Structured deep generative models for sampling on constraint manifolds in sequential manipulation

Anonymous Author(s)
Affiliation
Address
email

Abstract: Sampling efficiently on constraint manifolds is a core problem in robotics. We propose Deep Generative Constraint Sampling (DGCS), which combines a deep generative model for sampling close to a constraint manifold with a nonlinear constrained optimization to project onto the constraint manifold. The generative model is conditioned on the problem instance, taking a scene image as input, and it is trained with a dataset of solutions and a novel analytic constraint term. To further improve the precision and diversity of samples, we extend the approach to exploit a factorization of the constrained problem. We evaluate our approach in two problems of robotic sequential manipulation in cluttered environments. Experimental results demonstrate that our deep generative model produces diverse and precise samples and outperforms heuristic warmstart initialization.

Keywords: Generative Models, Nonlinear Optimization, Constraint Graph, Robotic Sequential Manipulation

1 Introduction

We consider the problem of sampling points on a constraint manifold, i.e. finding diverse solutions of a nonlinear mathematical program without costs. Such problems arise throughout robotics, in particular in solving sequential manipulation problems, as it will be the focus in our applications. We assume that the constraints are given in terms of piece-wise differentiable equalities and inequalities. This allows us to leverage constrained optimization methods to project a randomly sampled configurations onto the manifold. However, for highly non-linear constraints and disconnected solution manifolds, a local optimization method can get trapped in local optima and fail to find a feasible solution. Further, our aim is to generate a diverse and covering set of points on the constraint manifold, and to reduce the computation time needed by the local optimization. The crucial challenge therefore is to first *sample diversely and close to the constraint manifold*, so that the local optimizer can efficiently project to solutions covering the constraint manifold.

We follow a learning-based approach, where we assume that a dataset comprising of feasible points for various problem instances is available. Reusing precomputed data on similar problems is a promising approach [1, 2, 3]. A fundamental challenge is that the mapping between problem instance and the feasible manifold is extremely nonlinear and discontinuous [4, 5].

Our approach, therefore, is to train a generative neural model to map a problem instance – given in terms of a scene image – to a randomized sample close to the constraint manifold for that instance. An image-based prediction of solutions to manipulation problems has recently received attention [1, 6, 7, 8, 9] as it provides a fixed size parametrization (image resolution) that encodes the obstacles and the objects to be interacted, without need to engineer problem instance features. Importantly, it is a natural way to encode non-fixed number of objects.

Generative Adversarial Networks (GANs) [10, 11] and Variational Autoencoders (VAEs) [12] have proposed a powerful methodology for training such a generative model and shown great potential to represent complex distributions in high dimensional space. This work adopts the training objective and method of GANs, which is to minimize the divergence between the generative and target distributions. That is, given the training data being the diverse samples on the constraint manifold,

Submitted to the 5th Conference on Robot Learning (CoRL 2021). Do not distribute.

the deep generative model is trained to produce seeds that lie close to the manifold conditioned on the current problem, represented with an image of the scene. These seeds are then used as initialization (warmstart) of a subsequent optimization procedure that projects them to the manifold to find a feasible solution.

Despite the expressive power of function approximators and recent improvements in deep generative models, they still have limitations in generating samples from a highly nonlinear and multimodal distribution. Inspired by previous works exploiting an underlying sparse structure of robotic sequential manipulation problems [13, 14, 15, 16] we propose an extension of the above system that leverages a factorization of the constraint problem to model this multimodal distribution more efficiently. Sampling is decomposed into a sequence, where for each factor we can train a separate *conditional* generative model. Sequencing these conditional generative models increases sample efficiency of the training and the multi-modality of the distributions it can represent.

We demonstrate the system in the context of robotic sequential manipulation, where the problem is to find a sequence of feasible mode-switch configurations, e.g., robot's poses when it is picking or placing an object. These problems are defined by a set of continuous variables representing sequences of robot configurations, object poses, and relative transformations, with nonlinear, piecewise differentiable constraints, e.g., for grasping, placement, collision avoidance and kinematics, which give a natural factorization of the overall problem. The solutions of such problems are often not unique, but rather form a complex and disconnected manifold.

In summary, our core contributions are the following:

- We introduce Deep Generative Constraint Sampling (DGCS), that combines deep generative models with nonlinear optimization to efficiently generate diverse solutions of constraint problems. The generative model is trained conditional to a scene image to be applicable to varying problem instances. As an application, we produce mode-switch configurations in sequential manipulation.
- We extend the standard data-based training of the generative model with analytical features of the feasible manifold to achieve both, diversity and sample accuracy.
- We extend our system to exploit a factorization of the constraint problem, if given, to further improve modelling of multi-modal distributions with disconnected support and reduce sample complexity.

