International Journal of Computer Vision (2020) 128:1360–1374
https://doi.org/10.1007/s11263-019-01238-5


# Model-Based Robot Imitation with Future Image Similarity

A. Wu[1] · A. J. Piergiovanni[1] · M. S. Ryoo[1,2]

Received: 27 July 2018 / Accepted: 18 September 2019 / Published online: 11 October 2019


**Abstract**
We present a visual imitation learning framework that enables learning of robot action policies solely based on expert samples without any robot trials. Robot exploration and on-policy trials in a real-world environment could often be expensive/dangerous. We present a new approach to address this problem by learning a future scene prediction model solely from a collection of expert trajectories consisting of unlabeled example videos and actions, and by enabling action selection using *future image similarity*. In this approach, the robot learns to visually imagine the consequences of taking an action, and obtains the policy by evaluating how similar the predicted future image is to an expert sample. We develop an action-conditioned convolutional autoencoder, and present how we take advantage of future images for zero-online-trial imitation learning. We conduct experiments in simulated and real-life environments using a ground mobility robot with and without obstacles in reaching target objects. We explicitly compare our models to multiple baseline methods requiring only offline samples. The results confirm that our proposed methods perform superior to previous methods, including 1.5 × and 2.5 × higher success rate in two different tasks than behavioral cloning.

**Keywords** Robot action policy learning · Behavioral cloning · Model-based RL

## 1 Introduction

Learning robot control policies directly from visual input is a major challenge in robotics. Recently, progress has been made with reinforcement learning (RL) methods to learn action policies from images using convolutional neural networks (CNNs) (Levine et al. 2016; Zhu et al. 2016; Finn and Levine 2017; Sadeghi et al. 2017; Peng et al. 2018). Taking advantage of thousands of online trials as well as well-defined reward signals [e.g., thousands of robot hours (Levine et al. 2016)], these approaches provide promising results that CNNs within the RL framework make better state representation learning possible. Model-based RL works (Finn and Levine 2017) further demonstrated that learning a state-transition model from visual data is able to reduce the number of samples needed for RL. This 'state-transition' model capturing how state representations change conditioned on actions is also closely related to 'future image prediction' (or representation regression) models in computer vision (Vondrick et al. 2016; Oh et al. 2015; Denton and Fergus 2018).

Observing expert behavior can greatly accelerate learning. Humans are able to adapt and imitate a new ability without much practice and much more quickly than when learning from scratch. For this reason, the ability to learn from demonstrations (LfD) (Argall et al. 2009) is highly desirable for autonomous agents. Imitation learning is a form of policy learning, taking advantage of expert examples instead of explicit reward signals. There are two standard approaches to imitation learning: behavioral cloning (BC) and inverse reinforcement learning (IRL). Behavioral cloning is a supervised policy learning problem: the model is given the expert action taken in a state and attempts to replicate the state-action distribution. On the other hand, IRL attempts to infer the underlying reward signal from expert trajectories and

Communicated by Anelia Angelova, Gustavo Carneiro, Niko Sünderhauf, Jürgen Leitner.

Alan Wu and A. J. Piergiovanni these authors contributed equally to the paper.

✉ A. Wu
alanwu@iu.edu

A. J. Piergiovanni
ajpiergi@indiana.edu

M. S. Ryoo
mryoo@indiana.edu

[1] Indiana University, Bloomington, IN, USA

[2] Stony Brook University, Stony Brook, NY, USA

then derives action policies using reinforcement learning. This learns more generalized policies than behavioral cloning (Finn et al. 2016; Sutton and Barto 2017).

However, each of the three research directions mentioned (i.e., model-based RL, behavioral cloning, and IRL-based imitation learning) has its own limitation when applied to a real-world robot task. In a real-world robot scenario, it often is the case that a very limited number of expert examples is provided for the learning; robot explorations or on-policy trials could often be expensive and dangerous in a real-world environment. Behavioral cloning is able to obtain a policy solely based on such expert trajectories without any further trials. However, due to the amount of expert sample data being inevitably limited, encountering unseen states often leads to a covariate shift that can be irrecoverable without querying the expert (i.e., in unseen states, the predicted action can be unreliable or random) (Ross et al. 2017). On the other hand, while model-based RL and IRL-based imitation learning are able to learn policies that generalize better, both still require hundreds of real-world, on-policy trials for the learned policy to become accurate. This makes the learning process less practical for real-life autonomous agents in complex real-world environments, particularly when they need to interact with humans or when the environment is not controlled.

Ideally, we desire learning a real-world action policy from a limited number of expert trajectories without any further trials that still generalizes well to unseen states and actions. In this paper, we focus on the development of a convolutional model for such zero-trial policy learning. Our main idea is to enable (1) learning of a realistic future prediction (i.e., state-transition) model solely from a limited number of expert trajectories (i.e., without any robot exploration) with our CNN, and to make (2) better zero-trial action policy learning possible by taking advantage of such transition model.

We propose a model for imitation learning in continuous action spaces with completely offline learning (i.e., requiring no online trials) that produces better robot policies compared to behavioral cloning. We accomplish this through the use of raw visual sensory inputs (i.e., first person robot images) using the proposed method of *future image prediction*, shown in Fig. 1 and described in Sect. 4.1. Our future image prediction model is a CNN that predicts a realistic future image given a current image and the action taken. By learning a model predicting the consequences of taking an action, we are able to use it to learn a better action policy. In our approach (i.e., action selection using *future image similarity*), we learn the function that evaluates how similar the predicted future image is to expert images along with the future prediction model. Our action policy directly relies on this evaluation function as the critic, thereby selecting the action that leads to future images most similar to the expert images.

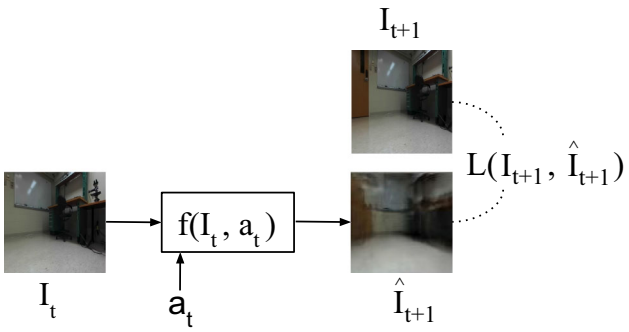


**Fig. 1** Our future prediction model $f(I_t, a_t)$ learns to predict a future image given the current state (i.e., image) and action (i.e., change in pose). This serves as the state transition model to evaluate actions in imitating expert behavior. Notice that the model was able to predict the door on the left of the scene to appear as the robot takes the action $a_t$ of turning left

We designed a CNN composed of a convolutional autoencoder for action-conditioned future image prediction. Given an input frame, the model maps it to a lower-dimensional intermediate representation, which abstracts scene information of the current image. The model then concatenates the action vector, and the convolutional decoder produces an image from this concatenated representation. We design and evaluate several CNN models for our action-conditioned autoencoder: (i) using a standard linear state-action representation, (ii) using a spatial state-action representation, and (iii) a stochastic version of the model by learning to generate a prior distribution.

We conduct our experiments using a ground mobility robot in both real-world office and simulator environments containing everyday objects (e.g., desks, chairs, monitors). The robot's goal is to approach a target object without hitting any obstacles. We do not use any artificial markers or labels for the target object in our trajectories. The learning was done entirely based on first-person videos taken from a robot's viewpoint without any annotations. This makes the future prediction model learning challenging as the entire frame changes based on the action taken, and not just a few pixels in the frame as in manipulator robots. Expert first-person robot videos were obtained via teleoperation.

The technical contributions of this paper are: (1) Learning of a realistic future image prediction (i.e., state-transition) model from only expert samples. We show it generalizes to non-expert actions and unseen states. (2) Using the learned state-transition model to improve behavioral cloning by taking advantage of the 'imagined' (i.e., predicted) consequences of non-expert actions. Our second component only relies on expert images, similar to zero-shot learning works (Torabi et al. 2018; Pathak et al. 2018).

## 2 Related Works

Convolutional neural networks are high capacity function approximators with the ability to learn intermediate representations, making them ideal for our models learning from high-dimensional visual inputs. Although CNNs are commonly used for the classification of images and videos (Simonyan and Zisserman 2014; Liu et al. 2016; Carreira and Zisserman 2017; Piergiovanni and Ryoo 2017), they can also be used for learning other types of functions [e.g., playing Atari games using Q-learning (Mnih et al. 2015)].

### 2.1 Behavioral Cloning and Imitation Learning

There are two main approaches to imitation learning: behavioral cloning (BC) and inverse reinforcement learning (IRL). Behavioral cloning is a supervised form of imitation learning; a simple but effective and direct solution. Given a state, or an observation such as an image, the model learns to produce an action matching one provided by an expert (Pomerleau 1991). The main challenge of BC is that it is very difficult to obtain supervised training data that covers all possible test conditions the robot may encounter and thus difficult for it to learn a model that generalizes to unseen states. Techniques such as using additional sensors (Pomerleau 1989; Giusti et al. 2016; Bojarski et al. 2016) or querying the expert when unseen states are encountered (Ross et al. 2017) have led to moderate success. Some have taken a variant approach to BC by only accessing expert observations and deriving resulting actions (Liu et al. 2017; Torabi et al. 2018), similar to inverse dynamics. Others have shown improvements to BC by injecting noise into the expert policy during data collection, forcing corrective action to disturbances (Laskey et al. 2017).

