Structured World Belief for Reinforcement Learning in POMDP

Gautam Singh [1] **Skand Peri** [1] **Junghyun Kim** [1] **Hyunseok Kim** [2] **Sungjin Ahn** [1 3]

Abstract

Object-centric world models provide structured representation of the scene and can be an important backbone in reinforcement learning and planning. However, existing approaches suffer in partially-observable environments due to the lack of belief states. In this paper, we propose Structured World Belief, a model for learning and inference of object-centric belief states. Inferred by Sequential Monte Carlo (SMC), our belief states provide multiple object-centric scene hypotheses. To synergize the benefits of SMC particles with object representations, we also propose a new object-centric dynamics model that considers the inductive bias of object permanence. This enables tracking of object states even when they are invisible for a long time. To further facilitate object tracking in this regime, we allow our model to attend flexibly to any spatial location in the image which was restricted in previous models. In experiments, we show that object-centric belief provides a more accurate and robust performance for filtering and generation. Furthermore, we show the efficacy of structured world belief in improving the performance of reinforcement learning, planning and supervised reasoning.

1. Introduction

There have been remarkable recent advances in object-centric representation learning (Greff et al., 2020). Unlike the conventional approaches that provide a single vector to encode the whole scene, the goal of this approach is to learn a set of modular representations, one per object in the scene, via self-supervision. Promising results have been shown for various tasks such as video modeling (Lin et al., 2020a), planning (Veerapaneni et al., 2019), and systematic

generalization to novel scenes (Chen et al., 2020).

Of particular interest is whether this representation can help Reinforcement Learning (RL) and planning. Although OP3 (Veerapaneni et al., 2019) and STOVE (Kossen et al., 2019) have investigated the potential, the lack of belief states and object permanence in the underlying object-centric dynamics models makes it hard for these models to deal with Partially Observable Markov Decision Processes (POMDP). While DVRL (Igl et al., 2018) proposes a neural approach for estimating the belief states using Sequential Monte Carlo (SMC) for RL in POMDP, we hypothesize that the lack of object-centric representation and interaction modeling can make the model suffer in more realistic scenes with multiple stochastic objects interacting under complex partial observability. It is, however, elusive whether these two associated structures, objectness and partial observability, can be integrated in synergy, and if yes then how.

In this paper, we present Structured World Belief (SWB), a novel object-centric world model that provides object-centric belief for partially observable environments, and methods for RL and planning utilizing the SWB representation. To this end, we first implement the principle of object permanence in our model by disentangling the *presence* of an object file in the belief representation from the *visibility* of the object in the observation. Second, we resolve a hard problem that arises when object permanence is considered under partial observability, i.e., consistently re-attaching to an object which can re-appear in a distant and non-deterministic position after a long occlusion. To solve this, we propose a learning mechanism called *file-slot matching*. Lastly, by integrating the object-centric belief into the auto-encoding SMC objective, our model can gracefully resolve tracking errors while maintaining diverse and likely explanations (or hypotheses) simultaneously. This makes our model work robustly with error-tolerance under the challenging setting of multiple and stochastic objects interacting under complex partial observability.

Our experiment results show that integrating object-centric representation with belief representation can be synergetic, and our proposed model realizes this harmony effectively. We show in various tasks that our model outperforms other models that have either only object-centric representation or only belief representation. We also show that the perfor-

mance of our model increases as we make the belief estimation more accurate by maintaining more particles. We show these results in video modeling (tracking and future generation), supervised reasoning, RL, and planning.

The contributions of the paper can be summarized as: (1) A new type of representation, SWB, combining object-centric representation with belief states. (2) A new model to learn and infer such representations and world models based on the integration of object permanence, file-slot matching, and sequential Monte Carlo. (3) A framework connecting the SWB representation to RL and planning. (4) Empirical evidence demonstrating the benefits and effectiveness of object-centric belief via SWB.

2. Background

Object-Centric Representation Learning aims to represent the input image as a set of latent representations, each of which corresponds to an object. The object representation can be a structure containing the encoding of the appearance and the position (e.g., by a bounding box) (Eslami et al., 2016; Crawford & Pineau, 2019b; Jiang & Ahn, 2020) as in SPACE (Lin et al., 2020b) or a single vector representing the segmentation image of an object (Greff et al., 2017; 2019; Burgess et al., 2019; Locatello et al., 2020). These methods are also extended to temporal settings, which we call object-centric world models (OCWM) (Kosiorek et al., 2018; Crawford & Pineau, 2019a; Jiang et al., 2019; Lin et al., 2020a; Veerapaneni et al., 2019). In the inference of OCWM, the model needs to learn to track the objects without supervision and the main challenge here is to deal with occlusions and interactions. These existing OCWMs do not provide object permanence and belief states. We discuss this problem in more detail in the next section.

Sequential Monte Carlo methods (Doucet et al., 2001) provide an efficient way to sequentially update posterior distributions of a state-space model as the model collects observations. This problem is called *filtering*, or *belief update* in POMDP. In SMC, a posterior distribution, or a belief, is approximated by a set of samples called *particles* and their associated *weights*. Upon the arrival of a new observation, updating the posterior distribution amounts to updating the particles and their weights. This update is done in three steps: (1) sampling a new particle via a proposal distribution, (2) updating the weights of the proposed particle, and (3) resampling the particles based on the updated weights.

Auto-Encoding Sequential Monte Carlo (AESMC) (Le et al., 2018; Naesseth et al., 2018) is a method that can learn the model and the proposal distribution jointly by simulating SMC chains. The model and the proposal distribution are parameterized as neural networks and thus the proposal distribution can be amortized. The key idea is to
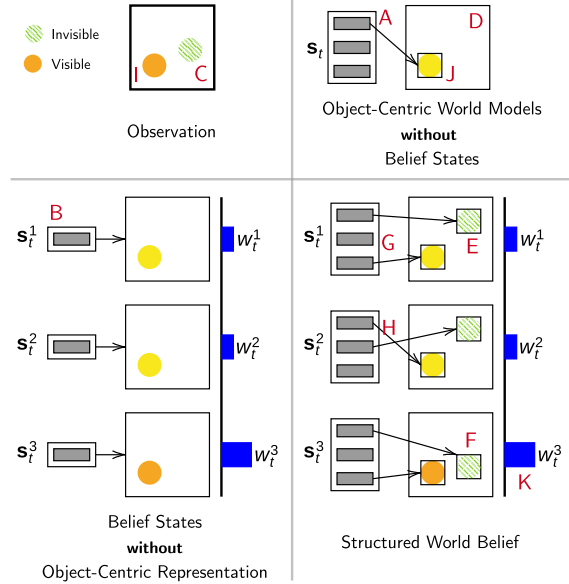

Figure 1. **Comparison of Representations.** The grey boxes denote object files and the rectangles (A, B and G) enclosing them denote a scene state or a particle. The figure demonstrates that: *i)* object-centric models without belief state maintain single hypothesis (A) for the scene. *ii)* AESMC maintains multiple hypotheses but the representation lacks object-wise structure (B). *iii)* Previous object-centric models only maintain files for visible objects (I and J) but delete them for invisible objects (C and D). *iv)* For invisible objects, SWB has object permanence and maintains multiple hypotheses over its files. The object files E and F for the occluded green object provide alternative hypotheses. *v)* In G and H, SWB provides belief over different files modeling the same objects. *vi)* If the inference distributions draws a poor sample through chance (for instance, mistaking orange for yellow in J), previous object-centric models cannot discard this sample. SWB can discard bad samples by assigning high weight to the good samples K.

use backpropagation via reparameterization to maximize the evidence lower bound of the marginal likelihood, which can easily be computed from the weights of the particles.

