LEARNING ROBOTIC MANIPULATION THROUGH VISUAL PLANNING AND ACTING

Anonymous authors
Paper under double-blind review

ABSTRACT

Planning for robotic manipulation requires reasoning about the changes a robot can affect on objects. When such interactions can be modelled analytically, as in domains with rigid objects, efficient planning algorithms exist. However, in both domestic and industrial domains, the objects of interest can be soft, or deformable, and hard to model analytically. For such cases, we posit that a data-driven modelling approach is more suitable. In recent years, progress in deep generative models has produced methods that learn to 'imagine' plausible images from data. Building on the recent Causal InfoGAN generative model, in this work we learn to imagine goal-directed object manipulation directly from raw image data of self-supervised interaction of the robot with the object. After learning, given a goal observation of the system, our model can generate an imagined plan – a sequence of images that transition the object into the desired goal. To execute the plan, we use it as a reference trajectory to track with a visual servoing controller, which we also learn from the data as an inverse dynamics model. In a simulated manipulation task, we show that separating the problem into visual planning and visual tracking control is more sample efficient and more interpretable than alternative data-driven approaches. We further demonstrate preliminary robot results on learning to imagine and execute deformable rope manipulation.

1 INTRODUCTION

The main difficulty in planning the manipulation of deformable objects is that, in contrast with rigid objects, there is no obvious mapping from an observation of the object to a compact representation in which planning can be performed. Thus, traditional task and motion planning approaches, which require manual design of the predicates, preconditions, and effects in the problem, are difficult to apply (McConachie et al. (2017); Srivastava et al. (2014)). In recent years, several studies have proposed a data-driven, self-supervised paradigm for robotic manipulation (Agrawal et al. (2016); Nair et al. (2017); Finn & Levine (2017)). In this approach, the robot 'plays' with the object using some random manipulation policy (e.g., randomly grasping or poking an object), and collects perceptual data about the interactions with the object. Later, machine learning is used to train a policy that performs the task directly from the perceptual inputs. By relying directly on data, these approaches overcome the modelling challenges of classical planning approaches, and scale to handle high-dimensional perceptual inputs such as raw images.

In this work we ask – can we learn to automatically *generate the visual plan and follow it* in a data-driven way? Concretely, given the current image of the system and some desired goal observation, we propose to generate a sequence of images that manipulate the object to the desired configuration, *without* any human guidance, and then use this plan in conjunction with an inverse model for actually manipulating the object.

However, learning visual planning directly from raw image data has so far been limited to very simple planning tasks, such as reaching or pushing rigid objects (Finn & Levine (2017); Ebert et al. (2017)). In this work, we take a step towards learning complex visual planning for robotic manipulation, by learning features that *are compatible* with a strong planning algorithm. At the basis of our approach is the recent NeurIPS 2018 work, the Causal InfoGAN (CIGAN) model from Kurutach et al. (2018). In CIGAN, a deep generative model is trained to predict the possible next states of the object, with a constraint that linear trajectories in the latent state of the model produce feasible observation sequences. Kurutach et al. (2018) used a CIGAN model for planning goal-directed tra-

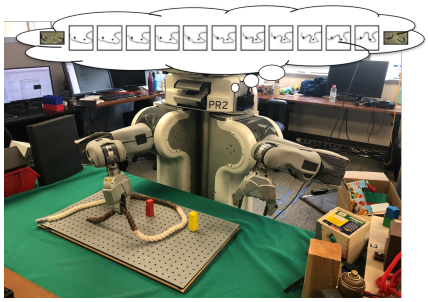

Figure 1: Visual planning and acting for rope manipulation. The PR2 robot first collects data through self-supervised random rope manipulation, and learns from this a generative model for possible visual transformations of the rope. Then, given a goal observation for the rope, we *plan* a visual trajectory of a possible manipulation sequence that reaches the goal (shown on top). Finally, visual servoing is used to execute the imagined plan.

jectories simply by linearly interpolating in the latent space, and then mapping the latent trajectory to observations for generating the visual plan.

Building on CIGAN, we propose a method for *visual planning and acting* (VPA), where sensory data obtained from self-supervised interaction is used to learn both a CIGAN model for visual planning and an inverse model for tracking a visual plan. After learning, given a goal observation for the system, we first use CIGAN to imagine a sequence of images that transition the system from its current configuration towards the goal. Then, we use the imagined trajectory as a reference for tracking using the inverse model.