Inverse reinforcement learning (IRL) techniques recover a reward signal from the expert trajectories which are state-action pairs (Ng and Jordan 2000; Abbeel and Ng 2004; Ziebart et al. 2008; Wulfmeier et al. 2015). The learned reward function is used with traditional reinforcement learning methods, which usually spend thousands of trials or more to learn action policies matching expert behavior. These methods have recently achieved success with high-dimensional data inputs (Finn et al. 2016; Ho et al. 2016). Other works take on an adversarial formulation to imitation learning, such as GAIL (Ho and Ermon 2016). However, GAIL requires an extensive trial-and-error learning period by its nature, suffers from high variance when training stochastic policies, and is not suitable for the scenarios with zero online trials, which this paper focuses on.

Baram et al. (2017) present a model-based generative adversarial imitation learning (MGAIL) approach that uses fewer expert examples and interactions with the environment than GAIL while achieving similar or better performance in classical simulated RL experiments (e.g., mountain car, hopper) through the use of a forward model and multi-step transitions. However, even with such low dimensional inputs, 10's of thousands of interactions with the environment were required to train the model. Learning with high dimensional images with mobility robots in a real-life environment would make such an approach extremely difficult or nearly prohibitive.

### 2.2 Model-Based Reinforcement Learning

One major drawback of reinforcement learning is the large number of online trials needed for the agent to learn the policy. Although some have trained their agent in a simulated environment (Sadeghi et al. 2017; Zhu et al. 2016) to minimize the amount of learning needed in a real-life environment, it is very challenging to design realistic simulation models that transfer well to accurate real-world policies.

Model-based RL, utilizing a state-transition model, is able to significantly reduce the number of on-policy samples needed for reinforcement learning. Recent works took advantage of convolutional models to learn image-level state transitions conditioned on agent/robot actions. For example, using action-conditioned models to predict future scenes (Oh et al. 2015; Chiappa et al. 2017) or robot states (Peng et al. 2018) has been employed in simulated environments. Oh et al. (2015) and Chiappa et al. (2017) learn transition models using recurrent networks to predict scenes for multiple steps into the future conditioned on a sequence of actions. Peng et al. (2018) learn an inverse model for multi-jointed humanoid robots to perform complex tasks such as flipping and spinning. The agent performs imitation learning by receiving rewards for following reference motions and task completions. Although the image inputs for these works were high-dimensional, they were limited to video games or simulated environments and did not need to address the highly stochastic nature of the real-world environment. In our robot experiments, we found that accurate future predictions were much more challenging to achieve in the real-life environment than the simulated environment. We discuss this more with the SSIM measures in Sect. 6.2.

### 2.3 Model-Based Imitation Learning

There are also approaches utilizing state-transition model learning (or its inverse model learning) for imitation learning. In particular, recent works (Torabi et al. 2018; Pathak et al. 2018) attempted learning of agent policies without expert action labels. The setting was that although state-action pairs from robot exploration trajectories (random or guided) could be used for the state-transition model learning, only the use of expert states (images in our case) were allowed for policy learning. This is similar to our learning process in the sense

that no action labels are used in the policy learning stage (i.e., critic learning).

Torabi et al. (2018) presented an approach of behavioral cloning from observation. They first allow the agent to explore its environment to learn an inverse dynamics model (i.e., learn the action causing the state to change from $s_t$ to $s_{t+1}$). Then they infer the action from the state transitions of expert trajectories, yielding state-action$_{inferred}$ pairs to train a generative model. Because they are focusing on a more challenging problem (i.e., they do not access expert actions directly for the policy learning), many of their experiments showed worse performance to standard behavioral cloning using both state and action information.

Forward consistency was a recently developed model that only relies on off-policy learning (Pathak et al. 2018). The model built on the premise that reaching a goal is more important than how it is reached - that the path does not need to be optimal. They first predict an action based on the current image, a goal image, and the previous action taken. As long as the predicted action leads to a state that is consistent with the ground truth state, regardless of the predicted action being different from a ground truth action, the predicted action is considered to be 'consistent' and the forward consistency network learns such behavior. This learning could be done without expert action information, as long as it already has a state-transition model learned from non-expert state-action pairs. Although the model requires an additional input (i.e., goal image) compared to standard behavioral cloning or imitation learning, it showed a potential in this direction.

### 2.4 Future Scene Prediction

Several existing works use CNNs to predict (i.e., regress) future human pose (Chao et al. 2016; Walker et al. 2017; Lee and Ryoo 2017) or object/pixel movements in a scene (Vondrick et al. 2016; Finn and Levine 2017; Walker et al. 2014). However, the background in most of these works is static and only a small portion of the frame (e.g., specific objects or a robot arm) actually changes. This is particularly limiting when applied to videos with dynamic camera movements, where pixels in the entire scene move and objects may newly appear or leave the scene frequently. Further work on movie scene prediction (Vondrick et al. 2016) contained some samples with camera motion, but many examples had a static camera. Liu et al. (2017) proposed a model able to reconstruct interpolated frames as well as extrapolated future frames. This was a voxel-based approach using multiple input frames to determine scene flows. However, these works were purely designed for computer vision problems rather than real-world robot applications, making them less suitable for first-person robot videos. Further, these models are not action-conditioned, limiting their ability to predict different futures.

There are also works on future prediction conditioned on a particular set of parameters, including robot action (Finn et al. 2016) and camera viewpoint (Tatarchenko et al. 2016; Dosovitskiy et al. 2017; Zhou et al. 2016). All these models use a similar approach where an image is fed into a CNN and the other parameters are concatenated directly with an intermediate representation of the CNN (e.g., at a fully connected layer). In this paper, we use future prediction architectures which concatenate a learned spatial form of the action representation to the intermediate convolutional layer. This is done by making a small sub-network learn a better representation of the action vector using reshaping and dense layers, before the concatenation. We propose an action-conditioned autoencoder CNN architecture for the future image prediction, and confirm it benefits the robot learning.

Babaeizadeh et al. (2017) proposed a stochastic version of future prediction, which was also tested with videos of a manipulator robot but without actual robot control. Denton and Fergus (2018) similarly introduced a stochastic state representation that can learn a prior distribution capturing the various outcomes in complex environments. The use of such state representation allows for generation of different future images from the same state, based on the learned probabilities of what can happen. We combine this stochastic state representation with the action-conditioned future predictor to generate realistic and clear future images. We find that this stochastic state is particularly beneficial for real-world data where the environment is quite complex, allowing us to jointly train the model in an end-to-end fashion for better policy learning.

## 3 Problem Formulation

We focus on the problem of 'zero-trial' imitation learning, which essentially is the setting of behavioral cloning. By 'zero-trial' we mean that we learn an action policy of the robot solely based on expert trajectories (i.e., human samples) without any on-policy robot trials. That is, the robot is prohibited from interacting with the environment to obtain its own samples to check/update the policy. The learned policy is directly applied to the robot in the real-world environment. Notably, this is different from approaches such as GAIL (Ho and Ermon 2016), which requires both expert trajectories and on-policy robot trials to learn the policy.

We formulate the problem of robot policy learning as the learning of convolutional neural network (CNN) parameters. Our agent (i.e., a robot) receives an image input $I_t$ at every time step $t$. Given $I_t$, our goal is to learn a policy, $\pi$, which given the current state, selects the optimal action to take. That is, the policy is modeled as a neural network, and takes the current image as input and produces an action $a = \pi(I_t)$. In the case of robot imitation learning, $\pi$ has to be learned from

expert trajectories (i.e., samples), which are often obtained by humans controlling the robot. Note that the action $a_t$ can be the direct motor control commands. In this work, we consider continuous state and action spaces where the states are simply images from the robot camera.

## 4 Zero-Trial Imitation with a Visual Model

We present a new zero-trial imitation learning method using *future image prediction*. Our future image prediction model is a CNN that predicts a realistic future image given its current image and an action. The model first learns how the image (from the robot camera) visually changes when the robot takes an action, and learns to evaluate the predicted images to obtain the action policy. The method is illustrated at a conceptual level in Fig. 2. Section 4.1 presents the details of our future image prediction model, designed as an action-conditioned autoencoder CNN. Our stochastic image prediction model learns a prior to capture the uncertainty in an environment and generates clearer images. Section 4.2 describes our approach of selecting actions by learning *future image similarity* by leveraging non-expert actions.

