Refining 6-DoF Grasps with Context-Specific Classifiers

Tasbolat Taunyazov[1], Heng Zhang[1], John Patrick Eala[1], Na Zhao[2], and Harold Soh[1,3]

Abstract— In this work, we present GraspFlow, a refinement approach for generating context-specific grasps. We formulate the problem of grasp synthesis as a sampling problem: we seek to sample from a context-conditioned probability distribution of successful grasps. However, this target distribution is unknown. As a solution, we devise a discriminator gradient-flow method to evolve grasps obtained from a simpler distribution in a manner that mimics sampling from the desired target distribution. Unlike existing approaches, GraspFlow is modular, allowing grasps that satisfy multiple criteria to be obtained simply by incorporating the relevant discriminators. It is also simple to implement, requiring minimal code given existing auto-differentiation libraries and suitable discriminators. Experiments show that GraspFlow generates stable and executable grasps on a real-world Panda robot for a diverse range of objects. In particular, in 60 trials on 20 different household objects, the first attempted grasp was successful 94% of the time, and 100% grasp success was achieved by the second grasp. Moreover, incorporating a functional discriminator for robot-human handover improved the functional aspect of the grasp by up to 33%.

I. INTRODUCTION

The right way to grasp an object is context-specific. It depends not only on the object being grasped, but also on the characteristics and state of the robot executing the grasp — certain grasp configurations (and subsequent lifting/manipulation) are achievable on some robots, yet not on others. In addition, grasps should be functional; a robot grasp that works well for a pick-and-place operation may be inappropriate for human-robot handover. How robots can synthesize grasps that satisfy such diverse context-dependent quality criteria remains a fundamental challenge in robotics.

In this work, we adopt a new perspective on the problem of grasp synthesis and treat the problem as one of sampling: we seek to sample quality grasps from a context-dependent target distribution p. The key problem is that we don't have access to p and it is unknown. We propose a method that *evolves*, or *refines*, grasps generated from a simpler base distribution q_0 (e.g., a deep generative model [1]) such that the grasps appear to be sampled from p. Our approach is based on theory of discriminator gradient flows [2]; we leverage a Fokker-Planck equation that represents the gradient flow — the steepest descent curve in the space of probability

[1]Authors are with the Dept. of Computer Science, National University of Singapore. Correspondence to Tasbolat Taunyazov `tasbolat@comp.nus.edu.sg` or Harold Soh `harold@comp.nus.edu.sg`

[2]Author is with Singapore University of Technology and Design. Email: `na_zhao@sutd.edu.sg`

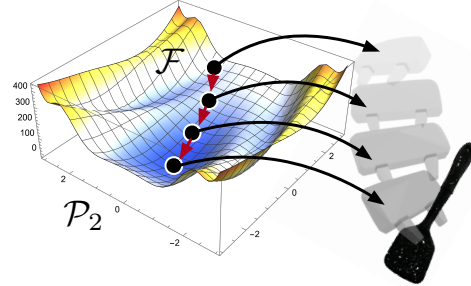[3]Smart Systems Institute, National University of Singapore

Fig. 1: Illustration of GRASPFLOW. Grasps shown are samples from probability distributions (black circles) along the gradient flow from the initial distribution q_0 to a target distribution p.

measures — obtained from minimizing a regularized f-divergence functional between q_0 and p (Fig. 1).

In this setup, the density ratio q_0/p emerges as the crucial quantity to be estimated. We propose an approximation using discriminators/classifiers trained (or designed) to evaluate grasp quality criteria. This enables a modular approach towards grasp synthesis; different discriminators can be applied to suit specific contexts. Moreover, while classifiers have become strongly associated with deep learning, our method does not specifically rely on neural methods. Practitioners are free to select classifiers which are best for the task (even handcrafted models), as long as they are differentiable. Our approach, which we call GRASPFLOW, is straightforward to implement, requiring only a few lines of code given modern auto-differentiation libraries and existing discriminators.

Experiments show that GRASPFLOW is able to generate stable and executable 6DOF grasps on a Franka-Emika Panda robot. In our first experiment, GRASPFLOW significantly improves grasps sampled from the GraspNet VAE [1] on a set of 20 different household objects (1800 grasps); success percentages rose from 42% to 82% after sample refinement. Note that this was achieved *without* having to discard grasps that violate robot kinematics as in prior work (e.g., [1]). In a second experiment involving 120 grasps, we demonstrate that successful *and* functional grasps can be obtained when a simple handover discriminator was incorporated into the setup; on two objects (a hammer and spatula), the proportion of functionally-appropriate grasping increased from 40% to 68%. In both experiments, a successful grasp was obtained within the first *two* attempts.

Relation to Prior Work. GRASPFLOW pertains to the study of robot grasping, which has a rich history and encompasses a variety of approaches [3]–[6]. Recent work has focused largely on data-driven methods using deep learning to generate

suitable grasps from low-level observations (e.g., RGB-D or point clouds) and derive models that generalize to novel objects [7]–[12]. For example, DexNet 2.0 [9] employs a trained convolutional neural network to rank and filter potential planar grasps. The closest related body of work to ours involve methods that synthesize (or refine) grasps via optimization of learnt functions. Compared to end-to-end deep models, these methods separate out the optimization/inference process from grasp evaluation. As such, they can generate a diverse set of grasps for a given object and filter out those that do not meet selected criteria; in contrast, single-stage end-to-end models typically have to be completely retrained for new quality criteria. Early work by [13] optimized a 6DOF grasp quality function via gradient descent and quasi-Newton methods. Later methods [14]–[16] viewed grasp synthesis/planning as probabilistic inference — [14] proposed to conduct maximum likelihood estimation of the grasp by optimizing the likelihood of grasp success conditioned upon the grasp pose. This probabilistic approach was later extended to incorporate priors, yielding a maximum a posteriori solution [15], [16]. GraspNet [1] refines grasps sampled from a variational autoencoder (VAE) by optimizing a quality function (similar in spirit to [13]), while later variants [17], [18] performed Monte-Carlo sampling using the discriminator to compute the acceptance ratio.

