Constrained Generative Sampling of 6-DoF Grasps

Jens Lundell[†], Francesco Verdoja[*], Tran Nguyen Le[*], Arsalan Mousavian[‡], Dieter Fox[‡,§] and Ville Kyrki[*]

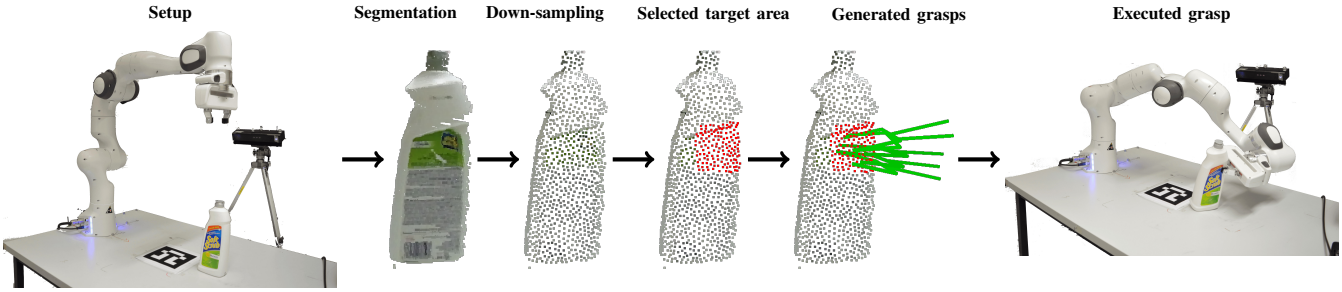


Fig. 1: An example grasp generated by VCGS on the target grasping area highlighted in red.

***Abstract*— Most state-of-the-art data-driven grasp sampling methods propose stable and collision-free grasps uniformly on the target object. For bin-picking, executing any of those reachable grasps is sufficient. However, for completing specific tasks, such as squeezing out liquid from a bottle, we want the grasp to be on a specific part of the object's body while avoiding other locations, such as the cap. This work presents a generative grasp sampling network, VCGS, capable of constrained 6-Degrees of Freedom (DoF) grasp sampling. In addition, we also curate a new dataset designed to train and evaluate methods for constrained grasping. The new dataset, called CONG, consists of over 14 million training samples of synthetically rendered point clouds and grasps at random target areas on 2889 objects. VCGS is benchmarked against GraspNet, a state-of-the-art unconstrained grasp sampler, in simulation and on a real robot. The results demonstrate that VCGS achieves a 10–15% higher grasp success rate than the baseline while being 2–3 times as sample efficient. Supplementary material is available on our project website.**

I. Introduction

Most state-of-the-art data-driven grasp sampling methods [1]–[4] focus on generating stable and collision-free grasps uniformly on the target object, which works well for completing bin-picking tasks. However, completing other tasks, such as squeezing the liquid from a bottle shown in Fig. 1, often requires grasping specific target areas. A possible approach to use state-of-the-art grasp sampling methods to generate grasps on specific target areas is to filter out grasps outside those areas. Unfortunately, this option is extremely sample-inefficient as it requires sampling many grasps to ensure that some high-quality ones are kept. Another option is to constrain grasp sampling to specific target regions [5]–[8]. In comparison to the first option, the second one has the promise of being much more sample efficient. Unfortunately, these constrained grasp sampling methods focus on generating grasps that either fulfill a specific task [7] or are located at semantically meaningful areas [5], [6], [8]. In this work, we do not make these assumptions and instead propose a general constrained grasp sampling method capable of focusing grasp sampling on *any* target area on the object, as demonstrated in Fig. 1.

Towards learning a constrained grasp sampler, we propose the Variational Constrained Grasp Sampler (VCGS): a new generative 6-DoF constrained grasp sampling method. VCGS takes as input a point cloud of the object to grasp and the target area and produces multiple 6-DoF grasps around the target area. We also curate a new dataset, CONG, to train VCGS. CONG consists of synthetically rendered point clouds of 2889 objects, and over 37 million grasps constrained to randomly sampled target areas.

We empirically evaluated VCGS in terms of grasp success rates and sample efficiency by benchmarking it against the state-of-the-art unconstrained 6-DoF grasp sampler GraspNet [3] on 126 objects in simulation and 12 objects in the real world. The experimental results demonstrate that the proposed constrained grasp sampler is 2–3 times as sample efficient as the unconstrained sampler while attaining 10–15% higher grasp success rates.

The main contributions of this work are:

- The Variational Constrained Grasp Sampler (VCGS): a novel constrained 6-DoF generative grasp sampling deep neural network.
- CONG: a large-scale grasping dataset including over 37 million grasps constrained to random target areas on 2889 objects.
- An extensive empirical evaluation of VCGS against the state-of-the-art 6-DoF GraspNet [3], demonstrating, both in simulation and on real hardware, that generating constrained grasps brings significant improvement in grasp success rates and sample efficiency.

[*] Intelligent Robotics Group, Department of Electrical Engineering and Automation, School of Electrical Engineering, Aalto University. Finland.
[†] KTH Royal Institute of Technology, Sweden jelundel@kth.se.
[‡] NVIDIA Corporation, USA
[§] Paul G. Allen School of Computer Science & Engineering, University of Washington, Seattle, USA

TABLE I: Comparison of constrained grasping datasets.