3. Structured World Belief

We propose to represent the state $\mathbf{b}_t$ of a POMDP at time t as Structured World Belief (SWB) (hereafter, we simply call it belief). As in SMC, SWB represents a belief $\mathbf{b}_t := (\mathbf{s}_t, \mathbf{w}_t)$ as a set of K particles $\mathbf{s}_t = (\mathbf{s}_t^1, \ldots, \mathbf{s}_t^K)$ and its corresponding weights $\mathbf{w}_t = (\mathbf{w}_t^1, \ldots, \mathbf{w}_t^K)$. However, unlike previous belief representations, we further structure a particle as a set of *object files*: $\mathbf{s}_t^k = (\mathbf{s}_{t,1}^k, \ldots, \mathbf{s}_{t,N}^k)$. An object file is a D-dimensional vector and N is the capacity of the number of object files.

Object Files. An object file (or simply a file) (Kahneman et al., 1992; Goyal et al., 2020) is a structure containing the following four items: (1) the ID $i_{t,n}^k$ of the object that the file is bound to, (2) the visibility $\mathbf{z}_{t,n}^{k,\text{vis}}$ denoting whether the

object is *visible* or not, (3) the current state (e.g., appearance and position) of the object $\mathbf{z}_{t,n}^{k,\text{obj}}$, and (4) an RNN hidden state $\mathbf{h}_{t,n}^k$ encoding the history of the object. All object files are initialized with ID as `null`. This represents that the object files are *inactive*, i.e., not bound to any object. When an object file binds to an object, a positive integer is assigned as the ID meaning that the object file is *activated* and the underlying RNN is initialized. The ID is unique across files $n = 1, \dots, N$ within a particle k and thus can be useful in downstream tasks or evaluation.

Object Permanence. In previous models (Jiang et al., 2019; Crawford & Pineau, 2019a; Lin et al., 2020a), whether a file is maintained or deleted is determined by the visibility of the object. When an object is entirely invisible, it does not contribute to the reconstruction error, and thus the file is deleted. That is, these models ignore the inductive bias of object permanence and this makes it difficult to maintain object-centric belief. To implement object permanence, our model decomposes the presence of an object in the belief and its visibility in the observation. When an object goes out of view, we only set its visibility to zero, but other information (ID, object state, and RNN state) is maintained. This allows for the file to re-bind to the object and set the visibility to 1 when the object becomes visible again.

Object Files as Belief. When an object becomes invisible, tracking its state becomes highly uncertain because there can be multiple possible states explaining the situation. However, in our model, the active but invisible files corresponding to that object are populated in a set of particles and maintain different states, providing a belief representation of multiple possible explanations. For example, an agent can avoid occluded enemies by predicting all possible trajectories under the occlusion.

4. Learning and Inference

4.1. Generative Model

The joint distribution of a sequence of files $\mathbf{s}_{1:T,1:N}$ and observations $\mathbf{x}_{1:T}$ can be described as:

$$p_\theta(\mathbf{x}_{1:T}, \mathbf{s}_{1:T}) = \prod_{t=1}^{T} p_\theta(\mathbf{x}_t|\mathbf{s}_t) \prod_{n=1}^{N} p_\theta(\mathbf{s}_{t,n}|\mathbf{s}_{t-1})$$

where the file transition $p_\theta(\mathbf{s}_{t,n}|\mathbf{s}_{t-1})$ is further decomposed to $p_\theta(\mathbf{z}_{t,n}|\mathbf{z}_{t-1}, \mathbf{h}_{t-1})p(i_{t,n})$. Here, $p_\theta(\mathbf{z}_{t,n}|\mathbf{z}_{t-1}, \mathbf{h}_{t-1})$ is the dynamics prior for $\mathbf{z}_{t,n}^{\text{obj}}$ and $\mathbf{z}_{t,n}^{\text{vis}}$ and $p(i_{t,n})$ is the prior on ID. The RNN is updated by $\mathbf{h}_{t,n} = \text{RNN}_\theta(\mathbf{z}_{t,n}, \mathbf{h}_{t-1,n})$.

4.2. Inference

In this section, we describe the inference process of updating $\mathbf{b}_{t-1}$ to $\mathbf{b}_t$ using input $\mathbf{x}_t$. At the start of an episode, SWB contains $K \times N$ inactive files and these are updated at every

---

Algorithm 1 File-Slot Matching and Glimpse Proposal

input Previous files $\mathbf{s}_{t-1}$, Slots $\mathbf{u}_{t,0:M}$
output Proposal region $\boldsymbol{o}_{t,n}^{k,\text{proposal}}$

 # 1. Interact each slot with all files
$$\mathbf{e}_m^{k,\text{interact}} = \sum_{n=1}^{N} f_\phi^{\text{interact}}(\mathbf{s}_{t-1,n}^k, \mathbf{u}_{t,m}) \qquad \forall k, m$$

 # 2. Compute probabilities to match file to slot
$$q_{n,m}^{k,\text{match}} = f_\phi^{\text{match}}(\mathbf{s}_{t-1,n}^k, \mathbf{u}_{t,m}, \mathbf{e}_m^{k,\text{interact}}) \qquad \forall k, n, m$$

 # 3. Obtain matched slot
$$m_{t,n}^k \sim \text{Categorical}(\cdot|q_{n,0:M}^{k,\text{match}}) \qquad \forall k, n$$

 # 4. Obtain Glimpse Proposal
$$\boldsymbol{o}_{t,n}^{k,\text{proposal}} = f_\phi^{\text{proposal}}(\mathbf{s}_{t-1,n}^k, \mathbf{u}_{t,m_{t,n}^k}) \qquad \forall k, n$$

---

time-step by the input image. This involves three steps: *i) file update* from $\mathbf{s}_{t-1,n}^k$ to $\mathbf{s}_{t,n}^k$ and *ii) weight update* from $\mathbf{w}_{t-1}$ to $\mathbf{w}_t$ and *iii) particle resampling*.

4.2.1. UPDATING OBJECT FILES

Image Encoder and Object Slots. To update the belief, our model takes an input image at each time step. For the image encoder, the general framework of the proposed model requires a network that returns a set of object-centric representations, called *object slots*, $\mathbf{u}_{t,1:M} = \{\mathbf{u}_{t,1}, \dots, \mathbf{u}_{t,M}\}$ for image $\mathbf{x}_t$. Here, M is the capacity of the number of object slots and we use m to denote the index of slots. To obtain the slots in our experiments, we use an encoder similar to SPACE. However, unlike SPACE that has a separate encoder for the glimpse patch, we directly use M CNN features of the full image which have the highest presence values. These presence values are not necessary if an encoder can directly provide M slots as in (Greff et al., 2017; Burgess et al., 2019; Greff et al., 2019). This image encoder is not pretrained but trained jointly end-to-end along with other components.