GRASPFLOW can be seen as unifying the sampling, probabilistic, and optimization approaches towards grasp synthesis. As such, it inherits many of the advantages above, yet resolves issues associated with each individual approach. GRASPFLOW's basic formulation is one of sampling, which enables the incorporation of uncertainty and provides a diversity of grasp candidates in proportion to their estimated probability of success — this can be difficult to achieve in a pure optimization framework. However, a standard sampling-based approach can be wasteful since many potential grasps are filtered away. Rather than discard samples, GRASPFLOW *refines* samples via a scheme derived by optimizing the sampling distribution; this method enables us to sample from a desired target distribution of successful grasps, which remains implicit in the setup. To our knowledge, GRASPFLOW is the first work to optimize for multiple criteria in such a manner.

II. GRASPFLOW: GRASP REFINEMENT VIA DISCRIMINATOR GRADIENT FLOW

In this section, we describe GRASPFLOW, a framework for generating grasp candidates that satisfy desired criteria. At a high-level, we seek to sample grasps $\mathbf{g}$ from a conditional distribution $p(\mathbf{g}|\mathbf{c}, \mathbf{o})$ where $\mathbf{c}$ and $\mathbf{o}$ are the desired quality criteria and robot observation, respectively. Unfortunately, p is generally unknown. Instead, we only have access to a distribution $q_0(\mathbf{g}|\mathbf{c}, \mathbf{o})$ that is easy to sample from. We will develop a gradient-flow formulation that will enable us to *transform* grasps sampled from $q_0(\mathbf{g}|\mathbf{c}, \mathbf{o})$ such that they appear to be coming from $p(\mathbf{g}|\mathbf{c}, \mathbf{o})$. We will begin with a brief introduction to gradient flows[1], followed by

[1]Please see [19] for a more thorough introduction.

a description of how gradient flows can be applied to grasp refinement using multiple classifiers, which enables sampling for a specified context. In this section, we will focus on conveying the main ideas behind our approach and delay discussing implementation details (e.g., grasp representation, the specific classifiers used in our setup) to the next section. Our presentation summarizes Ansari et al. [2], with additional comments regarding key differences when applying discriminator gradient flows to grasping.

Background on Gradient Flows. To provide intuition, we begin with the familiar notion of a Euclidean space with the 2-norm $(\mathcal{X}, \|\cdot\|_2)$. Given a smooth energy function $F : \mathcal{X} \to \mathbb{R}$, the curve $\{\mathbf{g}_t\}_{t \in \mathbb{R}_+}$ that follows the direction of steepest descent and minimizes the energy is called the *gradient flow*, $\mathbf{g}'(t) = -\nabla F(\mathbf{g}(t))$. In this work, we are interested in sampling from a *probability distribution* of successful grasps. Rather than Euclidean spaces, we are interested in steepest descent curves in the metric space of *probability measures*. Specifically, we will examine gradient flows in the 2-Wasserstein space $(\mathcal{P}_2(\Omega), \mathcal{W}_2)$, i.e., the space of probability measures with finite second moments $\mathcal{P}_2(\Omega)$ that is coupled with the Wasserstein metric $\mathcal{W}_p$. Given a functional $\mathcal{F} : \mathcal{P}_2(\Omega) \to \mathbb{R}$ in the 2-Wasserstein space, the gradient flow $\{q_t\}_{t \in \mathbb{R}_+}$ of $\mathcal{F}$ minimizes the value of $\mathcal{F}$. Recently, gradient flows been used in deep generative modeling [2], [20]–[22]. We adopt a similar scheme to [2], who introduced a gradient flow-based technique for refining samples (images and text) using a GAN-based discriminator. Here, we will refine robot grasps, which unlike the samples in [2], are conditionally-generated and have to be refined to satisfy quality criteria specific to robot grasping.

Grasp Refinement via Gradient Flow. We adopt a standard definition of a grasp, i.e., a set of contact points with an object j which restricts movement when external forces are applied [23]. We model $p(\mathbf{g}|j)$ as a uniform distribution over end-effector poses $\mathbf{g}$ that form grasps. In practice, two issues arise: (i) we do not know the object j but have access to an observation $\mathbf{o}$, and (ii) grasps may not satisfy desired criteria $\mathbf{c}$ (e.g., stable, functional). Hence, we seek to sample from the context-dependent distribution $p(\mathbf{g}|\mathbf{c}, \mathbf{o}) \propto p(\mathbf{c}|\mathbf{g}, \mathbf{o})p(\mathbf{g}|\mathbf{o})$ where we have applied Bayes rule and marginalized out the unobserved object, $p(\mathbf{g}|\mathbf{o}) = \sum_j p(\mathbf{g}|j)p(j|\mathbf{o})$. The distribution $p(\mathbf{g}|\mathbf{c}, \mathbf{o})$ is generally difficult (or impossible) to explicitly specify or compute.

We only have access to a distribution $q_0(\mathbf{g}|\mathbf{c}, \mathbf{o})$ that is easy to sample from, but may not generate grasps that fulfill the desired criteria. We refer to the variables $(\mathbf{c}, \mathbf{o})$ as the *context*. Depending on the chosen distribution, q_0 may disregard the context or parts of it. To reduce clutter, we will drop the explicit dependence on $\mathbf{c}$ and $\mathbf{o}$ but note that p is always conditioned upon the context.