2 Related Work

Generative Models in Robotics Recently, deep generative models have been successfully applied in robotics, specially in settings where the problem is represented directly with images or point clouds. For example, for generating grasp of complex objects directly from images [17, 18]. In the context of motion planning [19, 20, 21], and planning [22, 23, 24], generative models are used for sampling informative collision free configurations which greatly improve running time of sampling based algorithms. In this line of research the general aim is to train a network to directly predict solutions to a problem. However, non of these work addresses sampling on a high-dimensional constraint manifold, or predicting full multi-robot manipulation sequences. To achieve accurate sampling, we combine learned generative models with local optimization for constraint projection.

Warmstart in Nonlinear optimization In robotics, nonlinear optimization is used to sample from constraint manifolds and optimize trajectories [25, 26, 27]. The goal of recent data-based approaches is to learn a warmstart to reduce online computation time [2, 28]. In settings where the nonlinear programs can be represented with mixed integer constraints, [29, 30, 3, 31] learn an assignment to the integer constraints and run subsequent optimization. In comparison to our method, integer formulations use a discrimative model and are easier to train but is difficult to generalize to problems without a clear integer structure. Recently [32] applies GANs for inverse kinematics, adding forward kinematics into the network training. In contrast we use general analytical features in the cost term. Moreover our framework is conditioned on an image representation, and scales to sequential manipulation.

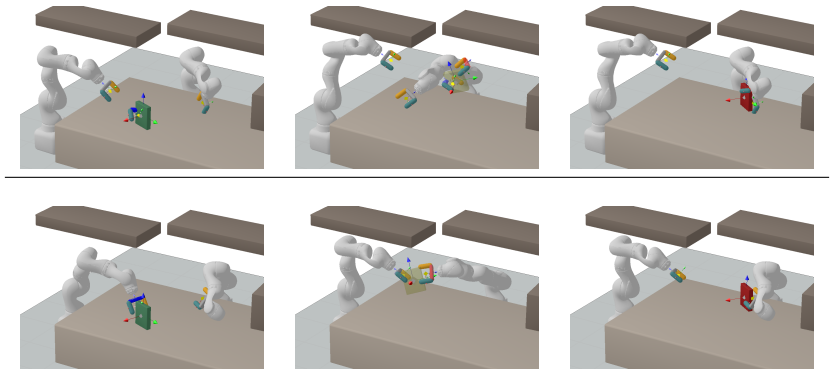


Figure 1: Sequence of mode-switches of the *handover* problem. Our sampling framework (DGCS) combines a deep generative model $x_0 \sim \mathbb{P}_\theta(\tau)$ that produces approximate samples conditioned on the scene (top row) with a nonlinear optimizer that projects them onto the constraint manifold (bottom row).

3 Sampling from a Constraint Manifold

The problem we address is to generate samples from a manifold $\mathcal{M}_\tau$ parametrized by a fixed dimensional (but potentially large, e.g. images) problem parameter τ,

$$\mathcal{M}_\tau = \{x \in \mathbb{R}^n \quad \text{s.t} \quad \phi_{\text{eq}}(x;\tau) = 0,\ \phi_{\text{ineq}}(x;\tau) \leq 0\}\ , \tag{1}$$

where $\phi_{\text{eq}}(x;\tau) : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^{n_e}$ and $\phi_{\text{ineq}}(x;\tau) : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^{n_i}$ are nonlinear vector-valued functions, that are piecewise differentiable. $\tau \in \mathbb{R}^m$ represents the current problem instance and parametrizes all the constraints. In the context of robotic manipulation, τ represents all the objects in the scene (position, size, shape...), x are robot and object degrees of freedom that are subject to constraints, and $\phi_{\text{eq}}, \phi_{\text{ineq}}$ are (in)equality constraints that describe the problem's objectives such as collision avoidance, grasping, kinematic, and pose constraints.

A generative model $x \sim \mathbb{P}(\tau)$ that produces feasible solutions (samples $x \in \mathcal{M}_\tau$) is built from two components (see Alg. 1): a randomized seed x_0 generation and a constrained optimization algorithm that projects the seed x_0 to the solution manifold (2).

$$\min_x ||x - x_0||^2 \text{ s.t } \phi_{\text{eq}}(x;\tau) = 0;\ \phi_{\text{ineq}}(x;\tau) \leq 0 \tag{2}$$

In our current implementation, we solve (2) approximately, by running a nonlinear optimizer from starting point x_0 on the feasibility problem: find x s.t $\phi_{\text{eq}}(x;\tau) = 0;\ \phi_{\text{ineq}}(x;\tau) \leq 0$. The initialization and internal regularization of the optimizer provides an implicit regularization with respect to x_0.

The projection step is a non-convex optimization problem that has no guarantees to produce a feasible sample; especially for a complex sequential manipulation problem, the optimization landscape often has substantial local optima induced by constraints, making non-linear projection prone to fail unless seeds are close to the solution manifold.