	Task-agnostic constraints	Number of objects	Number of grasps	Grasp representation	Input modality
Contact DB [9]	✗	50	3750	Contact map	RGB-D + Thermal
SG14000 [8]	✗	44	14K	SE(3)	RGB-D
TaskGrasp [7]	✗	191	250K	SE(3)	Point cloud
TOG-Net [10]	✗	18K	1.5M	SE(2)	D
CONG (Ours)	✓	2889	14M	SE(3)	Point cloud

II. Related Work

To contextualize our work, we next review constrained grasp sampling approaches. Thereafter, we review grasping datasets for training both constrained and unconstrained data-driven grasp samplers.

A. Constrained Grasp Sampling

Early research in constrained grasping focused on analytically identifying the most appropriate task-specific grasps from already sampled grasps [11], [12]. The central idea in those works was to formulate task-specific grasp quality metrics as optimization problems. Although the quality metrics are theoretically justifiable, calculating them requires known object models and already sampled grasps.

More recent works [6], [7], [10], [13]–[16] propose data-driven methods for learning task-specific grasping to circumvent the issues with analytical methods. In one of the earliest data-driven works, Song et al. [14], [15] proposed a probabilistic framework based on Bayesian networks for learning task constraints from object-action features. Another approach in [10] explored learning a classifier to determine whether a given grasp can fulfill a task.

The problem of constrained grasping can also be split into two parts: one model to detect object affordances or semantics, and another that suggests grasps on these detections [6]–[8], [13]. For instance, in [6], [8], [13], the object affordances were first detected and then used as constraints when stochastically searching for task-specific grasps. As the iterative stochastic search process can be time-consuming, Murali et al. [7] proposed training a generative model to predict multiple grasps on the target object directly and then using another model to rank the proposals according to their ability to fulfill a given task.

In this work, we draw inspiration from [7], but instead of proposing grasps all over the target object, we learn a model that only proposes grasps on specific target areas. These areas can represent, for instance, semantically meaningful locations on the target object, such as the handle of a cup or the bottle cap, but can also cover the entire object. As such, the proposed method can generate affordance or task-oriented grasps but is not restricted to that.

B. Grasping Datasets

Due to the immense popularity and good performance of data-driven grasping methods, many different grasping datasets have been curated to train and evaluate such methods [1], [3], [7], [9], [17]–[24]. These datasets differ in multiple aspects, including input modality: structured [1], [9], [17], [19]–[21], [23], [24] vs unstructured [3], [7], [18], [22]; grasp type: planar [1], [20], [21], [23], [24] vs 6-DoF [3], [7], [9], [17]–[19], [22]; and labels: simulation [3], [7], [18], [21], [22] vs analytic [1], [17], [19], [23] vs human annotated [9], [24].

Despite the abundance of grasping datasets, only a few exist for constrained grasping, as highlighted in Table I. Most of these constrained grasping datasets [7]–[9], are relatively small, including, at most, 191 objects and 250,000 grasps. Moreover, the datasets in [7]–[9] all require humans to create or label the grasps. However, one of the most pressing limitations of all the previously proposed constrained grasping datasets [7]–[9], [17] is that the grasps are conditioned on specific tasks, rendering them unusable for training constrained grasping policies that can generate grasps on *any* constrained area. In this work, we propose a new dataset called CONG, inspired from [7] but did neither require human labeling nor grasps tied to specific tasks. CONG, with over 14 million training examples on 2889 objects, is also orders of magnitude larger compared to [7]–[9].

III. Problem Statement

In this work, we address the problem of generating parallel-jaw grasp poses $\mathbf{G}$ on an object point-cloud $\mathbf{O} \in \mathbb{R}^{\mathrm{N}\times 3}$ such that when the gripper is closed, the grasps are both stable ($\mathrm{S} = 1$) and located at a target area $\mathbf{A} \subseteq \mathbf{O} \in \mathbb{R}^{\mathrm{M}\times 3}$. Here, N and M, where $\mathrm{M} \leq \mathrm{N}$, represent the number of points in a point cloud. As presented in Fig. 2, a grasp $\mathbf{G}$ is located at a target area $\mathbf{A}$ iff the grasp center point is at most at a Euclidean distance d from any point in $\mathbf{O}$.

Specifically, the target is to learn the joint distribution $\mathsf{P}(\mathbf{G}, \mathrm{S} \mid \mathbf{O}, \mathbf{A})$. To learn such a complex joint distribution, we propose factorizing it into two separate distributions: 1) a generative grasp sampler $\mathsf{P}(\mathbf{G} \mid \mathbf{O}, \mathbf{A})$ from which constrained grasps can be sampled, and 2) a grasp evaluator $\mathsf{P}(\mathrm{S} \mid \mathbf{O}, \mathbf{G})$ for evaluating the stability of each grasp $\mathbf{G}$. We approximate these distributions with parametric models $\mathcal{Q}_\theta(\mathbf{G} \mid \mathbf{O}, \mathbf{A}) \approx \mathsf{P}(\mathbf{G} \mid \mathbf{O}, \mathbf{A})$ and $\mathcal{M}_\phi(\mathrm{S} \mid \mathbf{O}, \mathbf{G}) \approx \mathsf{P}(\mathrm{S} \mid \mathbf{O}, \mathbf{G})$, where $\boldsymbol{\theta}$ and $\boldsymbol{\phi}$ are trainable parameters.