We show that separating the control task into visual planning and visual tracking leads to an interpretable decision making paradigm, which is also more sample efficient than data-driven methods which learn actions directly from images.

2 Visual Planning and Acting

Our approach is model-based, where we first use the data $\mathcal{D}$ to learn both a CIGAN model M_{CIGAN} and an inverse dynamics model M_{IM}. For any two start and goal observations o_{start}, o_{goal}, the CIGAN model M_{CIGAN} can generate a visual plan that transitions the system from start to goal, $o_{start}, o_1, \ldots, o_k, o_{goal}$. Since the CIGAN model is trained to generate feasible pairs of observations, we are guaranteed that the plan generated by a well-trained CIGAN model will be feasible, in the sense that the robot can actually execute it.

Our Visual Planning and Acting (VPA) method for solving the goal directed planning problem is a combination of planning and replanning using the CIGAN model M_{CIGAN}, and trajectory tracking using the inverse model M_{IM}. The VPA algorithm is given as follows:

1. **Plan:** as in Kurutach et al. (2018), given a pair, o_{start}, o_{goal}, use the CIGAN model M_{CIGAN} to generate a planned sequence of observations $o_{start}, o_1, ..., o_m, o_{goal}$.
2. **Act:** If the length of the plan m is zero, take an action u to reach the goal $u = M_{\text{IM}}(o_{start}, o_{goal})$, then stop. Else:
3. Take an action u to reach the first observation in the plan $u = M_{\text{IM}}(o_{start}, o_1)$ and take a new observation of the current system state o_{new}.
4. **Replan:** update o_{start} to be the current observation o_{new}, and go back to step 1.

The only data required is images taken from self-supervised manipulation of an object. Nevertheless, our method enjoys the **interpretability** of model based methods – at every step of our algorithm we have a visual plan of the proposed manipulation. We found that this allows us to reliably evaluate the performance of VPA before performing any robot experiment, significantly reducing robot-time and effort.

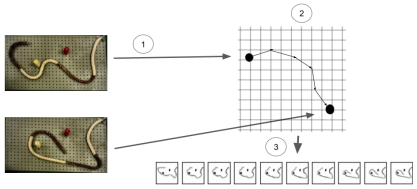

Figure 2: Illustration of how the CIGAN model plan step works. First, start and goal images are encoded to their latent representations (denoted here as points in the plane). Second, search is used to find a sequence of points in the latent space that connect the start to the goal, while obeying the latent space dynamics. Here we illustrate the result of A* search. Third, the plan in latent space is decoded into a sequence of images using the generator, resulting in a visual plan.

2.1 Context Conditional CIGAN Model

For many problems of interest, such as the rope manipulation with dynamic obstacles we experiment with, the domain can be decomposed into a manipulate-able, movable object (rope), and components which are fixed during manipulation (obstacles). In such cases, training the CIGAN model to generate both movable and fixed components is unnecessary, since only the movable component can change within an episode. Thus, we propose a modification of the CIGAN architecture that takes as input an observation of the fixed components as a *context vector*. We term this model a Context Conditional CIGAN (C^3IGAN). More information about this model can be found in the appendix in Section A.

3 Experiments

We demonstrate our method on three domains. The first is a two-block world in Mujoco (Todorov et al. (2012)). In this domain, we perform a comparison with batch off-policy RL – an alternative method for learning a control policy from data. The second domain contains a movable block with a static obstacle. In this domain, we show the need for planning when the inverse model fails to navigate around the obstacle, while VPA learns to do so. Finally, we deploy the algorithm on a PR2 robot to manipulate a deformable rope around obstacles. Within real world rope manipulation, we explore two similar variations of the domain: one with static obstacles in which we compare our method to that of Nair et al. (2017), and the other with dynamic obstacles in which we demonstrate the potential of generalizing to variations in the environment using C^3IGAN.

3.1 Two-Block Domain

In this domain, the task is to move two rigid blocks on a table to some goal location.

We compare VPA with an alternative data-driven approach based on model-free batch RL, namely, fitted Q-iteration (Riedmiller (2005)). This is a strong baseline, that makes use of both the action-labeled and unlabled data, and incorporates several recent techniques for image-based RL. However, as stated earlier, RL is known to have difficulties with large state spaces (image), reward specification, and sample efficiency. To demonstrate this, we also run RL with several *artificial benefits*: (1) simple state space – true positions of the blocks, (2) true reward – based on real distance to target, and (3) more data – 30k action-labeled samples. Our results, reported in Table 1 show that, surprisingly, VPA significantly outperforms RL even with the artificial benefits.

