OBJECT-ORIENTED MODEL LEARNING THROUGH MULTI-LEVEL ABSTRACTION

Anonymous authors
Paper under double-blind review

ABSTRACT

Object-based approaches for learning action-conditioned dynamics has demonstrated promise of strong generalization and interpretability. However, existing approaches suffer from structural limitations and optimization difficulties for common environments with multiple dynamic objects. In this paper, we present a novel self-supervised learning framework, called *Multi-level Abstraction Object-oriented Predictor* (MAOP), for learning object-based dynamics models from raw visual observations. MAOP employs three-level learning archicture that enables efficient dynamics learning for complex environments with a dynamic background. We also design a spatial-temporal relational reasoning mechanism to support instance-level dynamics learning and handle partial observability. Empirical results show that MAOP significantly outperforms previous methods in terms of sample efficiency and generalization over novel environments that have multiple controllable and uncontrollable dynamic objects and different static object layouts. In addition, MAOP learns semantically and visually interpretable disentangled representations.

1 INTRODUCTION

Model-based deep reinforcement learning (DRL) (Racanière et al., 2017; Chiappa et al., 2017; Finn & Levine, 2017) has recently attracted much attention for improving sample efficiency of DRL. One of the core problems for model-based reinforcement learning is to learn action-conditioned dynamics models by interacting with environments. Approaches have been proposed for such dynamics learning from raw visual perception, achieving remarkable performance in training environments (Oh et al., 2015; Watter et al., 2015; Chiappa et al., 2017).

To unlock sample efficiency of model-based DRL, learning action-conditioned dynamics models that generalize over unseen environments is critical yet challenging. (Finn et al., 2016) proposed an action-conditioned video prediction method that explicitly models pixel motion and thus is partially invariant to object appearances. Zhu & Zhang (2018) developed an object-oriented dynamics predictor, taking a further step towards generalization over unseen environments with different object layouts. However, due to structural limitations and optimization difficulties, these methods do not learn and generalize well for common environments with a dynamic background, which contain multiple moving objects in addition to controllable objects.

To address these limitations, this paper presents a novel self-supervised, object-oriented dynamics learning framework, called *Multi-level Abstraction Object-oriented Predictor* (MAOP). This framework simultaneously learns disentangled object representations and predicts object motions conditioned on their historical states, their interactions to other objects, and an agent's actions. To reduce the complexity of such concurrent learning and improve sample efficiency, MAOP employs a three-level learning architecture from the most abstract level of motion detection, to dynamic instance segmentation, and to dynamics learning and prediction. The more abstract learning level solves an easier problem and has lower learning complexity, and its output provides a coarse-grained guidance for the less abstract learning level, improving its speed and quality of learning convergence. This multi-level architecture is inspired by humans' multi-level motion perception from cognitive science studies (Johansson, 1975; Lu & Sperling, 1995; Smith et al., 1998) and multi-level abstraction search in constraint optimization (Zhang & Shah, 2016). In addition, we design a novel spatial-temporal relational reasoning mechanism, which includes a CNN-based Relation Net to reason about spatial

relations between objects and an Inertia Net to learn temporal effects. This mechanism offers a disentangled way to handle physical reasoning in the setting of partial observation.

Empirical results show that MAOP significantly outperforms previous methods in terms of sample efficiency and generalization over novel environments that have multiple controllable and uncontrollable dynamic objects and different object layouts. It can learn from few examples and accurately predict the dynamics of objects as well as raw visual observations in previously unseen environments. In addition, MAOP learns disentangled representations and gains fruitful semantically and visually interpretable knowledge, including meaningful object masks, accurate object motions, disentangled reasoning process, and discovery of controllable agent.

2 Related Work

Object-oriented reinforcement learning has received much research attention, which exploits efficient representations based on objects and their interactions. This learning paradigm is close to that of human cognition in the physical world and the learned object-level knowledge can be robustly generalized across environments. Early work on object-oriented RL requires explicit encodings of object representations and relations, such as *relational MDPs* (Guestrin et al., 2003), *OO-MDPs* (Diuk et al., 2008) and Object focused q-learning (Cobo et al., 2013). In this paper, we present an end-to-end, self-supervised neural network framework that automatically learns object representations and dynamics conditioned on actions and object relations from raw visual observations.

Action-conditioned dynamics learning aims to address one of the core problems for model-based DRL, that is, constructing an environment dynamics model. Several approaches have been proposed for learning how an environment changes in response to actions through unsupervised video prediction and achieve remarkable performance in training environments (Oh et al., 2015; Watter et al., 2015; Chiappa et al., 2017). Finn et al. (2016) develops a dynamics prediction method that explicitly models pixel motions and is partially invariant to object appearances, and its usage for model-based DRL is demonstrated with model predictive controller (Finn & Levine, 2017). Recently, Zhu & Zhang (2018) proposes an object-oriented dynamics learning paradigm that enables its learned model to generalize over unseen environments with different object layouts and be robust to changes of object appearances. However, this paradigm focuses environments with a single dynamic object. In this paper, we take a further step towards learning object-oriented dynamics model in more general environments with multiple controlled and uncontrollable dynamic objects. In addition, we design an instance-aware dynamics mechanism to support instance-level dynamics learning and handle partial observations.

Relation-based deep learning approaches make significant progress in a wide range of domains such as physical reasoning (Chang et al., 2016; Battaglia et al., 2016), computer vision (Watters et al., 2017; Wu et al., 2017), natural language processing Santoro et al. (2017), and reinforcement learning (Zambaldi et al., 2018; Zhu & Zhang, 2018). Relation-based nets introduce relational inductive biases into neural networks, which facilitate generalization over entities and relations and enables relational reasoning (Battaglia et al., 2018). (Zhu & Zhang, 2018) is similar to this paper, learning object-level dynamics conditioned on actions and object-to-object relations. In contrast to (Zhu & Zhang, 2018), this paper proposes a novel spatial-temporal relational reasoning mechanism, which includes an Inertia Net for learning temporal effects in addition to a CNN-based Relation Net for reasoning about spatial relations. This mechanism offers a disentangled way to handle physical reasoning in the setting of partial observability.

Instance Semantic Segmentation has been one of the fundamental problems in computer vision. Instance segmentation can be regarded as the combination of semantic segmentation and object localization. Many approaches have been proposed for instance segmentation, including DeepMask (Pinheiro et al., 2015), InstanceFCN (Dai et al., 2016), FCIS (Li et al., 2017), and Mask R-CNN He et al. (2017). Most models are supervised learning and require a large labeled training dataset. Liu et al. (2015) proposes a weakly-supervised approach to infer object instances in foreground by exploiting dynamic consistency in video. In this paper, we design a self-supervised, three-level instance segmentation approach for learning dynamic instance masks. At the most abstract level, the foreground detection module provides a coarse-grained guidance for producing region proposals at the instance segmentation level. The instance segmentation level then learns coarse instance segmentation. This coarse instance segmentation provides a guidance for learning accurate instance

masks at the dynamics learning level, whose instance segmentation not only considers object appearances and dynamics, but also motion prediction conditioned on object-to-object relations and actions.

3 MULTI-LEVEL ABSTRACTION OBJECT-ORIENTED PREDICTOR (MAOP)

In this section, we will present a novel self-supervised deep learning framework, aiming to learn object-oriented dynamics models that are able to generalize over unseen environments with different object layouts and multiple dynamic objects. Such an object-oriented dynamics learning approach requires simultaneously learning object representations and motions conditioned on their historical states, their interactions to other objects, and an agent's actions. This concurrent learning is very challenging for an end-to-end approach in complex environments with a dynamic background. Evidences from cognitive science studies (Johansson, 1975; Lu & Sperling, 1995; Smith et al., 1998) show that human beings are born with prior motion perception ability (in Cortical area MT) of perceiving moving and motionlessness, which enables learning more complex knowledge, such as object-level dynamics prediction. Inspired by these studies, we design a multi-level learning framework, called *Multi-level Abstraction Object-oriented Predictor* (MAOP), which incorporates motion perception levels to assist in dynamics learning.