We represent a grasp $\mathbf{G} = [\mathbf{q},\ \mathbf{p}]$ by a unit quaternion $\mathbf{q} \in \mathbb{R}^4$ and a 3-D position $\mathbf{p} \in \mathbb{R}^3$. Because we use quaternions, the grasp pose is represented by 7 scalars. We assume that

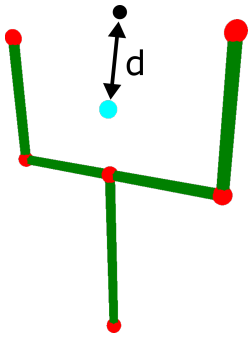


Fig. 2: The grasp in green, its point cloud representation in red, the center grasp point in cyan, and the distance d between it and the black point on the object. The center grasp point is set to the average of the two leftmost and the two rightmost points of the gripper.

all sampled grasps are reachable and that only one graspable object is present.

Solving the aforementioned problem requires the following: (i) the parametric models $\mathcal{Q}_\theta$ and $\mathcal{M}_\phi$ capable of approximating their target distributions, and (ii) a dataset for training the parametric models. In the next section, the parametric models are presented, and in Section V, the dataset used for training them is described.

IV. Method

In the previous section, the problem of learning to sample stable constrained grasps was separated into learning a parametric grasp sampler $\mathcal{Q}_{\boldsymbol{\theta}}$ and a parametric grasp evaluator $\mathcal{M}_{\boldsymbol{\phi}}$. In this section, we first present the constrained grasp sampler (Section IV-A) and then the grasp evaluator (Section IV-B).

A. Variational Constrained Grasp Sampler

We model the parametric grasp sampler $\mathcal{Q}_\theta(\mathbf{G} \mid \mathbf{O}, \mathbf{A})$ using a Conditional Variational Autoencoder (CVAE) [25], where the conditional variables are $\mathbf{O}$ and $\mathbf{A}$. The parametric grasp sampler, henceforth referred to as VCGS, is the decoder $\mathsf{p}_{\boldsymbol{\chi}}(\mathbf{G} \mid \mathbf{O}, \mathbf{A}, \mathbf{z})$ of the CVAE. The corresponding encoder $\mathsf{q}_{\boldsymbol{\psi}}(\mathbf{z} \mid \mathbf{O}, \mathbf{A}, \mathbf{G})$ is only used to train the encoder-decoder network using examples of high-quality grasps. $\boldsymbol{\psi}$ and $\boldsymbol{\chi}$ represent the trainable parameters and $\mathbf{z} \in \mathbb{R}^{\mathrm{L}}$ is a latent space variable of size L.

The backbone for both the encoder and decoder is PointNet++ [26]. Because of this choice, the network input has to be in the form of a point cloud $\mathbf{X} \in \mathbb{R}^{\mathrm{N}\times(3+\mathrm{K})}$, where each point $\mathbf{x} \in \mathbf{X}$ is represented by its 3D Euclidean position and, optionally, K additional real-valued or binary features. For both the encoder and decoder, $\mathbf{X}$ is the same as $\mathbf{O}$ but with an additional point-wise binary feature indicating if the point $\mathbf{x} \in \mathbf{X}$ belongs to the target area $\mathbf{A}$ or not. This construction is made possible because, as defined in Section III, $\mathbf{A} \subseteq \mathbf{O}$.

In addition to the extra binary input feature, the encoder also takes $\mathbf{g}$ as a point-wise feature. The input dimension then becomes $\mathrm{N} \times 11$, where N is the number of points in the point cloud, and the eleven features consist of the 3D position of each point, the binary feature indicating if the point belongs to the target area or not, and the 7-dimensional grasp pose representation. The decoder, on the other hand, does not include $\mathbf{g}$ but does include the latent space variable $\mathbf{z} \in \mathbb{R}^{\mathrm{L}}$ as an additional point-wise feature. Therefore, the decoder input dimension is $\mathrm{N} \times (4 + \mathrm{L})$, where L is the dimension of the latent space. In this work, the size of the latent space was set to 2 in accordance with prior work [3].

