From Images to Task Planning: How NLP Can Help Physical Reasoning

Son Tung Nguyen*, Ozgur S. Oguz*†, Danny Driess*† and Marc Toussaint†‡
*Machine Learning & Robotics Lab, University of Stuttgart, †Max Planck Institute for Intelligent Systems
‡Learning and Intelligent Systems Lab, TU Berlin, Germany

***Abstract*—This paper integrates a multi-modal learning formulation into classical AI planning. For sequential manipulation tasks, planning with visual reasoning is challenging for autonomous agents. We propose to optimize an autoencoder not only on a regular image reconstruction but also jointly on a natural language processing (NLP) task. In essence, a *discrete*, *spatially meaningful* latent representation is obtained that enables effective autonomous planning for sequential decision-making problems only using visual sensory data. We integrate our method into a full planning framework and verify its feasibility on the classic blocks world domain [9]. Our experiments show that using auxiliary linguistic data leads to better representations, thus improves planning capability.**

I. Introduction

As autonomous agents are endowed with well-studied and fine-tuned perception as well as single manipulation skills, the next frontier emerges as how to combine these skills to realize tasks in more realistic settings. This necessitates a planning capability to take sequential decisions. A typical planning problem often involves an input state, a desired output state and an action model. A classical planning algorithm then treats this problem as a graph search, exploiting the action model to explore starting from an initial state. While graph search algorithms are usually efficient and fast, the action model is often the bottleneck in realizing a fully automated planning framework, especially when the observations are high-dimensional. In order to effectively determine transition rules, the action model requires an invariant, structured state representation of the input state. However, capturing these representations for scene images is a challenging task.

State representations, which can be continuous or discrete, play a critical role for planning systems. While continuous representations allow natural end-to-end training of state encoder and action models, thus enabling overall consistency between these components, goal state verification cannot be performed exactly. Methods using continuous representations [18, 13] often deploy a metric function to measure how close the current state is with respect to the goal state. This leads to error aggregation, where short-term inaccuracies accumulate into bigger long-term errors, thus is not particularly suitable for long-horizon tasks. On the other hand, discrete state representations facilitate trivial goal state verification, hence particularly useful for graph exploration. To achieve this, recent studies by Asai and Fukunaga [4], Asai [3] proposed to add an activation layer to enforce discrete representations. However, this makes it difficult when integrating the state encoder into an end-to-end framework or with an additional training objective.

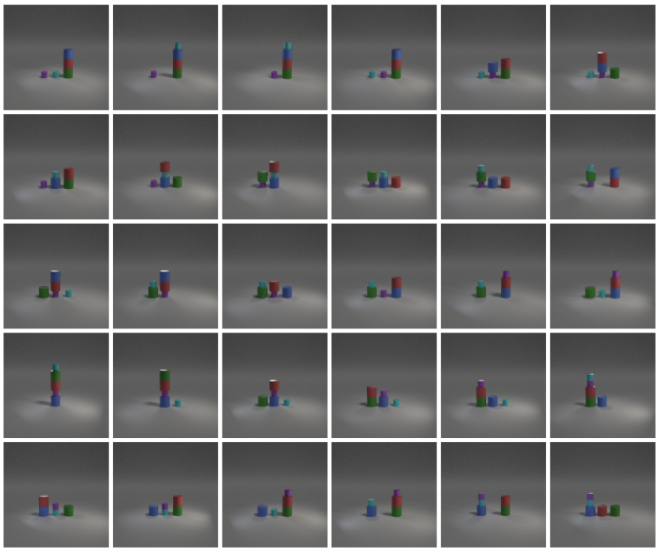

Fig. 1. Five samples of planning sequences (from *top* row to *bottom* row) using $\mathcal{M}_1$ where the *left-most* and *right-most* images are specified as initial and goal states, respectively.

We propose to use Vector-Quantized Variational Autoencoder (VQ-VAE) [21] as a state encoder. VQ-VAE maintains a dictionary of continuous embedding vectors, which acts as a database to pair a key (continuous latents) with a query (the input state). While the keys are differentiable, each of them is represented by a discrete ID. VQ-VAE is trained using likelihood and reconstruction loss functions. While likelihood function is necessary to keep the latent space similar to the image space, reconstruction loss optimizes to encode non-essential scene elements (for example: background or shadows). As a result, VQ-VAE becomes less robust when encoding semantically identical scenes.