To address such difficulty, we train a seeding distribution $\mathbb{P}_\theta(\tau)$ to approximate a reference distribution of feasible solutions $\mathbb{P}_r(\tau)$, so that it can generate diverse samples close to the parametric manifold $\mathcal{M}_\tau$ which the optimizer can easily project. An example is shown in Fig. 1.

Algorithm 1 Sample from a Constraint Manifold

Input: Problem parametrization τ, number of trials N
$L = \{\}$ empty list of samples
for $i = 1, 2, ..., N$ **do**
  Sample $x_0 \sim \mathbb{P}_\theta(\tau)$
  $x \leftarrow \Pi(x_0)$, Project to $\mathcal{M}_\tau$ with a nonlinear optimizer (2)
  if x feasible **then**
    append x to L
Output: List of valid samples L

4 Training deep generative models to sample constraint manifolds

Our deep generative distribution is $\tilde{x} \sim \mathbb{P}_\theta(\tau)$ with $\tilde{x} = G_\theta(z, \tau), z \sim \mathbb{P}_z$ where $\mathbb{P}_z$ is a multidimensional Gaussian and G_θ is a neural network with parameters θ.

In contrast to the standard application of adversarial generative models (images generation), in our setting we also have an analytical description of the support of the target distribution, namely the features $\phi(x; \tau) = [\phi_{\text{eq}}(x; \tau), \max(0, \phi_{\text{ineq}}(x, \tau))]$ that describe $\mathcal{M}_\tau$. We could consider a data-free gradient based optimization of the analytical constraint violation as

$$\min \mathop{\mathbb{E}}_{\tau} \mathop{\mathbb{E}}_{\tilde{x} \sim \mathbb{P}_\theta} ||\phi(\tilde{x}; \tau)||^2 \ . \tag{3}$$

However, this is extremely ill-posed and converges to the "deterministic mapping" local minimum $G_\theta(z, \tau) = G_{\theta,\tau} \forall z$, losing the potential to generate a diverse distribution.

To enforce sample diversity, we regularize with respect to a reference distribution $\mathbb{P}_r$ that is diverse and has its support on the manifold, i.e., $\mathbb{E}_{x \sim \mathbb{P}_r} ||\phi(x; \tau)||^2 = 0$. Namely, we formulate the problem as:

$$\min \mathop{\mathbb{E}}_{\tau} \ W(\mathbb{P}_\theta(\tau), \mathbb{P}_r(\tau)) + \beta \mathop{\mathbb{E}}_{\tilde{x} \sim \mathbb{P}_\theta} ||\phi(\tilde{x}; \tau)||^2 \ , \tag{4}$$

where W is the Wasserstein distance, and $\beta \in \mathbb{R}_{>0}$. Although the analytical term does not provide additional information than the support of $\mathbb{P}_r$, as will be shown in the experiment section, its contribution is important in the context of function approximation and stochastic gradient descent in non-convex optimization.

4.1 Wasserstein Distance and Adversarial Formulation

The Wasserstein-1 (Earth Mover distance) between two probability distributions $\mathbb{P}_r$ and $\mathbb{P}_\theta$ (5) is defined intuitively as the cost of geometrically moving the mass from one distribution to the other under an optimal transport plan,

$$W\left(\mathbb{P}_r, \mathbb{P}_\theta\right) = \inf_{\gamma \in \Pi(\mathbb{P}_r, \mathbb{P}_\theta)} \mathop{\mathbb{E}}_{(x,y) \sim \gamma} [\|x - y\|] \ , \tag{5}$$

where $\Pi(\mathbb{P}_r, \mathbb{P}_\theta)$ denotes the set of all joint distributions $\gamma(x, y)$ with marginals $\mathbb{P}_r$ and $\mathbb{P}_\theta$.

Compared to other notions of distance such as Jensen-Shannon Divergence or Total Variation, adversarial generative models with Wasserstein distances are known to have good practical stability and convergence and are less prone to mode collapse [11, 33]. Furthermore, the Wasserstein distance provides a nice interpretation in our application, because it takes into account the geometric distance between two distributions; the minimization of the geometric distance (norm in configuration space) to a distribution supported on the manifold improves the success rate of the subsequent projection procedure.