Figure 1 illustrates three levels of the MAOP framework: dynamics learning, dynamic instance segmentation, and motion detection. The dynamics learning level is an end-to-end, self-supervised neural network, aiming to learn object representations and instance-level dynamics, and predict the next visual observation conditioned on an agent's action. To guide the learning of the object representations and instance localization at the dynamics learning level, the dynamic instance segmentation level provides coarse instant mask proposals in a self-supervised manner. It only utilizes the spatial-temporal information of locomotion property and appearance pattern to capture the region proposals of dynamic instances. To facilitate the learning of dynamic instance segmentation, MAOP employs the more coarse-grained motion detection level, which detects changes in image sequences and provides guidance on proposing regions potentially containing dynamic instance. As the learning proceeds, the knowledge distilled from the more coarse-grained level are gradually refined at the more fine-grained level by considering additional information. When the training is finished, the coarse-grained levels of dynamic instance segmentation and motion detection will be removed at the testing stage.

The multi-level learning design of MAOP shares some similar idea with the multi-level abstraction search approach (MASA) in constraint optimization (Zhang & Shah, 2016), which also employs multi-level course-gained abstract problems and greatly improves the search quality and speed in the original complex problem space. Two observations may explain why such multi-level abstraction approaches work (Zhang & Shah, 2016). First, the optimization in a coarse-grained abstract problem considers an easier objective and less information and thus is computationally cheap, but its output still provides an effective guidance for solving the original problem. Second, a coarse-grained abstract problem often has a smoother search space surface, avoiding earlier convergence to inferior local optima in the original search space.

In the rest of this section, we will describe in detail the design of each level of MAOP and their connections.

3.1 OBJECT-ORIENTED DYNAMICS LEARNING

The semantics of the dynamics learning level can be formulated as learning an object-based dynamics model with the attention proposals generated from the dynamic instance segmentation level. The learning architecture is shown at the top level of Figure 1. It is an end-to-end neural network and trained in a self-supervised manner. It learns disentangled representations (including objects, relations and effects) and follows an object-oriented paradigm. It takes multiple-frame video images and an agent's actions as input, learns the dynamics of controllable and uncontrollable dynamic object instances conditioned on the actions and object-to-object relations, and produce the predictions of raw visual observations. The dynamics learning network has four major components: A) an Object Detector that decomposes the input image into objects; B) an Instance Localization module responsible for localizing dynamic instances; C) a Dynamics Net for learning the motion of each dynamic

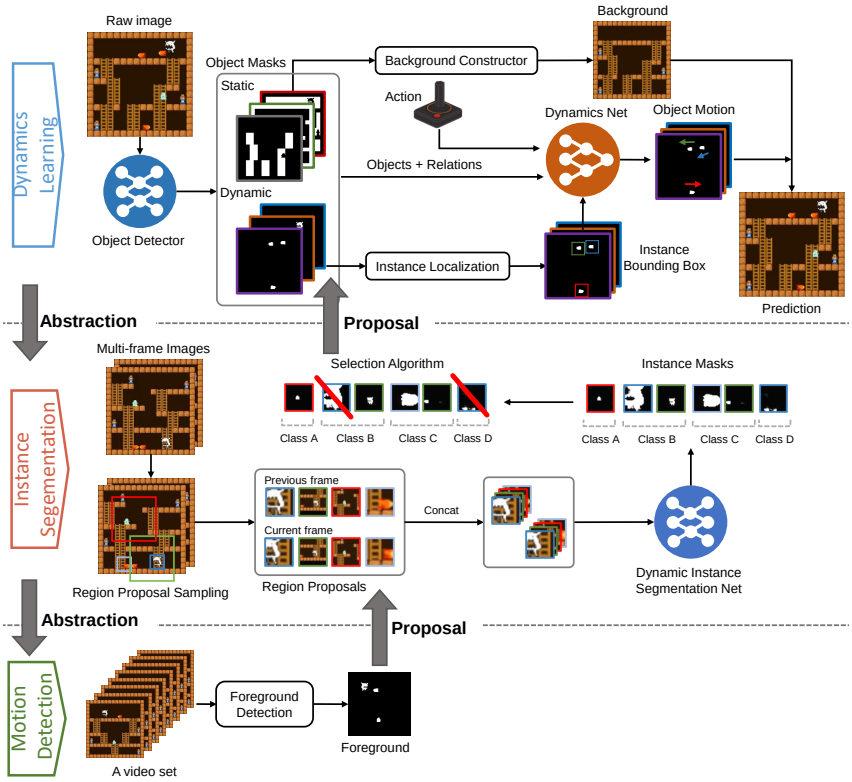

Figure 1: Multi-level abstraction framework from a top-down decomposition view.

instance conditioned on the effects from actions and object-to-object relations; and D) a Background Constructor that computes the background image from the learned static object masks.

Object Detector and Instance Localization Module. Object Detector learns to decompose the input image into object masks. An object mask describes the spatial distribution of an object, which forms the fundamental building block of our object-oriented framework. (Zhu & Zhang, 2018) presents a possible formulation of object masks mainly based on visual appearances by using CNN modules. However, this formulation suffers from limitations in scenes with multiple dynamic instances, because it is hard to distinguish two instances with a similar visual appearance but different motions and to divide them into two different masks. To address this issue, we introduce a new formulation of object masks, each of which either represents one dynamic instance or one class of static objects, and incorporate Object Detector with an Instance Localization Module to localize each dynamic instance to support instance-level dynamics learning. Instance localization is also a common technique in the area of region-based object detection (e.g. (Girshick et al., 2014), (Girshick, 2015), (Ren et al., 2015), (He et al., 2017)). The class-specific object masks in conjunction with instance localization build the bridge to connect visual perception (Object Detector) with dynamics learning (Dynamics Net), which allows learning objects from appearances, interactions, and dynamics.

Specifically, Object Detector receives image $\mathbf{I}^{(t)} \in \mathbb{R}^{H \times W \times 3}$ at timestep t and then outputs object masks $\mathbf{O}^{(t)} \in [0,1]^{H \times W \times n_O}$ (including dynamic object masks $\mathbf{D}^{(t)} \in [0,1]^{H \times W \times n_D}$ and static object masks $\mathbf{S}^{(t)} \in [0,1]^{H \times W \times n_S}$) , where n_D and n_S denotes the maximum class number of dynamic and static objects respectively, and $n_O = n_D + n_S$. Note that Object Detector uses the same CNN architecture with OODP. Then, Instance Localization module uses object masks produced by Object Detector to compute each single instance mask $\mathbf{M}^{(t)}_{:,:,i} \in [0,1]^{H_M \times W_M} (1 \leq i \leq n_M)$ through region proposal sampling and instance mask selection (similar with Section 3.2), where n_M denotes the maximum number of localized instances.

Dynamics Net. Dynamics Net is designed to be a disentangled knowledge learner, which learns instance-based motion effects of actions and object-to-object relations (using Relation Net) and

historical states (using Inertia Net), and then reasons about the motion of each dynamic instance based on these effects. Its architecture is illustrated in Figure 2. As shown in the left subfigure, instance-level dynamics learning is performed, which means the motion of each dynamic instance is individually computed. We take as an example the computation of the motion of the i-th instance $\mathbf{M}^{(t)}_{:,:,i}$ to show the detailed structure of the Effect Net module.

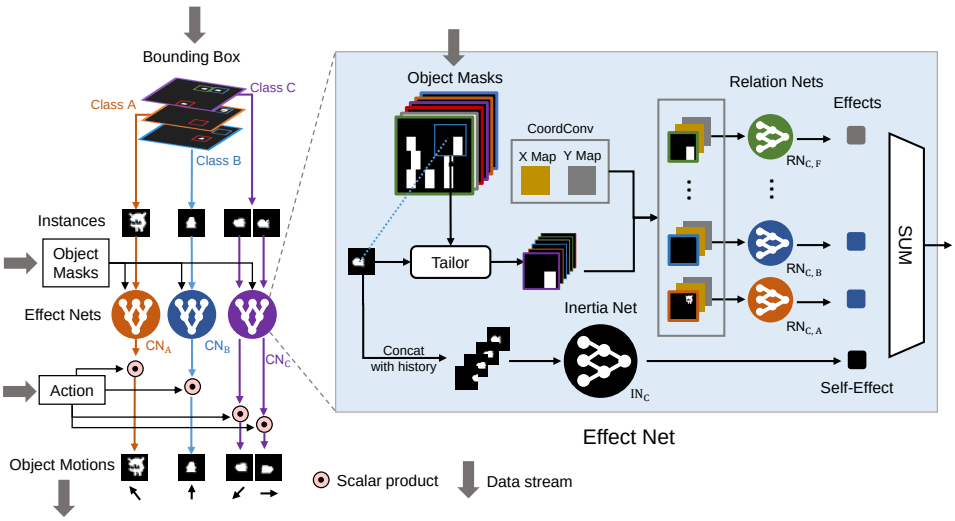

Figure 2: Architecture of Dynamics Net (left) and Effect Net (right). Different classes are distinguished by different letters (e.g., A, B, ... , F).