3.2 Block-Wall Domain

We further investigate the efficacy of our model on another simulated domain, now with planning more intuitively necessary to complete the task. In this domain, the agent has to manipulate a green block around a red vertical obstacle.

We compare two variations of our method against the baseline of using only an inverse model, as suggested by Nair et al. (2017). The first method executes 100 different generated plans from

Table 1: VPA vs. Batch RL on block moving task. We show the average final L2 distance to goal and the success rate to move two blocks to be within 0.5 radius to the goal when executed on 50 new tasks.

Method	L2 distance	Success Rate
VPA (2k)	**0.335 ±0.121**	**90%**
Batch RL (positions, real r, 2k)	0.657 ±0.701	76%
Batch RL (positions, real r, 30k)	0.675 ±0.739	74%
Batch RL (image, real r, 2k)	1.172 ±0.991	16%
Batch RL (image, real r, 30k)	1.186 ±0.940	42%
Batch RL (image, embedded r, 2k)	1.346 ±0.891	14%
Batch RL (image, embedded r, 30k)	1.445 ±1.096	18%

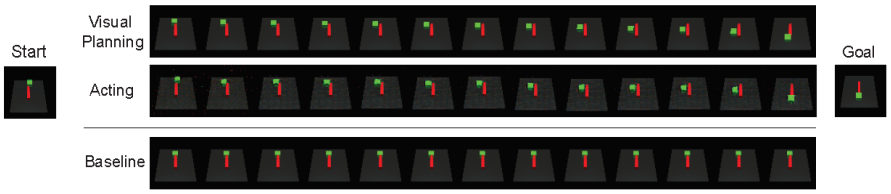

Figure 3: Comparison between VPA and an inverse model baseline. The baseline attempts to directly apply the inverse model on the goal, while our method employs the plan generated by CIGAN, shown center, to navigate from start to goal. Without a plan, the baseline blindly attempts to move the block downwards without knowledge that there is an obstacle in the way, whereas our model has gained the intuition that the block cannot pass through walls and thus any feasible plan must go around the obstacle.

CIGAN in the simulated environment and selects the minimum L2-distance execution. The second executes just 1 generated plan, which is selected using a combination of a classifier trained on the dataset and an object detector, where we use Mask-RCNN trained on a simple shape dataset and reward pairs of images that have the desired number of objects He et al. (2018).

Table 2: VPA vs. inverse model on block-wall task. The average final L2 distance to goal and the success rate to move one block in the block-wall domain to be within 0.5 radius of the goal.

Method	L2 distance	Success Rate
Baseline	0.459 ±0.433	45%
VPA (minimum selected out of 100 plans)	**0.023 ±0.033**	**100%**
VPA (autoselected 1 plan)	0.131 ±0.242	90%

3.3 Real Robot Rope Manipulation Domain

And finally, we bring our method out of simulation and into the real world by conducting experiments with a PR2 robot manipulating a flexible rope that is fixed on one end and can move between two obstacles. This domain is inspired by wire threading – an important industrial task that is extremely challenging for autonomous robots.

3.3.1 Static Obstacles

We begin our investigation by comparing our planning based method to same baseline of only using an inverse model without planning, as in the previous block and wall domain. With the additional constraint of obstacles, we conjecture that the inverse model, which is essentially reactive in its computation, will not suffice to plan movements that involve these obstacles.

In Figure 4, we demonstrate a setting where indeed planning is required to solve the task. In this example, going from start to goal requires traveling with the rope around the obstacle. It can be seen that our VPA method *plans* to go around the obstacle, which makes it feasible to solve the task by following the plan with the inverse model. Just using the M_{IM}, however, does not result in planning

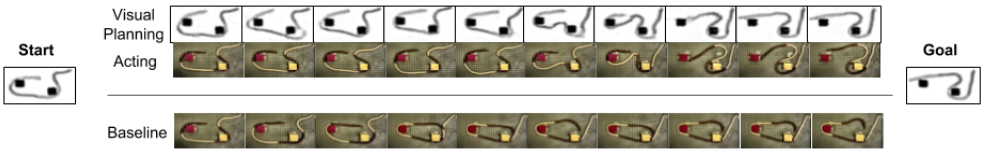

Figure 4: Comparison between VPA and an inverse model baseline. In this example, the rope is required to travel around an obstacle to reach the goal. The CIGAN-generated plan is presented in grayscale in the middle. The results after the PR2 robot successfully runs iterations of the M_{IM} with our VPA method by tracking the plan is shown above, and the baseline of only the inverse model, M_{IM} is shown below (similar to the method of Nair et al. (2017)). Note that VPA plans to go *around* the obstacle, leading to a successful plan execution, while the inverse model is not capable of such planning, due to its reactive, short-term nature, and therefore cannot complete the task. This example demonstrates the importance of planning for nontrivial manipulation tasks.