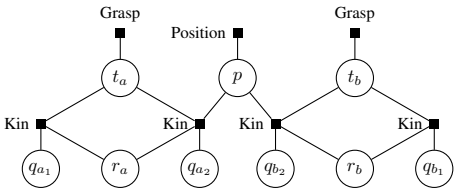
We minimize the objective function (4) with the Wasserstein GAN [11, 33] formulation. Using Kantorovich Duality, the original formulation (5) is transformed into a minimax game between the critic network D and the generator G, which are trained simultaneously with stochastic gradient descent. Specifically, the minimax problem (with our analytical feature) is:

$$\min_G \max_D \mathop{\mathbb{E}}_{x \sim \mathbb{P}_r} [D(x)] - \mathop{\mathbb{E}}_{\tilde{x} \sim \mathbb{P}_\theta} [D(\tilde{x})] - \lambda \mathop{\mathbb{E}}_{\hat{x} \sim \mathbb{P}_{\hat{x}}} \left[(\|\nabla_{\hat{x}} D(\hat{x})\| - 1)^2 \right] + \beta \mathop{\mathbb{E}}_{\tilde{x} \sim \mathbb{P}_\theta} \left[\|\phi\left(\tilde{x}; \tau\right)\|^2 \right] \ , \tag{6}$$

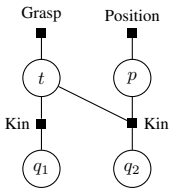
where $\hat{x}$ are samples between x and $\tilde{x}$. Our reference distribution $\mathbb{P}_r(\tau)$ comprises of a discrete set of solution/problem-pairs $D = \{(x_i, \tau_i)\}$. This data is computed offline by solving a large set of problems, with a focus on getting diverse samples.

5 Extension to Exploit Factorization

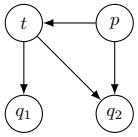
The previous sections treated x as a single variable. While the approach for training a single generative model $\tilde{x} = G_\theta(x; \tau), z \sim \mathbb{P}_z$ is powerful, we can further improve the scalability to large problems by exploiting a given factorization of x and sequentially decomposing the sampling procedure into a sequence of conditional samplings, as in Bayesian networks [34].



(a) Handover



(b) Pick and Place



(c) Pick and Place

Figure 2: Constraint graphs (2b, 2a) and sampling network (2c). Additional uni-variable collision constraints are applied to p's and q's.

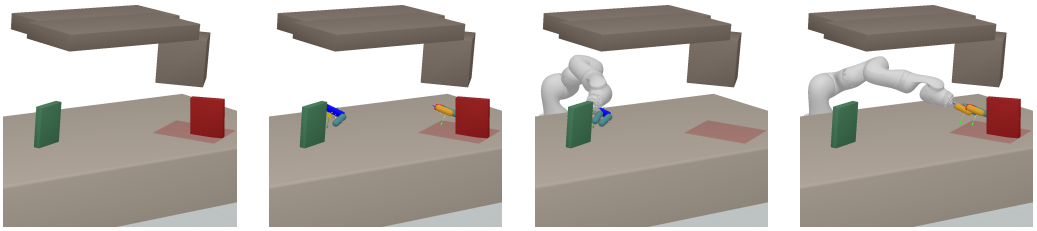


Figure 3: Sequence of learned deep generative of models for Pick and Place problems.

We assume the configuration $x = \{x_1, ..., x_N\}$ can be factored into N vector variables. Similarly, the constraints (equalities and inequalities) $\phi(x; \tau)$ are factored into L sets of constraints, $\{\phi_1, \phi_2, \ldots, \phi_L\}$ with $\phi_j : \mathbb{R}^{a_j} \to \mathbb{R}^{b_j}$ where $a_j,\ b_j \in \mathbb{N}$ are the dimensions of the domain and co-domain of the constraint ϕ_j, and each constraint ϕ_j depends only on a subset of variables.

$$x = \{x_1, \ldots, x_N\}\ , \quad \phi(x; \tau) = \{\phi_1, \ldots, \phi_L\}\ . \tag{7}$$

The set of constraints now defines a factored mathematical program without costs, also called constraint graph.

Such factorization naturally arise in many applications, where each variable has some semantic and geometric meaning. Fig. 2 shows some examples of such constraint graphs in the context of robotics sequential manipulation. Variables correspond to robot joint configuration q, mobile robots base pose r, relative transformations between objects and grippers t and object position p. These variables are then coupled by the kinematic chain, collision avoidance, grasp, and pose constraints. The pick-and-place graph in Fig. 2b is the fundamental building block of sequential manipulation. Longer and more complex tasks such as a handover (Fig. 2a) or an assembly can be represented by chaining the basic pick and place with additional constraints.

5.1 Directed Graphical Model and Sequential Sampling

The factored structure directly implies a factorization of the joint probability distribution $\mathbb{P}(\tau)$ from which we want to sample. As in Bayesian networks we can sequentialize sampling if we decide on an ordering of variables that corresponds to a directed acyclic graph. Instead of learning a single G_θ to produce a full assignment with $z \sim \mathbb{P}_z, x = G_\theta(z; \tau)$, we learn a *conditional* model for each factor by using the corresponding marginal distributions of the original data, and the subset of corresponding analytical features in the constraint graph.