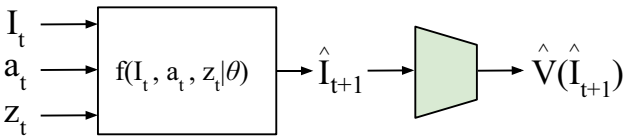


**Fig. 2** Illustration of the action policy network using a stochastic future image predictor $f(\cdot)$ and critic $\hat{V}$. The stochastic image predictor uses a learned prior $z_t$, allowing for better generation of images in real-life environments. The predicted image is used as input to the critic, which determines the value of the predicted image

### 4.1 Action-Conditioned Future Prediction CNN

Our CNN architecture is designed to predict future images based on the current image and an action. This can be viewed as a state-transition model used in model-based reinforcement learning, following a Markov decision process (MDP). Further, we learn a prior that captures the stochastic nature of real-world environments motivated by Denton and Fergus (2018). Our proposed model is shown in Fig. 3: the action-conditioned stochastic autoencoder CNN.

The model is composed of two main CNN components: the encoder $Enc$ and the decoder $Dec$. The encoder learns to map the current image to a latent representation, $z_I = Enc(I_t)$. We designed our autoencoder to explicitly preserve the spatial information in its intermediate representations, making $Enc$ and $Dec$ to be fully convolutional, which allows our intermediate scene representation to preserve the spatial shape.

We use another neural network component, $Act$, which learns a representation of the action, $z_a = Act(a_t)$. We also design our action representation CNN $Act$ to have several fully connected layers followed by a reshaping (i.e., spatial de-pooling) layer. This makes the output of $Act$ have the spatial dimensionality as the image representation, i.e., $Enc(I_t)$, which allows the two representations to be concatenated together. Further, this allows for the action representation learned to be different at each spatial location, in contrast to previous works (Oh et al. 2015; Finn et al. 2016).

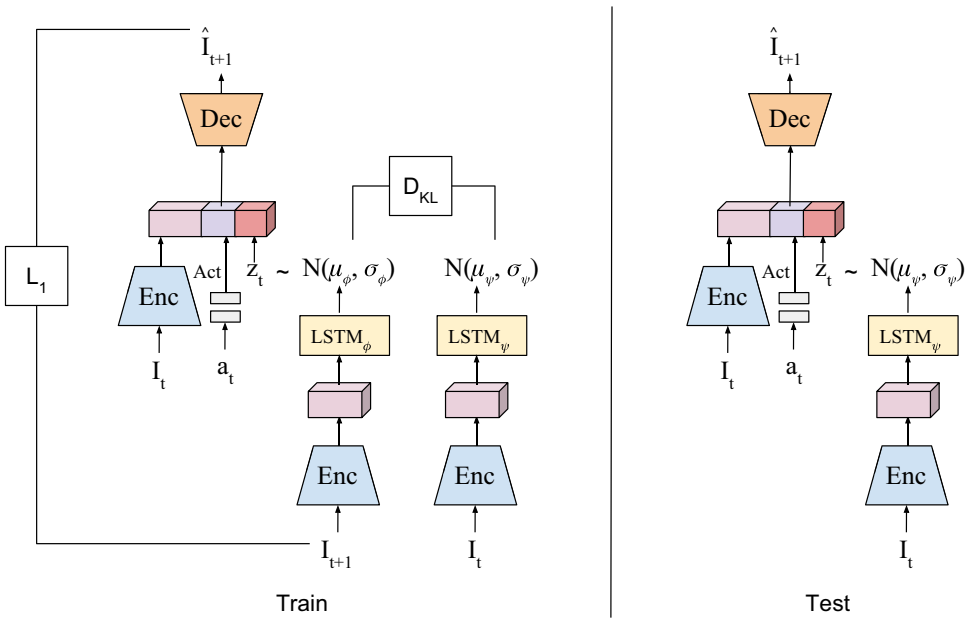


**Fig. 3** Illustration of the stochastic image predictor model. This model takes input of the image and action, but also learns to generate a prior, $z_t$, which varies based on the input sequence. This is further concatenated with the representation before future image prediction. The use of the prior allows for better modeling in stochastic environments and generates clearer images

Thirdly, the latent variable $z_t$ is sampled from a learned prior distribution $p_\psi$ also implemented as a neural network. The latent variable $z_t$, as explained in Denton and Fergus (2018), captures the stochastic information in videos that is lacking in deterministic models. A separate inference model $q_\phi$ is used only during training and removed during test. $q_\phi$ is forced to be close to $p_\psi$ using a KL-divergence term, which helps prevent $z_t$ from directly copying $I_t$. As the prior learns stochastic characteristics of video across time from a sequence of images, convolutional LSTMs $LSTM_\phi$ and $LSTM_\psi$ are used to capture the temporal elements. We use multiple frames as input during training that benefits the learning of the prior, but we use only a single image during test. Therefore, we omit the use of an LSTM in the encoder-decoder network.

The representations of the current image, action, and learned prior are concatenated together $[z_I, z_a, z_t]$ and used as input to the decoder which then reconstructs the future image:

$$\hat{I}_{t+1} = Dec([Enc(I_t), Act(a_t), z_t]) \quad (1)$$

As mentioned above, this can also be viewed as learning a state-transition model from robot videos,

$$f(I_t, a_t, z_t) = Dec([Enc(I_t), Act(a_t), z_t])$$

Figure 4 provides an example of the predicted images in real-life lab environment and simulation environment given different actions, visually showing we are able to generate realistic, unseen future images.

The model has two loss terms. It is trained to minimize $L_1$ error between the predicted image and the ground truth future image. It is also simultaneously trained to minimize the KL-divergence between the prior distribution $p_\psi$ and an inferred distribution $q_\phi$. $\beta$ is a hyperparameter set to 0.0001 as was done in Denton and Fergus (2018). We used an input sequence of T = 5 frames during training and a single frame during test. Multiple frame inputs during test could improve learning and thus a worthwhile pursuit in the future.

$$\mathcal{L}_\mathcal{F}(I_{1:T+1}) = \sum_{t=1}^{T} \Big( |\hat{I}_{t+1} - I_{t+1}| + \beta D_{KL}[q_\phi(z_t|I_{1:t+1}) \,||\, p_\psi(z_t|I_{1:t})] \Big) \quad (2)$$

Figure 3 shows our full model as well as its losses for the training. We emphasize that during training, we only have expert future images as a result of expert actions, greatly limiting the training data. However, as shown in Fig. 4, our learned model is able to generalize to unseen, non-expert actions by generating realistic and accurate images.

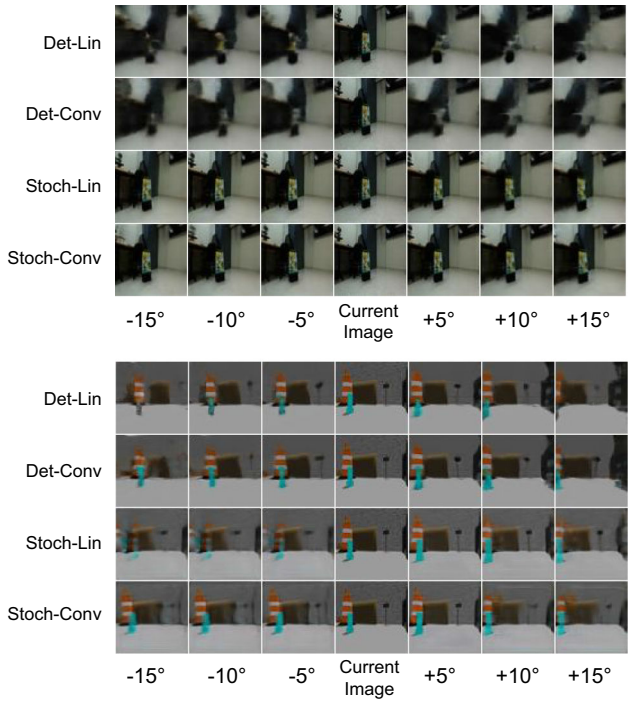


**Fig. 4** Predicted future images in the real-life lab (top) and simulation (bottom) environments taking different actions. Top two rows of each environment: deterministic model with linear and convolutional state representation, respectively. Bottom two rows: stochastic model with linear and convolutional state representation, respectively. Center image of each row is current image with each adjacent image to the left turning $-5°$ and to the right turning $+5°$

## 4.2 Imitation Learning with Future Image Similarity

To use our learned state-transition model for imitation learning, we train a 'critic' function to evaluate how similar a generated state-action pair is to an expert state-action pair. The idea is to train a CNN that can distinguish state-action pairs that look like expert samples from non-expert samples, similar to the learning of the 'discriminator' in a generative adversarial imitation learning (GAIL) (Ho and Ermon 2016). Learning such a CNN will allow for the selection of the optimal action given a state (e.g., image) by enumerating through several action candidates.

More specifically, we use the above future image prediction to generate images for many different candidate actions. We then learn a critic CNN that evaluates how similar the predicted future image is to the ground truth, expert future image. We train a CNN to model this function, $\hat{V}$, which performs this evaluation.

