Adversarial Grasp Objects

David Wang*[1], David Tseng*[1], Pusong Li*[1], Yiding Jiang*[1], Jeffrey Mahler[1], Ken Goldberg[1,2]

***Abstract*— Learning-based approaches to robust robot grasp planning can handle a wide variety of objects, but may be prone to failure on objects with subtle geometry. We define an "adversarial grasp object" as an object that is visually similar to an original object but decreases the predicted graspability resulting from a robot grasping policy. We study two algorithms for synthesizing adversarial grasp objects under a grasp reliability measure based on Dex-Net: 1) an analytic method that locally perturbs vertices on antipodal faces, and 2) a deep-learning based method using a variation of the Cross-Entropy Method (CEM) augmented with a generative adversarial network (GAN) to synthesize adversarial grasp objects represented by a Signed Distance Field through more global geometric changes. Experiments suggest that both algorithms consistently reduce graspability. The analytic algorithm is able to reduce graspability by 32%, 12%, and 32% on intersected cylinders, intersected prisms, and ShapeNet bottles, respectively, while maintaining shape similarity using geometric constraints. The GAN is able to reduce graspability by 22%, 36%, and 17% on the same objects.**

I. Introduction

Adversarial images [1], [2], [3], [4] are modified images that drastically alter the prediction made by a classifier while applying minimal perturbation to the original image. This paper defines "adversarial grasp objects," an analog of adversarial images in the domain of robust robot grasping. Similar to adversarial images, adversarial grasp objects reduce graspability while retaining geometric similarity to input objects.

Robust robot grasping of a large variety of objects can benefit a diverse range of applications, such as the automation of industrial warehousing and home decluttering. Recent research suggests that robot policies based on deep learning can grasp a variety of previously unseen objects [5], [6], [7], [8], but can be prone to failures on objects that may not be encountered during training [9].

Adversarial image generation techniques involve performing constrained gradient-based optimization algorithms on the image classification loss [1]. However, a central challenge in applying these algorithms to deep grasping policies is that grasping performance is not a differentiable function of the network output. Instead, the grasp planned by a policy is the result of scoring, ranking, and pruning a set of grasp candidates for each object. To address this, we explore analytic methods and derivative-free optimization. We present two algorithms for synthesizing adversarial objects: an algorithm that modifies objects by perturbating vertices on antipodal faces subject to geometric constraints to maintain similarity to the input object, and an algorithm for synthesizing adversarial 3D object models using 3D Generative Adversarial Networks (GANs) [10] and the Cross Entropy Method (CEM) for derivative-free optimization. The second algorithm extends recent advances in GANs to synthesize a 3D Signed Distance Function (SDF) representation for objects that minimizes the quality of available grasps. This paper contributes:

1) A formal definition of adversarial grasp objects.
2) An analytical algorithm to synthesize adversarial 3D objects for grasp planning from a given 3D object by performing constrained perturbations of vertices on antipodal faces.
3) A deep learning algorithm based on the Cross Entropy Method (CEM) for derivative-free optimization and deep Generative Adversarial Networks (GANs) that uses an SDF representation of 3D objects to generate a distribution of adversarial objects that look similar to objects from a prior distribution.

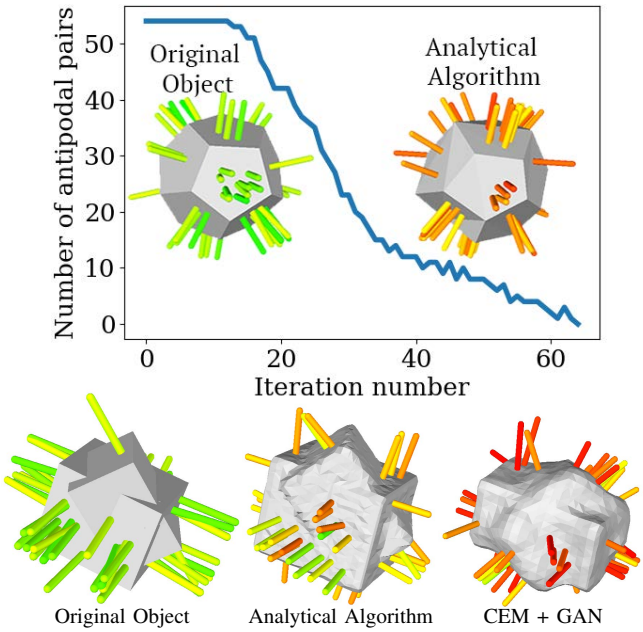

Fig. 1: Original objects vs. adversarial objects. The most robust 25 of 100 parallel-jaw grasps sampled on each object are displayed as grasp axes colored by relative reliability on a linear gradient from green to red. Top: Results from running an analytic algorithm on a dodecahedron mesh for 64 iterations, and a plot of the number of antipodal faces vs. the number of iterations of the algorithm. Bottom row: Results from applying an analytical algorithm and the CEM + GAN algorithm on a synthetically generated intersected prism.