VCGS is trained on the standard Evidence Lower Bound (ELBO) loss:

$$\mathcal{L}_{\text{VAE}} = \mathcal{L}(\mathbf{G}^*, \hat{\mathbf{G}}) + \alpha \mathcal{D}_{\text{KL}}[\mathsf{q}_\psi(\mathbf{z} \mid \mathbf{X}, \mathbf{G}^*),\ \mathcal{N}(\mathbf{0}, \mathbf{I})], \quad (1)$$

where α is a scalar, and $\mathcal{D}_{\text{KL}}$ is the KL-divergence between the latent space encoding $\mathbf{z}$ produced by q_ψ, and a zero-mean Gaussian distribution. The reconstruction loss $\mathcal{L}$ in (1) is defined as

$$\mathcal{L}(\mathbf{G}^*, \hat{\mathbf{G}}) = \left\| \mathrm{h}(\mathbf{G}^*) - \mathrm{h}(\hat{\mathbf{G}}) \right\|_1, \quad (2)$$

where $\mathbf{G}^*$ is a ground truth stable grasp, $\hat{\mathbf{G}}$ the generated grasp from the decoder p_χ, and $\mathrm{h} : \mathbb{R}^7 \to \mathbb{R}^{6\times 3}$ is a function that maps a 7D grasp pose into a point cloud representation of the gripper $\mathbf{P} \in \mathbb{R}^{6\times 3}$ as visualized in Fig. 2. The reason for mapping grasp poses to point clouds is that it combines both the translation and orientation components into a single loss function [3].

Both the encoder and the decoder are optimized during training, while only the decoder is used for grasp sampling. More specifically, to sample a set of grasps $\hat{\mathbf{G}}$, the first step is to draw multiple random latent samples $\mathbf{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$. Then, each of these samples is concatenated to a separate copy of the input point cloud together with the binary feature representing the target area. Finally, each copy of the point cloud is passed through the decoder, producing a separate grasp.

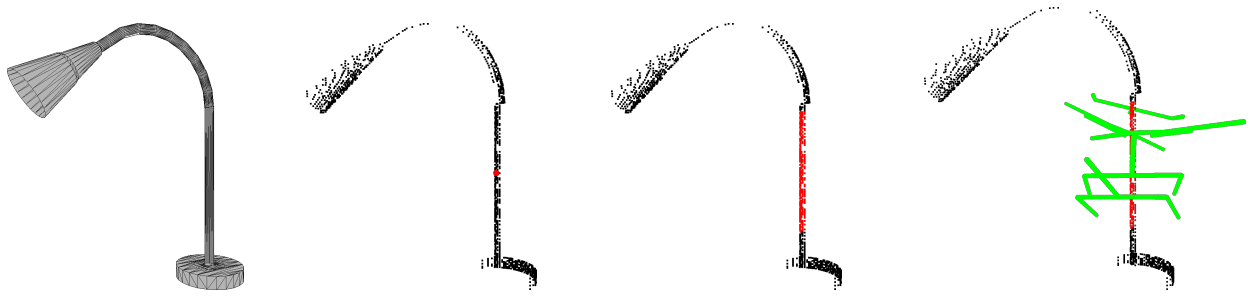
B. Grasp Evaluator

Because the constrained grasp sampler is only trained on stable grasps, it can learn to generate poor grasps between modes [3]. To avoid executing poor grasps, we train a grasp evaluator to distinguish between good and bad grasps by predicting the probability that a grasp $\mathbf{G}$ succeeds ($\mathrm{S} = 1$) on object $\mathbf{O}$. Formally, the grasp evaluator models the conditional probability $\mathsf{P}(\mathrm{S} \mid \mathbf{G}, \mathbf{O})$.

The grasp evaluator used in this work was originally proposed in [3]. It is formed as a deep network that uses the PointNet++ architecture [26] as the backbone. Therefore, the input to the evaluator network is also a point cloud $\mathbf{Y} \in \mathbb{R}^{(\mathrm{N}+6)\times(3+1)}$. However, in contrast to the input point cloud to VCGS, $\mathbf{Y}$ consists of an object point cloud $\mathbf{O} \in \mathbb{R}^{\mathrm{N}\times 3}$ concatenated with a grasp pose point cloud $\mathbf{P} \in \mathbb{R}^{6\times 3}$, and an additional point-wise binary feature to distinguish between these two point clouds.

The grasp evaluator network is trained on the binary cross-entropy loss

$$\mathcal{L}_E = -\mathrm{S}^* \log(\mathrm{S}) + (1 - \mathrm{S}^*) \log(1 - \mathrm{S}), \quad (3)$$



(a) (b) (c) (d)

Fig. 3: An example of how the dataset is curated. (a) From the object mesh, (b) a point cloud is rendered, and a query point, highlighted in red, is selected. Given the query point, (c) all neighbors within a specific radius from it are found, and (d) the grasps close to those points are stored.

where S^* is the ground-truth success of a grasp and S is the predicted success.

V. Dataset

To train VCGS, we need a large-scale dataset consisting of object point clouds $\mathbf{O}$, and successful grasps $\mathbf{G}^*$ on randomly sampled target areas $\mathbf{A}$. To curate such a dataset, we expand the recently large-scale Acronym dataset [18] to include randomly subsampled grasping areas. We name the new dataset CONG.