To train this CNN, we use pixel-wise $L_1$ difference between the predicted future image and the actual future image (i.e., $|I_{t+1} - \hat{I}_{t+1}|$) as our similarity measure. We train the CNN $\hat{V}(\hat{I}_{t+1}; w)$ governed by the parameters $w$:

$$w^* = \arg\min_w \| \hat{V}(\hat{I}_{t+1}; w) - |I_{t+1} - \hat{I}_{t+1}| \| \quad (3)$$

Once learned, given a current robot image $I_t$ and an action candidate $a_t$, we are able to evaluate how good the action candidate is solely based on the predicted future image $\hat{I}_{t+1}$ by computing

$$\hat{V}(\hat{I}_{t+1} = Dec([Enc(I_t), Act(a_t), z_t]); w^*) \quad (4)$$

The motivation for this approach is that actions taken by the robot in one state should result in a future image that is similar to the next expert image, if the imitation is being done correctly. $\hat{V}$ reflects this imitation constraint while only taking advantage of image frames. Note that this is possible because we have the explicit future image predictor which allows us to compare two different images (i.e., real expert images vs. model predicted images), enabling the training/learning of this visual imitation critic.

The training procedure of this method is illustrated in Fig. 5. We use our pre-trained future prediction model to obtain the predicted future image from each state-action pair $(I_t, a_t)$ in our training set of expert trajectories. We can then sample many random candidate actions, which contain many non-expert actions.

The main advantage of this approach is the ability to 'imagine' unseen future images resulting from non-expert actions taken in expert states. This allows the model to benefit from more training data without further interaction with the environment. Once learned, the optimal action at each state $I_t$ is selected by taking the max value from the randomly sampled candidate actions:

$$a_t = \arg\max_a \hat{V}(f(I_t, a, z_t)) \quad (5)$$

where

$$f(I_t, a, z_t) = Dec([Enc(I_t), Act(a), z_t]) \quad (6)$$

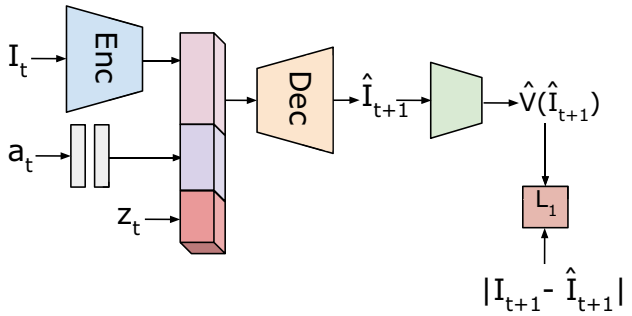


**Fig. 5** Training of the critic function $\hat{V}$ with the future prediction model for future image similarity. We generate many future images from the current image and various actions, and train the critic to match the $L_1$ difference between the predicted and ground truth image

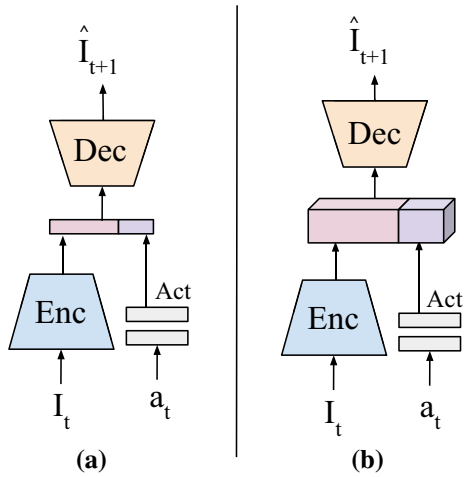


**Fig. 6** Illustration of the future prediction with a deterministic image predictor. The networks take an image and action as input, concatenate the learned representations, then generate the future image. **a** Linear state representation. **b** Convolutional state representation

## 5 Alternative Models

### 5.1 Ablations

To evaluate our design decisions, we compare our full model against other models that replace or remove some of the components. Specifically, we compared using linear state representations against convolutional state representations. Secondly, we examined the impact of the learned prior on our action policy. Without the learned prior, we simply have a deterministic image predictor model that is trained without the latent variable $z_t$. We can see in Fig. 4 that the predicted images from the deterministic models are blurrier, but wanted to determine if the blurriness had a significant impact on action policy. The models resulting from these ablations are shown in Fig. 6.

### 5.2 Baselines

In addition to different versions of our approach as mentioned above, we implemented a number of baselines: behavioral cloning, handcrafted critic learning, and forward consistency (Pathak et al. 2018). Similar to ours, all of these models do not perform any learning from interacting with the environment, as opposed to RL models. Using a CNN formulation, each baseline is described below.

#### 5.2.1 Behavioral Cloning

Behavioral cloning (BC) is a straightforward approach to imitation learning, which we formulate as a CNN. The network directly learns a function, $\hat{a}_t = \pi(I_t)$, given current state (i.e., image) as an input and outputs the correspond-

ing action. The BC model is trained using expert samples consisting of a sequence of image-action pairs in a supervised fashion. However, when a given image is not part of the seen expert samples, BC often does not generalize well and produces essentially random actions in those states (Laskey et al. 2017). This is particularly challenging for real-world robots, where training data is greatly limited. The model is trained to minimize the error between the expert action, $a_t$, and the CNN predicted action, $\hat{a}_t$, using the $L_1$ distance:

$$\mathcal{L}_{\mathcal{B}}(a_t, \hat{a}_t) = |a_t - \hat{a}_t| \tag{7}$$

During inference, the current image is used as an input to the network $\pi$ and the predicted action is taken by the robot.

#### 5.2.2 Handcrafted Critic

Borrowing the idea of taking advantage of non-expert actions behind our future similarity model, we trained a handcrafted critic CNN based on expert angle change as another baseline to distinguish state-action pairs that look like expert samples from non-expert samples. This critic allows the selection of the optimal action given a state by enumerating through action candidates, just like our main approach. This model is simpler in that we formulate this as supervised learning and we also handcraft the 'shape' of the outputs.

Given a current image $I_t$ (serving as the state), we apply a CNN $C$ to obtain its vector representation. In parallel, we use a fully-connected neural network, $Act$, to produce a vector representation of the action $a_t$. The two vectors are concatenated and several fully-connected layers follow to produce the critic for state-action pairs:

$$\hat{Q}(I_t, a_t) = F([C(I_t), Act(a_t)]) \tag{8}$$

We train the critic function $\hat{Q}$ in a supervised fashion by handcrafting its target values based on the angular distance between the expert action and each candidate action. In this approach, we use the change in angular pose $\delta\theta$ as our action. Although the translational change $(\delta x, \delta y)$ can be independent of angular change, for simplicity, we set the proposed translational change as a function of angular change $\delta\theta = \arctan(\delta y/\delta x)$ so that there would be only a single parameter to learn. That is, we define our supervision signal to be:

$$Q(I_t, a_t) = -|\delta\theta_{exp} - \delta\theta_{a_t}| \tag{9}$$

where $\delta\theta_{exp}$ is the expert action and $\delta\theta_{a_t}$ is the candidate action.

Unlike behavioral cloning, this critic function benefits from the training data with non-expert actions while still only taking advantage of offline samples. That is, the training data for the critic CNN may contain pairings of expert state and non-expert actions. This provides the network with significantly more data, allowing more reliable learning even without performing any robot trials. The network is trained to minimize the $L_1$ error between the critic CNN output, $\hat{Q}(I_t, a_t)$ and the (handcrafted) ground truth state-action value, $Q(I_t, a_t)$.

$$\begin{aligned}\mathcal{L}_{\mathcal{Q}} &= |\hat{Q}(I_t, a_t) - Q(I_t, a_t)| \\ &= |F([C(I_t), Act(a_t)]) - Q(I_t, a_t)|\end{aligned} \tag{10}$$

During inference, we select the action that gives the maximum critic value (i.e., the action that is expected to be most similar to the expert action):

$$a_t = \arg\max_a \hat{Q}(I_t, a) \tag{11}$$

Since our action space is continuous, we randomly sample 20 candidate actions from a uniform distribution from $-30°$ to $+30°$ relative to the robot's current pose to learn the critic function. During inference, we evaluate each of the candidate actions for the current state, and select the one that maximizes the critic function. For a higher dimensional pose change such as by an aerial vehicle, the candidates would be sampled from a spherical cap instead of an arc for a ground mobility robot, and the handcrafted target value would become much more complex. Therefore, it would be desirable to have a critic function that does not require any handcrafting.

#### 5.2.3 Forward Consistency

We also experimentally compared our proposed method against the model with Forward Consistency loss from Pathak et al. (2018). Our setting is similar to Pathak et al. (2018) in that both learn state-transition models first based on off-policy trajectories and then learn the action policy network (i.e., GSP in Pathak et al. (2018) and future image similarity critic in ours) without any expert action information by relying only on expert images. The difference is that they used exploration trajectories for the training of their transition model while we used fewer expert trajectories. We took advantage of the code from the authors of the paper, and made it compatible with our problem setting by feeding expert trajectories instead of exploration trajectories for the training of their state transition model, as well as supplying a goal image of each task as required by their model.