To overcome this challenge, we propose to jointly train the state encoder with a visual-question answering (VQA) framework in a pre-training step. Under this setting, our goal is to encode image features which are important not only for reconstruction but also for question-answering. In particular, when the questions are about spatial relations, like those in the CLEVR dataset [15], the visual reasoning ability of the state encoder is supported.

We evaluated our state encoder in the classic blocks world problem [2, 9]. To evaluate the effectiveness of our encoder, we prepared two action models. The first action model is an oracular model that observes all the possible state transitions, then maps these transitions into a graph-like structure, similar to [4]. Although this model is impractical in real use cases, we intend to demonstrate that our learned representations are feasible for planning. This is confirmed by achieving 100% planning accuracy with the oracular model acting on our learned representations. The second action model is stochastic and observes only a part of the transitions. We evaluate how this model can infer the next state representation using the current representation. We are able to gain approximately 8.2% in accuracy by using the VQA framework, which confirms a higher quality of latent representation compared to using only the reconstruction loss.

The main contributions of this work are:

- We implement Vector-Quantized Variational Autoencoder (VQ-VAE) to obtain discrete state representations of *large* scene images which are feasible for classical planning.
- We integrate an auxiliary question-answering loss during a pre-training step to restrict the latent space, and further improve visual reasoning capability of the framework.

II. Related Work

Learning to act in latent space: Recent studies have integrated generative models and self-supervised learning into reinforcement learning formulations. A variational autoencoder (VAE) is used to encode raw image observations into a latent space where a policy is trained [10, 18]. During an iteration, this encoder can be trained before the policy [18], or jointly trained with the policy [10]. The latter case allowed the agent to encode relevant features for training but required a hand-designed reward function. In [18], the agent continuously samples and achieves a goal state by a simple distance reward function (in the latent space), thus facilitates the self-improvement of its general-purpose skills without a hand-designed reward. However, both methods lack an action model, hence can not sample the resulting state after applying an action, rendering them inapplicable to classical planning.

Symbolic planning: Our work is inspired by the use of symbolic planning [3, 7, 11, 13] for manipulation. Symbolic planning allows us to disentangle the input scene, thus reduces the amount of training data. However, symbolic planners do not work out of the box with continuous state representations. Huang et al. [13] overcame this challenge by relaxing the discrete requirement of symbolic planners. This method applies only the actions with the highest chance of satisfying preconditions. Since these actions are not necessarily the optimal actions, the resulting plan is suboptimal. On the other hand, there is a different line of research which tried to obtain discrete state representations to be compatible with existing classical planning. Asai and Fukunaga [4] added a Gumbel-softmax [14, 17] layer to enforce discrete outputs. While being plannable, these representations are not ideal to be integrated in any end-to-end framework, thus making them difficult to train with any additional objectives.

Inductive prior: Annotated data is often expensive and not always available. One of the most commonly proposed solutions is unsupervised learning of a pre-trained representation. The main idea is trying to make a good use of largely available unannotated datasets (images [12], natural languages [1, 6], offline exploratory data [22]), obtaining inductive priors which empower pre-trained models. In Andreas et al. [1], which is a close work to ours, the training phase is divided into two phases. In *pre-trained phase* both a sampler of missing language data in the downstream task and a shared parameter space are learned, and in *concept-learning phase* the parameter space is fine-tuned on the downstream task. During the concept-learning phase, the sampler is used to sample any absent parameter in the downstream dataset. The main difference between Andreas et al. [1] and our work is that we merge pre-trained and concept-learning phases into a single step. Therefore, our framework is unified and naturally solves the problem of absent parameters in the downstream task.

III. Problem definition

We focus on solving planning problems for sequential robotics manipulation tasks autonomously. Our objective is to achieve physical reasoning by only visual perception and without human-designed logical rules for high-level action selection. The goal is to find an action sequence $a_{1:N}$, given an image of the scene x_0, and a goal state description v_{goal}, e.g., *red box is above yellow box*,

$$\begin{aligned} \underset{a_{1:N}}{\arg\min} \sum_{i=1}^{N} \mathcal{H}(z_{i-1}, a_i) \\ \text{s.t. } z_0 = \mathcal{G}(x_0), \quad z_N = \mathcal{G}(\mathcal{F}(v_{\text{goal}})), \\ z_{i+1} = \mathcal{M}(z_i, a_i), \quad a_i \in \mathbf{A} \end{aligned} \tag{1}$$