We introduce a special slot $\mathbf{u}_{t,0}$, called *null slot* (represented as a zero vector), and append it to the object slots, i.e., $\mathbf{u}_t = \mathbf{u}_{t,0} \cup \mathbf{u}_{t,1:M}$. We shall introduce a process where each object file finds a matching object slot to attend to. The null slot will act as a blank slot to which an object file can match if it is not necessary to attend to any of the object slots $\mathbf{u}_{t,1:M}$, for example, if the object is active in the file but invisible in the input image.

Learning to (Re)-Attach by File-Slot Matching. A natural way to update the files is to obtain new information from the relevant area in the input image. In previous works (Crawford & Pineau, 2019a; Jiang et al., 2019; Lin et al., 2020a), this was done by limiting the attention to the neighborhood of the position of the previous object file because an object may not move too far in the next frame. However, this approach easily fails in the more realistic
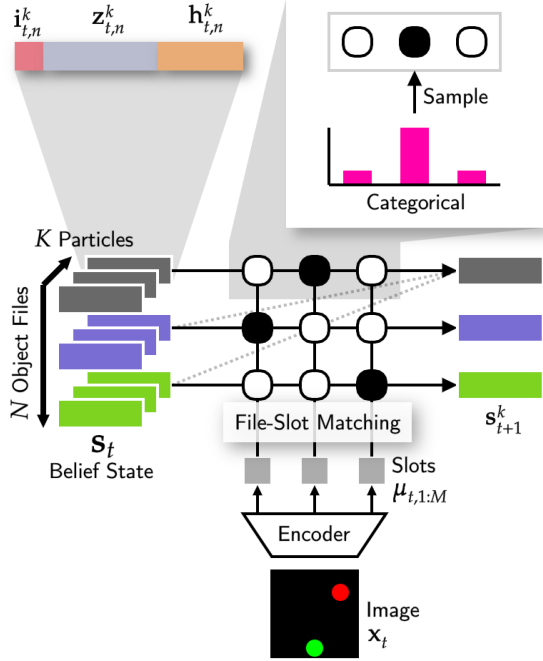
Figure 2. **A Simplified View of Inference in SWB.** For each object-file, we sample a matching slot and use it to update the object file. In this update, the object files also interact with each other through a graph neural network (shown in dotted lines).

setting of partial observability tackled in this work. For example, when the duration of invisibility of an object is long, the object can reappear in a non-deterministic surprise location far from the previously observed location. Therefore, finding a matching object that can re-appear anywhere in the input image is a key problem.

To address this, we propose a mechanism for learning-to-match the object files and the object slots. Crucially, our model learns to infer the matching implicitly via the AESMC learning objective without any inductive bias. The aim of the matching mechanism is to compute the probability $q_{n,m}^k$ that file n matches with slot m. To compute this, if each file looks at the candidate slot alone, multiple files can match with the same slot causing false detections or misses during tracking. To prevent this, a file should not only look at the candidate slot but also at other files that are competing to match with this slot. For this, we compute an interaction summary $\mathbf{e}_m^{k,\text{interact}}$ of interaction between a slot $\mathbf{u}_{t,m}$ and all the files $\mathbf{s}_{t-1,1:N}^k$ via a graph neural network. After doing this for all m in parallel, we compute the probability of matching $q_{n,m}^k$ using an MLP that takes as input the interaction summary, the file n, and the slot m. For each file, we then apply softmax over the slots. This constructs a categorical distribution with $M+1$ output heads. From this distribution, we sample the index $m_{t,n}^k$ of the matching slot for the file n. The randomness in this sampling, when

combined with SMC inference, helps maintain multiple hypotheses. The match index $m_{t,n}^k = 0$ denotes a match to the null slot.

State Inference. Given the matched slot, it is possible to directly use the encoding of the slot $\mathbf{u}_{t,m_{t,n}^k}$ provided by the encoder to update file n. However, we found it more robust to use the slot $\mathbf{u}_{t,m_{t,n}^k}$ and the file $\mathbf{s}_{t-1,n}^k$ to first infer a glimpse area via an MLP network. The image crop from the glimpse area is then encoded as $\mathbf{e}_{t,n}^k$ and used to infer the object visibility, the object state, and the RNN state of the file n:

$$\mathbf{z}_{t,n}^{k,\text{vis}} \sim \text{Bernoulli}_\phi(\mathbf{z}_{t,n}^{k,\text{vis}}|\mathbf{s}_{t-1,n}^k, \mathbf{e}_{t,n}^k),$$
$$\mathbf{z}_{t,n}^{k,\text{obj}} \sim q_{\theta,\phi}(\mathbf{z}_{t,n}^{k,\text{obj}}|\mathbf{s}_{t-1,n}^k, \mathbf{e}_{t,n}^k, \mathbf{z}_{t,n}^{k,\text{vis}}),$$
$$\mathbf{h}_{t,n}^k = \text{RNN}_\theta(\mathbf{z}_{t,n}^k, \mathbf{h}_{t-1,n}^k).$$

Imagination. A crucial advantage of our object-centric representation is that we can control the belief at the individual object-level rather than the full scene level. This is particularly useful when we update the state of an invisible object file while other files are visible. For the invisible object, the input image does not provide direct information for updating the state of the invisible file and thus it is more effective to use the prior distribution for the invisible files instead of letting the posterior re-learn the dynamics of the prior. We call this update based on the prior as the imagination phase. Note that the objects governed by the prior can still interact with other objects, both visible and invisible. To allow this transition between imagination and attentive tracking at the individual object level, we factorize $q_{\theta,\phi}(\mathbf{z}_{t,n}^{k,\text{obj}}|\mathbf{s}_{t-1}^k, \mathbf{x}_t, \mathbf{z}_{t,n}^{k,\text{vis}})$ according to the visibility:

$$q_\phi(\mathbf{z}_{t,n}^{k,\text{obj}}|\mathbf{s}_{t-1}^k, \mathbf{x}_t)^{\mathbf{z}_{t,n}^{k,\text{vis}}} p_\theta(\mathbf{z}_{t,n}^{k,\text{obj}}|\mathbf{s}_{t-1}^k)^{1-\mathbf{z}_{t,n}^{k,\text{vis}}}.$$

Discovery and Deletion. The discovery of an object is the state transition from inactive to active which is learned through the file-slot matching. An active file carries over its assigned ID until it becomes inactive. Even if a file is invisible for a long time, we do not explicitly delete an object as long as the file state stays active. Rather, these files are updated via imagination. Alternatively, we can introduce the invisibility counter, and when the file capacity is near full, we can delete the file which has been invisible for the longest time.