Our goal is to obtain grasp candidates from p rather than q_0. To achieve this, we first consider how we can transform q_0 into p. Similar to how we might optimize samples in a Euclidean space by minimizing a function, we minimize the

entropy-regularized f-divergence functional:

$$\mathcal{F}_p^f(q) \triangleq \underbrace{\int f\left(q_0(\mathbf{g})/p(\mathbf{g})\right)p(\mathbf{g})d\mathbf{g}}_{f\text{-divergence } \mathbb{D}_f[p(\mathbf{g})\|q_0(\mathbf{g})]} + \gamma \underbrace{\int q_0(\mathbf{g})\log q_0(\mathbf{g})d\mathbf{g}}_{\text{negative entropy } -\mathcal{H}(q_0(\mathbf{g}))},$$
$$(1)$$

where the f-divergence term $\mathbb{D}_f[p\|q_0]$ captures the "distance" between a density q_0 and our target p. The gradient flow of $\mathcal{F}_p^f(q)$ in the Wasserstein space is given by the Fokker-Planck equation (FPE),

$$\partial_t q_t(\mathbf{g}) = \nabla_{\mathbf{g}} \cdot \left(q_t(\mathbf{g})\nabla_{\mathbf{g}}f'\left(q_t(\mathbf{g})/p(\mathbf{g})\right)\right) + \gamma\Delta_{\mathbf{gg}}q_t(\mathbf{g})$$
$$(2)$$

where f' is the derivative of the chosen f-divergence, and $\nabla_{\mathbf{g}}\cdot$ and $\Delta_{\mathbf{gg}}$ denote the divergence and the Laplace operators, respectively. Eq. (2) gives us the gradient flow of the density q_t as it is transformed from q_0 to p, but working directly with the distributions q_t is difficult. Ideally, we would like to transform *samples*, i.e., the candidate grasps.

From Distributions to Samples. The FPE above has an equivalent Stochastic Differential Equation (SDE) formulation [24]:

$$d\mathbf{g}_t = \underbrace{-\nabla_{\mathbf{g}}f'\left(q_t(\mathbf{g}_t)/p(\mathbf{g}_t)\right)dt}_{\text{drift}} + \underbrace{\sqrt{2\gamma}d\mathbf{w}_t}_{\text{diffusion}}, \quad (3)$$

which describes the evolution of *samples* following the distributions in Eq. (2). This SDE

can be numerically solved using the Euler-Maruyama method [25]:

$$\mathbf{g}_{\tau_{n+1}} = \mathbf{g}_{\tau_n} - \eta\nabla_{\mathbf{g}}f'\left(q_{\tau_n}(\mathbf{g}_{\tau_n})/p(\mathbf{g}_{\tau_n})\right) + \sqrt{2\gamma\eta}\boldsymbol{\xi}_{\tau_n}, \quad (4)$$

where $\boldsymbol{\xi}_{\tau_n} \sim \mathcal{N}(\mathbf{0},\mathbf{I})$. The τ_n are discretized time-steps where we have partitioned the time interval $[0,N]$ into equal intervals of size η. Unlike the FPE, Eq. (4) provides a method for changing grasp candidates drawn from q_0 to grasps from p.

Approximating the Density Ratio with Context Classifiers. To transform the samples, Eq. (4) requires evaluation of the density-ratio $q_{\tau_n}(\mathbf{g}_{\tau_n})/p(\mathbf{g}_{\tau_n})$. [2] approximated this density ratio using the *density-ratio trick* [26], i.e., a differentiable discriminator trained to distinguish between samples from q_0 and samples from p,

$$\frac{q_{\tau_n}(\mathbf{g}_{\tau_n})}{p(\mathbf{g}_{\tau_n})} \approx \frac{1 - p(d = 1|\mathbf{g}_{\tau_n})}{p(d = 1|\mathbf{g}_{\tau_n})} \quad (5)$$

where $d = 1$ is the label given to samples from p (and $d = 0$ for samples from q_0). This type of classifier may be familiar to readers acquainted with GANs; similar classifiers are learned during GAN training to tell apart the real and generated samples. In our setting however, we do *not* have ready access to a classifier of this nature. Indeed, gathering a large number of successful real-world grasps to train such a discriminator for a variety of contexts would be prohibitively expensive.

Instead, we propose a different approximation using a set of classifiers developed for assessing grasp quality. Specifically, we approximate the density ratio,

$$\frac{q_{\tau_n}(\mathbf{g}_{\tau_n})}{p(\mathbf{g}_{\tau_n})} \approx \frac{1 - \prod_i p(c_i = 1|\mathbf{g}_{\tau_n},\mathbf{o})}{\prod_i p(c_i = 1|\mathbf{g}_{\tau_n},\mathbf{o})} \quad (6)$$

This form assumes that the different quality criteria are conditionally independent *given the grasp and observation*, which is not unreasonable if $\mathbf{g}$ and $\mathbf{o}$ already comprise the information necessary to determine whether a given criterion c_i is fulfilled. Also, the classifiers $p(c_i|\mathbf{g},\mathbf{o})$ are not trained to distinguish samples from q and p; an implicit assumption is that q is similar to the distribution of grasps used for training the classifiers and that do not fulfill all the quality criteria. If this property does not hold for a specific sampler q, we can amend the flow using a corrector term. We did not find it necessary to employ this corrector in our experiments, presumably because the grasps generated from many samplers, even state-of-the-art deep generative models, are unlikely to fulfill all desired criteria.