where $\mathcal{H}$ defines the optimality criterion, z_0, z_N is the initial and final latent states encoded by $\mathcal{G}$, $\mathcal{M}$ is an action model, $\mathcal{F}$ is a mapping from the text space to the image space, and $\mathbf{A}$ is a fixed set of actions (Fig. 5). Note that this formulation is generalizable to whether the goal state is in the image or the text space. We aim for this generalization since textual (or verbal) goal state v_{goal} is reasonable in real-world settings. We do not focus on deriving such $\mathcal{F}$ in this work, instead we relax this constraint and use x_{goal}, i.e., $\mathcal{G}(\mathcal{F}(v_{\text{goal}})) \simeq \mathcal{G}(x_{\text{goal}})$.

IV. Methodology

First, we describe how to realize $\mathcal{G}$ in section A. Then, we describe two possible action models $\mathcal{M}_1$ and $\mathcal{M}_2$. These action models are used to integrate $\mathcal{G}$ into a planning framework to solve for equation (1).

A. State encoder

To realize $\mathcal{G}$, we propose to use a Vector Quantized - Variational AutoEncoder (VQ-VAE) [21] to directly encode raw images into propositions under a multi-modal training scheme. The latent vector z in VQ-VAE is an ordered set of L

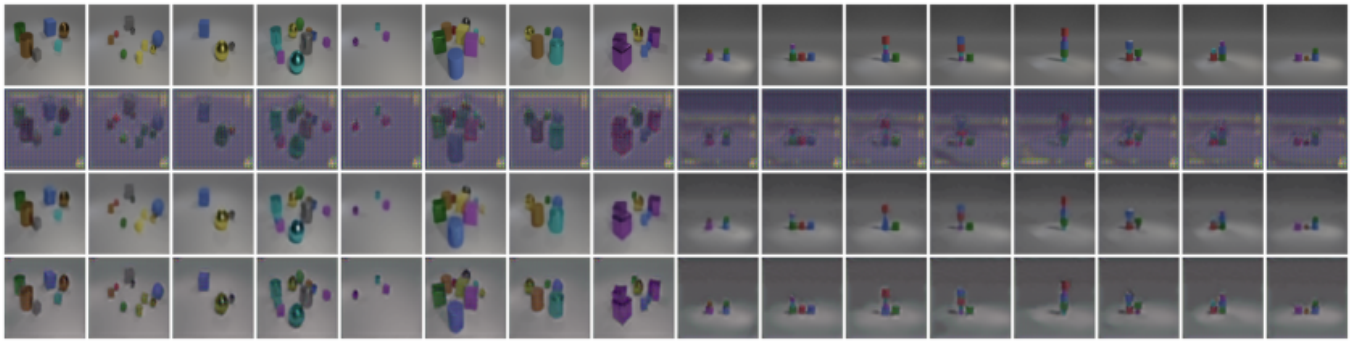

Fig. 2. Qualitative results of our reconstructed images when trained under three schemes. From top to bottom: *first row:* real images, *second row:* reconstructed images using only D_{KL} loss, *third row:* reconstructed images using only $\mathcal{L}_{\text{rec}}$ loss, *fourth row:* reconstructed images using both losses.

differentiable, pre-defined, continuous embedding vectors e_i, i.e., $z = \{e_i \mid e_i \in \mathbf{R}^{K \times D}, 1 \leq i \leq K\}$, $|z| = L$, where e_i are D-dimensional selected from the embedding space of size K and $L < K$. Observe that z can be uniquely represented by indices of embedding vectors e_i in the embedding space, therefore facilitating trivial goal state verification while keeping the stochastic nature as e_i are continuous. This is crucial for our training settings where z is jointly optimized with an auxiliary linguistic task.

In order to make use of linguistic data, we propose to jointly train $\mathcal{G}$ (which is a VQ-VAE) in a visual-question answering (VQA) framework (Fig. 6). Given an image x, a question $w_{1:T}$ which is a sequence of T words w_i, the VQA framework returns an answer by forming a distribution over a fixed set of answer $\mathbf{Y}$, i.e., $p(y_j|\cdot)$ where $1 \leq j \leq |\mathbf{Y}|$. Under this setting, our goal is to encode image features which are important not only for reconstruction but also for question-answering. Hence, when the questions are about spatial relations, like those in the CLEVR dataset [15], the autoencoder focuses more on the semantically essential scene elements.