4.2.2. WEIGHT UPDATE AND RESAMPLING

The object-centric decomposition in our model allows the following decomposition of the weight computation:

$$w_t^k \propto w_{t-1}^k \cdot p_\theta(\mathbf{x}_t|\mathbf{z}_t^k) \prod_{n=1}^N w_{t,n}^{k,\text{file}} \cdot w_{t,n}^{k,\text{ID}} \cdot w_{t,n}^{k,\text{match}}$$

where

$$w_{t,n}^{k,\text{file}} = \frac{p_\theta(\mathbf{z}_{t,n}^{k,\text{obj}}|\mathbf{s}_{t-1}^k)}{q_{\theta,\phi}(\mathbf{z}_{t,n}^{k,\text{obj}}|\mathbf{s}_{t-1}^k, \mathbf{x}_t, \mathbf{z}_{t,n}^{k,\text{vis}})} \frac{p_\theta(\mathbf{z}_{t,n}^{k,\text{vis}}|\mathbf{s}_{t-1}^k)}{q_\phi(\mathbf{z}_{t,n}^{\text{vis},k}|\mathbf{s}_{t-1}^k, \mathbf{x}_t)},$$

$$w_{t,n}^{k,\text{ID}} = \frac{p(i_{t,n}^k)}{q_\phi(i_{t,n}^k|\mathbf{s}_{t-1}^k, \mathbf{x}_t)},$$

$$w_{t,n}^{k,\text{match}} = \frac{p(m_{t,n}^k)}{q_\phi(m_{t,n}^k|\mathbf{s}_{t-1}^k, \mathbf{x}_t)}.$$

To allow object files to match to any slot without an inductive bias such as spatial locality in previous works, we apply a uniform prior $p(m_{t,n}^k)$ on the matching index latent. Using these updated weights, we then perform resampling.

4.3. Learning Objective

Our learning objective is the same Evidence Lower Bound (ELBO) used in AESMC which can be computed by: $\mathcal{L}_{\theta,\phi} = \frac{1}{T}\sum_{t=1}^T \log\sum_{k=1}^K w_t^k$. To optimize the objective via gradient descent, we backpropagate through the random variables via the reparameterization trick (Kingma & Welling, 2013) and for discrete random variables, we use REINFORCE (Williams, 1992). We found the straight-through estimator to be less stable than REINFORCE.

5. Structured World Belief Agent

To use our structured world belief for the downstream tasks such as RL, planning, and supervised learning, we need to encode the particles and weights into a belief vector. For this, we first encode each object file in the belief using a shared MLP and obtain $N \times K$ such file encodings. Then, we apply mean-pooling over objects to obtain K particle encodings. To each particle encoding, we concatenate the particle weight. These are fed to another shared MLP to obtain K particle encodings, to which we again apply a mean-pooling and apply a final MLP to obtain the final encoding of the structured world belief.

6. Related Works

Neural Belief Representations: There are several works that use belief states for RL and planning in POMDP (Kaelbling et al., 1998). DRQN (Hausknecht & Stone, 2015) and ADRQN (Zhu et al., 2017) use RNN states as input to the Q-network. RNN history as the sufficient statistics of the belief has also been used in (Moreno et al., 2018) for supervised reasoning tasks, in (Guo et al., 2018; Gregor et al., 2019; Guo et al., 2020) for sample-efficient RL, and in (Gangwani et al., 2020) for imitation learning. PF-RNNs (Ma et al., 2019) incorporate belief as hidden state distribution in the RNN. (Tschiatschek et al., 2018; Wang et al., 2019; Piché et al., 2018) use belief states for policy optimization in planning. One of the closest works to SWB is that of

DVRL (Igl et al., 2018). DVRL combines AESMC (Le et al., 2018)-based world model with A2C (Mnih et al., 2016), an actor-critic policy gradient method. AlignNet (Creswell et al., 2020) uses attention matrix in order to align objects in a scene and track them over time. It also incorporates an object slot memory module in order to capture object permanence demonstrated for short-term occlusions. Unlike SWB, it does not provide structured belief for the invisible objects and cannot perform long term future generation. R-NEM (van Steenkiste et al., 2018) uses a spatial mixture model to obtain structured representation of a scene. While R-NEM implicitly handles object permanence, it does not provide an explicit belief over object states.

Object-centric approaches in RL and planning: Object-centric representations have been considered for use in Reinforcement Learning (Scholz et al., 2014; Diuk et al., 2008; Mohan & Laird, 2011; Goel et al., 2018; Keramati et al., 2018; Devin et al., 2018). Recent works have focused on self-supervised learning objectives to infer object-centric representations and benefit Reinforcement Learning. OP3 (Veerapaneni et al., 2019) uses a temporal spatial-mixture model to obtain disentangled representations while O2P2 (Janner et al., 2019) uses the ground truth object segmentation mask to learn a predictive model for control. COBRA (Watters et al., 2019) uses MONet (Burgess et al., 2019) to obtain object-centric representations. Transporter (Kulkarni et al., 2019) uses KeyNet (Jakab et al., 2018) to extract keypoints for sample-efficient RL. Davidson & Lake (2020) also show that object representations are useful for generalization and sample-efficiency in model-free RL. Liang et al. (2015) and Agnew & Domingos (2020) use handcrafted object features for RL.

7. Experiments

The goal of the experiments is to *i)* evaluate the belief state provided by SWB for modeling the true object states *ii)* evaluate the performance of SWB when it is used as a world model for reinforcement learning and planning and *iii)* evaluate the performance of SWB when the inferred belief states are used as inputs for supervised learning tasks.

Environments. In all the environments, the objects can disappear for 25 to 40 time-steps creating the long-term invisibility that we tackle in this work. We experiment using the following environments: *i) 2D Branching Sprites.* This is a 2D environment with moving sprites. The object paths split recursively, with objects randomly taking one branch at every split. Each sprite is associated with a pair of colors and the color switches periodically. We have two versions of this data set: one in which objects can spawn and another in which objects remain fixed. *ii) 2D Branching Sprites Game.* We turn the 2D Branching Sprites environment into a game in which the aim of the agent is to select a lane on

which an invisible object is present. The reward and penalty depend on object color. This makes it important for agent to have accurate belief state over both position and appearance. *iii) 2D Maze Game.* In this game, the objective of the agent is to reach a goal and avoid the enemies. *iv) 3D Food Chase Game.* In this game, the objective of the agent is to chase and eat the food and avoid the enemies. For more details of each environment see Appendix C.2.

In these environments, by having color change and randomness in object dynamics, we test SWB in modeling multiple attributes of invisible objects. The dynamics of color change in 2D Branching Sprites also test the ability of the model in having memory of the past colors. Lastly, the objects can have identical appearances which makes it important for the model to utilize both the position and appearance based cues to solve object matching problem.

7.1. World Belief Modeling

7.1.1. REPRESENTATION LOG-LIKELIHOOD

We first evaluate the accuracy of the current belief, or in other words, the filtering distribution conditioned on observations seen so far. We expect SWB to provide accurate hypotheses about possible object positions and appearance even when objects are not visible. We note that our task is significantly more challenging than the previous object-centric world models because unlike these works, the objects are fully invisible for much longer time steps.

Model and Baselines. We evaluate our model SWB with $K = 10$ particles which provides both object-centric representation and belief states. We compare against GSWM (Lin et al., 2020a) which provides object-centric world representation but does not provides belief states. We also compare against AESMC (Le et al., 2018) which provides belief states but not object-centric world representation. Because AESMC does not provide object-wise positions or segments, we train a supervised position estimator to extract position from the AESMC belief representation. Note that the supervised learning signal is only used to train this estimator and is not propagated to AESMC. We also compare against the non-belief version of SWB i.e. SWB with $K = 1$ which can be seen as GSWM augmented with object-permanence. This enables it to keep files for invisible objects but the belief state is composed of only one structured state. To also study the effect of gradually increasing the number of particles we compare against SWB with $K = 3$.