around the obstacle, which leads to a failure in execution. Further plans can be seen in the appendix in Figure 8

3.3.2 Dynamic Obstacles

In this section we demonstrate the potential of C^{3}IGAN in generalizing to unseen environments. To this end, we modified the rope manipulation domain to include *dynamic*, smaller obstacles, which were intermittently moved (manually) while collecting the training data. These changes render this variation harder than the previous one for our vision-based planning method. Our hope is that our model can imagine, plan, and execute rope manipulation in domains with obstacle configurations that were not explicitly seen during training.

In terms of success rate, we qualitatively inspected the plans and found that approximately 15% were visually accurate representations of rope manipulation. The most common failure cases are inaccurate encoding, leading to a misspecified goal image, or the rope breaking during the trajectory. We believe that more data and further improvements to C^{3}IGAN would significantly improve these results. From the visually correct plans, the inverse model was able to successfully execute 20%. This is somewhat worse than the results of Nair et al. (2017), which we attribute to the order of magnitude smaller data set we used, and our additional obstacles. We emphasize that even though our success rates are not high, our method is interpretable, and many failure cases can be caught by visual inspection, without running the robot. We see these results as a proof of concept for a new paradigm for robot manipulation.

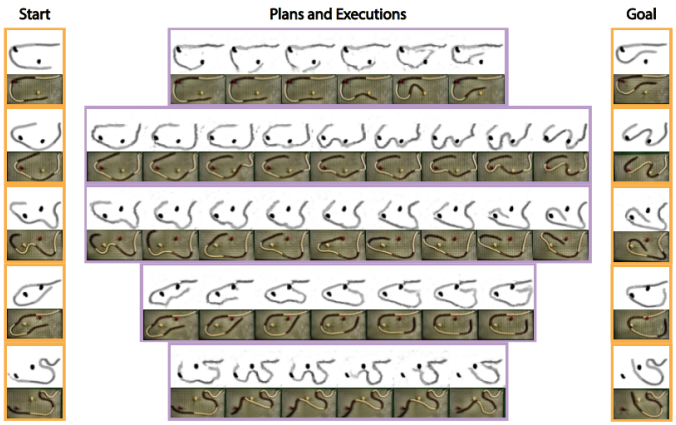

Figure 5: 5 examples of VPA executed on the rope domain. The first 4 are successful runs, and the last 1 is where a plan is generated to reach the goal, but the action policy is not strong enough to carry it out. Looking at one column at a time, the top image is the start state and the bottom is the goal state. In the middle, the grayscale images are the visualized plan, and the colored images are the actual results of the rope when we run the inverse model to have the PR2 take actions.

REFERENCES

P. Agrawal, A. Nair, P. Abbeel, J. Malik, and S. Levine. Learning to poke by poking: Experiential learning of intuitive physics. 2016.

F. Ebert, C. Finn, A. X. Lee, and S. Levine. Self-supervised visual planning with temporal skip connections. pp. 344–356, 2017.

C. Finn and S. Levine. Deep visual foresight for planning robot motion. pp. 2786–2793, 2017.

Kaiming He, Georgia Gkioxari, Piotr Dollar, and Ross Girshick. Mask r-cnn. *arxiv preprint arxiv:1703.06870*, 2018.

T. Kurutach, A. Tamar, G. Yang, S. Russell, and P. Abbeel. Learning plannable representations with causal infogan. *arXiv preprint arXiv:1807.09341*, 2018.

Dale McConachie, Mengyao Ruan, and Dmitry Berenson. Interleaving planning and control for deformable object manipulation. In *International Symposium on Robotics Research (ISRR)*, 2017.

Ashvin Nair, Dian Chen, Pulkit Agrawal, Phillip Isola, Pieter Abbeel, Jitendra Malik, and Sergey Levine. Combining self-supervised learning and imitation for vision-based rope manipulation. In *Robotics and Automation (ICRA), 2017 IEEE International Conference on*, pp. 2146–2153. IEEE, 2017.