As shown in the right subfigure of Figure 2, we first use a sub-differentiable tailor module to enable the inference of dynamics focusing on the relations with neighbour objects. This module crops a w-size "horizon" window from the concatenated masks of all objects $\mathbf{O}^{(t)}$ centered on the expected location of $\mathbf{M}^{(t)}_{:,:,i}$, where w denotes the maximum effective range of relations. The details of the sub-differentiable cropping process can be found in (Zhu & Zhang, 2018). Then, the cropped object masks are respectively concatenated with the constant x-coordinate and y-coordinate meshgrid map and fed into the corresponding Relation Nets (RN) according to their classes. We use $\mathbf{C}^{(t)}_{:,:,i,j}$ to denote the cropped mask that crops the j-th class object $\mathbf{O}^{(t)}_{:,:,j}$ centered on the expected location of the i-th dynamic instance (the class it belongs to is denoted as $c_i, 1 \leq c_i \leq n_D$). The effects of object class j on class c_i, $E^{(t)}(c_i, j) \in \mathbb{R}^{2\times n_a}$ (n_a denotes the number of actions) is calculated as,

$$E^{(t)}(c_i, j) = \mathrm{RN}_{c_i,j}\Big(\mathrm{concat}\big(\mathbf{C}^{(t)}_{:,:,i,j}, \mathrm{Xmap}, \mathrm{Ymap}\big)\Big).$$

Note that there are total $n_D \times n_O$ RNs for $n_D \times n_O$ pairs of object classes that share the same architecture but not their weights. To handle the partial observation problem, we add an Inertia Nets (IN) to learn the self-effects of an object class through historical states, that is,

$$E^{(t)}_{\mathrm{self}}(c_i) = \mathrm{IN}_{c_i}\Big(\mathrm{concat}\big(\mathbf{M}^{(t)}_{:,:,i}, \mathbf{M}^{(t+1)}_{:,:,i}, \mathbf{M}^{(t+2)}_{:,:,i}, \ldots, \mathbf{M}^{(t+H)}_{:,:,i}\big)\Big).$$

where H is the history length and there are total n_D INs for n_D dynamic object classes that share the same architecture but not their weights. To predict the motion vector $\boldsymbol{m}^{(t)}_i \in \mathbb{R}^2$ for the i-th dynamic instance, all these effects are summed up and then multiplied by the one-hot coding of action $\boldsymbol{a}^{(t)} \in \{0, 1\}^{n_a}$, that is,

$$\boldsymbol{m}^{(t)}_i = \Big(\big(\sum_j E^{(t)}(c_i, j)\big) + E^{(t)}_{\mathrm{self}}(c_i)\Big) \cdot a^{(t)}.$$

Background Constructor. This module uses static object masks learned by Object Detector and composes the static background of its observations, which is then used with predicted motions of

dynamic instances to predict the next visual observation. As Object Detector can decompose its observation into objects in an unseen environment with a different object layout, Background Constructor is able to generate a corresponding static background and support the visual observation prediction in novel environments. Specifically, Background Constructor maintains an external background memory $\mathbf{B} \in \mathbb{R}^{H \times W \times 3}$ which is continuously updated (via moving average) by the static object mask learned by Object Detector. The updating formula is given by,

$$\mathbf{B}^{(t)} = \begin{cases} \alpha \mathbf{B}^{(t-1)} + (1-\alpha)\mathbf{I}^{(t)} \sum_i \mathbf{S}^{(t)}_{:,:,i}, & t > 0; \\ 0, & t = 0, \end{cases}$$

where $\mathbf{I}^{(t)}$ is the current frame and α is the decay rate.

Prediction and Training Loss. At the output end of our model, the prediction of the next frame is produced by merging the learned object motions and the background $\mathbf{B}^{(t)}$. We use a similar STN-based merging process as (Zhu & Zhang, 2018). The pixels of a dynamic region indicated by the i-th instance mask can be calculated as $\mathbf{I}^{(t)} * \mathbf{M}^{(t)}_{:,:,i}$, where $*$ denotes masking an image with a mask. Then we use Spatial Transformer Network (STN) Jaderberg et al. (2015) to apply the learned instance motion vector $\boldsymbol{m}^{(t)}_i$ on this dynamic region (denoted by $\text{STN}(\mathbf{I}^{(t)} * \mathbf{M}^{(t)}_{:,:,i}, \boldsymbol{m}^{(t)}_i)$). First, we transform all the dynamic instances to obtain the pixels of dynamic instances in the next frame, that is, $\hat{\mathbf{I}}^{(t+1)}_D = \sum_i \text{STN}(\mathbf{I}^{(t)} * \mathbf{M}^{(t)}_{:,:,i}, \boldsymbol{m}^{(t)}_i)$. Second, we compute the rest pixels from background $\mathbf{B}^{(t)}$, that is, $\hat{\mathbf{I}}^{(t+1)}_S = \big(1 - \sum_i \text{STN}(\mathbf{M}^{(t)}_{:,:,i}, \boldsymbol{m}^{(t)}_i)\big) * \mathbf{B}^{(t)}$. Thus, the prediction of the next frame $\hat{I}^{(t+1)}$, is calculated by,

$$\hat{I}^{(t+1)} = \hat{I}^{(t+1)}_D + \hat{I}^{(t+1)}_S = \sum_i \text{STN}(\mathbf{I}^{(t)} * \mathbf{M}^{(t)}_{:,:,i}, \boldsymbol{m}^{(t)}_i) + \big(1 - \sum_i \text{STN}(\mathbf{M}^{(t)}_{:,:,i}, \boldsymbol{m}^{(t)}_i)\big) * \mathbf{B}^{(t)}.$$

In this paper, we use l_2 pixel loss to restrain image prediction error, $\mathcal{L}_{\text{prediction}} = \big\| \hat{I}^{(t+1)} - I^{(t+1)} \big\|^2_2$. To get earlier feedback before reconstructing images and facilitate the training process, we add a highway loss here, $\mathcal{L}_{\text{highway}} = \sum_i \big\| (\bar{u}_i, \bar{v}_i)^{(t)} + \boldsymbol{m}^{(t)}_i - (\bar{u}_i, \bar{v}_i)^{(t+1)} \big\|^2_2$, where $(\bar{u}_i, \bar{v}_i)^{(t)}$ is the excepted location of i-th instance mask $\mathbf{M}^{(t)}_{:,:,i}$. Finally, we add a proposal loss to utilize the dynamic instance proposals provided by the abstracted problem to guide our optimization, which is given by, $\mathcal{L}_{\text{proposal}} = \big\| \sum_i (\mathbf{O}^{(t)}_{:,:,i} - \mathbf{P}^{(t)}_{:,:,i}) \big\|^2_2$, where $\mathbf{P}$ denotes the region proposals of object masks. The total loss is given by combining the previous losses with different weights,

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{highway}} + \lambda_1 \mathcal{L}_{\text{prediction}} + \lambda_2 \mathcal{L}_{\text{proposal}}$$

3.2 Dynamic Instance Segmentation

The level of dynamic instance segmentation aims to generate region proposals of dynamic instances in a self-supervised manner to guide the learning of object masks and provide initial signals for instance localization at the dynamics learning level. This level captures dynamic instances only based on the spatial-temporal information of locomotion property and appearance pattern without using labeled data, and thus may generate noisy coarse region proposals of dynamic instances. The process of dynamic instance segmentation takes a sequence of image frames as input and is guided by the foreground region proposals generated by the Motion Detection level.