To process the textual data, we use a relation network [19], $p(y_j|\cdot) = RN(x, w_{1:T}) = RN_x$ ($w_{1:T}$ omitted for brevity). Our full training objective becomes:

$$\mathcal{L}_{\mathcal{G}} = \alpha \mathcal{L}_{\text{rec}}(x_{\text{rec}}, x) + \beta D_{\text{KL}}(RN_{x_{\text{rec}}} \parallel RN_x) + \mathcal{L}_{\text{lh}} \quad (2)$$

where $\mathcal{L}_{\text{rec}}$ is the mean squared error loss, D_{KL} is the Kullback–Leibler divergence loss, and $\mathcal{L}_{\text{lh}}$ is the log-likelihood loss originally presented in [21]. We keep the log-likelihood loss in all of our experiments.

B. Action model

In order to solve for equation (1), an action model $\mathcal{M}$ is missing. We evaluate $\mathcal{G}$ with two action models.

1) Oracular action model: The oracular action model $\mathcal{M}_1$ observes all the possible state transitions in the state space, therefore $\tilde{z}_{i+1} = \mathcal{M}_1(z_i, a_i)$ is exactly equal to z_{i+1} where z_{i+1} and $\tilde{z}_{i+1}$ are the ground-truth and predicted resulting state of applying an action a_i to a state z_i.

2) Stochastic action model: Our stochastic action model $\mathcal{M}_2$, which is inspired by Asai and Fukunaga [4], has two components: *action policy* π and *action discriminator* $\mathcal{D}$.

- The π network is trained on transition pairs $(\mathcal{G}(x_i), a_i, \mathcal{G}(x_{i+1}))$ where a_i is the action which transitions the scene image x_i to x_{i+1}. π samples the result state after applying an action to one state, i.e., $\tilde{z}_{i+1} = \pi(z_i, a_i)$ and $z_i = \mathcal{G}(x_i)$. To train π, we use a contrastive learning objective function similar to [16]:

$$\mathcal{L}_{\pi} = d(\tilde{z}_{i+1}, z_{i+1}) + \max(0, \gamma - d(\tilde{z}_{i+1}, z^{\text{neg}}_{i+1})) \quad (3)$$

  where d is the squared difference, γ is a hyper parameter ($\gamma = 1$ during our experiments [16]), and z^{neg}_{i+1} are negative examples, sampled from the same training batch.
- The $\mathcal{D}$ network decides which action is applicable given a state, i.e., $\mathcal{D}(z_i) = \{u_j \mid 1 \leq j \leq |\mathbf{A}|\}, u_j \in \{0, 1\}\}$ where $u_j = 1$ if a_j is applicable to state z_i. To train $\mathcal{D}$, we use binary cross-entropy (BCE) loss in a multi-label classification manner:

$$\mathcal{L}_{\mathcal{D}} = \text{BCE}(\mathcal{D}(z_i), \mathbf{U}) \quad (4)$$

  where $\mathbf{U} = \{u_j\}$.

We found that the $\mathcal{D}$ component often performs much better than the π component (apprx. 94% vs. 65% accuracy, respectively) on the test set. Since the accuracy of π appears to be the bottleneck of $\mathcal{M}_2$, our analysis focuses on this component.

V. Evaluation

We evaluate our method based on three criterion: *reasoning*, *planning*, and *efficiency*. Our experiments are based on the following setting: (1) we train the state encoder $\mathcal{G}$ using the CLEVR dataset [15], (2) we use $\mathcal{G}$ to encode the scene images of the photorealistic Blocks world dataset [2] to evaluate our overall planning framework.

Reasoning: We analyze our VQA framework with three different objective functions (Table I). We trained the framework with CLEVR dataset [15] for 150 epochs over the full training set. The training set consists of 70 000 images and 699 989 questions. With the D_{KL} objective, our state encoder reasoning shows better performance than using only $\mathcal{L}_{\text{rec}}$ on the held-out

TABLE I
REASONING AND RECONSTRUCTION PERFORMANCE

Loss function	QA accuracy	Reconstruction error
KL-Divergence only	**0.932923**	0.118789
Reconstruction only	0.676227	**0.000464**
Both	0.931829	0.002303