An overview of the process to curate CONG is presented in Fig. 3. Formally, the process is divided into four steps:

(i) Place the object at the origin in a randomized orientation and render a point cloud $\mathbf{O} \in \mathbb{R}^{\mathrm{N}\times 3}$ from it.
(ii) Sample $\mathbf{I} \in \mathbb{R}^{\mathrm{K}\times 3}$ query points from $\mathbf{O}$, where $\mathrm{K} \ll \mathrm{N}$, using the Farthest Point Sampling (FPS) algorithm.
(iii) For each query point $\mathbf{x}_i \in \mathbf{I}$, find all neighboring points $\mathbf{A}_i$ that are within a uniformly sampled radius $r_i \sim \mathcal{U}[0,\ \mathrm{R}]$ from $\mathbf{x}_i$, where R is the diagonal length of the mesh's bounding box.
(iv) Find all grasps $\mathbf{G}$ for object $\mathbf{O}$ in the Acronym dataset [18] where the distance between the center grasp point and any point in $\mathbf{A}_i$ is at most d.

For the steps above, we defined $\mathrm{N} = 1024$, $\mathrm{K} = 50$, and $\mathrm{d} = 2\,\mathrm{cm}$. The center grasp point is defined as in Fig. 2.

We ran the above procedure on 2889 objects from the Acronym dataset [18]. For each of the 2889 objects, we rendered 100 point clouds $\mathbf{O}$ of the object, and for each of these point clouds, we sampled $\mathbf{I}$. The resulting dataset contains over 14 million examples with an average of 257 grasps per target area $\mathbf{A}_i$. Of this dataset, 123 objects were randomly selected for the simulated grasping experiment, and the rest were used for training.

VI. Experiments

The two questions we want to answer in the experiments are:

1) What is the grasp success rate of constrained grasping?
2) How much more sample efficient is a constrained grasp sampler than an unconstrained one for target-driven grasping?

We answer these two questions with two experiments: one in simulation and one using real robotic hardware. In all experiments, VCGS was benchmarked against GraspNet [3]. Both methods were trained on the same objects from Acronym [18] to ensure a just comparison.

To evaluate grasps, we used the grasp success rate metric, which is the ratio of successful grasps to total grasp attempts. We counted a grasp as successful if the object was successfully picked up and remained within the gripper during a predefined manipulation motion. To only test grasps on the target area, all those whose center grasping point was further away than a distance d to any point in the target area were removed. As in Section V, we set $d = 2\,\mathrm{cm}$.

A. Simulated Robotic Grasping

In the simulation experiments, we wanted to test the best grasp, not the best reachable one. Therefore, to ensure all sampled grasps were reachable, we used a free-floating Franka Emika Panda gripper to grasp a free-floating object, as depicted in Fig. 4. To grasp an object, an open Franka Emika Panda gripper was placed at the grasp pose, and then the finger closed slowly until either the object was grasped or the two fingers touched. If the object was between the fingers of the gripper, the grasp was evaluated by turning gravity on and executing a predefined linear acceleration motion followed by an angular acceleration motion. The grasp was successful if the object remained within the gripper during all motions.

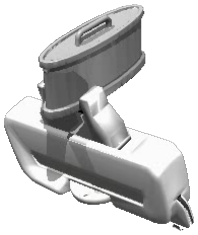


Fig. 4: An example grasp from the simulation.

The simulation experiments were carried out using the publicly available Isaac Gym simulator [27] on 123 randomly held out objects from the Acronym dataset [18]. To observe the objects, we used a simulated depth sensor.

Two different simulation experiments were conducted to determine the effect constrained grasp sampling had on grasp success rates. In the first experiment, called *Unconstrained sampling*, grasps around the objects were sampled, and no

TABLE II: Simulation results when evaluating the 10 highest scoring grasps. In cases where less than 10 grasps were kept, all were evaluated. ↑: higher the better, ↓: lower the better.

	Unconstrained sampling						Constrained sampling									
	VCGS		GraspNet				VCGS		GraspNet				GraspNetTaI			
# of grasps sampled	100	500	100	500	5K	10K	100	500	100	500	5K	10K	100	500	5K	10K
Grasp success rate (%) ↑	**56**	**56**	45	45	47	45	53	**54**	39	43	46	46	34	33	33	34
Ratio of grasps kept (%) ↑	100	100	100	100	100	100	**93**	**93**	30	30	30	30	34	34	34	34
Inference time (s) ↓	**0.07**	0.29	**0.07**	0.31	5.63	10.27	0.07	0.27	0.07	0.27	3.9	8.55	**0.06**	0.27	3.93	8.11

specific target area was set. To sample unconstrained grasps with VCGS, all the points were set as the target area, *i.e.*, $\mathbf{A} = \mathbf{O}$. In the second simulation experiment, called *Constrained sampling*, only grasps at the target area were allowed. We also included an additional baseline in the second experiment called GraspNet Target as Input (TaI) that used only the target area as input to the grasp sampling network.

To analyze the effect constrained sampling had on sample efficiency, we varied the number of grasps sampled. For VCGS, we sampled 100 and 500 grasps per object or target area, while for GraspNet, we sampled 100, 500, 5000, and 10000 grasps. Out of these grasps, we only executed the 10 highest-scoring grasps according to the grasp evaluator. The same procedure as in Section V was used for generating the target areas. That is, from the rendered point cloud, we first sampled 10 query points using FPS. Then, for each query point, a target area was constructed by finding all neighboring points within a uniformly sampled radius from the query point. The bounds on the uniform distribution were 0 and R, where R is the diagonal length of the mesh's bounding box.