Metrics. We evaluate our belief using ground truth values of object-wise positions and object segments. We use the predicted particles to perform kernel density estimation of the distribution over these attributes and evaluate the log-likelihood of the true states. We compute MOT-A (Milan et al., 2016) metric components (such as false positives and

switches) by counting them for each particle k and taking weighted average using the particle weights w_t^k. For more details, see Appendix C.1.

Tracking. We infer and evaluate the current belief on episodes of length 50. For this experiment, we consider the 2D Branching Sprites environment with no object spawning. The objects alternate between being visible and invisible, leading to a periodic variation in number of objects visible at different points of time in tracking. In Fig. 3, we see that by increasing the number of particles from $K = 1$ to 3 and 10, we are able to improve our modeling of density over the object states. Our improvements are more pronounced when the objects are invisible. This is because more particles lead to a diverse belief state that is less likely to miss out possible states of the environment, thereby showing improvement in density estimation. Furthermore, we see that previous object-centric models (evaluated as GSWM) suffer in performance when the objects become invisible due in large part to deletion of their object files. Lastly, we see that in comparison to models providing unstructured belief (evaluated as AESMC with $K = 10$), our belief modeling is significantly more accurate. This is enabled by our modeling of structure which allows us to accurately track several possible hypothesis for states of objects.

Generation. In this task, we study how well our model predicts the density over future states of the scene conditioned on the provided observations $\mathbf{x}_{1:t}$. That is, we seek to evaluate the density $p(\mathbf{s}_{t+\Delta}|\mathbf{x}_{1:t})$ where $t + \Delta$ is a future time-step. Due to partial observability, our generation task is more challenging than what previous object-centric world models perform. In previous models, this generation task is equated to predicting future samples from the last inferred scene state $p(\mathbf{s}_{t+\Delta}|\mathbf{s}_t)$. However, when occlusion and object invisibility are present, a single scene state $\mathbf{s}_t$ cannot capture diverse possible states in which the environment can be at time t when the generation is started.

In Fig. 3, we study the effect of number of particles used in SWB for maintaining belief $K = 1$ and $K = 3$ at the start of generation. During generation, note that we draw the same number of future roll outs to eliminate the effect of number of roll-outs in our kernel density estimation. We observe that for SWB $K = 1$ and $K = 3$, the generation starts with a poor density estimation which gets worse as the roll-out proceeds. Fig. 3 compares our future generation against baselines GSWM and AESMC. Since GSWM deletes files when objects are invisible, we made sure that objects were visible in the conditioning period. Because of this, there is no requirement of good belief at the start of the generations. Despite this, we note the generation in GSWM to be worse than that of SWB. We also ensured that same number of generation samples were drawn for all the models compared. Generation in AESMC suffers severe deterioration over time
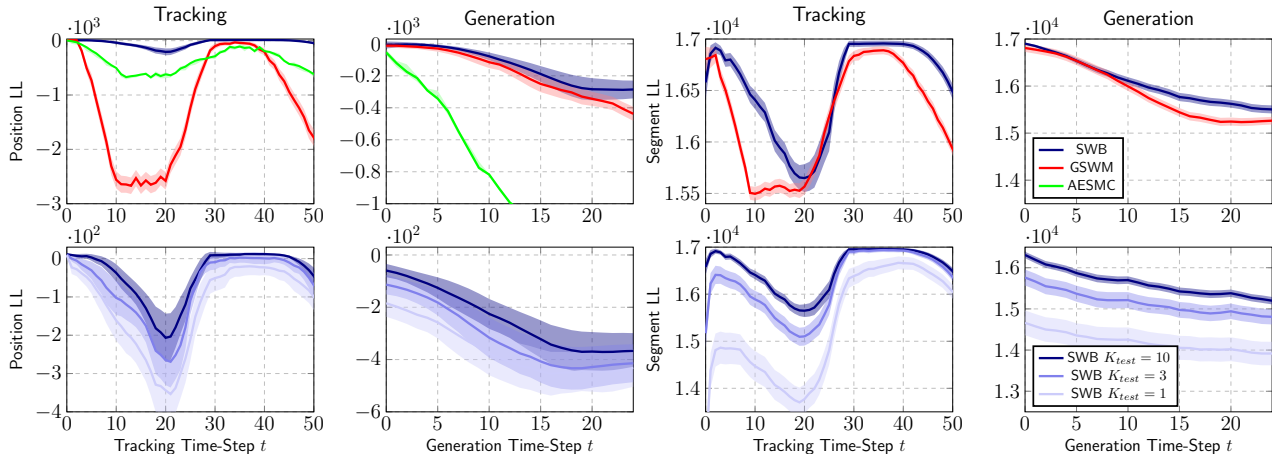
Figure 3. Comparison of SWB in terms of position log-likelihood and segment log-likelihood with the baselines: GSWM, AESMC and SWB $K = 1$ and 3 on 2D Branching Sprites without new object spawning during the episode.
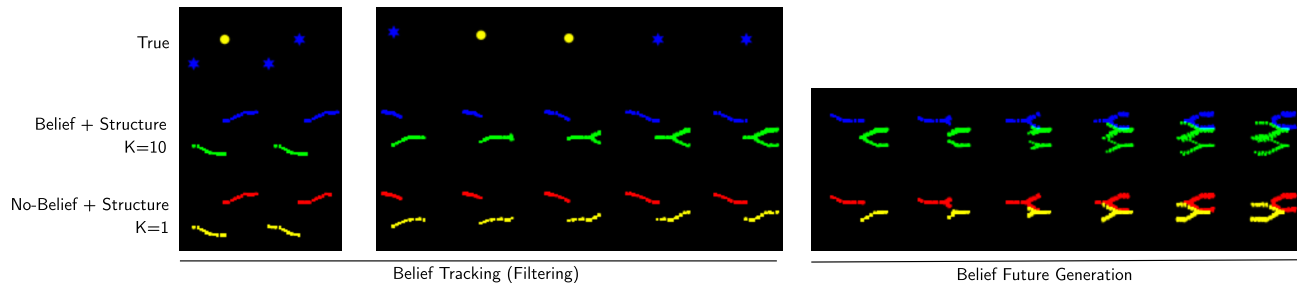

Figure 4. **Comparison between $K = 1$ and $K = 10$ for Belief Tracking and Generation in SWB in 2D Branching Sprites.** The top row shows observations and the middle and the bottom rows visualize 8 most recent positions of each active object file. In the beginning, two objects are shown and model assigns object files to each object. Subsequently, one object becomes invisible and other remains visible. We note that the belief using $K = 10$ properly represents multi-modal hypotheses for the position of the invisible object in comparison to $K = 1$. We also note that imagination is performed only for the invisible object while the state of the visible object is inferred via the matched slot from the input image. In future generation of the belief, we note that starting generation from a belief with $K = 10$ particles allows us to generate more complete set of possible futures (see green object file). On the contrary, when the belief is maintained using $K = 1$, the set of generated future states do not capture all modes of the possible futures. (see yellow object file)

because of the lack of object structure.