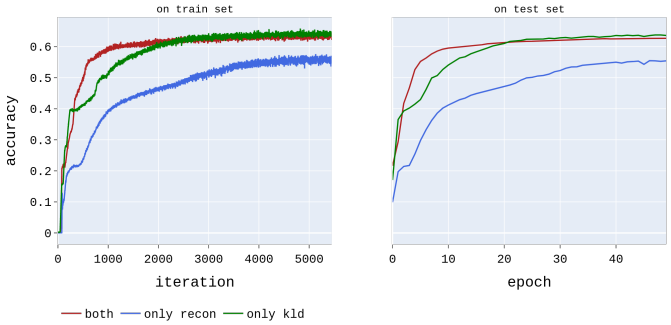

Fig. 3. Accuracy of the stochastic action model overtime on the training (left) and test (right) sets using representations acquired by only D_{KL} loss (green line), $\mathcal{L}_{\text{rec}}$ loss (blue line), and both (red line).

test set. Figure 2 shows reconstructed images in each case. We observe that when using only D_{KL} loss, the framework tends to neglect unnecessary details (e.g. background, shadows) as reflected by its larger reconstruction error but with a high QA accuracy.

Planning with $\mathcal{M}_1$: Following Asai and Fukunaga [4], we show that our learned representation is compatible with planning algorithms, for example, with a breadth-first search graph exploration. Note that one can achieve a planning accuracy of 100% with $\mathcal{M}_1$ on one condition: each state is *uniquely* encoded. With VQ-VAE, we fulfill this condition, thus achieve the absolute planning accuracy using $\mathcal{M}_1$ (Fig. 1 and Fig. 9). Similar to [5], we challenged this *uniqueness* under noisy settings. Each scene image is perturbed with noise sampled from a standard normal distribution (Fig. 8) $\mathcal{N}(0,1)$, i.e., $\tilde{x}_i = x_i + \alpha u$ where α is a noise magnitude and $u \sim \mathcal{N}(0,1)$. Our latent representations are less affected by noise, as shown by smaller deviation between $\mathcal{G}(x_i)$ and $\mathcal{G}(\tilde{x}_i)$ (Fig. 7), when trained with D_{KL} loss. However, the effects are magnified under heavier noise ($\alpha > 0.01$).

Planning with $\mathcal{M}_2$: We evaluate our learned representations in a stochastic manner with another action model, $\mathcal{M}_2$. We first encode all the transitions in the problem domain of 5 objects and 3 stacks and use these transitions to train $\mathcal{M}_2$. After training, we evaluate $\mathcal{M}_2$ using the transitions in the problem domain of 4 objects and 3 stacks. We report the accuracy by comparing the predicted latent $\tilde{z}_{\text{to}} = \mathcal{M}_2(z_{\text{from}}, a)$ to the ground-truth latent z_{to}, i.e., accuracy $= \frac{1}{M}\frac{1}{N}\sum\sum_{i=1}^{N} u_i$ where M is the size of the test set, z_m^n is the n-th bit of the latent z_m ($1 \leq n \leq N$), and $u_i = 1$ if $\tilde{z}_{\text{to}}^i = z_{\text{to}}^i$ and 0 otherwise. For this action model $\mathcal{M}_2$, our experiment shows that the latent representation trained with D_{KL} objective enables easier learning of transition rule even when $\mathcal{G}$ does not observe the action data [2] (Fig. 3).

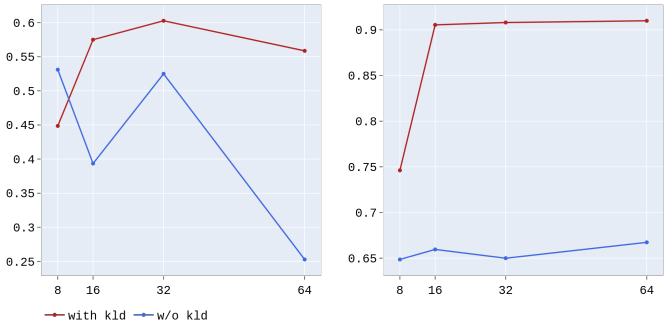

Fig. 4. Planning (*left*) and QA (*right*) accuracies using different latent sizes.

Efficiency: We investigated the effect of the latent vector dimensionality, i.e., $\dim(\mathcal{G}(x)) = (o, 32, 32)$ where $o \in \{8, 16, 32, 64\}$. We kept $o = 64$ for all the previous experiments. Due to computational time constraints, we used a subset of the CLEVR dataset [15] consisting of only 50 000 questions. We train our VQA framework using this smaller version under two settings: optimizing *only* $\mathcal{L}_{rec}$ *loss* and *both* D_{KL} *and* $\mathcal{L}_{rec}$ *loss*. With D_{KL} loss, our state encoder achieves better QA and planning accuracy even for smaller latent sizes (Fig. 4).