The experimental results are presented in Table II, and an example grasp is shown in Fig. 4. Based on these results, we can draw multiple conclusions. First, for constrained grasping, VCGS kept over three times more grasps than GraspNet, demonstrating the benefit of constraining grasp sampling already in the input to the network. Moreover, GraspNet TaI did not generate more successful grasps than GraspNet, highlighting the benefit of using global object information even when sampling grasps locally at specific target regions.

Secondly, for GraspNet, the number of sampled grasps mainly affects the grasp success rate for constrained sampling, but the effect tapers with the number of grasps. We hypothesize that this finding demonstrates that an unconstrained grasp sampler must sample orders of magnitude more grasps to increase the probability that some good ones end up in the target area. This hypothesis is also supported by the fact that for the same experiment, the ratio of grasps kept for GraspNet is the same no matter if 100, 500, 5000, or 10000 grasps were sampled. However, by sampling orders of magnitude more grasps, as in the case of 5000 or 10000 grasps, the inference time increases by 80–140 times compared to only sampling 100 grasps.

The last conclusion to draw is that VCGS achieves the highest grasp success rates in both constrained and, more interestingly, unconstrained grasping. We hypothesize that the lower unconstrained grasp success rate for GraspNet is because it generates many more grasps on the unobserved part of the object than VCGS. The reason for generating grasps on unobserved parts is that these could admit better grasps than the observed parts. However, the grasp success prediction for such grasps is also less reliable and could lead to more misclassified grasps.

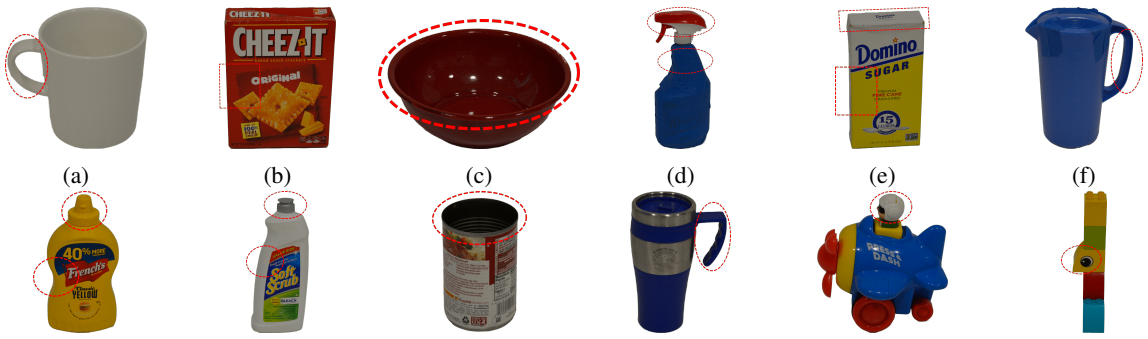
All in all, the results from the simulation experiment highlight the benefits of constraining grasp sampling. Next, we investigate if similar benefits are present in real-world robotic grasping.

B. Real Robotic Grasping

In the real robotic experiments, we explored if VCGS can achieve a higher grasp success rate and be more sample efficient than GraspNet when grasping real-world objects. To this end, we used the setup shown in Fig. 1 that included a Franka Emika Panda robot to execute the grasps, a Kinect 2.0 to capture the point clouds, and an Aruco marker [29] for extrinsic calibration.

As the simulation experiments demonstrated that GraspNet performs better than GraspNet TaI, we only evaluated GraspNet and VCGS, both of which were trained on synthetic data only, on the 12 different objects and the 16 different target areas presented in Fig. 5. Most of these areas were semantically meaningful and included, for instance, object handles (Fig. 5a, Fig. 5f, and Fig. 5j), rims (Fig. 5c and Fig. 5i), and caps (Fig. 5g and Fig. 5h). Each object was placed at a predefined position (see Fig. 1) but in two different orientations toward the camera: at $0°$ and $90°$. In total, this setup amounted to 32 grasp trials per method. A grasp was successful if the robot picked up the object and moved it to the start pose without dropping it.

To further explore the effect sample size has on grasp success rates, we only sampled 10 grasps at a time per target area. For each batch of 10 grasps, only grasps for which the center grasp point was, at most, 2 cm away from any point in the target area were kept. Each kept grasp was then scored using the evaluation network, and the highest-



(a) (b) (c) (d) (e) (f)

(g) (h) (i) (j) (k) (l)

Fig. 5: The 10 objects used in the real-world experiment. All objects, except (a), (j), (k), and (l), are from the YCB object dataset [28]. The dashed red lines depict the target grasping area for each object.

scoring reachable grasp was executed. If all grasps were removed or no reachable ones were found, another 10 grasps were sampled. The resampling process was repeated until a maximum of 50 grasps had been sampled. If none of the 50 grasps was executed, we terminated the trial and considered the grasp as unsuccessful. Based on this process, the theoretically least number of grasps to be sampled for all objects and target areas together was 320.

The results are presented in Table III, and an example grasp is shown in Fig. 1. These results demonstrate that VCGS achieves a 15% higher grasp success rate and is 1.8 times more sample efficiency than GraspNet. Both results are also statistically significant. Furthermore, VCGS only had to sample 40 more grasps than the theoretical lower bound of 320 grasps.