Martin Riedmiller. Neural fitted q iteration–first experiences with a data efficient neural reinforcement learning method. In *European Conference on Machine Learning*, pp. 317–328. Springer, 2005.

Siddharth Srivastava, Eugene Fang, Lorenzo Riano, Rohan Chitnis, Stuart Russell, and Pieter Abbeel. Combined task and motion planning through an extensible planner-independent interface layer. In *Robotics and Automation (ICRA), 2014 IEEE International Conference on*, pp. 639–646. IEEE, 2014.

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

A C^{3}IGAN MODEL

In C^{3}IGAN, as shown in Figure 6, the generator takes in as input z, s, s', c, where the context c represents an image of the fixed components in the domain, so in our case, the obstacles (obst). The generated observations are added (pixel-wise) to c before they are passed onto the discriminator. In this way the generator is trained to generate only the movable part in the scene. Thus, the generator is now only in charge of generating the images of the rope, and not the obstacles. By relieving the generator of the responsibility of generating a fixed backdrop that is fixed throughout a trajectory, it can focus on the nuances of the object whose movement we actually want to control. The generator in this model is also able to generalize to new obstacle domains not seen during training. It is clear how this model could extend to other applications where there is a fixed background to interact with, such as a maze or other physical barriers.

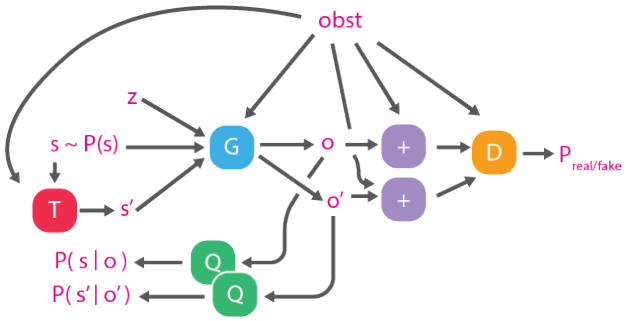

Figure 6: C^{3}IGAN Model Architecture

B ADDITIONAL RESULTS

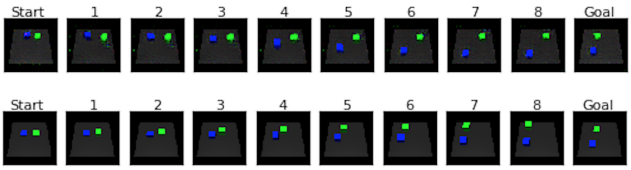

Figure 7: Top image: Imagined plan by Causal InfoGAN in two block domain. Start and Goal image are both $o_{closest}$ to the actual o_{start} and o_{goal}, which are shown right below them. Bottom image: Images showing the actual successful results of running entire VPA pipeline on Mujoco

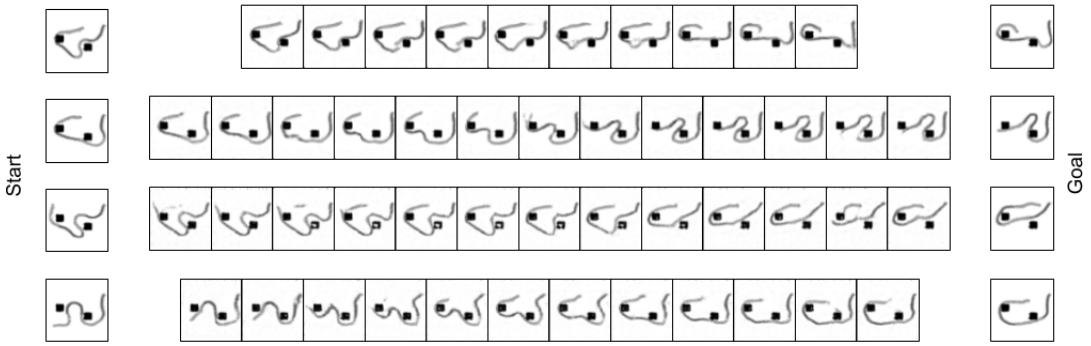

Figure 8: Demonstration of CIGAN plans for the rope domain with fixed obstacles. The top row shows the start states and the bottom row shows the goal states of 4 different problems given to the CIGAN. The middle 4 columns show sample generated plans. Note the realistic transitions of the rope around the obstacles, which obey physical properties of the rope such as being stretched when pulled from the end.