VI. CONCLUSION

We introduced a novel method to train a state encoder and integrated it into a classical planner. We are able to obtain latent representations that are higher quality in terms of visual reasoning, and thus feasible for planning. We also presented a full *symbolic* planning framework to readily unify with the state encoder. Our experiments demonstrate better performance of the whole system when pre-trained with the proposed multi-modal scheme.

The limitation of our framework lies at the π component of the action model. While doing sufficiently well in our toy experiment [2], π has to perform better to be effective under more complex settings. Another aspect, which renders our framework semi-automatic, is that a set of actions $\mathbf{A}$ must be defined explicitly. A fully autonomous agent has to learn applicable actions and their effects itself. This work takes the first steps towards achieving this goal.

While classical AI planners are efficient and optimal, high-dimensional state representations, such as visual data, hinders their applicability on real-world robotic manipulation problems [8]. Our work hopefully sheds light on how to obtain such representations by taking advantage of auxiliary (linguistic) data. The presented planner is ready to realize high-level plans, and can be integrated into a motion planner (e.g., [20]) for robotic sequential manipulation tasks.

REFERENCES

[1] Jacob Andreas, Dan Klein, and Sergey Levine. Learning with latent language. In *NAACL-HLT*, pages 2166–2179. Association for Computational Linguistics, 2018.

[2] Masataro Asai. Photo-Realistic Blocksworld Dataset. *arXiv preprint arXiv:1812.01818*, 2018.

[3] Masataro Asai. Unsupervised grounding of plannable first-order logic representation from images. In *ICAPS*, pages 583–591. AAAI Press, 2019.

[4] Masataro Asai and Alex Fukunaga. Classical planning in deep latent space: Bridging the subsymbolic-symbolic boundary. In *AAAI*, pages 6094–6101. AAAI Press, 2018.

[5] Masataro Asai and Hiroshi Kajino. Towards stable symbol grounding with zero-suppressed state autoencoder. In *ICAPS*, pages 592–600. AAAI Press, 2019.

[6] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT (1)*, pages 4171–4186. Association for Computational Linguistics, 2019.

[7] Danny Driess, Ozgur S. Oguz, and Marc Toussaint. Hierarchical task and motion planning using logic-geometric programming (HLGP). *RSS Workshop on Robust TAMP*, 2019.

[8] Danny Driess, Ozgur S. Oguz, Jung-Su Ha, and Marc Toussaint. Deep visual heuristics: Learning feasibility of mixed-integer programs for manipulation planning. In *Proc. of the International Conference on Robotics and Automation (ICRA)*, 2020.

[9] Naresh Gupta and Dana S. Nau. On the complexity of blocks-world planning. *Artif. Intell.*, 56(2-3):223–254, 1992.

[10] David Ha and Jürgen Schmidhuber. Recurrent world models facilitate policy evolution. In *NeurIPS*, pages 2455–2467, 2018.

[11] Valentin N. Hartmann, Ozgur S. Oguz, Danny Driess, Marc Toussaint, and Achim Menges. Robust task and motion planning for long-horizon architectural construction planning. URL http://arxiv.org/abs/2003.07754.

[12] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross B. Girshick. Momentum contrast for unsupervised visual representation learning. *CoRR*, abs/1911.05722, 2019.

[13] De-An Huang, Danfei Xu, Yuke Zhu, Animesh Garg, Silvio Savarese, Li Fei-Fei, and Juan Carlos Niebles. Continuous relaxation of symbolic planner for one-shot imitation learning. In *IROS*, pages 2635–2642. IEEE, 2019.

[14] Eric Jang, Shixiang Gu, and Ben Poole. Categorical reparameterization with gumbel-softmax. In *ICLR (Poster)*. OpenReview.net, 2017.

[15] Justin Johnson, Bharath Hariharan, Laurens van der Maaten, Li Fei-Fei, C. Lawrence Zitnick, and Ross B. Girshick. CLEVR: A diagnostic dataset for compositional language and elementary visual reasoning. In *CVPR*, pages 1988–1997. IEEE Computer Society, 2017.

[16] Thomas N. Kipf, Elise van der Pol, and Max Welling. Contrastive learning of structured world models. In *ICLR*. OpenReview.net, 2020.