TABLE III: The experimental results along with the test statistics and p-values of a pair-wise one-sided Wilcoxon signed-rank test. ↑: higher the better, ↓: lower the better.

	GraspNet	VCGS	VCGS vs GraspNet
Grasp success rate (%) ↑	59.4	75.0	T=10.0, p< .05*
# of grasps sampled ↓	660	360	T=148.5, p< .001**

VII. Limitations

In the experimental evaluation, we identified two limitations. The first limitation is that only position constraints can be enforced, while others, such as orientation or reachability constraints, cannot. Although the network did learn implicitly from data which orientations are meaningful to grasp an object successfully, in some scenarios, it would be helpful also explicitly to constrain the orientation of the grasps. As an example, imagine grasping a USB cable for insertion. For this type of object and task, it would make sense to constrain the grasp orientation to not obscure the part of the USB that will be inserted. It would also be helpful to incorporate the robot's reachability as another constraint, as this could further increase sample efficiency by only suggesting reachable areas to grasp.

The second limitation is refining sampled grasps using the grasp evaluator as was explored in [3]. In that work, sampled grasps were locally refined by moving them in a direction that improved the success probability measured by the grasp evaluator. Unfortunately, as noticed during the experimental evaluation of [3], such a refining process often moves grasps outside the target area. A possible solution to mitigate this issue would also be to condition the grasp evaluator on the target area, *i.e.*, $\mathsf{P}(\mathsf{S} \mid \mathbf{G}, \mathbf{O}, \mathbf{A})$, or to constrain the magnitude of the refinement not to leave the target area.

VIII. Conclusion

We presented VCGS, a generative method for constrained 6-DoF grasps sampling, and CONG, a new dataset for learning and evaluating constrained grasp samplers. Constrained grasping has so far been restricted to finding task- or affordance-specific grasps. VCGS is instead structured to find grasps on any target area whether the area carries task or affordance semantics. The key idea to achieve such general constrained grasping capabilities was to embed the target area as input features for the network and train the network on a dataset that included random target areas of varying sizes. We compared VCGS to GraspNet, a state-of-the-art generative unconstrained 6-DoF grasp sampler, in simulation and the real world. The results demonstrate that VCGS achieves a 10–15% higher grasp success rate and is 2–3 times more sample efficient than GraspNet.

All in all, the work presented here extends constraint handling in modern neural grasping to arbitrary contact location constraints compared to existing works that constrain grasps primarily from the task compatibility viewpoint. This extension is beneficial as most real-life manipulation tasks pose some constraints on the grasps sampled. However, most tasks require a complex set of constraints to be satisfied. Thus, an important avenue for further research is inferring the constraints from a complex task.

Acknowledgment

We gratefully acknowledge the support of NVIDIA Corporation with the donation of the Titan Xp GPU used for this research.

REFERENCES

[1] J. Mahler, J. Liang, S. Niyaz, M. Laskey, R. Doan, X. Liu, J. Aparicio, and K. Goldberg, "Dex-Net 2.0: Deep Learning to Plan Robust Grasps with Synthetic Point Clouds and Analytic Grasp Metrics," in *Robotics: Science and Systems XIII*. Robotics: Science and Systems Foundation, Jul. 2017.

[2] D. Morrison, J. Leitner, and P. Corke, "Closing the Loop for Robotic Grasping: A Real-time, Generative Grasp Synthesis Approach," in *Robotics: Science and Systems XIV*. Robotics: Science and Systems Foundation, Jun. 2018.

[3] A. Mousavian, C. Eppner, and D. Fox, "6-DOF GraspNet: Variational Grasp Generation for Object Manipulation," in *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*. Seoul, Korea (South): IEEE, Oct. 2019, pp. 2901–2910.

[4] M. Sundermeyer, A. Mousavian, R. Triebel, and D. Fox, "Contact-GraspNet: Efficient 6-DoF Grasp Generation in Cluttered Scenes," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*, May 2021, pp. 13 438–13 444.

[5] M. Kokic, D. Kragic, and J. Bohg, "Learning Task-Oriented Grasping From Human Activity Datasets," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 3352–3359, Apr. 2020.

[6] R. Detry, J. Papon, and L. Matthies, "Task-oriented grasping with semantic and geometric scene understanding," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Sep. 2017, pp. 3266–3273.

[7] A. Murali, W. Liu, K. Marino, S. Chernova, and A. Gupta, "Same Object, Different Grasps: Data and Semantic Knowledge for Task-Oriented Grasping," in *Proceedings of the 2020 Conference on Robot Learning*. PMLR, Oct. 2021, pp. 1540–1557.

[8] W. Liu, A. Daruna, and S. Chernova, "Cage: Context-aware grasping engine," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2020, pp. 2550–2556.

[9] S. Brahmbhatt, C. Ham, C. C. Kemp, and J. Hays, "ContactDB: Analyzing and Predicting Grasp Contact via Thermal Imaging," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 8709–8719.