[1] Dept. of Electrical Engineering and Computer Science; {dmwang, davidtseng, alanpusongli, yiding.jiang, jmahler, goldberg}@berkeley.edu
[2] Dept. of Industrial Engineering and Operations Research;
[1,2] The AUTOLab at UC Berkeley (automation.berkeley.edu)
*Authors contributed equally to this work

4) Experiments studying adversarial grasp objects of several categories (bottles, intersected cylinders, and intersected prisms) generated by the two algorithms for the Dexterity Network (Dex-Net) 1.0 robust grasp planner, which plans parallel-jaw grasps based on a robust quasi-static point contact model [11].

II. Related Work

Adversarial Images. Adversarial images [1], [2], [3], [4] are inputs with a small added perturbation that can change the output of an image classifier, and the problem of finding adversarial images is typically formulated as a constrained optimization problem that can be approximately solved using gradient-based approaches [1]. Yang et al. developed a method to perturb the texture maps of 3D shapes such that their projections onto 2D image space can fool classifiers [12]. We build on this line of research by studying adversarial examples in the context of generating adversarial 3D objects for robotic grasping.

Grasp Planning. Grasp planning considers the problem of finding a gripper configuration that maximizes the probability of grasp success. Approaches generally fall into one of three categories: analytic [13], empirical [14], and hybrid methods.

Analytic approaches typically assume knowledge of the object and gripper state, including geometry, pose, and material properties, and consider the ability to resist external wrenches [13] or constrain the object's motion [15], possibly under perturbations to model robustness to sensor noise. Examples include GraspIt! [16], OpenGRASP [17], and the Dexterity Network (Dex-Net) 1.0 [11]. To satisfy the assumption of known state, analytic methods typically assume a perception system based on registration: matching sensor data to known 3D object models in the database [18], [19], [20], [21], [22], [23]. However, these systems do not scale well to novel objects and may be computationally expensive during execution.

Empirical approaches use machine learning to develop models that map from robotic sensor readings directly to success labels from humans or physical trials. Research in this area has largely focused on associating human labels with graspable regions in RGB-D images [5], [24], [25] or using self-supervision to collect labels from successes and failures on a physical system [6], [7]. A downside of empirical methods is that data collection may be time-consuming and prone to errors.

Hybrid approaches make use of analytic models to automatically generate large training datasets for machine learning models [26], [27]. Recent results suggest that these methods can be used to rapidly train grasping policies to plan grasps on point clouds that generalize well to novel objects on a physical robot [8], [9], [28]. In this paper, we consider synthesizing adversarial 3D objects for the analytic supervisor used to train these hybrid grasp planning methods.

Generative Models. Deep generative models map a simple distribution, such as a multivariate Gaussian distribution, to a much more complex distribution, such as natural images. Common deep generative models fall into likelihood-based models (i.e., the Variational Auto-Encoder (VAE) [29] and PixelCNN [30]) and likelihood-free models (i.e., various formulations of Generative Adversarial Networks (GANs) [10]). During training of a GAN, a discriminator tries to distinguish the generated samples apart from the samples from the real data while a generator tries to generate samples to confuse the discriminator. Generative models have also been previously used in the domain of robot grasping, where Veres et al. [31] used conditional generative models to synthesize grasps from RGB-D images, and Bousmalis et al. [28] used GANs for simulation-to-reality transfer learning.

On the other hand, applications of deep generative models to 3D data are relatively under-explored. Some notable works in this area include the 3D GAN work by Wu et al. [32], which uses a GAN on the latent code learned by a variational autoencoder to generate 3D reconstruction from an image, and the signed distance-based, higher-detail object generation by Jiang et al. [33], where the low frequency components and high frequency components are generated by two separate networks. We expand upon previous efforts in this direction by incorporating recent advances in GANs for 2D image data.