To obtain our final refinement process, we combine (3) and (6) to give,

$$\mathbf{g}_{\tau_{n+1}} = \mathbf{g}_{\tau_n} - \eta\nabla_{\mathbf{g}_{\tau_n}}f'\left(\frac{1 - \prod_i p(c_i = 1|\mathbf{g}_{\tau_n},\mathbf{o})}{\prod_i p(c_i = 1|\mathbf{g}_{\tau_n},\mathbf{o})}\right)$$
$$+ \sqrt{2\gamma\eta}\boldsymbol{\xi}_{\tau_n}, \quad (7)$$

The sample evolution equation above is simple to implement given auto-differentiation libraries and affords a degree of modularity—different classifiers can be selected and combined to jointly refine grasps for a particular application

Summary and Practical Use. In the GRASPFLOW framework, we first choose an base sampler q_0 along with classifiers $p(c_i|\mathbf{g},\mathbf{o})$ representing our desired quality criteria. We also select a f-divergence; in our experience, the KL-divergence $f(r) = \log(r)$ ($f'(r) = \log(r) + 1$)) generally works well, but other divergences can be applied. We then sample a candidate grasp $\mathbf{g}_{\tau_0} \sim q_0$, and then update the grasp using Eq. (7) for N time steps. The hyperparameters η, N, and γ can be tuned for performance. Given that η emerges as part of a numerical solution for the SDE (3), it is generally set to be small (e.g., $10^{-3} - 10^{-5}$). N should be chosen based on the sampler q_0; larger N is needed for samplers that are generate poor grasps (i.e., q_0 is far from p). Finally, γ can be set to $10^{-2} - 10^{-4}$ to allow for diversity in the generation. In our experiments, we found GRASPFLOW to perform similarly within reasonable parameter ranges.

III. GRASPFLOW FOR STABLE, EXECUTABLE, AND FUNCTIONAL 6DOF GRASPS

Thus far, we have described GRASPFLOW in a relatively abstract manner. Here, we will describe how GRASPFLOW can be used to obtain desired 6-DoF grasps for a Franka-Emika Panda arm (Fig. 2). Each hand pose or grasp $\mathbf{g}$ is represented by its rotation and translation $(\mathbf{r},\mathbf{t}) \in SE(3)$ where $\mathbf{r} \in SO(3)$ and $\mathbf{t} \in \mathbb{R}^3$. We mainly focus on obtaining stable and executable grasps using (i) a stability classifier
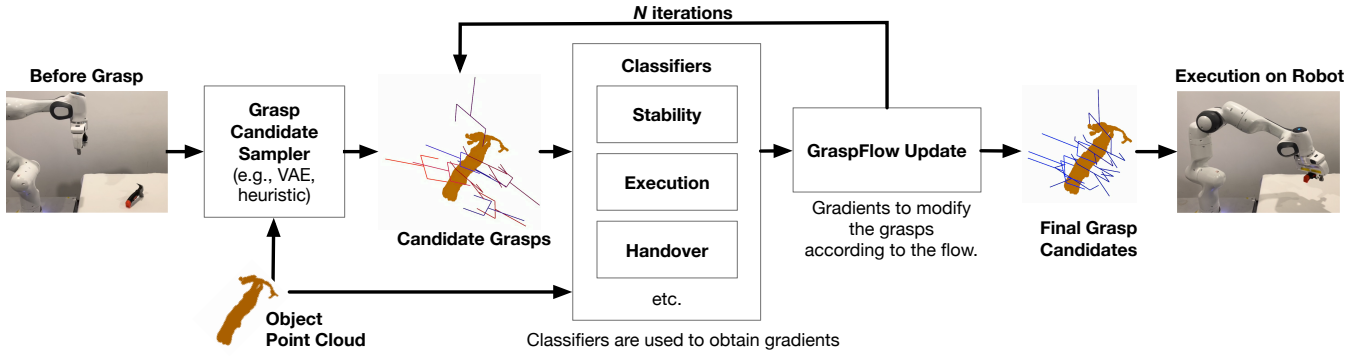
Fig. 2: Overview of GRASPFLOW for 6DOF grasping. GRASPFLOW refines sampled grasp candidates over N iterations using gradients obtained from the classifiers. Here, a batch of grasps are refined and candidates can be ranked using the classifiers before execution.

trained in simulation to distinguish stable from unstable grasps, and (ii) a handcrafted "execution" classifier that estimates whether the grasp can be performed by the Panda. Finally, we perform preliminary experiments with (iii) a functional classifier that classifies grasps for robot-human handover. Alternative classifiers can be used without changing the overall framework.

A. Stability Classifier

Our data-driven grasp stability classifier is based on the GraspNet evaluator [1], which uses PointNet [27]. The input to the classifier is a point cloud comprising the object point cloud (obtained from 4 views) and the gripper point cloud, with a feature label to distinguish the two types.

Simulator and Grasp Data. We collected grasp data using NVIDIA Isaac Gym [28] which provides a highly parallelizable simulation of realistic grasps for various objects. Our simulation environment consists of a free-floating Panda gripper and an object mesh with no gravity. We used ShapeNet [29] and 3DNet [30] objects from 10 categories, with 20 objects in each category (200 unique objects). The categories are mugs, hammer, bottles, boxes, cylinders, scissors, spatula, fork, pans and bowls. First, we sampled candidate grasps using the object's shape geometry as described in [1] and randomly sampling within a bounding box that fully covers the object's shape. Any candidate in collision with the object was labelled as negative. The candidate was also negative if the closing volume between robot fingers had zero intersection with the object. Any remaining candidates were labelled via simulation using the Panda gripper moving through a predefined set of motions (forward/backward movements by 10 cm and 30 degree rotations around pitch axis). Positive grasps were those that successfully held on to the object after these motions. Using this methodology, we initially collected 12.5 million grasps (9.1% positive labels).