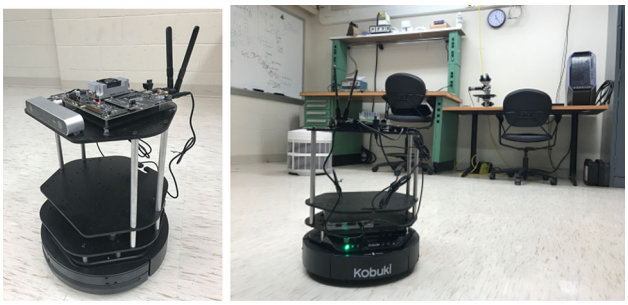


**Fig. 7** Images of our ground mobility robot (Turtlebot 2) and its environment

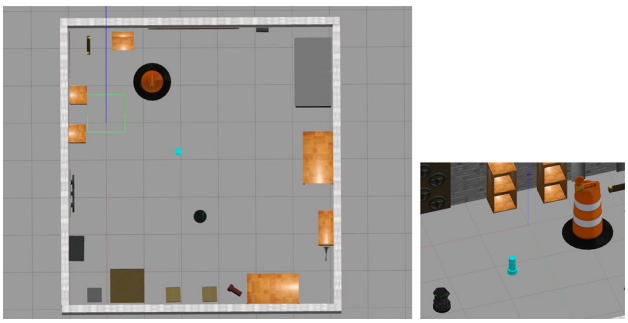

**Fig. 8** Our simulation environment in Gazebo: birdseye view (L) and side view (R). An obstacle (blue cylinder) is placed between the robot and the target (construction barrel)

## 6 Experiments

We performed a series of experiments with our method in two different environments, a real-life lab setting (Fig. 7) and a simulation setting (Fig. 8) in Gazebo, each with a different task. In the lab environment, the robot performed an approach task where the robot must move towards a target object. In the simulation environment, the robot also must approach a target but needed to avoid an obstacle placed in between. The latter is a more challenging task as the view of the target from the robot's perspective is partially obstructed by the obstacle and the robot must learn to avoid the obstacle without explicitly detecting it.

### 6.1 Datasets

We collected two datasets of ground mobility robot trajectories: one in a real-world room with everyday objects (e.g., desks, chairs), without any markers, and another in a simulated environment with different obstacles (one at a time), target object, and similar everyday objects. A trajectory is a sequence of video frames obtained during the robot execution, and each frame is annotated with the expert action taken in the state (translational $(x, y)$ and rotational $(\theta)$ coordinates). For each trajectory of the lab dataset, a human expert controlled the robot to move toward a target object selected from a total of 7 different objects. For the simulated dataset, the expert controlled the robot to move toward a single target while avoiding an obstacle object selected from 2 different objects. The initial robot placement and the target location were varied leading to varied background scenery. This is a very challenging setting in the aspect that we never explicitly annotate any target or obstacle objects, and there are other distinct everyday objects in the environment such as chairs and other lab equipment.

For the lab dataset, on average, we collected 35,100-frame trajectories for each target. We split our dataset into 32 training trajectories and 3 test trajectories per target, a total of 224 training trajectories and 21 test trajectories. For the simulation dataset, on average, we collected 100,400-frame trajectories for each obstacle. We split our dataset into 91 training trajectories and 9 test trajectories per obstacle, a total of 182 training trajectories and 18 test trajectories. The objective of our expert trajectory collection was to train our robot with the imitation learning framework. We perform both offline evaluation of the robot action model and online experiments of real-time robot trials, and report the results in the below subsections.

**Table 1** Assessment of image predictor models using SSIM with our real-world lab and simulation datasets. The higher the SSIM score is, the more accurate the prediction is

| Model | Lab dataset | Sim dataset |
|---|---|---|
| Det linear state | $0.7029 \pm 0.0086$ | $0.8856 \pm 0.0036$ |
| Det conv state | $0.7211 \pm 0.0082$ | $0.8983 \pm 0.0037$ |
| Stoch linear state | $0.7255 \pm 0.0145$ | $0.9251 \pm 0.0083$ |
| Stoch conv state | $0.7436 \pm 0.0153$ | $0.9220 \pm 0.0091$ |

### 6.2 Image Predictor Evaluation

Because our method depends on the quality of the predicted images, we compare the quality of the image predictors using structural similarity (SSIM) (Wang et al. 2004; Denton and Fergus 2018; Babaeizadeh et al. 2017). Table 1 compares image quality of 2000 generated images of the action-conditioned deterministic and stochastic models, showing both linear and convolutional state representations. Figures 9 and 10 allow for visual comparison of the different models, demonstrating that using the learned stochastic prior leads to clearer images. We note that the real-world lab dataset benefits more from the convolutional state representation since it has richer scene appearance and more inherent noise that needs to be abstracted into the state representation. On the other hand, the simulation data is much cleaner, sufficient to abstract with lower-dimensional representations. Our stochastic models perform superior to the deterministic models in both datasets.

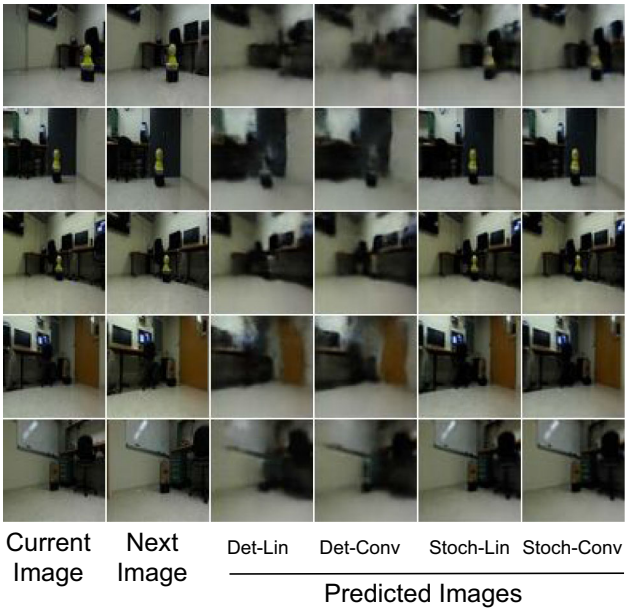


**Fig. 9** Sample predicted images from the lab dataset. From left to right: current image; true next image; deterministic linear; deterministic convolutional; stochastic linear; stochastic convolutional. High level description of action taken for each row starting from the top: turn right; move forward; move forward slightly; move forward and turn left; move forward and turn left

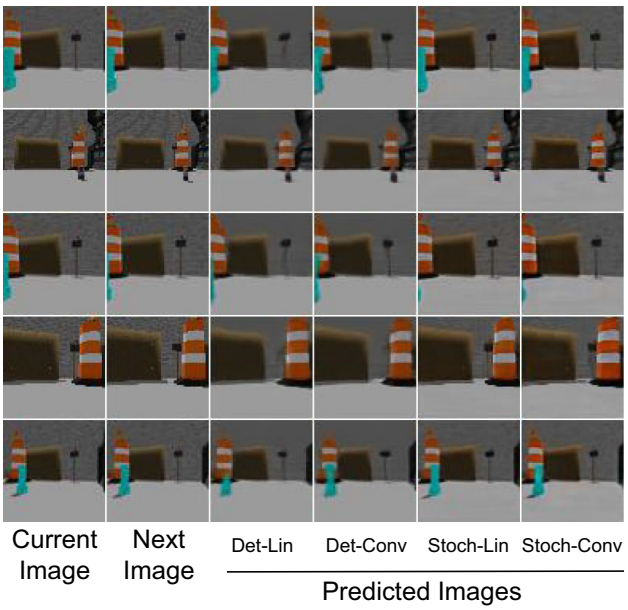


**Fig. 10** Sample predicted images from the simulation dataset. From left to right: current image; true next image; deterministic linear; deterministic convolutional; stochastic linear; stochastic convolutional. High level description of action taken for each row starting from the top: move forward and turn right; turn right slightly; turn right; move forward slightly; turn left slightly

**Table 2** Real-time robot target approach experiment results reporting the task success rate

| Model | Targ1 (%) | Targ2 (%) | Targ3 (%) | Mean (%) |
|---|---|---|---|---|
| Clone | 50 | 33 | 28 | 37 |
| Critic learning-hand | 67 | 61 | 50 | 59 |
| Forward consistency | 50 | 39 | 28 | 39 |
| *Future image similarity-Sect. 4.2* | | | | |
| Det linear state | 67 | 56 | 56 | 59 |
| Det conv state | 78 | 78 | 67 | 74 |
| Stoch linear state | 94 | 89 | 89 | 91 |
| Stoch conv state | 100 | 89 | 94 | 94 |

**Table 3** Simulated robot obstacle avoidance experiment results reporting the task success rate

| Model | Obstacle1 (%) | Obstacle2 (%) | Mean (%) |
|---|---|---|---|
| Clone | 41 | 48 | 44 |
| Critic learning-hand | 59 | 52 | 55 |
| Forward consistency | 37 | 41 | 39 |
| *Future image similarity-Sect. 4.2* | | | |
| Deterministic state | 44 | 37 | 41 |
| Stochastic state | 67 | 59 | 63 |

### 6.3 Robot Evaluations

We conducted a set of real-time experiments with a ground mobility robot in complex environments—real-world lab and simulation—to illustrate the implementation of the approaches. For each model in the lab environment, we ran 18 trials for each of three different target objects. For the simulation environment, we ran 27 trials for two different obstacle objects. For the action-conditioned models, each model was given 60 action candidates to choose from, distributed from $-30°$ to $+30°$ relative to the current angular pose of the robot. For each trial, we altered the target object location and the robot starting pose, with the constraint that the target object is within the initial field of view of the robot. We allowed the robot to take up to 30 steps to reach within 0.5 m of the target. We considered a trial successful if the robot captured an image of the target within the 0.5 m. If the robot did not reach the target within 30 steps or if the robot went out of bounds, such as hitting a wall or running into a table, we counted the trial as a failure.

