Alexander L Strehl and Michael L Littman. "An analysis of model-based interval estimation for Markov decision processes". In: *Journal of Computer and System Sciences* 74.8 (2008), pp. 1309–1331.

[164] Wen Sun et al. "Dual Policy Iteration". In: *arXiv preprint arXiv:1805.10755* (2018).

[165] Richard S Sutton. "Dyna, an integrated architecture for learning, planning, and reacting". In: *ACM SIGART Bulletin* 2.4 (1991), pp. 160–163.

[166] Richard S Sutton. "Integrated architectures for learning, planning, and reacting based on approximating dynamic programming". In: *Machine Learning Proceedings 1990.* Elsevier, 1990, pp. 216–224.

[167] Richard S Sutton et al. "Dyna-style planning with linear function approximation and prioritized sweeping". In: *arXiv preprint arXiv:1206.3285* (2012).

[168] Umar Syed and Robert E Schapire. "A game-theoretic approach to apprenticeship learning". In: *Advances in neural information processing systems.* 2008, pp. 1449–1456.

[169] István Szita and Csaba Szepesvári. "Model-based reinforcement learning with nearly tight exploration complexity bounds". In: *Proceedings of the 27th International Conference on Machine Learning (ICML-10).* 2010, pp. 1031–1038.

[170] Aviv Tamar, Dotan Di Castro, and Ron Meir. "Integrating a partial model into model free reinforcement learning". In: *Journal of Machine Learning Research* 13.Jun (2012), pp. 1927–1966.

[171] Haoran Tang, Rein Houthooft, et al. "# Exploration: A study of count-based exploration for deep reinforcement learning". In: *Neurips.* 2017, pp. 2753–2762.

[172] Yichuan Charlie Tang and Ruslan Salakhutdinov. "Multiple Futures Prediction". In: *NIPS.* 2019.

[173] Voot Tangkaratt et al. "Model-based policy gradients with parameter-based exploration by least-squares conditional density estimation". In: *Neural networks* 57 (2014), pp. 128–140.

[174] Emanuel Todorov, Tom Erez, and Yuval Tassa. "Mujoco: A physics engine for model-based control". In: *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on.* IEEE. 2012, pp. 5026–5033.

[175] Sergey Tulyakov et al. "MoCoGAN: Decomposing Motion and Content for Video Generation". In: *CVPR.* 2018.

[176] Sergey Tulyakov et al. "Mocogan: Decomposing motion and content for video generation". In: *ICLR.* 2018.

[177] Thomas Unterthiner et al. "Towards Accurate Generative Models of Video: A New Metric & Challenges". In: *CoRR* (2018).

[178] Ruben Villegas et al. "Decomposing Motion and Content for Natural Video Sequence Prediction". In: *ICLR.* 2017.

[179] Ruben Villegas et al. "Decomposing motion and content for natural video sequence prediction". In: *ICLR.* 2017.

[180] Ruben Villegas et al. "High Fidelity Video Prediction with Large Stochastic Recurrent Neural Networks". In: *NeurIPS.* Ed. by Hanna M. Wallach et al. 2019.

[181] Ruben Villegas et al. "Learning to Generate Long-term Future via Hierarchical Prediction". In: *ICML.* 2017.

[182] Ruben Villegas et al. "Learning to generate long-term future via hierarchical prediction". In: *ICML*. 2017.

[183] Jacob Walker et al. "The pose knows: Video forecasting by generating pose futures". In: *ICCV*. 2017.

[184] Dequan Wang et al. "Deep Object-Centric Policies for Autonomous Driving". In: *ICRA*. IEEE. 2019, pp. 8853–8859.

[185] Jack M. Wang, David J. Fleet, and Aaron Hertzmann. "Gaussian Process Dynamical Models". In: *NeurIPS*. 2005.

[186] Shaojun Wang et al. "The latent maximum entropy principle". In: *Proceedings IEEE International Symposium on Information Theory,* IEEE. 2002, p. 131.

[187] Ting-Chun Wang et al. "High-Resolution Image Synthesis and Semantic Manipulation with Conditional GANs". In: *CVPR*. 2018.

[188] Yunbo Wang et al. "Eidetic 3D LSTM: A Model for Video Prediction and Beyond". In: *ICLR*. 2019.

[189] Yunbo Wang et al. "PredRNN: Recurrent Neural Networks for Predictive Learning using Spatiotemporal LSTMs". In: *NIPS*. 2017.

[190] Zhou Wang et al. "Image quality assessment: from error visibility to structural similarity". In: *IEEE transactions on image processing* (2004).

[191] Nevan Wichers et al. "Hierarchical Long-term Video Prediction without Supervision". In: *ICML*. 2018.

[192] Nevan Wichers et al. "Hierarchical Long-term Video Prediction without Supervision". In: *ICML*. 2018.

[193] Ronald J Williams and Jing Peng. "Function optimization using connectionist reinforcement learning algorithms". In: *Connection Science* 3.3 (1991), pp. 241–268.

[194] Chris Xie et al. "Model-based reinforcement learning with parametrized physical models and optimism-driven exploration". In: *2016 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2016, pp. 504–511.

[195] Saining Xie, Xun Huang, and Zhuowen Tu. "Top-down learning for structured labeling with convolutional pseudoprior". In: *ECCV*. 2016.

[196] Xu Xie et al. "Learning Virtual Grasp with Failed Demonstrations via Bayesian Inverse Reinforcement Learning". In: *IROS*. 2019.

[197] Jingwei Xu, Bingbing Ni, and Xiaokang Yang. "Video Prediction via Selective Sampling". In: *NeurIPS*. 2018.

[198] Jingwei Xu et al. "Deep Kinematics Analysis for Monocular 3D Human Pose Estimation". In: *CVPR*. 2020.

[199] Jingwei Xu et al. "Structure Preserving Video Prediction". In: *CVPR*. 2018.

[200] Yichao Yan et al. "Skeleton-Aided Articulated Motion Generation". In: *ACM MM*. 2017.

[201] Yufei Ye et al. "Compositional Video Prediction". In: *ICCV*. 2019.

[202] Zhichao Yin and Jianping Shi. "GeoNet: Unsupervised Learning of Dense Depth, Optical Flow and Camera Pose". In: *CVPR*. 2018.

[203] Michael C Yip and David B Camarillo. "Model-less feedback control of continuum manipulators in constrained environments". In: *IEEE Transactions on Robotics* 30.4 (2014), pp. 880–889.

[204] Hang Zhang et al. "Context encoding for semantic segmentation". In: *CVPR*. 2018.

[205] Richard Zhang et al. "The unreasonable effectiveness of deep features as a perceptual metric". In: *arXiv preprint* (2018).

[206] Weiyu Zhang, Menglong Zhu, and Konstantinos G. Derpanis. "From Actemes to Action: A Strongly-Supervised Representation for Detailed Action Understanding". In: *ICCV*. 2013.

[207] Guangxiang Zhu, Zhiao Huang, and Chongjie Zhang. "Object-oriented dynamics predictor". In: *Advances in Neural Information Processing Systems*. 2018, pp. 9804–9815.