MOT. In Fig. 6, we show MOT-A metrics (Milan et al., 2016) and compare SWB for different number of particles and against GSWM. We note that SWB $K = 10$ is the most robust and makes the least errors. This is because by having more particles, even if the inference network makes a few errors in file-slot matching in few particles, those particles receive low weights and get pruned. We see that when we use fewer particles ($K = 3$ and 1), the error rate increases because the model cannot recover from the errors. We hypothesize that this is also the reason why GSWM has higher error rate than ours in various metrics. A notable point is the high *ascend* rate in GSWM when a new file gets created for an object which already has a file. This is because GSWM deletes the original file when the object becomes invisible and needs to create a new file when the object re-appears.

Object Spawning, Object Permanence and Robust Generalization. In Fig. 7, we investigate how our model behaves when objects spawn during the episode, how it generalizes to a larger number of objects in the scene, object files and particles than used in training. For this we trained our model on 2D Branching Sprites with upto 2 objects but we test in a setting in which up to 4 objects can spawn. In Fig. 7, we note that when we test our model with $K = 10$ and $N = 4$ which is the same as training configuration, our model is able to generalize well to 4 objects with small under-counting. The slight under-counting is expected because the model has never seen 4 objects during training. We note that GSWM, due to lack of object permanence, is unable to attach existing files to objects which re-appear after being invisible. Hence, we see an increasing error in the number of object files created in comparison to the true object count. Next, we increase the number of object files in our model from 4 to 8. We see that our model generalizes
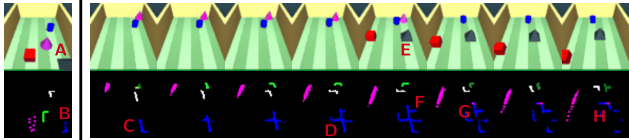
Figure 5. **Structured Belief in 3D Food Chase**. Top row shows observations and bottom row visualizes 8 most recent position states of each active object file. The color of the file denotes ID. The black pyramid (A) is tracked by a blue file (B). The pyramid becomes invisible and file continues to track it (C through D). As it remains invisible, the file produces possible hypotheses of its trajectory (D) which we see as splitting of paths. Hence, SWB can selectively perform imagination for the invisible object and bottom-up inference of visible objects. In E, The pyramid becomes visible and the file re-attaches to it (F). We see that the magenta object file also tries to attach to the pyramid (G). But this hypothesis gets low weight and the blue object file wins the competition (H).
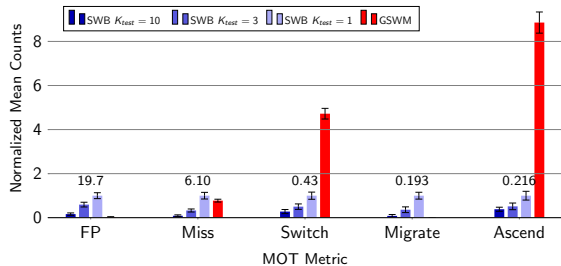

Figure 6. Effect of number of particles used in tracking on the MOT error metrics on Branching Sprites dataset (No-Spawn) with episode length 100. The values are normalized with respect to SWB $K_{\text{test}} = 1$. Values corresponding to 1 are labelled. The counts are first summed over length of the episode and then averaged over episodes. *FP* refers false positive detections. *Miss* denotes missed detections. *Switch* denotes ID swaps. *Migrate* denotes when an existing file gets assigned to a new object. *Ascend* denotes when a new file is created for an object which already has a file.

well and remains robust with slight over-counting. Lastly, we see that we can eliminate this over-counting by increasing the number of particles from $K = 10$ to $K = 30$. This shows that increasing the number of particles K beyond the training value can make inference more robust.

Ablation of SWB. We ablate two design choices of SWB. First, we test whether glimpse encoding is necessary for good inference. Second, we test whether file-slot matching indeed requires a summary representation of all the files. We report the results of ablation of SWB in Appendix A.1.

7.2. Belief-Based Control

We test how structured belief improves performance in reinforcement learning and planning. We experiment on three games: 2D Branching Sprites game, 2D Maze game and the 3D Food Chase game. We evaluate the performance in terms of average total reward.
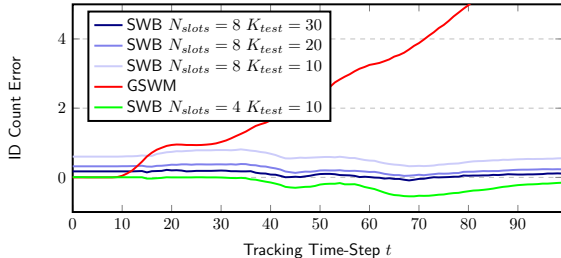

Figure 7. Error in number of spawned IDs against true number of objects over the tracking time-steps on 2D Branching Sprites with 4 object spawns during the episode. Negative and positive values denote under-counting and over-counting, respectively.

Model	2D Maze	3D Food Chase
Random Policy	-5.69 ± 2.13	-3.39 ± 5.58
AESMC $K = 10$	-3.52 ± 1.23	3.71 ± 8.32
AESMC $K = 30$	-2.37 ± 2.58	5.10 ± 7.83
SWB $K = 1$	-2.71 ± 1.53	8.69 ± 9.81
SWB $K = 10$	-2.15 ± 2.02	10.88 ± 9.90
SWB $K = 30$	$\mathbf{-1.32 \pm 2.30}$	$\mathbf{12.13 \pm 8.59}$

Table 1. Performance of Monte Carlo planning on 2D Maze and 3D Food Chase using SWB and AESMC as the world model with different number of particles K used to maintain belief.

7.2.1. REINFORCEMENT LEARNING

For reinforcement learning, we train the policy using A2C that has access to the belief states provided by a world model. We compare the world models: *i)* AESMC, *ii)* non-belief version of SWB i.e. SWB $K = 1$, *iii)* SWB with $K = 3$ and *iv)* SWB with $K = 10$. For SWB, we pre-train the world model with a random exploration policy. By doing this the world model becomes task agnostic. For AESMC, we train the world model jointly with the reward loss as done in DVRL (Igl et al., 2018). We will refer to this agent simply as DVRL. Note that by doing this, we are comparing against a stronger baseline with a task-aware world model.

In Fig. 8, we first analyse the performance of DVRL by training the agent on the fully and partially observable version of the 2D Branching Sprites game. We note that even though the agent learns well in the fully-observable game, it fails in the partially-observable version. This suggests that even when AESMC receives gradients from the reward loss, it fails to extract the information about invisible objects required for performing well in the game. Next, in all three games, we show that even a non-belief version of SWB significantly improves agent performance compared to the unstructured belief state of AESMC. This shows the benefit of structured representation which is enabled by our inductive bias of object permanence. In all three games, as we increase the number of particles from 1 to 10, the performance improves further. We hypothesize that this gain results from better modeling of density over the true states of the objects as shown in Fig. 3 and Fig. 6.
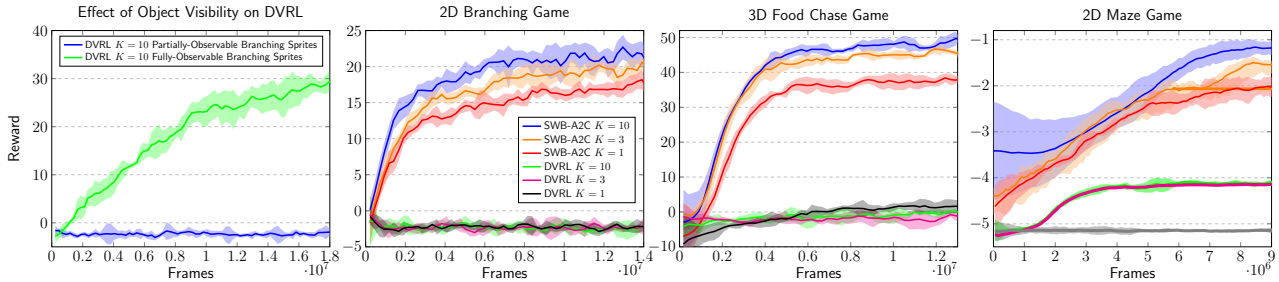
Figure 8. Comparison with DVRL in terms of average reward of A2C when SWB is used as a world model. We report performance for different values of K. The SWB world models for 2D Branching Sprites, 3D Food Chase game and 2D Maze Game were pre-trained using frames collected through 200K, 200K and 500K interactions with the environment, respectively, using a random policy.