Improving the Discriminator. Unfortunately, we found the standard classifier trained with the collected data above was unable to refine grasps well, particularly when the grasps were too far or too close to the object. Fortunately, the gradient flow formulation provides guidance into potential causes: refinement can fail when the classifier does not accurately
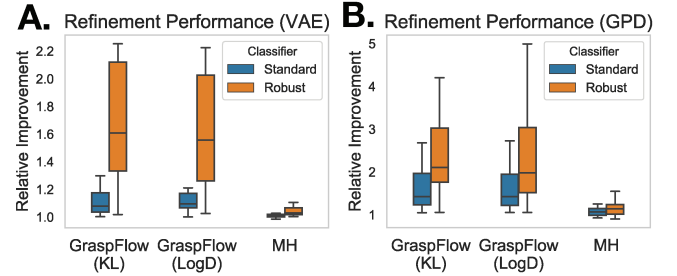

Fig. 3: Refinement Performance for Stability evaluated using NVIDIA Isaac Gym.

reflect the density ratio and the resultant derivatives. This is likely to happen with learned classifiers in regions with low-data (in either class) where overfitting can occur.

To mitigate this problem, we train our stability classifier with data augmentation (with random rotations and translations of the entire point cloud data) to "fill in" gaps in the data. Note that this data augmentation is performed in addition to the negative mining suggested in prior work [1]. We also conducted *positive* mining by perturbing good samples to ensure a sufficient number of positive samples. In total, we trained the model with 31.6 million grasps, which included 8.3 million (26%) positive grasps. In addition, we found it important to include an auxiliary grasp pose reconstruction loss. This loss encouraged the top-level latent representation of the classifier to encode information about the grasp, which we believe facilitates generalization and better gradients; the classifier is given grasp information in a point cloud representation and forcing the network to recover the grasp pose mitigates over-fitting to the noisy object point cloud.

Simulation Results. With these changes, our *robust* classifier was significantly better at refinement than the standard model. Fig. 3.A summarizes the relative improvement after refinement for 20 unseen ShapeNet objects. Here, the relative improvement captures the ratio of stable grasps before and after refinement per object (hence, values above 1 indicate a higher number of stable grasps). The initial 5000 grasps (per object) were sampled using the GraspNet VAE and stability was evaluated using NVIDIA Isaac Gym. We refined grasps

using GRASPFLOW with the KL and logD divergences (50 iterations), and a Metropolis-Hastings (MH) method (135 iterations to match computational time) [1], [17].

The boxplots show that the refinement with the robust classifier resulted in more stable grasps compared to the standard classifier. In addition, prior work had noted a simple MH approach obtained similar performance to refinement via gradients [17]. However, we observe the opposite: GRASPFLOW was better able to improve grasps compared to MH given the same computational budget. This discrepancy can potentially be explained by considering the importance of a robust classifier for refinement.

GRASPFLOW can refine grasps from alternative samplers such as GPD [8]. Unlike the VAE, the GPD sampler uses heuristics to generate antipodal grasps. Fig. 3.B again shows better refinement when using the robust classifier. In fact, the relative improvement here is larger as the GPD grasps were poorer for certain object classes; in total, the average proportion of stable grasps generated (before refinement) by the GraspNet VAE was 21% (SD=11.3) compared to 14.7% (SD=15.4) for GPD.

B. Execution Classifier

Unlike the stability classifier above, our execution classifier is theory-driven; it was developed using knowledge of robot kinematics. Here, our goal is to obtain grasps that avoid singular configurations of the robot. We will leverage manipulability ellipsoids, which are well-studied within the control community [31], [32]. For an open kinematic chain robot, the volume of the manipulability ellipsoid [33] is defined as:

$$\omega(\boldsymbol{\theta}) = \sqrt{\det \mathbf{J}(\boldsymbol{\theta})\mathbf{J}(\boldsymbol{\theta})^{\mathbf{T}}} \geqslant 0 \qquad (8)$$

where $\boldsymbol{\theta}$ is robot's joint configuration and $\mathbf{J}(\boldsymbol{\theta})$ is Jacobian matrix. The volume $\omega(\boldsymbol{\theta})$ spans the Cartesian speed of robot's end-effector and thus, it indicates singular or near-singular configurations of the robot [34]. As the eigenvalues of the manipulability ellipsoid are the reciprocal of the eigenvalues of the force ellipsoid, the volume also suggests how much force can be exerted on robot's end-effector. As such, it is desirable for the robot to operate in the configuration space that has a sufficiently large $\omega(\boldsymbol{\theta})$. Our goal is to find a robot's joint configuration that ensures a minimum ω_{th}, i.e., $\omega(\boldsymbol{\theta}) \geqslant \omega_{\text{th}}$. Using (8), we can define a classifier that gives the likelihood of executable grasp given robot's joint configuration, θ:

$$p(c_e = 1|\boldsymbol{\theta}) = \sigma(C(\omega(\boldsymbol{\theta}) - \omega_{\text{th}})) \qquad (9)$$

where $\sigma(\cdot)$ is the logistic function and C is a scale coefficient. For the Panda robot, we set $C = 100$ and $\omega_{\text{th}} = 0.04$. Since our grasp $\mathbf{g}$ is represented in Cartesian space coordinates, we obtain the joint space configuration $\boldsymbol{\theta}$ using an analytical inverse kinematics (IK) solver [35].

Grasps that were not reachable according to the IK solver were mapped to the closest reachable grasp in the joint space. We compute the Jacobian matrix in Eqn. (8) using the Newton-Euler method [36].