The dynamic instance segmentation module is composed of three parts. The first part generates region proposals which define the set of candidate regions for the following instance segmentation. The second part is Dynamic Instance Segmentation Network which employs object appearance and dynamics consistency to derive the instance segmentation. The third part is instance mask selection which is similar to non-maximum suppression (NMS) (Ren et al., 2015). This part tries to eliminate redundant and noisy instance masks. Although the entire process is analogous to the region-based object detector method (Liu et al., 2018), it focuses on generalizing detection results over different object layouts and exploits instance-level coherence without using labeled data.

Region proposal sampling. The advance of region proposal techniques (e.g. sliding window (Dalal & Triggs, 2005), selective search (Uijlings et al., 2013), region proposal network (RPN) (Ren et al., 2015)) is critical for the development of object detection methods (Liu et al., 2018). It is challenging

to quickly generate accurate region proposals without using labeled data. To address this challenge, we design a novel multi-scale learning-free sampling algorithm for region proposal. This algorithm generates multi-scale region proposals with a full coverage over the input mask. It effectively utilizes foreground mask proposals generated from the Motion Detection level to encourage its attentions on the dynamic regions. This algorithm is also used to localize every single instance in the dynamic object masks learned by Object Detector, which is described in Section 3.1. The detailed algorithm is described in Appendix A.

Dynamic Instance Segmentation Network (DISN). The dynamic instance segmentation network receives the sampled instance candidate proposals and segments instances and background by integrating appearance and dynamics property. It consists of two parts: Foreground Detector and Instance Detector. Foreground Detector learns to decompose the region proposal into foreground mask $\mathbf{F}^{(t)} \in [0,1]^{H_R \times W_R}$ and background mask $\mathbf{B}^{(t)} = 1 - \mathbf{F}^{(t)}$, where H_R and W_R denotes the size of sampled proposal region. To guide learning of the Foreground Detector, we use the foreground region proposals $\mathbf{F}^{(t)}_{\text{foreground}}$ from the third abstraction by adding a l_2 loss, $\mathcal{L}_{\text{foreground}} = \|\mathbf{F}^{(t)} - \mathbf{F}^{(t)}_{\text{foreground}}\|_2^2$. The architecture of Foreground Detector is similar to binary-class Object Detector. The follow-up Instance Detector also uses the same architecture to decompose the foreground mask $\mathbf{F}^{(t)}$ into different semantic masks $\mathbf{M}^{(t)} \in [0,1]^{H_R \times W_R \times n_O}$, where n_O is the maximum number of object classes. Since DISN leans semantic segmentations in the region proposals of instance candidates, the learned semantic masks are exactly instance masks. We denote the i-th instance by $\mathbf{M}^{(t)}_{:,:,i}$. CNNs offer the ability to identify object appearance, and we also introduce an instance-consistent loss to consider the dynamics property in the meanwhile, which is defined as,

$$\mathcal{L}_{\text{instance}} = l_{\text{consistent}}(\mathbf{I}^{(t-1)}, \mathbf{I}^{(t)}, \mathbf{B}^{(t-1)}, \mathbf{B}^{(t)}) + \sum_k l_{\text{consistent}}(\mathbf{I}^{(t-1)}, \mathbf{I}^{(t)}, \mathbf{M}^{(t-1)}_{:,:,k}, \mathbf{M}^{(t)}_{:,:,k}).$$

This loss function is based on two frames of images $\mathbf{I}^{(t-1)}, \mathbf{I}^{(t)}$, and the function $l_{\text{consistent}}(\cdot)$ (Equation 1) formally specifies instance temporal-spatial coherence.

Instance mask selection. Instance mask selection is designed to remove the redundant and noisy instance masks. Inspired by Non-maximum suppression (NMS) and selective search (SS) (Uijlings et al., 2013) in region-based object detection (e.g. (Girshick et al., 2014), (Girshick, 2015), (Ren et al., 2015), (He et al., 2017)), we propose a novel instance mask selection, which integrates the instance coherence quantification, overlapping discount and multi-scale detection approach to screen out high-consistency and non-overlapping instance masks from a large number of instance mask candidates. The details of the selection algorithm are discussed in Appendix B.

Instance-consistent loss and selection score function. We introduce a novel instance-consistent loss that considers the spatial-temporal motion consistency of instances. This loss is used to train DISN and a similar variant (see Appendix B) serves as a metric score for selecting instance masks. We use STN to estimate the spatial transformation of an instance in two adjacent frames. We denote $\mathbf{T}' = \text{STN}(\mathbf{T}, \boldsymbol{v})$ as applying the motion vectors $\boldsymbol{v}$ and on the input tensor $\mathbf{T}$ and $\mathbf{T}'$ is the transformed tensor. Conversely, the inverse function is denoted as $\boldsymbol{v} = \text{STN}^{-1}(\mathbf{T}, \mathbf{T}')$ where STN^{-1} function estimates the motion vector $\boldsymbol{v}$ from the two adjacent frames. The instance-consistent loss $l_{\text{consistent}}(\mathbf{I}, \mathbf{I}', \mathbf{M}, \mathbf{M}')$ is defined as,

$$l_{\text{consistent}}(\mathbf{I}, \mathbf{I}', \mathbf{M}, \mathbf{M}') = \|\text{STN}(\mathbf{I}, \text{STN}^{-1}(\mathbf{M}, \mathbf{M}')) - \mathbf{I}'\|_2^2 \cdot \max(\text{STN}(\mathbf{M}, \text{STN}^{-1}(\mathbf{M}, \mathbf{M}')), \mathbf{M}'), \quad (1)$$

where $\max()$ is element-wise max and we choose the translation form of STN for our experiment.

3.3 Motion Detection

At this level, we employ foreground detection to detect changing regions from a sequence of image frames and provide coarse dynamic region proposals for assisting in dynamic instance segmentation. In our experiments, we use a simple unsupervised foreground detection approach proposed by (Lo & Velastin, 2001). Our MAOP framework is also compatible with many advanced unsupervised foreground detection methods (Lee, 2005; Zhou et al., 2013; Guo et al., 2014; Maddalena et al., 2008) that are more efficient or more robust to moving camera. These complex unsupervised foreground detection methods have the potential to improve the performance but are not the focus of this work.

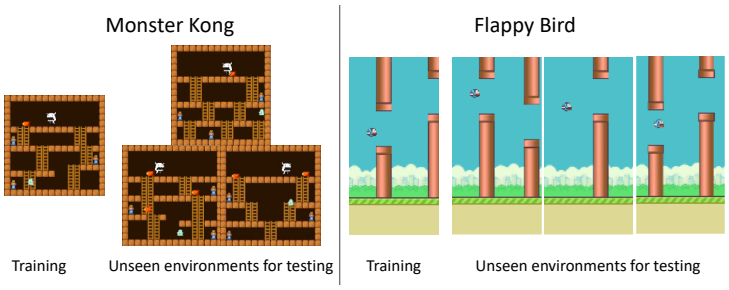

Figure 3: Examples of 1-to-3 generalization experiments.

4 EXPERIMENTS

We evaluate our model on two games, *Monster Kong* and *Flappy Bird*, from the Pygame Learning Environment (Tasfi, 2016), which allows us to test generalization ability over various scenes with different layouts. Here, the *Monster Kong* is the advanced version of that used by Zhu & Zhang (2018) which has a more general and complex setting. The monster wanders around and breathes out fires randomly, and the fires also move with some randomness. The agent explores with actions *up*, *down*, *left*, *right*, *jump* and *noop*. All these dynamic objects interact with the environments and objects according to their own physics engine. Moreover, gravity and jump model has a long-term dynamics effects, leading to a partial observation problem. To test whether our model can truly learn the underlying physical mechanism behind the visual observations and perform relational reasoning, we set the k-to-m generalization experiment, where we use k different environments for training and m different unseen environments for testing. *Flappy Bird* is a side-scroller game, where a bird flies between columns of green pipes with action *jump* and *noop*. Since the unseen environments will be similar with the training ones without limitation of samples in this game, we limit the samples for training. Two experimental settings are shown in Figure 3. We use random exploration on *Monster Kong*, and an expert guided random exploration on *Flappy Bird* because in this domain a totally random exploration will lead to an early death of the agent even at the very beginning. We compare MAOP with state-of-the-art action-conditioned dynamics learning baselines, AC Model (Oh et al., 2015), CDNA (Finn et al., 2016), and OODP (Zhu & Zhang, 2018).