We illustrate the benefits of this factorization with the Pick-and-Place problem. The joint distribution for the Pick-and-Place problem is factorized into (we omit conditioning on τ for clarity)

$$\mathbb{P}(p, t, q_1, q_2) = \mathbb{P}(p)\, \mathbb{P}(t|p)\, \mathbb{P}(q_1|t)\, \mathbb{P}(q_2|t, p)\ , \tag{8}$$

which exploits conditional independence between (q_1, p) given t and (q_1, q_2) given t. We exploit this structure by training a sequence of conditional samplers $p \sim \mathbb{P}_p$, $t \sim \mathbb{P}_t(p)$, $q_1 \sim \mathbb{P}_{q_1}(t)$, $q_2 \sim \mathbb{P}_{q_2}(t, p)$, see Fig. 3. In the case of mobile robots, we can consider the robot base position r as a variable in the graph and use the sampling order $p \to r \to t \to q$.

As a side note, using marginal distributions of a jointly consistent dataset is necessary, as only the marginal contains useful information of whether a partial assignments will admit a full solution. For

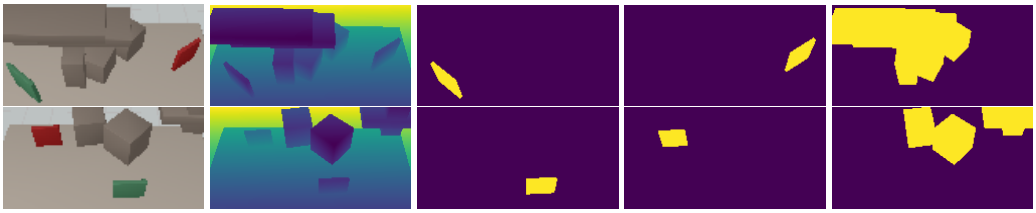


Figure 4: Problem instances in *Assembly* and *Handover*

example, when sampling $p \sim \mathbb{P}_p$, we want to fulfil the two following constraints (conditioned on the scene): a) position$(p) = 0$; and b) $\exists(t, q_1, q_2)$ s.t (t, q_1, q_2, p) is feasible. The second constraint cannot be evaluated efficiently, but is modelled by the marginal distribution of the data.

5.2 Note on the advantage of factorization for modelling mulimodality

Generating a distribution with disconnected supports with a deep generative model that receives a continuous input noise $z \sim \mathbb{P}_z$ would require infinite gradients in G_θ and can only be done approximately. In these cases, training is unstable and very sensitive to hyperparameters and architecture.

We can model disconnected distribution more effectively by factoring the full joint probability as a sequence of smaller, conditional modules, as is confirmed by our experimental results in Section 6.3. The sequencing still requires that each module is able to produce some degree of multimodality. However, once a module in the sequence receives a disconnected input in form of conditioning, it can successfully produce a disconnected output. As we chain modules with the ability to generate a small amount of disconnected components given a continuous input, the number of possible disconnected components of the output distribution grows exponentially with the number of modules in sequence.

Furthermore, from a practical perspective, training smaller modules turned out to be more effective. For instance, we observed that the analytical feature $||\phi(x; \tau)||^2$ of the joint problem can provide badly-conditioned gradients when evaluated far from the manifold. This issue is alleviated when considering only subsets of constraints and variables.

In our preliminary experiments, we also evaluated GAN frameworks that have a mechanism to model disconnected distribution [35, 36] (and require an estimate of the number of disconnected components) but did not found significant improvement.

6 Experiments

6.1 Image based Problem representation

We use an image-based representation of the problem instance τ that consists of the depth image and masks. Specifically, $\tau = \{d, m_1, m_2, m_3\}$, where d is the depth image, and m_1, m_2, m_3 are three masks that contain, respectively, information from the initial object pose, goal pose or placement region, and obstacles, see Fig. 4. In the factored approach, each generative module receives as input only the "relevant" masks, e.g. the sampler for the robot pick configuration receives a mask of the obstacles and the initial configuration, but not the goal pose.

The main strength of the image representation is that it can generalize to different object shapes and changing number of obstacles and shapes. Moreover, a depth camera is easily available, approximate masks can be computed with image segmentation techniques and provides a good representation of sequential manipulation problems on tabletop scenarios.

6.2 Scenarios

We consider three different manipulation tasks that involve object manipulation with stable grasps.

- *Pick-and-Place*: A robot has to pick an object and place it on a goal rectangular region of the table. Fig. 3.
- *Assembly*: two mobile robots have to pick an object each and assemble them. The assembly in not unique, but modelled as a manifold with constraints on rotation and position: objects

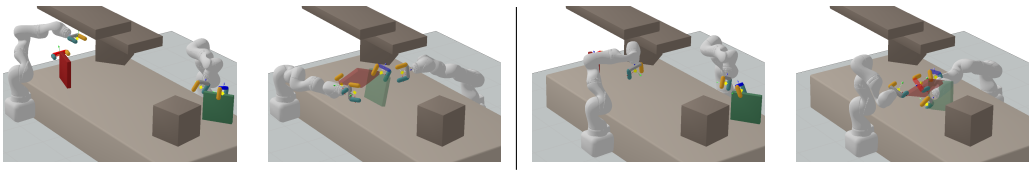


Figure 5: Samples from the deep generative model in the same instance of the *assembly* scenario. Each sample is shown with two keyframes (pick and assembly). Both seeds lead to feasible solution.