III. Problem Statement

A. Adversarial Grasp Objects

Let $\mathcal{X}$ be the set of all 3D objects. Let π be a robot grasping policy mapping a 3D object $\mathbf{x} \in \mathcal{X}$ specified as a 3D triangular mesh to a grasp action $\mathbf{u}$. In this work, we only consider a parallel-jaw grasping policy. We assume that the policy can be represented as:

$$\pi(\mathbf{x}) \triangleq \underset{\mathbf{u} \in \mathcal{U}(\mathbf{x})}{\operatorname{argmax}} \; Q(\mathbf{x}, \mathbf{u}) \tag{III.1}$$

where $\mathcal{U}(\mathbf{x})$ denotes the set of all reachable grasp candidates on $\mathbf{x}$, and Q is a quality function measuring the reliability or probability of success for a candidate grasp $\mathbf{u}$ on object $\mathbf{x}$.

We define the graspability $g(\mathbf{x}, \pi)$ of $\mathbf{x}$ with respect to π as a measure of how well the policy can robustly grasp the object. We measure graspability by the γ-percentile of grasp quality [34]:

$$g(\mathbf{x}, \pi) \triangleq \mathbb{P}_{\gamma}(Q(\mathbf{x}, \mathbf{u})) \tag{III.2}$$

We then consider the problem of generating an adversarial grasp object: a 3D object that systematically reduces graspability under a grasping policy with constrained changes to the input geometry. Let $\sigma(A, B)$ for subsets $A, B \subset \mathcal{X}$ be a binary-valued shape similarity constraint between the two subsets of objects. We study the following optimization problem, which defines an adversarial grasp object $\mathbf{x}^*$:

$$\mathbf{x}^* = \underset{\mathbf{x} \in \mathcal{X}}{\operatorname{argmin}} \; g(\mathbf{x}, \pi) \text{ subject to } \sigma(\{\mathbf{x}\}, S) = 1 \tag{III.3}$$

where $S \subset \mathcal{X}$ is a subset of objects that the generated object should be similar in shape to.

B. Robust Grasp Analysis

In this paper, we optimize adversarial examples with respect to the Dexterity Network (Dex-Net) 1.0 grasping policy [11]. In this setting, the action set $\mathcal{U}(\mathbf{x})$ is a set of antipodal points on the object surface that correspond to a reachable grasp, where a pair of opposite contact points v_1, v_2 are antipodal if the line between the v_1, v_2 lie entirely within the friction cones [11]. The quality function Q measures the robust wrench resistance, or the ability of a grasp to resist a target wrench under perturbations to the object pose, gripper pose, friction, and wrench under a soft-finger point contact model [9].

When calculating g, both the reward and policy are based on the Dex-Net 1.0 robust grasp quality metric and the associated maximal quality grasping policy. Within the Dex-Net 1.0 robust quality metric, $Q(\mathbf{x}, \mathbf{u})$ is defined as:

$$Q(\mathbf{x}, \mathbf{u}) \triangleq \mathbb{E}_{\mathbf{u}' \sim p(\cdot|\mathbf{u}), \mathbf{x}' \sim p(\cdot|\mathbf{x})}[R(\mathbf{x}', \mathbf{u}')] \tag{III.4}$$

where $p(\mathbf{u}'|\mathbf{u})$ and $p(\mathbf{x}'|\mathbf{x})$ denote distributions over possible perturbations conditioned on $\mathbf{x}$ and grasp $\mathbf{u}$, and R represents a measure of grasp quality if the grasp is executed exactly as given; that is, executed with zero uncertainty in object and gripper pose. In this case, we use the epsilon metric by Ferrari and Canny with a soft-finger point contact model [35].

To calculate $g(\mathbf{x}, \pi)$ in practice, both the expected value over the distributions of object and grasp pose $p(\mathbf{x}'|\mathbf{x})$ and $p(\mathbf{u}'|\mathbf{u})$ and the γ-percentile are calculated using sample estimates [36]. To do this, we first uniformly sample a constant number of antipodal grasps across the surface of the object. We then approximate the robustness for each grasp by sampling perturbations in object and gripper pose and taking the average grasp quality over all sampled configurations.

The empirical robust grasp quality is:

$$\hat{Q}(\mathbf{x}, \mathbf{u}) = \frac{1}{N} \sum_{i=1}^{N} R(\mathbf{x}_i, \mathbf{u}_i) \tag{III.5}$$

where $\{\mathbf{u}_i\}_{i=1}^N$, $\{\mathbf{x}_i\}_{i=1}^N$ are i.i.d. samples drawn from $p(\mathbf{u}'|\mathbf{u})$ and $p(\mathbf{x}'|\mathbf{x})$ respectively.