4.1 GENERALIZATION AND SAMPLE EFFICIENCY

To make a sufficient comparison with the previous methods on the generalization ability and sample efficiency of object dynamics learning and image prediction, we conduct 1-5, 2-5 and 3-5 generalization experiments with a variety of evaluation indices on *Monster Kong*. We use n-error accuracy to measure the performance of object dynamics prediction, which is defined as the proportion that the difference between the predicted and ground-true agent locations is less than n pixel. We also add an extra pixel-based measurement (denoted by object RMSE), which compares the pixel difference near dynamic objects between the predicted and ground-truth images. To evaluate the image prediction, we adopt a typical image prediction loss RMSE. In Figure 4 and C7 (Appendix C), we plot the learning curve for better visualization of the comparison.

As shown in Table 1, MAOP significantly outperforms other methods in all experiment settings in terms of generalization ability and sample efficiency of both object dynamics learning and image prediction. It can achieve 0.84 0-error accuracy, even with a single training environment, which suggests MAOP is good at relational reasoning. Although AC Model achieves high accuracy in training environments, its performance in unseen scenes is much worse, which is probably because its pixel-level inference easily leads to overfitting. CDNA performs better than AC Model in those uncontrolled objects, but still cannot deal with complicated interactions in lack of knowledge on object-to-object relations. By the structural limitation of OODP, it has innate difficulty on frames with multiple dynamic objects. We also test our model on *Flappy Bird*, where we limit the training samples to 100 and 300 to form a sufficiently challenging generalization task. As shown in Table 2, our performance is similar with that on *Monster Kong*. Our generalization ability and sample efficiency significantly outperform other baselines. Surprisingly, only 100 samples are enough to reach almost perfect 1-error accuracy.

	Models	Training environments								Unseen environments							
		1-5†		1-5		2-5		3-5		1-5†		1-5		2-5		3-5	
		Agent	All	Agent	All	Agent	All	Agent	All	Agent	All	Agent	All	Agent	All	Agent	All
0-error accuracy	MAOP	0.67	0.80	0.88	0.87	0.86	0.87	0.80	0.83	**0.60**	**0.77**	**0.81**	**0.84**	**0.85**	**0.87**	**0.80**	**0.85**
	OODP	0.24	0.17	0.18	0.16	0.22	0.17	0.26	0.20	0.20	0.16	0.20	0.15	0.18	0.15	0.26	0.18
	AC Model	0.04	0.59	0.92	0.97	0.80	0.93	0.77	0.92	0.01	0.18	0.46	0.48	0.30	0.48	0.45	0.66
	CDNA	0.30	0.66	0.41	0.76	0.42	0.78	0.44	0.74	0.31	0.55	0.37	0.59	0.40	0.71	0.41	0.70
1-error accuracy	MAOP	0.90	0.91	0.97	0.94	0.97	0.93	0.96	0.93	**0.86**	**0.90**	**0.96**	**0.93**	**0.97**	**0.93**	**0.95**	**0.93**
	OODP	0.49	0.29	0.32	0.23	0.34	0.23	0.35	0.25	0.39	0.25	0.34	0.22	0.32	0.21	0.34	0.22
	AC Model	0.07	0.63	0.98	0.99	0.95	0.98	0.94	0.98	0.02	0.34	0.65	0.69	0.52	0.67	0.66	0.77
	CDNA	0.42	0.84	0.48	0.86	0.48	0.86	0.51	0.87	0.45	0.82	0.45	0.83	0.47	0.84	0.48	0.86
2-error accuracy	MAOP	0.95	0.94	0.99	0.96	0.99	0.95	0.98	0.94	**0.95**	**0.94**	**0.98**	**0.95**	**0.99**	**0.95**	**0.98**	**0.95**
	OODP	0.67	0.47	0.44	0.37	0.46	0.32	0.49	0.39	0.60	0.43	0.48	0.34	0.43	0.31	0.46	0.36
	AC Model	0.10	0.64	0.99	0.99	0.98	0.99	0.97	0.98	0.04	0.34	0.73	0.74	0.64	0.73	0.77	0.81
	CDNA	0.50	0.86	0.52	0.87	0.53	0.88	0.54	0.88	0.53	0.85	0.47	0.84	0.50	0.86	0.51	0.87
Object RMSE	MAOP	31.99		26.65		31.68		30.33		**34.14**		**29.78**		**31.32**		**30.80**	
	OODP	65.51		66.44		66.66		64.73		67.39		67.41		67.78		64.95	
	AC Model	62.02		18.88		22.39		21.30		85.46		57.41		55.45		43.48	
	CDNA	53.89		34.99		35.26		35.94		56.31		45.34		37.59		37.80	
Image RMSE	MAOP	6.90		5.64		6.68		6.46		**7.90**		**8.60**		8.73		**6.55**	
	OODP	14.70		15.08		14.89		14.42		15.42		24.68		26.39		14.52	
	AC Model	15.99		4.12		4.78		4.69		44.92		39.46		38.07		38.12	
	CDNA	11.47		7.41		7.58		7.68		12.23		9.87		**8.10**		8.16	

Table 1: Prediction performance on *Monster Kong*. k-m means the k-to-m generalization problem. † indicates training with only 1000 samples. ALL represents all dynamic objects.

	Models	Training environments				Unseen environments			
		1-5†		1-5‡		1-5†		1-5‡	
		Agent	All	Agent	All	Agent	All	Agent	All
0-error accuracy	MAOP	**0.84**	**0.90**	**0.87**	**0.93**	**0.83**	**0.89**	**0.83**	**0.92**
	OODP	0.01	0.29	0.01	0.32	0.01	0.18	0.02	0.15
	AC Model	0.39	0.64	0.48	0.75	0.03	0.18	0.04	0.23
	CDNA	0.13	0.78	0.41	0.84	0.10	0.77	0.16	0.79
1-error accuracy	MAOP	**0.99**	**1.00**	**0.97**	**0.97**	**0.99**	**0.99**	**0.98**	**0.97**
	OODP	0.05	0.52	0.04	0.56	0.06	0.39	0.07	0.39
	AC Model	0.48	0.80	0.57	0.87	0.07	0.37	0.14	0.45
	CDNA	0.26	0.82	0.57	0.89	0.22	0.81	0.36	0.84
2-error accuracy	MAOP	**1.00**	**1.00**	**0.99**	**0.99**	**1.00**	**1.00**	**0.99**	**0.98**
	OODP	0.14	0.66	0.12	0.67	0.16	0.59	0.16	0.56
	AC Model	0.53	0.85	0.63	0.90	0.12	0.53	0.24	0.64
	CDNA	0.37	0.84	0.66	0.92	0.36	0.84	0.49	0.87

Table 2: Performance of the object dynamics prediction on 1-5 generalization problem in *Flappy Bird*. † and ‡ indicates training with only 100 and 300 samples.

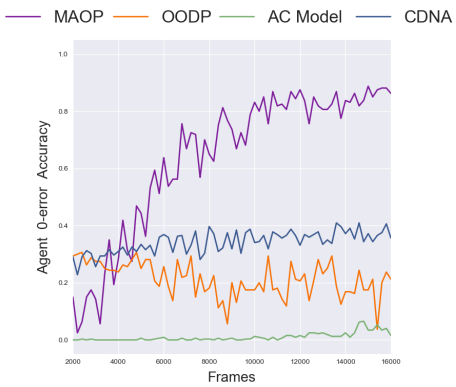

Figure 4: Learning curves.

Figure 5: Our discovery of the controllable agent.

4.2 INTERPRETABLE REPRESENTATIONS AND KNOWLEDGE

MAOP takes a step towards interpretable deep learning and disentangled representation learning. Through interacting with environments, it learns fruitful visually and semantically interpretable knowledge in an unsupervised manner, which contributes to unlock the "black box" of neural networks and open the avenue for further researches on object-based planning, object-oriented model-based RL, and hierarchical learning.

Visual interpretability. To demonstrate the visual interpretability of MAOP in unseen environments, we visualize the learned masks of dynamic and static objects. We highlight the attentions of the object masks by multiplying the raw images by the binarized masks. Note that MAOP does not require the actual number of objects but a maximum number and some learned object masks may be redundant. Thus, we only show the informative object masks. As shown in Figure 6, our model captures all the key objects in the environments including the controllable agents (the cowboy and the bird), the uncontrollable dynamic objects (the monster, fires and pipes), and the static objects that have effects on the motions of dynamic objects (ladders, walls and the free space). We also observe that model can learn disentangled object representations and distinguish the objects by both appearance and dynamic property.