[17] Chris J. Maddison, Andriy Mnih, and Yee Whye Teh. The concrete distribution: A continuous relaxation of discrete random variables. In *ICLR (Poster)*. OpenReview.net, 2017.

[18] Ashvin Nair, Vitchyr Pong, Murtaza Dalal, Shikhar Bahl, Steven Lin, and Sergey Levine. Visual reinforcement learning with imagined goals. In *NeurIPS*, pages 9209–9220, 2018.

[19] Adam Santoro, David Raposo, David G. T. Barrett, Mateusz Malinowski, Razvan Pascanu, Peter W. Battaglia, and Tim Lillicrap. A simple neural network module for relational reasoning. In *NIPS*, pages 4967–4976, 2017.

[20] Marc Toussaint, Kelsey R. Allen, Kevin A. Smith, and Joshua B. Tenenbaum. Differentiable physics and stable modes for tool-use and manipulation planning. In *Robotics: Science and Systems*, 2018.

[21] Aäron van den Oord, Oriol Vinyals, and Koray Kavukcuoglu. Neural discrete representation learning. In *NIPS*, pages 6306–6315, 2017.

[22] Ge Yang, Amy Zhang, Ari S. Morcos, Joelle Pineau, Pieter Abbeel, and Roberto Calandra. Plan2vec: Unsupervised representation learning by latent plans. *CoRR*, abs/2005.03648, 2020.

APPENDIX

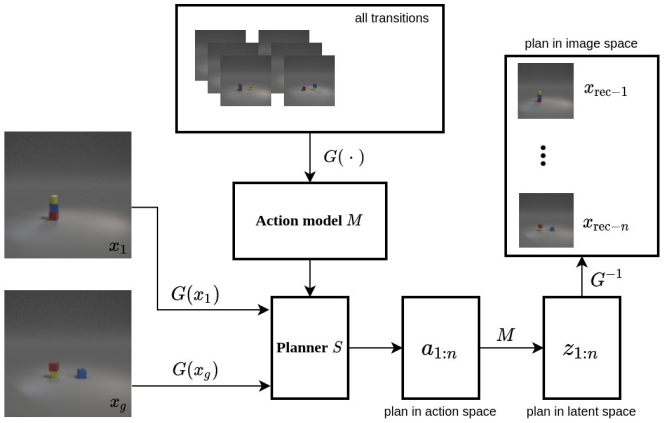

Fig. 5. An overview of our planning framework as formalized in Eqn. 1.

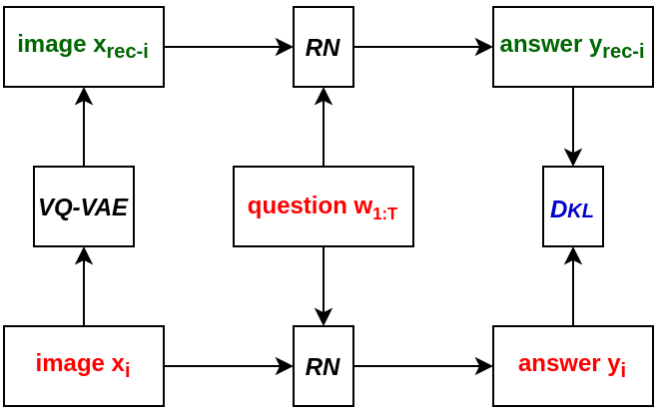

Fig. 6. An overview of a forward pass through our VQA framework from inputs (depicted in red color) to outputs (depicted in green color).

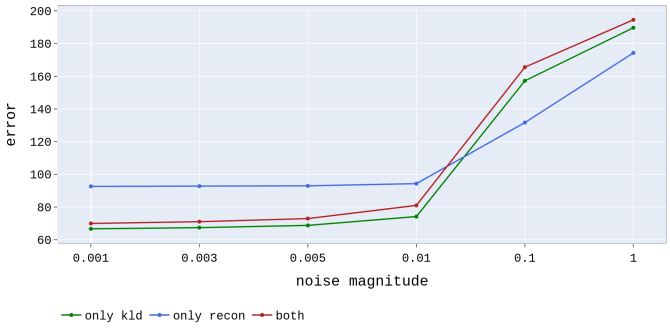

Fig. 7. Absolute difference of latent representations when encoded under *noisy* versus *normal* setting.