C. Handover Classifier

We developed a simple handover classifier that leverages point-wise part segmentation labels using a pre-trained deep model [37]. We then applied expert knowledge to select appropriate clusters to grasp for handover. Similar to the execution classifier, we compute likelihood of a positive handover grasp via a logistic classifier, $p(c_{\text{h}} = 1|\mathbf{g}) = \sigma(C\left((\mathbf{p}_{\text{centroid}} - \mathbf{g}_{\text{translation}})^2 - p_{\text{th}}\right))$ where p_{centroid} is a centroid of the target cluster, $\mathbf{g}_{\text{translation}}$ is translation of the grasp and C is a scale parameter and p_{th} is the minimum set distance to the grasp. We set $p_{\text{th}} = 4$ cm and $C = 10$ in our experiments.

IV. REAL-WORLD EXPERIMENTS

In this section, we report on experiments designed to validate our main claim, i.e., that GRASPFLOW synthesizes successful grasps for real robots. We focused on obtaining stable and executable grasps for a 7-DoF Franka-Emika Panda robot equipped with an Intel RealSense RGB-D camera mounted on the arm, and include preliminary findings on obtaining functional grasps. Our experimental setup is shown in Fig. 4.A and was constructed using ROS [38]; our code is available at `https://github.com/clear-nus/graspflow`. We used 20 unique household objects obtained from the YCB dataset [39] or from a local supermarket. The objects are from known categories but were previously *unseen*.

Procedure. First, an object was placed within the workspace of the robot. Then, the robot moves to 4 different poses around the object to collect point clouds using the RGB-D camera. At each pose, it stops for 5 seconds to minimize the effect of the noise. We filter out the background point cloud using [40] and collect only the point cloud for the object of interest. Then, we map these point clouds into the world frame and combine them into a single dense point cloud.

Grasp Generation. Given an object's point cloud, we use the GraspNet VAE to generate 200 grasp candidates. These candidates are evaluated using our stability classifier and we selected the top-10 highest scoring grasps; these grasps constitute our "Base" samples. We then applied GRASPFLOW either with the stability classifier (S) or a combination of the stability and execution classifiers (S+E) to the Base samples. GRASPFLOW parameters were set as $T = 50$, $\eta_{\text{trans}} = 10^{-5}$, $\eta_{\text{euler}} = 10^{-4}$, and $\gamma = 10^{-4}$ across all objects. We repeated the same procedure three times for each object; in each one of these "trials", the object was placed in a random pose.

Grasp Evaluation. In total, our experiment comprised 60 trials (20 objects, 3 trials each) and in each trial, we executed 10 grasps for each method (Base, S, S+E). In total, the robot executed 1800 grasps. We use Moveit! [41] with the RRT connect planner [42] to plan a trajectory to the grasp. A grasp was considered successful if the robot managed to grasp the object, lift it 20cm upwards, and hold it for 2 seconds. Otherwise, the grasp was labelled as a failure. We further distinguished failures as either a grasp failure or a robot execution failure (an error due to singularities).
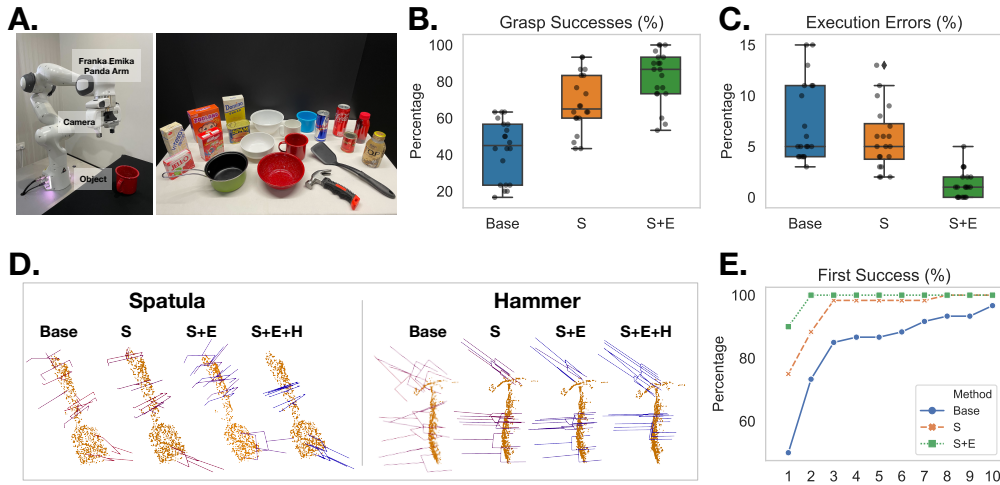
Fig. 4: (**A**) Panda robot along with the objects used for grasping. (**B**) Percentage of successful grasp (stable and executable grasps). (**C**) Percentage of robot execution errors. (**D**) Example refined grasps using the different classifiers. (**E**) Number of grasps until the first successful grasp.

TABLE I: Handover Grasp Success

Object	S+E	S+E+H
Hammer	36.6%	60%
Spatula	43.3%	76.7%

A. Results

Does GRASPFLOW refinement improve grasp stability and execution? In short, yes. The boxplot in Fig. 4.B shows that the proportion of successful grasps (averaged across 20 different objects with 30 grasps each) increased from 42.8% (SD=15.9) to 68.2% (SD=15.5) when the stability (S) refinement was applied. The success percentage further increased to 82% (SD=14.0) when both the stability and execution classifiers were used (S+E). These differences are statistically significant (assessed via three paired t-tests with Bonferroni-adjusted $\alpha = 0.0033$ per test, p-value $< 10^{-5}$ across the pairwise differences). Likewise, Fig. 4.C shows the percentage of robot errors experienced decreased sharply from 7.35% (SD=3.9) to 1.3% (SD=1.3) when the execution classifier was used with the stability classifier ($t(29) = 7.06$, $p < 10^{-5}$). In contrast, refinement with the stability classifier only had a mild effect on error reduction; this result supports the need to refine using multiple criteria for the given context.