	Seeds			Solutions		
	Coverage	Precision	Error	Coverage	Precision	Success Rate
Big NN	0.81±0.09	0.7±0.11	8.38±1.82	0.58±0.12	0.39±0.06	0.46
Big NN + analytical	0.79±0.12	0.53±0.14	1.21±1.35	0.75±0.13	0.41±0.13	0.43
Structure NN	0.6±0.08	0.62±0.1	8.09±1.35	0.41±0.05	0.44±0.06	0.56
Structure NN + analytical	0.57±0.08	0.47±0.13	1.46±0.79	0.44±0.07	0.28±0.08	0.78

Table 1: Ablation Study

have to be perpendicular and make stable contact with predefined faces of the cubes (similar to a T shape). Figure 5.

- *Handover*: Two mobile robots have to collaborate to move the object from the initial to the goal position doing a handover. Figure 1.

All these tasks are performed on a cluttered table with a varying number (between 3 and 5) of obstacles. The grasp between the gripper and the object is modelled with a two finger gripper (e.g. Franka Panda Gripper) that constraints the position and orientation. The training dataset consists of 4000 pairs of problem-solution computed offline with user-defined sampling sequence to ensure diversity. The problems consists of 64x128 images (input of network) and their corresponding environment.

6.3 Ablation Study

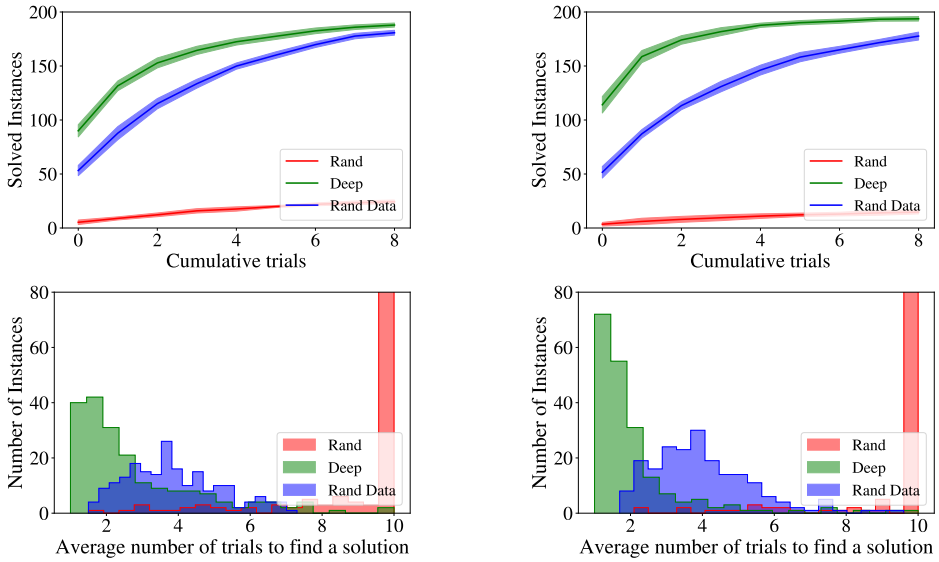
The *Pick-and-Place* scenario is used for an ablation study of the proposed generative model. We analyze the contribution of the factored structure (*Big NN* vs *Structure NN*) and analytical error term (+*analytical*). We evaluate precision of seed samples (output from deep generative models), and coverage of the seeds and solutions. Specifically, we first generate 4000 samples with the generative model and then project them with the nonlinear optimizer. We report the following metrics:

- *Constraint violation*: $||\phi(\tilde{x};\tau)||^2$
- *Precision*: average nearest neighbour distance to a reference dataset. It models how close samples are to the real data (lower is better).
- *Coverage*: average nearest neighbour distance from the reference dataset to the computed samples. It describes how good is the reference dataset covered by the learned distribution (lower is better).

Results are shown in Table 1. When analyzing the seeds (output of the generative model) our two contributions are essential to achieve small constraint violation and good coverage. In terms of solutions, only the networks with structure provide good coverage and seeds from the model with structure and analytical term have higher probability of leading to a solution (success rate).