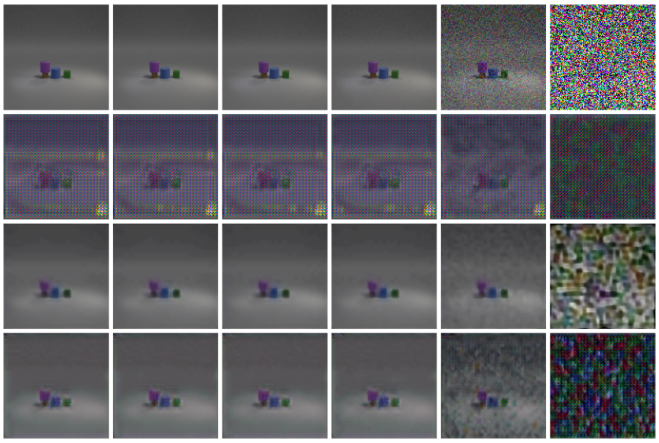

Fig. 8. Reconstruction quality under different magnitudes of addition noise. From *left* to *right*: six noise levels: 0.001, 0.003, 0.005, 0.01, 0.1, and 1. From *top* to *bottom*: real images, reconstructed images using only D_{KL} loss, using only $\mathcal{L}_{\text{rec}}$ loss, and both.

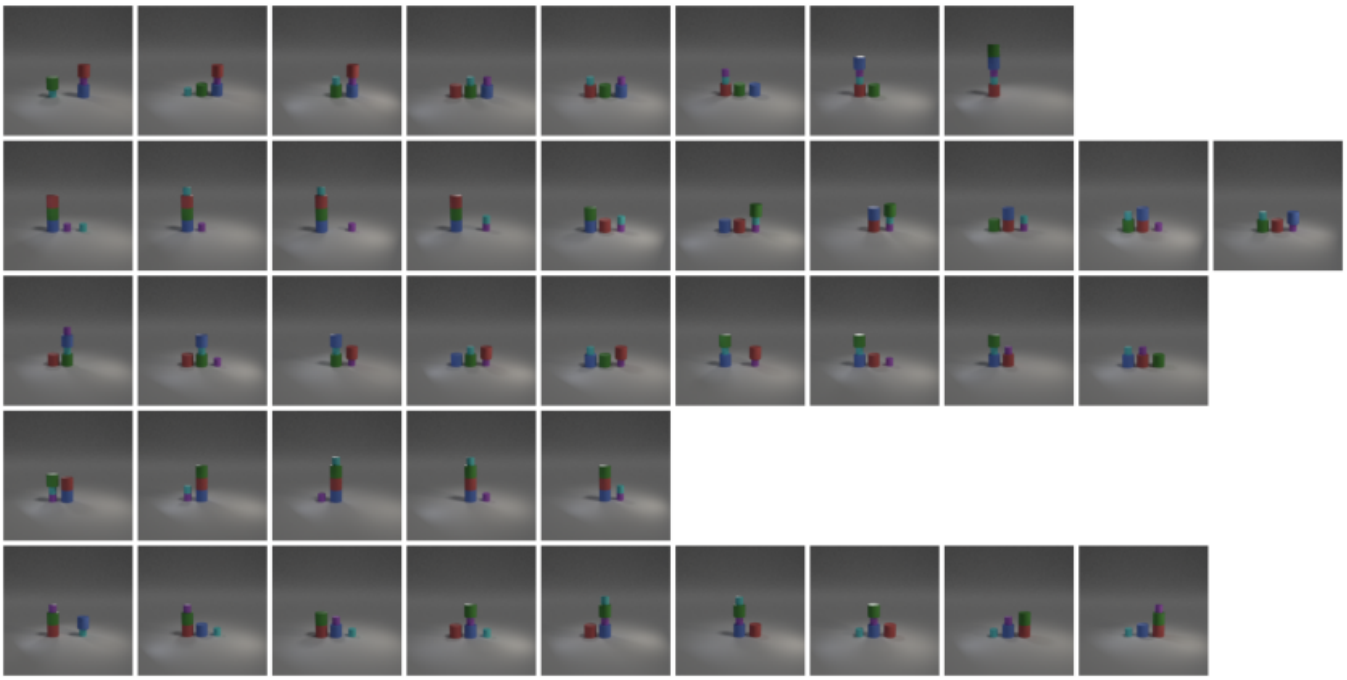

Fig. 9. Some more plan sequences using $\mathcal{M}_1$.