Does incorporating the Handover discriminator enable functional grasping? To answer this question, we conducted an experiment using the hammer and spatula objects to determine if GRASPFLOW was able to generate grasps that were not only stable and executable, but also functional. We used the same experimental protocol, with the additional success criterion that the executed grasp should be suitable for handover (ascertained by a human participant). The results are summarized in the Table I where (S+E+H) represents GRASPFLOW with all three classifiers. We observe that incorporating the handover classifier increased the percentage of functional grasps by up to 33%.

How quickly can we obtain a successful grasp? Fig.4.E shows the percentage of trials (out of 60) when the first successful grasp was obtained. Refinement with stability and execution classifiers lead to the first grasp being successful 94% of the time and grasping was 100% successful by the second grasp. For the handover experiment, 100% grasp success (i.e., stable, executable, and functional) was achieved by the second grasp.

V. CONCLUSIONS AND FUTURE WORK

This work presents GRASPFLOW, an alternative form of grasp synthesis where grasps are refined/evolved to satisfy multiple criteria via differentiable discriminators. As experiments show, GRASPFLOW generates more grasps candidates that are stable, executable, and functional compared to baseline methods.

Limitations and Future Work. There are a number of ways that GRASPFLOW could be further improved. While our results are positive, it remains unclear how robust GRASPFLOW is to noise in the discriminator gradients. Here, we used three classifiers and how the methodology scales to a large number of discriminators is an open question; the criteria landscape may be highly nonlinear with multiple minima, which can hamper sample evolution. The modularity afforded by the conditional independence assumption improves scaling in a computational sense, but the assumption may not hold with a large number of criteria. One possible workaround is to combine classifiers (with extra training) when the labels may be conditionally dependent. Further experiments are needed to examine such a setup. Finally, there are other contexts that are ripe for exploration—future work can look into applying GRASPFLOW with other methods such as DexNet, functional grasping beyond handover, and also other contexts, e.g., grasping in clutter, with multi-fingered end-effectors, or for in-hand manipulation.

REFERENCES

[1] A. Mousavian, C. Eppner, and D. Fox, "6-dof graspnet: Variational grasp generation for object manipulation," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 2901–2910.

[2] A. F. Ansari, M. L. Ang, and H. Soh, "Refining deep generative models via discriminator gradient flow," in *International Conference on Learning Representations (ICLR)*, 2021.

[3] J. Bohg, A. Morales, T. Asfour, and D. Kragic, "Data-driven grasp synthesis—a survey," *IEEE Transactions on Robotics*, vol. 30, no. 2, pp. 289–309, 2013.

[4] K. B. Shimoga, "Robot grasp synthesis algorithms: A survey," *The International Journal of Robotics Research*, vol. 15, no. 3, pp. 230–266, 1996.

[5] A. Sahbani, S. El-Khoury, and P. Bidaud, "An overview of 3d object grasp synthesis algorithms," *Robotics and Autonomous Systems*, vol. 60, no. 3, pp. 326–336, 2012.

[6] K. Kleeberger, R. Bormann, W. Kraus, and M. F. Huber, "A survey on learning-based robotic grasping," *Current Robotics Reports*, pp. 1–11, 2020.

[7] 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, 2015.

[8] A. ten Pas, M. Gualtieri, K. Saenko, and R. Platt, "Grasp pose detection in point clouds," *The International Journal of Robotics Research*, vol. 36, no. 13-14, pp. 1455–1473, 2017.

[9] J. Mahler, J. Liang, S. Niyaz, M. Laskey, R. Doan, X. Liu, J. A. Ojea, and K. Goldberg, "Dex-net 2.0: Deep learning to plan robust grasps with synthetic point clouds and analytic grasp metrics," in *Proc. of Robotics: Science and Systems (RSS)*, 2017.

[10] S. Levine, P. Pastor, A. Krizhevsky, J. Ibarz, and D. Quillen, "Learning hand-eye coordination for robotic grasping with deep learning and large-scale data collection," *The International Journal of Robotics Research*, vol. 37, no. 4-5, pp. 421–436, 2018.

[11] V. Satish, J. Mahler, and K. Goldberg, "On-policy dataset synthesis for learning robot grasping policies using fully convolutional deep networks," *IEEE Robotics and Automation Letters*, vol. 4, no. 2, pp. 1357–1364, 2019.

[12] D. Morrison, P. Corke, and J. Leitner, "Closing the Loop for Robotic Grasping: A Real-time, Generative Grasp Synthesis Approach," in *Proc. of Robotics: Science and Systems (RSS)*, 2018.

[13] Y. Zhou and K. Hauser, "6dof grasp planning by optimizing a deep learning scoring function," in *Robotics: Science and systems (RSS) workshop on revisiting contact-turning a problem into a solution*, vol. 2, 2017, p. 6.

[14] Q. Lu, K. Chenna, B. Sundaralingam, and T. Hermans, "Planning multi-fingered grasps as probabilistic inference in a learned deep network," in *Robotics Research*. Springer, 2020, pp. 455–472.

[15] Q. Lu, M. Van der Merwe, B. Sundaralingam, and T. Hermans, "Multifingered grasp planning via inference in deep neural networks: Outperforming sampling by learning differentiable models," *IEEE Robotics & Automation Magazine*, vol. 27, no. 2, pp. 55–65, 2020.

[16] M. Van der Merwe, Q. Lu, B. Sundaralingam, M. Matak, and T. Hermans, "Learning continuous 3d reconstructions for geometrically aware grasping," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2020, pp. 11 516–11 522.