6.4 Benchmark: Generative Model in Nonlinear optimization

The *Assembly* and *Handover* scenarios are used to compare our generative model against two baseline methods for warmstarting (seeding) nonlinear optimizers. We analyze the number of solved problems and the number of necessary optimization runs. Measuring the number of solved nonlinear programs is an indirect way to evaluate coverage and sample quality, as both are fundamental to solve a diverse set of problems with a nonlinear optimizer and prevent convergence to infeasible points. We compare our complete model (deep generative model with structure and analytical error term), in short *deep*, with baselines:



(a) Handover (b) Assembly

Figure 6: Top row: Cumulative Number of solved problems. Bottom Row: Histogram of the estimated average of the number of trials necessary to solve an instance.

- *Rand:* Randomized initial guess around a reference position.
- *Rand Data:* Choosing samples x from the training dataset at random. The initial point is thus a feasible sample for another problem of the same family. This is actually a strong baseline, because it provides diverse informative initial seeds.

We evaluate the generative model (deep generative model + optimization) on 200 problems from the test dataset. We first report the cumulative number of problems solved as we increase the number of optimization trials (each trial has and independent starting point). The experiments are repeated 10 times and we report mean and variance, Fig. 6. We also evaluate how many optimization trials are necessary to solve each of the test instances, and report the histogram of the mean value. For this plot, unsolved problems are assumed to be solved with 10 trials (maximum number trials). Fig. 6.

In both scenarios, the proposed deep generative model outperforms the baseline warmstart, significantly reducing the average number of trials to solve the instances. The computational overhead of evaluating the neural network is small (we produce 10 samples in 8 ms with a GPU), while evaluating 200 problems with a maximum of 10 tries takes around 300-500s (most of the time is spent in optimization runs that converge to infeasible points).

7 Conclusion

In this work, we propose Deep Generative Constraint Sampling (DGCS), a new approach to sample from a constraint manifold to tackle problems in robotic sequential manipulation. Our framework combines a deep generative sampling model, conditioned on an image based representation of the problem, and a nonlinear optimizer to project samples onto the manifold. We further extended the approach to exploit a given factorization of the problem, by training a sequence of conditional generative models rather than a full joint generator. Our empirical results confirm that the inclusion of analytic constraints in the training of the generative model, as well as exploiting the factorization of a given problem significantly improve the efficiency, diversity and precision of the sampling approach. Using the trained generative models we can outperform baselines and heuristic warmstart strategies.

Our current framework combines generative sampling using a neural network with subsequent projection using constrained optimization. A promising future direction is to explore whether it is possible to embed the optimization algorithm as a last layer of the generative model, while keeping good coverage and avoiding mode collapse.

References

[1] D. Driess, J.-S. Ha, and M. Toussaint. Deep visual reasoning: Learning to predict action sequences for task and motion planning from an initial scene image. In *Robotics: Science and Systems 2020 (RSS 2020)*. RSS Foundation, 2020.

[2] N. Mansard, A. DelPrete, M. Geisert, S. Tonneau, and O. Stasse. Using a memory of motion to efficiently warm-start a nonlinear predictive controller. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2986–2993. IEEE, 2018.

[3] A. Cauligi, P. Culbertson, B. Stellato, D. Bertsimas, M. Schwager, and M. Pavone. Learning mixed-integer convex optimization strategies for robot planning and control. In *2020 59th IEEE Conference on Decision and Control (CDC)*, pages 1698–1705. IEEE, 2020.

[4] K. Hauser. Learning the problem-optimum map: Analysis and application to global optimization in robotics. *IEEE Transactions on Robotics*, 33(1):141–152, 2016.

[5] G. Tang and K. Hauser. Discontinuity-sensitive optimal control learning by mixture of experts. *CoRR*, abs/1803.02493, 2018. URL http://arxiv.org/abs/1803.02493.

[6] D. Driess, J.-S. Ha, R. Tedrake, and M. Toussaint. Learning geometric reasoning and control for long-horizon tasks from visual input. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2021.

[7] A. Xie, F. Ebert, S. Levine, and C. Finn. Improvisation through physical understanding: Using novel objects as tools with visual foresight. In *Proc. of Robotics: Science and Systems (R:SS)*, 2019.

[8] F. Ebert, C. Finn, A. X. Lee, and S. Levine. Self-supervised visual planning with temporal skip connections. In *Conference on Robot Learning*, 2017.

[9] C. Paxton, Y. Barnoy, K. D. Katyal, R. Arora, and G. D. Hager. Visual robot task planning. In *International Conference on Robotics and Automation (ICRA)*, pages 8832–8838. IEEE, 2019.

[10] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. *Advances in neural information processing systems*, 27:2672–2680, 2014.

[11] M. Arjovsky, S. Chintala, and L. Bottou. Wasserstein gan, 2017.

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

[13] C. R. Garrett, T. Lozano-Pérez, and L. P. Kaelbling. Sampling-based methods for factored task and motion planning. *CoRR*, abs/1801.00680, 2018. URL http://arxiv.org/abs/1801.00680.