In Tables 2 and 3, we show the results of the models we tested in the real-world lab and simulation environments, respectively. Our future similarity models performed superior to standard behavioral cloning and forward consistency (Pathak et al. 2018), demonstrating the advantage of our action-conditioned future prediction for imitation learning.

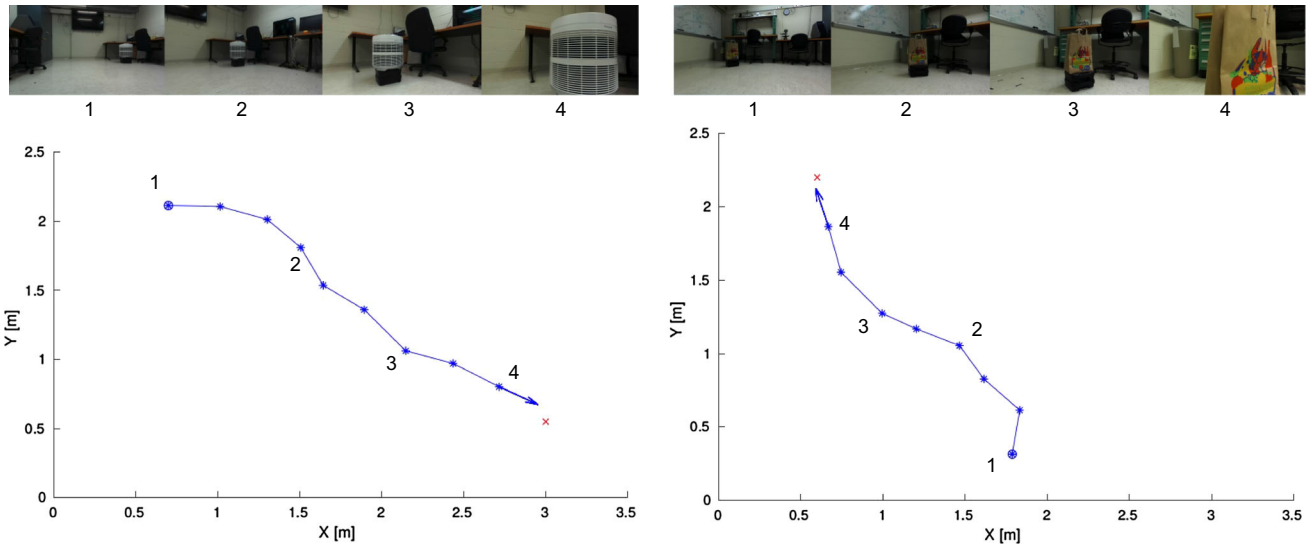


**Fig. 11** Examples of successful trajectories with the visible frames. The objective of the robot was to approach the air filter (left) and grocery bag (right) target objects

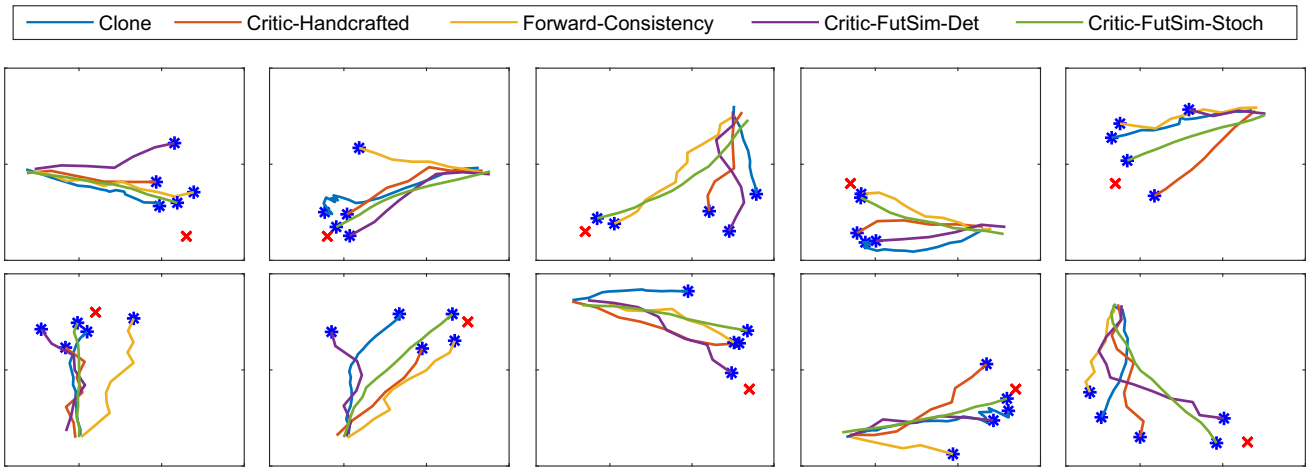


**Fig. 12** Example trajectories from the real-time robot experiments. The red '$X'$ marks the location of the target object and the blue *marks the end of each robot trajectory. Note that this is a challenging setting, since (1) we only provide a limited number of training examples, (2) it was done in a real-world environment with diverse objects, and (3) we did not provide any annotation of the target object

Our lab experiments indicate that retaining spatial information through convolutional state representation leads to better performance. The component that makes the greatest contribution for improved performance is the learned stochastic prior. Note that we only tested the linear state representation of our stochastic model in the lab environment since it already achieved a very high success rate with little margin left for improvement, which led us to omit testing the convolutional representation of our stochastic model. Since we found that the main contributor of improvement came from learning the stochastic prior, for our obstacle avoidance task in the simulation environment, we compare only future similarity models using the linear state representation as shown in Table 3.

Figure 11 illustrates the frames our robot obtains and processes during its task, and Fig. 12 explicitly compares the robot trajectories of 5 different models. Each model requires less than 30 ms to process a single image, measured using a single Titan X (Maxwell) GPU, which allows us run our experiments using the Jetson TX2 mounted on our mobile robot.

We also compared the trajectories of the various methods to the expert for 10 different robot-target placements in the lab environment. To measure the similarity between the expert and each of the methods, we use dynamic time warp-

**Table 4** Similarity scores of various models, comparing their trajectories to the expert. Lower is better

| Model | DTW score |
|---|---|
| Clone | 28.26 |
| Critic learning-hand | 9.97 |
| Forward consistency | 18.36 |
| FutureSim-det | 11.22 |
| FutureSim-stoch | 5.98 |

ing (DTW) also used in Vakanski et al. (2012) as a similarity score to assess how close the learned model performed the approach task compared to the expert. DTW is useful for measuring time series of different lengths such as our trajectories by finding the distance of a warp path $W$. If we compare two trajectories ($X_{1:M}$ and $Y_{1:N}$), their similarity using DTW is determined by finding the minimum-distance warp path: $Dist(W) = \sum_{k=1}^{K} Dist(w_{ki}, w_{kj})$, where $i$ is a point on path $X$, $j$ is a point on path $Y$, $k$ is a point on $W$, and $K = M * N$. $Dist(w_{ki}, w_{kj})$ is the distance between the two data point indices (one from $X$ and one from $Y$) in the $k$th element of the warp path (Salvador and Chan 2004). Table 4 shows that our method achieved the best similarity score.

To assess the stability of the various methods, we show multiple rollouts of several robot-target configurations in Fig. 13. We ran each method five times for each robot-target placement of two different target objects. Each configuration is overlaid by the expert trajectory in black. Qualitatively, we see that the stochastic model was the most stable.

Our method is robust to distractor objects. We conducted an experiment where we placed a distractor object in the scene about equidistant to the robot as the target object. We varied the target object, the distractor object, and their locations with respect to the robot. In Fig. 14 (top), we show examples of the target in the lab environment with and without distractors. Even without any labeling of the target or distractor objects, our model is able to ignore the distractor and navigate to the target. A similarity score of 4.94 was achieved for the trajectories shown in Fig. 14 (bottom) using the stochastic model, reflecting consistent small deviation from the expert.