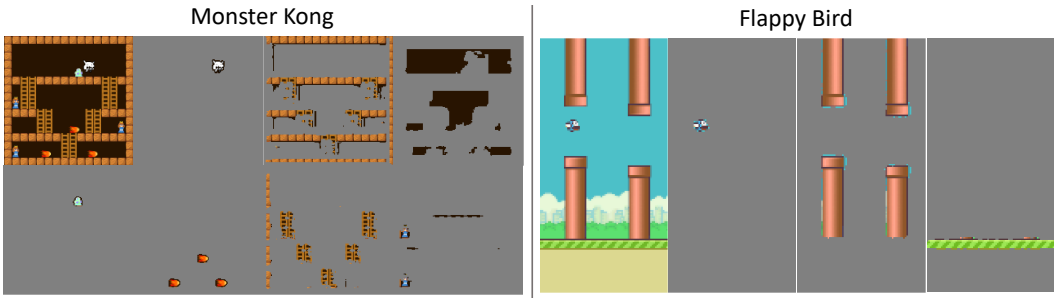

Figure 6: Visualization of the masked images in unseen environments. Left is the raw image.

Discovery of the controllable agent. With the learned knowledge in MAOP, we can easily uncover the action-controlled agent from all the dynamic objects, which is useful semantic information that can be used in heuristic algorithms. Specifically, the object that has the maximal variance of total effects over actions is the action-controlled agent. Denote the total effects as $E_i = (\sum_j E(c_i, j)) + E_{\text{self}}(c_i), E_i \in \mathbb{R}^{2\times n_a}$, the label of the action-controlled agent is calculated as, $\arg\max_i Var(E_i)$. The histogram in Figure 5 plotting the ground-truth label distribution of our discovered action-controlled agents clearly demonstrates that our discovery of the controllable agent achieves perfect 100% accuracy.

Dynamical interpretability. To show the dynamical interpretability behind image prediction, we test our predicted motions by comparing RMSEs between the predicted and ground-truth motions in unseen environments (Table C3 in Appendix C). Intriguingly, most predicted motions are quite accurate, with the RMSEs less than 1 pixel. Such a visually indistinguishable error also verifies our dynamics learning.

5 CONCLUSIONS AND FUTURE WORK

This paper presents a self-supervised multi-level learning framework for learning action-conditioned object-based dynamics. This framework is example-efficient and generalizes object dynamics and prediction of raw visual observations to complex unseen environments with multiple dynamic objects. The learned dynamics model potentially enables an agent to directly plan or efficiently learn for unseen environments. Although a random policy or an expert's policy is used for exploration in our experiments, our framework can support smarter exploration strategies, e.g., curiosity-driven exploration. Our future work includes extending our model for deformation prediction (e.g., object appearing, disappearing and non-rigid deformation) and incorporating a camera motion prediction network module for applications such as FPS games and autonomous driving.

REFERENCES

Peter Battaglia, Razvan Pascanu, Matthew Lai, Danilo Jimenez Rezende, et al. Interaction networks for learning about objects, relations and physics. In *Advances in Neural Information Processing Systems*, pp. 4502–4510, 2016.

Peter W Battaglia, Jessica B Hamrick, Victor Bapst, Alvaro Sanchez-Gonzalez, Vinicius Zambaldi, Mateusz Malinowski, Andrea Tacchetti, David Raposo, Adam Santoro, Ryan Faulkner, et al. Relational inductive biases, deep learning, and graph networks. *arXiv preprint arXiv:1806.01261*, 2018.

Michael B Chang, Tomer Ullman, Antonio Torralba, and Joshua B Tenenbaum. A compositional object-based approach to learning physical dynamics. *arXiv preprint arXiv:1612.00341*, 2016.

Silvia Chiappa, Sébastien Racaniere, Daan Wierstra, and Shakir Mohamed. Recurrent environment simulators. *International Conference on Learning Representations*, 2017.

Luis C Cobo, Charles L Isbell, and Andrea L Thomaz. Object focused q-learning for autonomous agents. In *Proceedings of the 2013 international conference on Autonomous agents and multi-agent systems*, pp. 1061–1068. International Foundation for Autonomous Agents and Multiagent Systems, 2013.

Jifeng Dai, Kaiming He, Yi Li, Shaoqing Ren, and Jian Sun. Instance-sensitive fully convolutional networks. In *European Conference on Computer Vision*, pp. 534–549. Springer, 2016.

Navneet Dalal and Bill Triggs. Histograms of oriented gradients for human detection. In *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, volume 1, pp. 886–893. IEEE, 2005.

Carlos Diuk, Andre Cohen, and Michael L Littman. An object-oriented representation for efficient reinforcement learning. In *Proceedings of the 25th international conference on Machine learning*, pp. 240–247. ACM, 2008.

Chelsea Finn and Sergey Levine. Deep visual foresight for planning robot motion. In *Robotics and Automation (ICRA), 2017 IEEE International Conference on*, pp. 2786–2793. IEEE, 2017.

Chelsea Finn, Ian Goodfellow, and Sergey Levine. Unsupervised learning for physical interaction through video prediction. In *Advances in Neural Information Processing Systems*, pp. 64–72, 2016.

Ross Girshick. Fast r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pp. 1440–1448, 2015.

Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 580–587, 2014.

Carlos Guestrin, Daphne Koller, Chris Gearhart, and Neal Kanodia. Generalizing plans to new environments in relational mdps. In *Proceedings of the 18th international joint conference on Artificial intelligence*, pp. 1003–1010. Morgan Kaufmann Publishers Inc., 2003.

Xiaojie Guo, Xinggang Wang, Liang Yang, Xiaochun Cao, and Yi Ma. Robust foreground detection using smoothness and arbitrariness constraints. In *European Conference on Computer Vision*, pp. 535–550. Springer, 2014.

Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *Computer Vision (ICCV), 2017 IEEE International Conference on*, pp. 2980–2988. IEEE, 2017.

Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International conference on machine learning*, pp. 448–456, 2015.

Max Jaderberg, Karen Simonyan, Andrew Zisserman, et al. Spatial transformer networks. In *Advances in Neural Information Processing Systems*, pp. 2017–2025, 2015.

Gunnar Johansson. Visual motion perception. *Scientific American*, 232(6):76–89, 1975.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980, 2014. URL http://arxiv.org/abs/1412.6980.

Dar-Shyang Lee. Effective gaussian mixture learning for video background subtraction. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, (5):827–832, 2005.

Yi Li, Haozhi Qi, Jifeng Dai, Xiangyang Ji, and Yichen Wei. Fully convolutional instance-aware semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 4438–4446, 2017.

Buyu Liu, Xuming He, and Stephen Gould. Multi-class semantic video segmentation with exemplar-based object reasoning. In *IEEE Winter Conference on Applications of Computer Vision (WACV)*, pp. 1014–1021. IEEE, 2015.

Li Liu, Wanli Ouyang, Xiaogang Wang, Paul Fieguth, Jie Chen, Xinwang Liu, and Matti Pietikäinen. Deep learning for generic object detection: A survey. *arXiv preprint arXiv:1809.02165*, 2018.

BPL Lo and SA Velastin. Automatic congestion detection system for underground platforms. In *Intelligent Multimedia, Video and Speech Processing, 2001. Proceedings of 2001 International Symposium on*, pp. 158–161. IEEE, 2001.

Zhong-Lin Lu and George Sperling. The functional architecture of human visual motion perception. *Vision research*, 35(19):2697–2722, 1995.

Lucia Maddalena, Alfredo Petrosino, et al. A self-organizing approach to background subtraction for visual surveillance applications. *IEEE Transactions on Image Processing*, 17(7):1168, 2008.

Junhyuk Oh, Xiaoxiao Guo, Honglak Lee, Richard L Lewis, and Satinder Singh. Action-conditional video prediction using deep networks in atari games. In *Advances in Neural Information Processing Systems*, pp. 2863–2871, 2015.