Model	$K = 1$	$K = 3$	$K = 5$	$K = 10$
AESMC	29.3 ± 2.1	37.6 ± 3.0	47.2 ± 1.2	54.2 ± 5.2
SWB	$\mathbf{30.1 \pm 2.4}$	$\mathbf{43.1 \pm 2.2}$	$\mathbf{60.8 \pm 3.2}$	$\mathbf{63.4 \pm 2.6}$

Table 2. Comparison of accuracy using SWB and AESMC on the supervised reasoning task for different number of particles K.

7.2.2. PLANNING

To investigate how SWB can improve planning, we use a simple planning algorithm that simulates future roll-outs using random actions up to a fixed depth and predicts an average discounted sum of rewards. We then pick and execute the action that led to the largest average reward. We evaluate the following world models: AESMC and SWB with $K = 1, 10$ and 30. By varying K, we study the gains of performing planning starting from multiple hypotheses. For fair comparison, we keep the number of simulated futures same for all K. In Table 1, we observe that AESMC achieves lower score than ours due to poorer generation. Furthermore, having more particles in SWB helps performance. We hypothesize that these are due to the future generation accuracy evaluated in Fig. 3.

7.3. Belief-Based Supervised Reasoning

To investigate the benefits of providing belief as input for supervised learning, we re-purpose the 2D Maze environment and assign a value to each object color. We also add more than one goal cells. The task is to predict the total sum of all object values present on the goal states. For this, a network takes the belief as input and learns to predicts the sum using a cross-entropy loss. We do not back-propagate the gradients to the world model. In Table 2, we observe the benefits of structured representation to unstructured belief provided by AESMC. We also observe gains when the number of particles are increased.

7.4. Analysis of AESMC

We study AESMC along *i)* number of objects in the scene and *ii)* whether the object dynamics are random or deter-
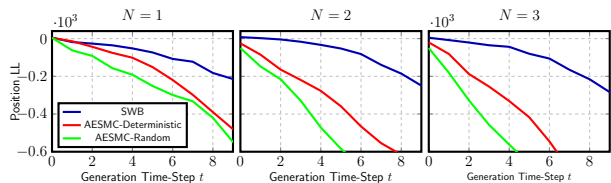

Figure 9. Position log-likelihood on 2D Bouncing Sprites comparing SWB and AESMC on varying number of objects (N) and effect of randomness in environment dynamics on AESMC.

ministic. We provide details about the experimental setup in Appendix C.3. We note that as the number of objects increase, performance of SWB remains robust. However, AESMC deteriorates due to lack of the structured dynamics modeling via objects and modeling the scene as a single encoding vector.

8. Conclusion

We presented a representation learning model combining the benefits of object-centric representation and belief states. Our experiment results on video modeling, RL, planning, and supervised reasoning show that this combination is synergetic, as belief representation makes object-centric representation error-tolerant while the object-centric representation makes the belief more expressive and structured. It would be interesting to extend this work to agent learning in complex 3D environments like the VizDoom environment.

Acknowledgements

References

Agnew, W. and Domingos, P. Relevance-guided modeling of object dynamics for reinforcement learning. *arXiv preprint*, 2020.

Burgess, C. P., Matthey, L., Watters, N., Kabra, R., Higgins, I., Botvinick, M., and Lerchner, A. Monet: Unsupervised scene decomposition and representation. *arXiv preprint arXiv:1901.11390*, 2019.

Chen, C., Deng, F., and Ahn, S. Object-centric representation and rendering of 3d scenes. *arXiv preprint arXiv:2006.06130*, 2020.

Crawford, E. and Pineau, J. Exploiting spatial invariance for scalable unsupervised object tracking. *arXiv preprint arXiv:1911.09033*, 2019a.

Crawford, E. and Pineau, J. Spatially invariant unsupervised object detection with convolutional neural networks. In *Proceedings of AAAI*, 2019b.

Creswell, A., Nikiforou, K., Vinyals, O., Saraiva, A., Kabra, R., Matthey, L., Burgess, C., Reynolds, M., Tanburn, R., Garnelo, M., et al. Alignnet: Unsupervised entity alignment. *arXiv preprint arXiv:2007.08973*, 2020.

Davidson, G. and Lake, B. M. Investigating simple object representations in model-free deep reinforcement learning. *Cog. Sci.*, 2020.

Devin, C., Abbeel, P., Darrell, T., and Levine, S. Deep object-centric representations for generalizable robot learning. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, 2018.

Diuk, C., Cohen, A., and Littman, M. L. An object-oriented representation for efficient reinforcement learning. In *International conference on Machine learning*, 2008.

Doucet, A., De Freitas, N., and Gordon, N. An introduction to sequential monte carlo methods. In *Sequential Monte Carlo methods in practice*, pp. 3–14. Springer, 2001.

Eslami, S. A., Heess, N., Weber, T., Tassa, Y., Szepesvari, D., and Hinton, G. E. Attend, infer, repeat: Fast scene understanding with generative models. In *Advances in Neural Information Processing Systems*, pp. 3225–3233, 2016.

Gangwani, T., Lehman, J., Liu, Q., and Peng, J. Learning belief representations for imitation learning in pomdps. In *Uncertainty in Artificial Intelligence*, 2020.

Goel, V., Weng, J., and Poupart, P. Unsupervised video object segmentation for deep reinforcement learning. *Advances in Neural Information Processing Systems*, 2018.

Goyal, A., Lamb, A., Gampa, P., Beaudoin, P., Levine, S., Blundell, C., Bengio, Y., and Mozer, M. Object files and schemata: Factorizing declarative and procedural knowledge in dynamical systems. *arXiv preprint arXiv:2006.16225*, 2020.

Greff, K., van Steenkiste, S., and Schmidhuber, J. Neural expectation maximization. In *Advances in Neural Information Processing Systems*, pp. 6691–6701, 2017.

Greff, K., Kaufmann, R. L., Kabra, R., Watters, N., Burgess, C., Zoran, D., Matthey, L., Botvinick, M., and Lerchner, A. Multi-object representation learning with iterative variational inference. *arXiv preprint arXiv:1903.00450*, 2019.

Greff, K., van Steenkiste, S., and Schmidhuber, J. On the binding problem in artificial neural networks. *arXiv preprint arXiv:2012.05208*, 2020.