[10] K. Fang, Y. Zhu, A. Garg, A. Kurenkov, V. Mehta, L. Fei-Fei, and S. Savarese, "Learning task-oriented grasping for tool manipulation from simulated self-supervision," *The International Journal of Robotics Research*, vol. 39, no. 2-3, pp. 202–216, Mar. 2020.

[11] C. Borst, M. Fischer, and G. Hirzinger, "Grasp planning: How to choose a suitable task wrench space," in *IEEE International Conference on Robotics and Automation, 2004. Proceedings. ICRA '04. 2004*, vol. 1, Apr. 2004, pp. 319–325 Vol.1.

[12] R. Haschke, J. Steil, I. Steuwer, and H. Ritter, "Task-oriented quality measures for dextrous grasping," in *2005 International Symposium on Computational Intelligence in Robotics and Automation*, Jun. 2005, pp. 689–694.

[13] M. Kokic, J. A. Stork, J. A. Haustein, and D. Kragic, "Affordance detection for task-specific grasping using deep learning," in *2017 IEEE-RAS 17th International Conference on Humanoid Robotics (Humanoids)*, Nov. 2017, pp. 91–98.

[14] D. Song, K. Huebner, V. Kyrki, and D. Kragic, "Learning task constraints for robot grasping using graphical models," in *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Oct. 2010, pp. 1579–1585.

[15] D. Song, C. H. Ek, K. Huebner, and D. Kragic, "Task-Based Robot Grasp Planning Using Probabilistic Inference," *IEEE Transactions on Robotics*, vol. 31, no. 3, pp. 546–561, Jun. 2015.

[16] R. Antonova, M. Kokic, J. A. Stork, and D. Kragic, "Global Search with Bernoulli Alternation Kernel for Task-oriented Grasping Informed by Simulation," in *Proceedings of The 2nd Conference on Robot Learning*. PMLR, Oct. 2018, pp. 641–650.

[17] H.-S. Fang, C. Wang, M. Gou, and C. Lu, "GraspNet-1Billion: A Large-Scale Benchmark for General Object Grasping," in *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. Seattle, WA, USA: IEEE, Jun. 2020, pp. 11 441–11 450.

[18] C. Eppner, A. Mousavian, and D. Fox, "ACRONYM: A Large-Scale Grasp Dataset Based on Simulation," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*, May 2021, pp. 6222–6227.

[19] J. Lundell, E. Corona, T. Nguyen Le, F. Verdoja, P. Weinzaepfel, G. Rogez, F. Moreno-Noguer, and V. Kyrki, "Multi-FinGAN: Generative Coarse-To-Fine Sampling of Multi-Finger Grasps," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*, May 2021, pp. 4495–4501.

[20] A. Depierre, E. Dellandréa, and L. Chen, "Jacquard: A Large Scale Dataset for Robotic Grasp Detection," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Oct. 2018, pp. 3511–3516.

[21] T. N. Le, J. Lundell, F. J. Abu-Dakka, and V. Kyrki, "Deformation-Aware Data-Driven Grasp Synthesis," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 3038–3045, Apr. 2022.

[22] C. Eppner, A. Mousavian, and D. Fox, "A billion ways to grasps - an evaluation of grasp sampling schemes on a dense, physics-based grasp data set," in *Proceedings of the International Symposium on Robotics Research (ISRR)*, Hanoi, Vietnam, 2019.

[23] D. Morrison, P. Corke, and J. Leitner, "EGAD! An Evolved Grasping Analysis Dataset for Diversity and Reproducibility in Robotic Manipulation," *IEEE Robotics and Automation Letters*, vol. 5, no. 3, pp. 4368–4375, Jul. 2020.

[24] I. Lenz, H. Lee, and A. Saxena, "Deep learning for detecting robotic grasps," *The International Journal of Robotics Research*, vol. 34, no. 4-5, pp. 705–724, Apr. 2015.

[25] K. Sohn, H. Lee, and X. Yan, "Learning Structured Output Representation using Deep Conditional Generative Models," in *Advances in Neural Information Processing Systems*, vol. 28. Curran Associates, Inc., 2015.

[26] C. R. Qi, L. Yi, H. Su, and L. J. Guibas, "PointNet++: Deep Hierarchical Feature Learning on Point Sets in a Metric Space," in *Advances in Neural Information Processing Systems*, vol. 30. Curran Associates, Inc., 2017.

[27] V. Makoviychuk, L. Wawrzyniak, Y. Guo, M. Lu, K. Storey, M. Macklin, D. Hoeller, N. Rudin, A. Allshire, A. Handa, and G. State, "Isaac Gym: High Performance GPU-Based Physics Simulation For Robot Learning," Aug. 2021.

[28] B. Calli, A. Singh, A. Walsman, S. Srinivasa, P. Abbeel, and A. M. Dollar, "The YCB object and Model set: Towards common benchmarks for manipulation research," in *2015 International Conference on Advanced Robotics (ICAR)*, Jul. 2015, pp. 510–517.

[29] S. Garrido-Jurado, R. Muñoz-Salinas, F. J. Madrid-Cuevas, and M. J. Marín-Jiménez, "Automatic generation and detection of highly reliable fiducial markers under occlusion," *Pattern Recognition*, vol. 47, no. 6, pp. 2280–2292, Jun. 2014.