[17] 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)*. IEEE, 2020, pp. 6232–6238.

[18] W. Yang, C. Paxton, A. Mousavian, Y.-W. Chao, M. Cakmak, and D. Fox, "Reactive human-to-robot handovers of arbitrary objects," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2021, pp. 3118–3124.

[19] F. Santambrogio, "{Euclidean, metric, and Wasserstein} gradient flows: an overview," *Bulletin of Mathematical Sciences*, vol. 7, no. 1, 2017.

[20] A. Liutkus, U. Simsekli, S. Majewski, A. Durmus, and F.-R. Stöter, "Sliced-Wasserstein flows: Nonparametric generative modeling via optimal transport and diffusions," in *ICML*, 2019.

[21] Y. Gao, Y. Jiao, Y. Wang, Y. Wang, C. Yang, and S. Zhang, "Deep generative learning via variational gradient flow," in *ICML*, 2019.

[22] Y. Gao, J. Huang, Y. Jiao, and J. Liu, "Learning implicit generative models with theoretical guarantees," *arXiv preprint arXiv:2002.02862*, 2020.

[23] A. Bicchi and V. Kumar, "Robotic grasping and contact: A review," in *Proceedings 2000 ICRA. Millennium conference. IEEE international conference on robotics and automation. Symposia proceedings (Cat. No. 00CH37065)*, vol. 1. IEEE, 2000, pp. 348–353.

[24] H. Risken, *Fokker-Planck equation*. Springer, 1996.

[25] W.-J. Beyn and R. Kruse, "Numerical methods for stochastic processes," *Lecture Book (in preparation)*, 2011.

[26] M. Sugiyama, T. Suzuki, and T. Kanamori, *Density ratio estimation in machine learning*. Cambridge University Press, 2012.

[27] C. R. Qi, H. Su, K. Mo, and L. J. Guibas, "Pointnet: Deep learning on point sets for 3d classification and segmentation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 652–660.

[28] V. Makoviychuk, L. Wawrzyniak, Y. Guo, M. Lu, K. Storey, M. Macklin, D. Hoeller, N. Rudin, A. Allshire, A. Handa, *et al.*, "Isaac gym: High performance gpu-based physics simulation for robot learning," *arXiv preprint arXiv:2108.10470*, 2021.

[29] A. X. Chang, T. Funkhouser, L. Guibas, P. Hanrahan, Q. Huang, Z. Li, S. Savarese, M. Savva, S. Song, H. Su, *et al.*, "Shapenet: An information-rich 3d model repository," *arXiv preprint arXiv:1512.03012*, 2015.

[30] W. Wohlkinger, A. Aldoma, R. B. Rusu, and M. Vincze, "3dnet: Large-scale object class recognition from cad models," in *2012 IEEE international conference on robotics and automation*. IEEE, 2012, pp. 5384–5391.

[31] T. Yoshikawa, "Manipulability and redundancy control of robotic mechanisms," in *Proceedings. 1985 IEEE International Conference on Robotics and Automation*, vol. 2. IEEE, 1985, pp. 1004–1009.

[32] J. Haviland and P. Corke, "A purely-reactive manipulability-maximising motion controller," *arXiv preprint arXiv:2002.11901*, 2020.

[33] K. M. Lynch and F. C. Park, *Modern robotics*. Cambridge University Press, 2017.

[34] M. Rubagotti, T. Taunyazov, B. Omarali, and A. Shintemirov, "Semi-autonomous robot teleoperation with obstacle avoidance via model predictive control," *IEEE Robotics and Automation Letters*, vol. 4, no. 3, pp. 2746–2753, 2019.

[35] Y. He and S. Liu, "Analytical inverse kinematics for franka emika panda–a geometrical solver for 7-dof manipulators with unconventional design," in *2021 9th International Conference on Control, Mechatronics and Automation (ICCMA)*. IEEE, 2021, pp. 194–199.

[36] G. Sutanto, A. Wang, Y. Lin, M. Mukadam, G. Sukhatme, A. Rai, and F. Meier, "Encoding physical constraints in differentiable newton-euler algorithm," in *Learning for Dynamics and Control*. PMLR, 2020, pp. 804–813.

[37] Y. Wang, Y. Sun, Z. Liu, S. E. Sarma, M. M. Bronstein, and J. M. Solomon, "Dynamic graph cnn for learning on point clouds," *Acm Transactions On Graphics (tog)*, vol. 38, no. 5, pp. 1–12, 2019.

[38] M. Quigley, K. Conley, B. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, A. Y. Ng, *et al.*, "Ros: an open-source robot operating system," in *ICRA workshop on open source software*, vol. 3, no. 3.2. Kobe, Japan, 2009, p. 5.

[39] B. Calli, A. Singh, J. Bruce, A. Walsman, K. Konolige, S. Srinivasa, P. Abbeel, and A. M. Dollar, "Yale-cmu-berkeley dataset for robotic manipulation research," *The International Journal of Robotics Research*, vol. 36, no. 3, pp. 261–268, 2017.

[40] R. B. Rusu and S. Cousins, "3d is here: Point cloud library (pcl)," in *2011 IEEE international conference on robotics and automation*. IEEE, 2011, pp. 1–4.

[41] S. Chitta, I. Sucan, and S. Cousins, "Moveit![ros topics]," *IEEE Robotics & Automation Magazine*, vol. 19, no. 1, pp. 18–19, 2012.

[42] J. J. Kuffner and S. M. LaValle, "Rrt-connect: An efficient approach to single-query path planning," in *Proceedings 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings (Cat. No. 00CH37065)*, vol. 2. IEEE, 2000, pp. 995–1001.