[14] F. Lagriffoul, D. Dimitrov, J. Bidot, A. Saffiotti, and L. Karlsson. Efficiently combining task and motion planning using geometric constraints. *The International Journal of Robotics Research*, 33(14):1726–1747, 2014.

[15] K. Fang, Y. Zhu, A. Garg, S. Savarese, and L. Fei-Fei. Dynamics learning with cascaded variational inference for multi-step manipulation. *arXiv preprint arXiv:1910.13395*, 2019.

[16] J. O. de Haro, V. N. Hartmann, O. S. Oguz, and M. Toussaint. Learning efficient constraint graph sampling for robotic sequential manipulation. *CoRR*, abs/2011.04828, 2020. URL https://arxiv.org/abs/2011.04828.

[17] A. Mousavian, C. Eppner, and D. Fox. 6-dof graspnet: Variational grasp generation for object manipulation. *CoRR*, abs/1905.10520, 2019. URL http://arxiv.org/abs/1905.10520.

[18] A. Murali, A. Mousavian, C. Eppner, C. Paxton, and D. Fox. 6-dof grasping for target-driven object manipulation in clutter. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 6232–6238. IEEE, 2020.

[19] B. Ichter, J. Harrison, and M. Pavone. Learning sampling distributions for robot motion planning. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 7087–7094. IEEE, 2018.

[20] J.-S. Ha, Y.-J. Park, H.-J. Chae, S.-S. Park, and H.-L. Choi. Adaptive path-integral autoencoders: Representation learning and planning for dynamical systems. *Advances in Neural Information Processing Systems*, 31:8927–8938, 2018.

[21] G. Sutanto, I. M. R. Fernández, P. Englert, R. K. Ramachandran, and G. S. Sukhatme. Learning equality constraints for motion planning on manifolds, 2020.

[22] T. Kurutach, A. Tamar, G. Yang, S. Russell, and P. Abbeel. Learning plannable representations with causal infogan, 2018.

[23] B. Kim, L. P. Kaelbling, and T. Lozano-Pérez. Guiding the search in continuous state-action spaces by learning an action sampling distribution from off-target samples. *CoRR*, abs/1711.01391, 2017. URL http://arxiv.org/abs/1711.01391.

[24] A. Simeonov, Y. Du, B. Kim, F. R. Hogan, J. Tenenbaum, P. Agrawal, and A. Rodriguez. A long horizon planning framework for manipulating rigid pointcloud objects, 2020.

[25] M. A. Toussaint, K. R. Allen, K. A. Smith, and J. B. Tenenbaum. Differentiable physics and stable modes for tool-use and manipulation planning. 2018.

[26] I. Mordatch, E. Todorov, and Z. Popović. Discovery of complex behaviors through contact-invariant optimization. *ACM Transactions on Graphics (TOG)*, 31(4):1–8, 2012.

[27] A. W. Winkler, C. D. Bellicoso, M. Hutter, and J. Buchli. Gait and trajectory optimization for legged systems through phase-based end-effector parameterization. *IEEE Robotics and Automation Letters*, 3(3):1560–1567, 2018.

[28] W. Merkt, V. Ivan, and S. Vijayakumar. Leveraging precomputation with problem encoding for warm-starting trajectory optimization in complex environments. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 5877–5884. IEEE, 2018.

[29] R. Deits, T. Koolen, and R. Tedrake. Lvis: learning from value function intervals for contact-aware robot controllers. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 7762–7768. IEEE, 2019.

[30] D. Bertsimas and B. Stellato. Online mixed-integer optimization in milliseconds, 2020.

[31] A. M. Wells, N. T. Dantam, A. Shrivastava, and L. E. Kavraki. Learning feasibility for task and motion planning in tabletop environments. *IEEE robotics and automation letters*, 4(2): 1255–1262, 2019.

[32] T. S. Lembono, E. Pignat, J. Jankowski, and S. Calinon. Generative adversarial network to learn valid distributions of robot configurations for inverse kinematics and constrained motion planning. *CoRR*, abs/2011.05717, 2020. URL https://arxiv.org/abs/2011.05717.

[33] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. C. Courville. Improved training of wasserstein gans. In *Advances in neural information processing systems*, pages 5767–5777, 2017.

[34] D. Koller and N. Friedman. *Probabilistic graphical models: principles and techniques*. MIT press, 2009.

[35] X. Chen, Y. Duan, R. Houthooft, J. Schulman, I. Sutskever, and P. Abbeel. Infogan: Interpretable representation learning by information maximizing generative adversarial nets. *CoRR*, abs/1606.03657, 2016. URL http://arxiv.org/abs/1606.03657.

[36] M. Khayatkhoei, M. Singh, and A. Elgammal. Disconnected manifold learning for generative adversarial networks. *CoRR*, abs/1806.00880, 2018. URL http://arxiv.org/abs/1806.00880.