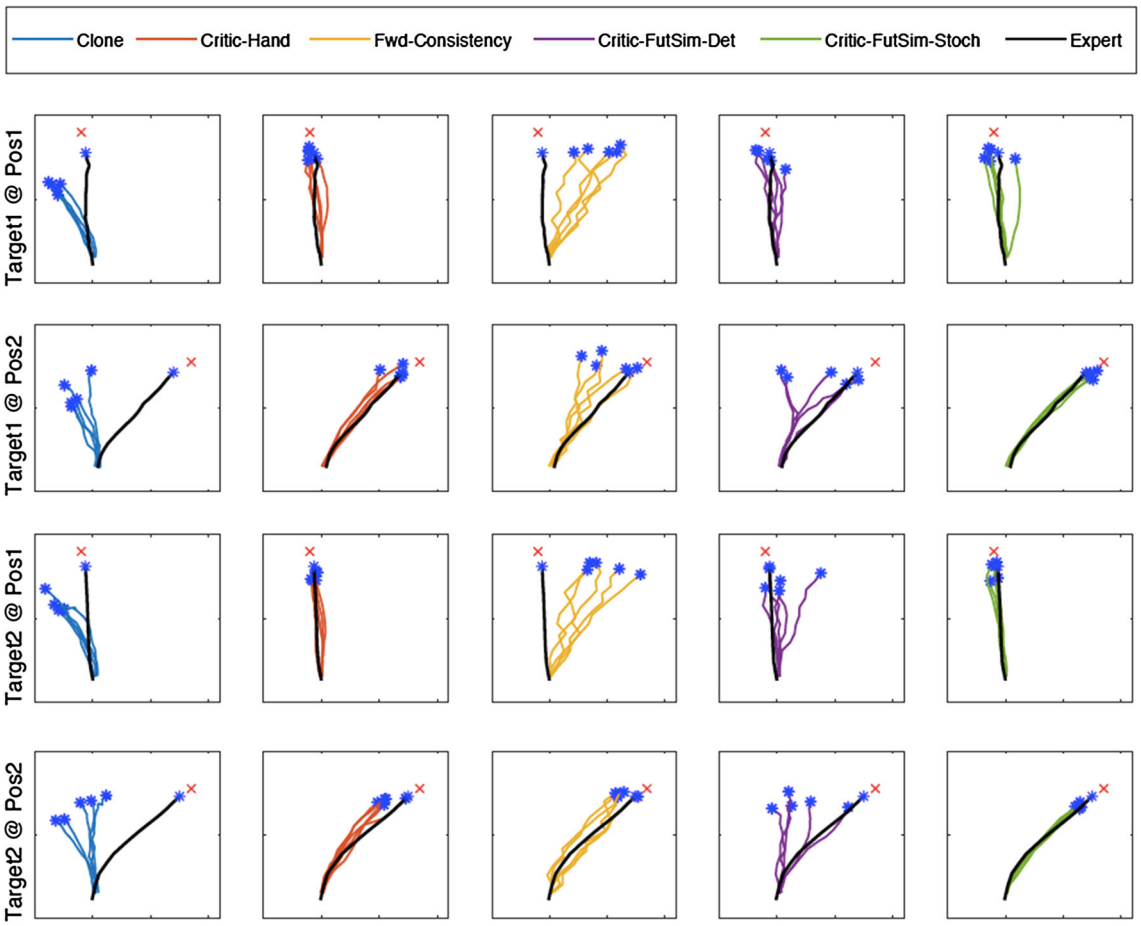


**Fig. 13** Runs showing the stability of the methods evaluated. Five runs of each method are shown in each box overlaid by the expert trajectory in black. Our method is very stable

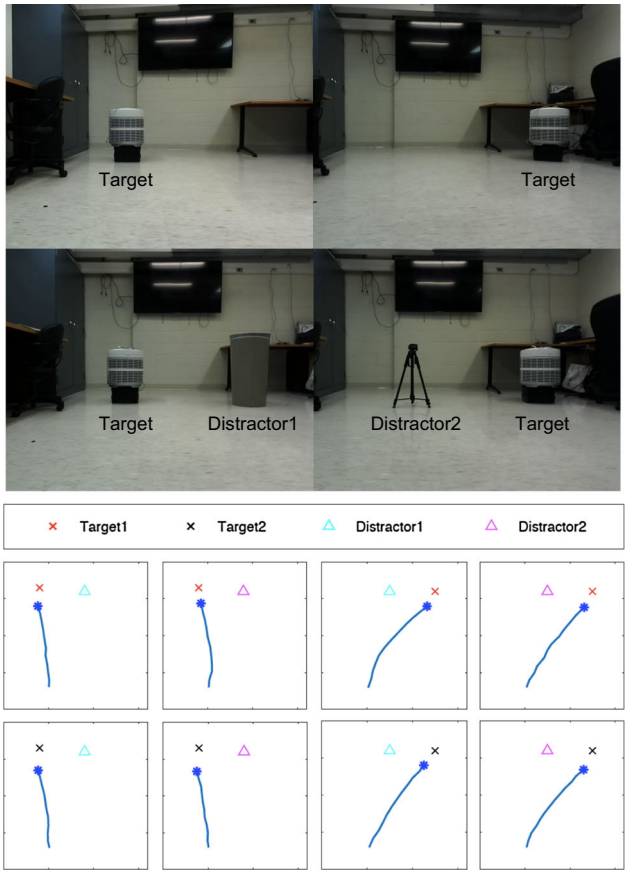


**Fig. 14** Trajectories with various distractors in place. Our method is immune to distractors, following the same trajectory each time. Top: Example frames without and with the distractor object, showing the distractor is quite large and visually similar to the target object. Bottom: Trajectories of the distractor test

## 7 Conclusion

We presented several CNN-based approaches for robot imitation learning. The goal was to make a real-time robot learn to execute a series of actions in real-world and simulation environments (e.g., an office) solely based on its visual input without any simulated or real-world trials. The learning was done without providing any explicit label showing what the target object looks like or allowing the robot to use other sensors or pose estimation techniques. Furthermore, it was done only based on a small set of example videos.

A key part of our action policy model is the future image predictor, which benefits from a learned prior to capture stochastic components of real-world videos and, in turn, generates clearer images. With higher quality images, we are able to learn a better value function to determine optimal actions. Our model achieved 2.5 × higher success rate in the real-life approach task than behavioral cloning and forward consistency models, and 1.5 × in the simulation obstacle avoidance task. Our model was also able to better imitate the expert, achieving trajectory similarity scores 3 × better than the forward consistency model and 4.5 × better than behavioral cloning.

We designed and experimentally compared multiple CNN models for imitation learning, including the standard CNN-based behavioral cloning method and the methods using action-conditioned convolutional future image prediction (i.e., a visual state transition model). We showed the superior performance of image based future similarity method.

## Appendix

### Implementation Details

We implemented the CNN models using the PyTorch library. The encoder/decoder networks followed the architecture of DCGAN (Radford et al. 2015), using their discriminator as our encoder CNN and their generator as our decoder CNN. Specifically, the encoder has 6 convolutional layers with a $3 \times 3$ kernel and stride of 2. The network layers have 64, 128, 256, 512, 512, 128 channels. Our input images are resized to $64 \times 64$, resulting in a feature map of size $128 \times 3 \times 3$. For the linear-representation model shown in Fig. 6a, we reshape this to be a vector of size $128 \cdot 3 \cdot 3$ then use a fully-connected layer to reduce the dimensionality to 4096. Our action network has two layers to increase the dimensionality to 64 then 256.

In the convolutional-representation model used in Fig. 6b, we leave the representation as-is. Our actions are 3-dimensional vectors for robot pose $(x, y, \theta)$, which are used as input to the action network. The action network has two layers that produces a 576-dimensional vector which we reshape to a spatial tensor of size $64 \times 3 \times 3$. We concatenate this tensor along the channel axis of the convolutional representation, which is then used as input to the decoder. The convolutional future prediction model contains 5 convolution layers with a $3 \times 3$ kernel and a stride of 1. The layers contain 256, 512, 512, 256, 128 channels.

Our decoder contains 6 deconvolutional layers for upsampling. All have a $3 \times 3$ kernel and a stride of 2. In the deconvolutional layer, a stride of 2 doubles in output size. The layers contain 512, 512, 256, 128, 64, 3 channels. The last layer is followed by a tanh activation function. All other layers in all networks were followed by batch normalization and used the LeakyReLU activation function with the negative slope set to 0.2. We minimize our loss function with gradient descent using the Kingma and Ba (2014) solver and learning rate set to 0.001.

The LSTMs are implemented similar to Denton and Fergus (2018). $LSTM_\phi$ and $LSTM_\psi$ are both single layer LSTMs with 256 cells in each layer. Each network has a linear embedding layer and a fully connected output layer. At inference, the output of $LSTM_\psi$ is concatenated to $z_I$

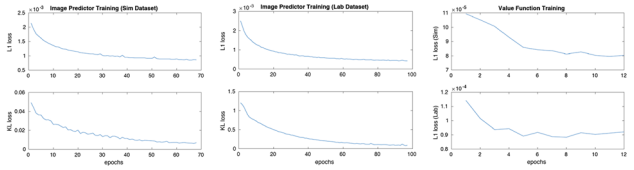


**Fig. 15** Training curves for the image predictor for the simulated and lab datasets with learning rate = 0.001 and batch size = 60, and the value function with learning rate = 5E−6

and $z_a$, and fed to the decoder. The output dimensionalities of the LSTM networks are g = 128 and $\mu_\phi = \mu_\psi = 64$.