Pedro O Pinheiro, Ronan Collobert, and Piotr Dollár. Learning to segment object candidates. In *Advances in Neural Information Processing Systems*, pp. 1990–1998, 2015.

Sébastien Racanière, Théophane Weber, David Reichert, Lars Buesing, Arthur Guez, Danilo Jimenez Rezende, Adrià Puigdomènech Badia, Oriol Vinyals, Nicolas Heess, Yujia Li, et al. Imagination-augmented agents for deep reinforcement learning. In *Advances in Neural Information Processing Systems*, pp. 5694–5705, 2017.

Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems*, pp. 91–99, 2015.

Adam Santoro, David Raposo, David G Barrett, Mateusz Malinowski, Razvan Pascanu, Peter Battaglia, and Tim Lillicrap. A simple neural network module for relational reasoning. In *Advances in neural information processing systems*, pp. 4974–4983, 2017.

Andrew T Smith, Mark W Greenlee, Krish D Singh, Falk M Kraemer, and Jürgen Hennig. The processing of first-and second-order motion in human visual cortex assessed by functional magnetic resonance imaging (fmri). *Journal of Neuroscience*, 18(10):3816–3830, 1998.

Norman Tasfi. Pygame learning environment. https://github.com/ntasfi/PyGame-Learning-Environment, 2016.

Jasper RR Uijlings, Koen EA Van De Sande, Theo Gevers, and Arnold WM Smeulders. Selective search for object recognition. *International journal of computer vision*, 104(2):154–171, 2013.

Manuel Watter, Jost Springenberg, Joschka Boedecker, and Martin Riedmiller. Embed to control: A locally linear latent dynamics model for control from raw images. In *Advances in neural information processing systems*, pp. 2746–2754, 2015.

Nicholas Watters, Daniel Zoran, Theophane Weber, Peter Battaglia, Razvan Pascanu, and Andrea Tacchetti. Visual interaction networks. In *Advances in Neural Information Processing Systems*, pp. 4540–4548, 2017.

Jiajun Wu, Erika Lu, Pushmeet Kohli, Bill Freeman, and Josh Tenenbaum. Learning to see physics via visual de-animation. In *Advances in Neural Information Processing Systems*, pp. 152–163, 2017.

Vinicius Zambaldi, David Raposo, Adam Santoro, Victor Bapst, Yujia Li, Igor Babuschkin, Karl Tuyls, David Reichert, Timothy Lillicrap, Edward Lockhart, et al. Relational deep reinforcement learning. *arXiv preprint arXiv:1806.01830*, 2018.

Chongjie Zhang and Julie A Shah. Co-optimizating multi-agent placement with task assignment and scheduling. In *IJCAI*, pp. 3308–3314, 2016.

Xiaowei Zhou, Can Yang, and Weichuan Yu. Moving object detection by detecting contiguous outliers in the low-rank representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(3):597–610, 2013.

Guangxiang Zhu and Chongjie Zhang. Object-oriented dynamics predictor. *arXiv preprint arXiv:1806.07371*, 2018.

APPENDIX A REGION PROPOSAL SAMPLING

To demonstrate the detailed pipeline in the region proposal sampling, we take as an example considering the foreground mask $\mathbf{F}^{(t)}$ as our input mask. We use the multi-scale bounding boxes and multiple full coverage sampling methods to guarantee that each pixel of the foreground mask is covered in each scale coverage and then resize the region proposals into the same scale. We apply our sampling algorithm on the $\mathbf{I}^{(t)}$ and get the byproduct region proposal from $\mathbf{I}^{(t-1)}$ as well.

Algorithm 1 Region proposal sampling.

Require: foreground mask $\mathbf{F}^{(t)} \in [0,1]^{H \times W}$, foreground threshold α, density threshold β, the number of region proposal scales n_S, the number of full coverage T .

Initialize proposal set $\mathbb{P} = \emptyset$.
for $l = 1 \dots n_S$ **do**
  Select scale dx, dy depend on the level l.
  for $t = 1 \dots T$ **do**
    Initialize foreground set $\mathbb{S} = \{(i,j) | \mathbf{F}^{(t)}_{i,j} > \alpha\}$.
    while $\mathbb{S} \neq \emptyset$ **do**
      Sample a pixel coordinate (x, y) from $\mathbb{S}$.
      Get a box $\mathbb{B} = \{(i,j) | \, |i - x| \leq dx, |j - y| \leq dy\}$.
      if $\sum_{(i,j) \in \mathbb{B}} \mathbf{F}_{i,j} > \beta$ **then**
        Insert $\mathbb{B}$ into the proposal set $\mathbb{P} \leftarrow \mathbb{P} \cup \{\mathbb{B}\}$.
      end if
      Update the remain foreground set $\mathbb{S} \leftarrow \mathbb{S} \setminus \mathbb{B}$.
    end while
  end for
end for
return $\mathbb{P}$

APPENDIX B INTANCE MASK SELECTION

Algorithm 2 Intance mask selection.

Require: Overlapping threshold α; empty threshold β; the number of region proposal scales n_S; the number of region proposals in one scale n_C; the height and width sets of the multi-scale region proposals $\mathbb{H}_R, \mathbb{W}_R$ where $\mathbb{H}_R = \{H_{R_i} | i = 1, \ldots, n_S\}$ and $\mathbb{W}_R = \{W_{R_i} | i = 1, \ldots, n_S\}$; image and mask set of different scale $\mathbb{M}_S = \{(\mathbf{M}_{S_i}^{(t-1)}, \mathbf{M}_{S_i}^{(t)}) | \mathbf{M}_{S_i}^{(t-1)}, \mathbf{M}_{S_i}^{(t)} \in [0,1]^{H_{R_i} \times W_{R_i} \times n_C}, i = 1, \ldots, n_S\}$ and $\mathbb{I}_S = \{(\mathbf{I}_{S_i}^{(t-1)}, \mathbf{I}_{S_i}^{(t)}) | \mathbf{I}_{S_i}^{(t-1)}, \mathbf{I}_{S_i}^{(t)} \in [0,1]^{H_{R_i} \times W_{R_i} \times 3 \times n_C}, i = 1, \ldots, n_S\}$; selection score function $g(\cdot)$ of mask $\mathbf{M} \in [0,1]^{H_{R_u} \times W_{R_u}}$ where $u \in \{1, \ldots, n_S\}$ which is mentioned in section B; a variant of NMS (Ren et al., 2015) $f_{\mathbf{NMS}'}(\cdot)$ where $f_{\mathbf{NMS}'}(\cdot)$ calculates the overlapping area by mask instead of bounding box.

Return: The mask set of single dynamic instance $\mathbb{M}' = \{\mathbf{M}_i | \mathbf{M}_i \in [0,1]^{H_{R_x} \times W_{R_x}}; i = 1, \ldots, n_I; x \in \{1, \ldots, n_S\}\}$ where n_I denotes the number of instances.

for $i = 1, \ldots, n_S$ **do**
  Let $g'(\mathbf{M}_{S_i}^{(t)}) = g(\mathbf{M}_{S_i}^{(t-1)}, \mathbf{M}_{S_i}^{(t)}, \mathbf{I}_{S_i}^{(t-1)}, \mathbf{I}_{S_i}^{(t)})$ and define

$$g'\left(\mathbf{M}_x^{(t)} \odot \mathbf{M}_y^{(t)}\right) = g\left(\left(\mathbf{M}_x^{(t-1)} \odot \mathbf{M}_y^{(t-1)}\right), \left(\mathbf{M}_x^{(t)} \odot \mathbf{M}_y^{(t)}\right), \left(\mathbf{I}_x^{(t-1)} \odot \mathbf{I}_y^{(t-1)}\right), \left(\mathbf{I}_x^{(t)} \odot \mathbf{I}_y^{(t)}\right)\right)$$