Gregor, K., Rezende, D. J., Besse, F., Wu, Y., Merzic, H., and van den Oord, A. Shaping belief states with generative environment models for rl. In *Advances in Neural Information Processing Systems*, pp. 13475–13487, 2019.

Guo, D., Pires, B. A., Piot, B., Grill, J.-b., Altché, F., Munos, R., and Azar, M. G. Bootstrap latent-predictive representations for multitask reinforcement learning. *arXiv preprint arXiv:2004.14646*, 2020.

Guo, Z. D., Azar, M. G., Piot, B., Pires, B. A., and Munos, R. Neural predictive belief representations. *arXiv preprint arXiv:1811.06407*, 2018.

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

Igl, M., Zintgraf, L., Le, T. A., Wood, F., and Whiteson, S. Deep variational reinforcement learning for pomdps. *arXiv preprint arXiv:1806.02426*, 2018.

Jakab, T., Gupta, A., Bilen, H., and Vedaldi, A. Unsupervised learning of object landmarks through conditional image generation. In *Advances in neural information processing systems*, pp. 4016–4027, 2018.

Janner, M., Levine, S., Freeman, W. T., Tenenbaum, J. B., Finn, C., and Wu, J. Reasoning about physical interactions with object-oriented prediction and planning. In *International Conference on Learning Representations*, 2019.

Jiang, J. and Ahn, S. Generative neurosymbolic machines. In *Advances in Neural Information Processing Systems*, 2020.

Jiang, J., Janghorbani, S., De Melo, G., and Ahn, S. Scalor: Generative world models with scalable object representations. In *International Conference on Learning Representations*, 2019.

Kaelbling, L. P., Littman, M. L., and Cassandra, A. R. Planning and acting in partially observable stochastic domains. *Artificial intelligence*, 1998.

Kahneman, D., Treisman, A., and Gibbs, B. J. The reviewing of object files: Object-specific integration of information. *Cognitive psychology*, 24(2):175–219, 1992.

Karkus, P., Hsu, D., and Lee, W. S. Qmdp-net: Deep learning for planning under partial observability. *arXiv preprint arXiv:1703.06692*, 2017.

Keramati, R., Whang, J., Cho, P., and Brunskill, E. Strategic object oriented reinforcement learning. 2018.

Kingma, D. P. and Welling, M. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

Kosiorek, A., Kim, H., Teh, Y. W., and Posner, I. Sequential attend, infer, repeat: Generative modelling of moving objects. In *Advances in Neural Information Processing Systems*, pp. 8606–8616, 2018.

Kossen, J., Stelzner, K., Hussing, M., Voelcker, C., and Kersting, K. Structured object-aware physics prediction for video modeling and planning. *arXiv preprint arXiv:1910.02425*, 2019.

Kulkarni, T. D., Gupta, A., Ionescu, C., Borgeaud, S., Reynolds, M., Zisserman, A., and Mnih, V. Unsupervised learning of object keypoints for perception and control. In *Advances in neural information processing systems*, pp. 10724–10734, 2019.

Le, T. A., Igl, M., Rainforth, T., Jin, T., and Wood, F. Auto-encoding sequential monte carlo. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=BJ8c3f-0b.

Liang, Y., Machado, M. C., Talvitie, E., and Bowling, M. State of the art control of atari games using shallow reinforcement learning. *arXiv preprint arXiv:1512.01563*, 2015.

Lin, Z., Wu, Y.-F., Peri, S. V., Jiang, J., and Ahn, S. Improving generative imagination in object-centric world models. In *International Conference on Machine Learning*, pp. 4114–4124, 2020a.

Lin, Z., Wu, Y.-F., Peri, S. V., Sun, W., Singh, G., Deng, F., Jiang, J., and Ahn, S. Space: Unsupervised object-oriented scene representation via spatial attention and decomposition. In *International Conference on Learning Representations*, 2020b.

Locatello, F., Weissenborn, D., Unterthiner, T., Mahendran, A., Heigold, G., Uszkoreit, J., Dosovitskiy, A., and Kipf, T. Object-centric learning with slot attention, 2020.

Ma, X., Karkus, P., Hsu, D., and Lee, W. S. Particle filter recurrent neural networks. *AAAI*, 2019.

MattChanTK. gym-maze. https://github.com/MattChanTK/gym-maze, 2020.

Milan, A., Leal-Taixé, L., Reid, I., Roth, S., and Schindler, K. Mot16: A benchmark for multi-object tracking. *arXiv preprint arXiv:1603.00831*, 2016.

Mnih, V., Badia, A. P., Mirza, M., Graves, A., Lillicrap, T., Harley, T., Silver, D., and Kavukcuoglu, K. Asynchronous methods for deep reinforcement learning. In *International conference on machine learning*, 2016.

Mohan, S. and Laird, J. An object-oriented approach to reinforcement learning in an action game. In *Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, 2011.

Moreno, P., Humplik, J., Papamakarios, G., Pires, B. A., Buesing, L., Heess, N., and Weber, T. Neural belief states for partially observed domains. In *NeurIPS 2018 workshop on Reinforcement Learning under Partial Observability*, 2018.

Naesseth, C., Linderman, S., Ranganath, R., and Blei, D. Variational sequential monte carlo. In *International Conference on Artificial Intelligence and Statistics*, pp. 968–977. PMLR, 2018.

Piché, A., Thomas, V., Ibrahim, C., Bengio, Y., and Pal, C. Probabilistic planning with sequential monte carlo methods. In *International Conference on Learning Representations*, 2018.

Scholz, J., Levihn, M., Isbell, C., and Wingate, D. A physics-based model prior for object-oriented mdps. In *International Conference on Machine Learning*, 2014.

Todorov, E., Erez, T., and Tassa, Y. Mujoco: A physics engine for model-based control. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 5026–5033. IEEE, 2012.

Tschiatschek, S., Arulkumaran, K., Stühmer, J., and Hofmann, K. Variational inference for data-efficient model learning in pomdps. *arXiv preprint arXiv:1805.09281*, 2018.

van Steenkiste, S., Chang, M., Greff, K., and Schmidhuber, J. Relational neural expectation maximization: Unsupervised discovery of objects and their interactions. *arXiv preprint arXiv:1802.10353*, 2018.

Veerapaneni, R., Co-Reyes, J. D., Chang, M., Janner, M., Finn, C., Wu, J., Tenenbaum, J. B., and Levine, S. Entity abstraction in visual model-based reinforcement learning. *arXiv preprint arXiv:1910.12827*, 2019.

Wang, Y., Liu, B., Wu, J., Zhu, Y., Du, S. S., Fei-Fei, L., and Tenenbaum, J. B. Dualsmc: Tunneling differentiable filtering and planning under continuous pomdps. *arXiv preprint arXiv:1909.13003*, 2019.

Watters, N., Matthey, L., Bosnjak, M., Burgess, C. P., and Lerchner, A. Cobra: Data-efficient model-based rl through unsupervised object discovery and curiosity-driven exploration. *arXiv preprint arXiv:1905.09275*, 2019.

Williams, R. J. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8(3-4):229–256, 1992.

Zhu, P., Li, X., Poupart, P., and Miao, G. On improving deep reinforcement learning for pomdps. *arXiv preprint arXiv:1704.07978*, 2017.