## Training Information

Our training curves for the image predictor model and the critic are shown in Fig. 15. For the image predictor of both datasets, we set the learning rate = 0.001 and batch size = 60. The $\beta$ multiplier for the KL loss was set to 0.0001 in our experiments. The learning rate of the value function was set to 5E−6. The weights of the image predictor were held constant when training the value function.

## References


Abbeel, P., & Ng, A. Y. (2004) Apprenticeship learning via inverse reinforcement learning. In *International conference on machine learning (ICML)*.

Argall, B. D., Chernova, S., Veloso, M., & Browning, B. (2009). A survey of robot learning from demonstration. *Robotics and Autonomous Systems*, *31*, 469–483.

Babaeizadeh, M., Finn, C., Erhan, D., Campbell, R. H., & Levine, S. (2017). Stochastic variational video prediction. In *CoRR*. http://arxiv.org/abs/1710.11252.

Baram, N., Anschel, O., Caspi, I., & Mannor, S. (2017). End-to-end differentiable adversarial imitation learning. In *International conference on machine learning (ICML)* (pp. 390–399).

Bojarski, M., Del Testa, D., Dworakowski, D., Firner, B., Flepp, B., Goyal, P., Jackel, L. D., Monfort, M., Muller, U., Zhang, J., et al. (2016) End to end learning for self-driving cars. arXiv:1604.07316.

Carreira, J., & Zisserman, A. (2017). Quo vadis, action recognition? A new model and the kinetics dataset. In *IEEE conference on computer vision and pattern recognition (CVPR)*.

Chao, Y. W., Yang, J., Price, B., Cohen, S., & Deng, J. (2016). Forecasting human dynamics from static images. In: *IEEE conference on computer vision and pattern recognition (CVPR)*.

Chiappa, S., Racanière, S., Wierstra, D., & Mohamed, S. (2017). Recurrent environment simulators. In *CoRR*. http://arxiv.org/abs/1704.02254.

Denton, E., & Fergus, R. (2018). Stochastic video generation with a learned prior. In *CoRR*. arXiv:1802.07687.

Dosovitskiy, A., Springenberg, J. T., Tatarchenko, M., & Brox, T. (2017). Learning to generate chairs, tables and cars with convolutional networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *39*(4), 692–705.

Finn, C., & Levine, S. (2017). Deep visual foresight for planning robot motion. In *IEEE international conference on robotics and automation (ICRA)*. IEEE (pp. 2786–2793).

Finn, C., Goodfellow, I. J., & Levine, S. (2016). Unsupervised learning for physical interaction through video prediction. In *CoRR*. http://arxiv.org/abs/1605.07157

Finn, C., Levine, S., & Abbeel, P. (2016). Guided cost learning: Deep inverse optimal control via policy optimization. arXiv:1603.00448.

Giusti, A., Guzzi, J., Cireşan, D. C., He, F.-L., Rodríguez, J. P., Fontana, F., et al. (2016). A machine learning approach to visual perception of forest trails for mobile robots. *IEEE Robotics and Automation Letters*, *1*(2), 661–667.

Ho, J., & Ermon, S. (2016). Generative adversarial imitation learning. In *Advances in neural information processing systems (NIPS)*.

Ho, J., Gupta, J., & Ermon, S. (2016). Model-free imitation learning with policy optimization. arXiv:1605.08478.

Kingma, D. P., & Ba, J. (2014). Adam: A method for stochastic optimization. arXiv:1412.6980.

Laskey, M., Lee, J., Hsieh, W., Liaw, R., Mahler, J., Fox, R., & Goldberg, K. (2017). Iterative noise injection for scalable imitation learning. arXiv:1703.09327.

Lee, J., & Ryoo, M. S. (2017). Learning robot activities from first-person human videos using convolutional future regression. In *IEEE/RSJ international conference on intelligent robots and systems (IROS)*.

Levine, S., Pastor, P., Krizhevsky, A., & Quillen, D. (2016). Learning hand-eye coordination for robotic grasping with large-scale data collection. In *International symposium on experimental robotics* (pp. 173–184). Springer.

Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C. Y., & Berg, A. C. (2016). SSD: Single shot multibox detector. In *European conference on computer vision (ECCV)*.

Liu, Y., Gupta, A., Abbeel, P., & Levine, S. (2018). Imitation from observation: learning to imitate behaviors from raw video via context translation. arXiv:1707.03374.

Liu, Z., Yeh, R. A., Tang, X., Liu, Y., & Agarwala, A. (2017). Video frame synthesis using deep voxel flow. In *IEEE international conference on computer vision (ICCV)*.

Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., et al. (2015). Human-level control through deep reinforcement learning. *Nature*, *518*(7540), 529.

Ng, A. Y., & Jordan, M. I. (2000). Inverse reinforcement learning. In *International conference on machine learning (ICML)*.

Oh, J., Guo, X., Lee, H., Lewis, R. L., & Singh, S. (2015). Action-conditional video prediction using deep networks in atari games. In *CoRR*. arXiv:1507.08750.

Pathak, D., Mahmoudieh, P., Luo, G., Agrawal, P., Chen, D., Shentu, Y., Shelhamer, E., Malik, Y., Efros, A. A., & Darrell, T. (2018). Zero-shot visual imitation. arXiv:1804.08606.

Peng, X. B., Abbeel, P., Levine, S., & van de Panne, M. (2018). Deepmimic: Example-guided deep reinforcement learning of physics-based character skills. In *ACM SIGGRAPH*.

Piergiovanni, A. J., & Ryoo, M. S. (2018). Learning latent super-events to detect multiple activities in videos. In *IEEE conference on computer vision and pattern recognition (CVPR)*.

Pomerleau, D. A. (1989). Alvinn: An autonomous land vehicle in a neural network. In *Advances in neural information processing systems (NIPS)* (pp. 305–313).

Pomerleau, D. A. (1991). Efficient training of artificial neural networks for autonomous navigation. *Neural Computation*, *3*(1), 88–97.

Radford, A., Metz, L., & Chintala, S. (2015). Unsupervised representation learning with deep convolutional generative adversarial networks. arXiv:1511.06434.

Ross, S., Gordon, G., & Bagnell, D. (2011). A reduction of imitation learning and structured prediction to no-regret online learning. In *International conference on artificial intelligence and statistics* (pp. 627–635).

Sadeghi, F., Toshev, A., Jang, E., & Levine, S. (2017). Sim2real view invariant visual servoing by recurrent control. arXiv:1712.07642.

Salvador, S., & Chan, P. (2004). Fastdtw: Toward accurate dynamic time warping in linear time and space. *Intelligent Data Analysis*, *11*(5), 561–580.

Simonyan, K., & Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. arXiv:1409.1556.

Sutton, R. S., & Barto, A. G. (2018). *Reinforcement learning: An introduction*. Cambridge: MIT press.

Tatarchenko, M., Dosovitskiy, A., & Brox, T. (2016). Multi-view 3D models from single images with a convolutional network. In *European conference on computer vision (ECCV)*.

Torabi, F., Warnell, G., & Stone, P. (2018). Behavioral cloning from observation. arXiv:1805.01954.

Vakanski, A., Mantegh, I., Irish, A., & Janabi-Sharifi, F. (2012). Trajectory learning for robot programming by demonstration using hidden markov model and dynamic time warping. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, *42*(4), 1039–1052.

Vondrick, C., Pirsiavash, H., & Torralba, A. (2016). Anticipating visual representations from unlabeled video. In *IEEE conference on computer vision and pattern recognition (CVPR)* (pp. 98–106).

Walker, J., Gupta, A., & Hebert, M. (2014). Patch to the future: Unsupervised visual prediction. In *IEEE conference on computer vision and pattern recognition (CVPR)* (pp. 3302–3309).

Walker, J., Marino, K., Gupta, A., & Hebert, M. (2017). The pose knows: Video forecasting by generating pose futures. In *IEEE international conference on computer vision (ICCV)* (pp. 3352–3361).

Wang, Z., Bovik, A. C., Sheikh, H. R., & Simoncelli, E. P. (2004). Image quality assessment: From error visibility to structural similarity. *IEEE Transactions on Image Processing*, *13*(4), 600–612.

Wulfmeier, M., Ondruska, P., & Posner, I. (2015). Deep inverse reinforcement learning. arXiv:1507.04888.

Zhou, T., Tulsiani, S., Sun, W., Malik, J., & Efros, A. A. (2016). View synthesis by appearance flow. In *European conference on computer vision (ECCV)* (2016) (pp. 286–301).

Zhu, Y., Mottaghi, R., Kolve, E., Lim, J. J., Gupta, A., Fei-Fei, L., & Farhadi, A. (2016). Target-driven visual navigation in indoor scenes using deep reinforcement learning. arXiv:1609.05143.

Ziebart, B. D., Maas, A., Bagnell, J. A., & Dey, A. K. (2008). Maximum entropy inverse reinforcement learning. In *PAAAI conference on artificial intelligence (AAAI)*.