  where $\odot$ denotes any operation.
  Select the remaining proposal set $\mathbb{S}_x = f_{\mathbf{NMS}'}(\mathbf{M}_{S_i}^{(t)}, g'(\cdot), \alpha, \beta)$ where $\mathbb{S}_x \subseteq \{1, \ldots, n_C\}$ and $\mathbb{M}_i = \{\mathbf{M}_{S_{i_{:,:,x}}}^{(t)} | x \in \mathbb{S}_x\}$.
end for
Initialize mask set $\mathbb{M}' = \emptyset$.
for $i = 1, \ldots, n_S$ **do**
  for $\mathbf{M}_x \in \mathbb{M}_i$ **do**
    if $\left(\sum_{\mathbf{M}_{x'} \in \mathbb{M}'} g'(\min(\mathbf{M}_x, \mathbf{M}_{x'})) > g'(\sum_{\mathbf{M}_{x'} \in \mathbb{M}'} \min(\mathbf{M}_x, \mathbf{M}_{x'}))\right)$ **then**
      for $\mathbf{M}_z \in \mathbb{M}'$ **do**
        Calculate the overlapping area $k_{\text{overlap}} = \sum_h^{H_{R_i}} \sum_w^{W_{R_i}} \min(\mathbf{M}_x, \mathbf{M}_z)_{h,w}$.
        Calculate the whole area $k_{\text{whole}} = \sum_h^{H_{R_i}} \sum_w^{W_{R_i}} \mathbf{M}_{z_{h,w}}$.
        if $k_{\text{overlap}} > \alpha k_{\text{whole}}$ **then**
          Remove $\mathbf{M}_z$ from the mask set $\mathbb{M}' \leftarrow \mathbb{M}' \setminus \mathbf{M}_z$.
        end if
      end for
      Insert $\mathbf{M}_x$ to the mask set $\mathbb{M}' \leftarrow \mathbb{M}' \cup \{\mathbf{M}_x\}$.
    end if
  end for
end for
return $\mathbb{M}'$.

selection score function. We introduce an instance-consistent score function that measures the motion consistentcy quantity of instance appearance and shape. The score function is similar to the instance-consistent loss mentioned in the section 3.2, which denotes $g(\mathbf{I}, \mathbf{I}', \mathbf{M}, \mathbf{M}')$ where $\mathbf{I}, \mathbf{I}', \mathbf{M}, \mathbf{M}'$ are the images and masks in two adjacent frames, which is defined as,

$$\begin{aligned} g(\mathbf{I}, \mathbf{I}', \mathbf{M}, \mathbf{M}') \quad = \quad & \|\text{STN}(\mathbf{I}, \text{STN}^{-1}(\mathbf{M}, \mathbf{M}')) - \mathbf{I}'\|_2 \cdot \\ & \max(\text{STN}(\mathbf{M}, \text{STN}^{-1}(\mathbf{M}, \mathbf{M}')), \mathbf{M}') + \\ & \|\text{STN}(\mathbf{M}, \text{STN}^{-1}(\mathbf{M}, \mathbf{M}')) - \mathbf{M}'\|_2. \end{aligned}$$

And the lower score indicates the better dynamic instance mask.

APPENDIX C TABLES AND FIGURES

Model	Monster Kong				Flappy Bird	
	1-5†	1-5	2-5	3-5	1-5†	1-5‡
MAOP	1.96	0.34	0.38	0.42	0.30	0.34

Table C3: Average motion prediction error in two experiment environments. † and ‡ correspond to the same sample restriction experiments in Table 1 and 2

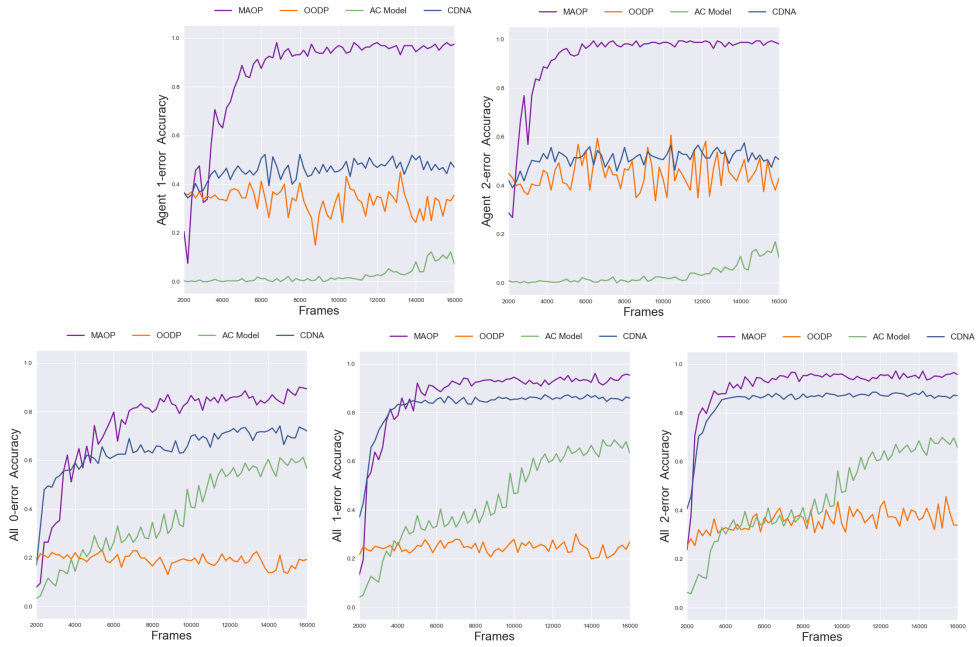

Figure C7: More learning curves.

APPENDIX D IMPLEMENTATION DETAILS

The architecture of the CNNs in Object Detector and Instance Detector is the same as that in OODP (Zhu & Zhang, 2018). Denote $Conv(F, K, S)$ as the convolutional layer with the number of filters F, kernel size K and stride S. Let $R(), S()$ and $BN()$ denote the ReLU layer, sigmoid layer and batch normalization layer Ioffe & Szegedy (2015).

The 5 convolutional layers in Object Detector can be indicated as $R(BN(Conv(16, 5, 2)))$, $R(BN(Conv(32, 3, 2)))$, $R(BN(Conv(64, 3, 1)))$, $R(BN(Conv(32, 1, 1)))$, and $BN(Conv(1, 3, 1))$, respectively.

The 5 convolutional layers in Instance Detector can be indicated as $R(BN(Conv(32, 5, 2)))$, $R(BN(Conv(32, 3, 2)))$, $R(BN(Conv(32, 3, 1)))$, $R(BN(Conv(32, 1, 1)))$, and $BN(Conv(1, 3, 1))$, respectively.

The architecture of Foreground Detector is similar to binary-class Object Detector and the 5 convolutional layers in Foreground Detector can be indicated as $R(BN(Conv(32, 5, 2)))$, $R(BN(Conv(32, 3, 2)))$, $R(BN(Conv(32, 3, 1)))$, $R(BN(Conv(32, 1, 1)))$, and $S(BN(Conv(1, 3, 1)))$, respectively.

The CNNs in Dynamics Net are connected in the order: $R(BN(Conv(16, 3, 2)))$, $R(BN(Conv(32, 3, 2)))$, $R(BN(Conv(32, 3, 2)))$, and $R(BN(Conv(32, 3, 2)))$. The last convolutional layer is reshaped and fully connected by the 64-dimensional hidden layer and the 2-dimensional output layer successively.

The CNNs in Inertia Net has the same architecture and hyperparameters as that in Dynamics Net.

The hyperparameters and parameters for training MAOP in *Monster Kong* and *Flappy Bird* are listed as follows:

- The images in *Monster Kong* and *Flappy Bird* are down-sampled to size $160 \times 160 \times 3$ and $160 \times 80 \times 3$, respectively.
- In MAOP, the weights for $\mathcal{L}_{\text{highway}}$, $\mathcal{L}_{\text{prediction}}$, $\mathcal{L}_{\text{proposal}}$, $\mathcal{L}_{\text{instance}}$, and $\mathcal{L}_{\text{foreground}}$ are 1, 100, 1, 1, and 10, respectively. In addition, all the l_2 losses are divided by HW to keep invariance to the image size.
- Batch size is 16 and the maximum number of training steps is set to be 1×10^5.
- The size of the horizon window w is 33 in *Monster Kong*.
- The size of the horizon window w is 41 in *Flappy Bird*.
- The optimizer is Adam (Kingma & Ba, 2014) with learning rate 1×10^{-3}.
- The maximum number of static and dynamic masks is 4 and 12, respectively in *Monster Kong*.
- The maximum number of static and dynamic masks is 4 and 12, respectively in *Flappy Bird*.
- To augment the number of interactions of instances, we random sample two region proposals and merge them into a new region proposal which has double size.