The empirical graspability $\hat{g}(\mathbf{x}, \pi)$ is estimated by taking the discrete γ-percentile of $\hat{Q}(\mathbf{x}, \mathbf{u})$ for all sampled grasps.

IV. Analytical Method: Constrained Vertex Perturbation

We consider an analogous approach for modifying an existing 3D triangular mesh $\mathbf{x} \in \mathcal{X}$ to decrease the graspability of $\mathbf{x}$. Let the mesh $\mathbf{x}$ be specified by a set of vertices $\mathcal{V} = \{v_1, v_2, \dots v_n\} \subset \mathbb{R}^3$ and a set of faces $\mathcal{F} = \{f_1, f_2, \dots f_m\}$, where each face f_i is the triangle defined by three distinct elements of $\mathcal{V}$. Also, let $F_a = \{(f_i, f_j), \dots (f_p, f_q)\}$ be the set of pairs of antipodal faces, and let the unit normal of face f_i be denoted by $\mathbf{n_i} \in \mathbb{R}^3$. Finally, let the antipodality angle φ between two faces be defined as $\varphi(f_i, f_j) = \arccos(-\mathbf{n_i}^T \mathbf{n_j})$.

A. Case Study: Simple Shapes

Dex-Net 1.0's graspability metric specifically considers the robustness of a parallel jaw grasp, which requires antipodal point pairs and can be susceptible to small pose variations. Thus, we consider the following iterative algorithm for analytically perturbing vertices to reduce the number of antipodal point pairs. Intuitively, we are attempting to decrease $|F_a|$, the number of antipodal faces, by maximizing φ between all antipodal faces. In each iteration, we compute F_a. For each vertex v of each face in F_a (e.g., all vertices incident to a face in F_a), we consider perturbations in directions defined by $\mathcal{W}$, a set of unit vectors $\{\mathbf{w_1}, -\mathbf{w_1}, \mathbf{w_2}, -\mathbf{w_2}, \mathbf{w_3}, -\mathbf{w_3}\}$, where $\mathbf{w_1}$, $\mathbf{w_2}$, and $\mathbf{w_3}$ are randomly selected and orthogonal, forming a basis for $\mathbb{R}^3$. The intuition is to search along all three directions by adding both a positive and negative perturbation. We then test perturbations $v' = v + \delta \mathbf{w}$ for each $\mathbf{w} \in \mathcal{W}$, where $\delta \in \mathbb{R}^+$ is a constant. We select the v' that maximizes $\sum_i \varphi_i$, the sum of the antipodality angles between all antipodal pairs in F_a that contain a face that is incident to v. Results from applying this algorithm to a sample dodecahedron mesh to systematically decrease the number of antipodal faces can be seen in Fig 1. The angle of the friction cone was set to $\arctan(0.17)$ in this case study.

B. Sampling-Based Algorithm

The previous algorithm can be effective on simple meshes, but because it has time complexity $\mathcal{O}(k \cdot m^2)$, where k is the number of iterations and m is the number of faces, it is less feasible to run this on very complex meshes with thousands of faces and vertices. Thus, we propose a sampling-based version of the above algorithm to avoid the overhead of computing the full set of antipodal faces. Consider the same mesh $\mathbf{x}$ as above. We want to perturb vertices while constraining the movement such that the surface normals of adjacent faces do not deviate by more than some angle α. This corresponds to the shape similarity constraint σ in Equation III.3, and in this case, $S = \{\mathbf{x}\}$, the original object itself.

In each iteration, we sample a pair of antipodal faces f_a and f_b, where $a, b \in \{1, 2, \dots m\}$. We then randomly sample one of the vertices v_k of f_a and f_b. Let $\mathcal{I} \subset \{1, 2, \dots n\}$ denote the set of indices of the faces adjacent to v_k. Again, we consider a set of 6 directions $\mathcal{W}$ sampled using the procedure described above, and for each direction $\mathbf{w} \in \mathcal{W}$, we compute the perturbation $\delta_w \in \mathbb{R}^+$ such that the antipodality angle φ between faces f_a and f_b is maximized subject to the constraints that $\cos^{-1}(\mathbf{n_i}^T \mathbf{n_i'}) < \alpha$ for all $i \in \mathcal{I}$, where $\mathbf{n_i'} \in \mathbb{R}^3$ denotes the unit surface normal of face f_i after moving vertex v_k to $v_k + \delta_w \mathbf{w}$. Then, we take the minimum perturbation δ_w found along each of the 6 directions as the actual perturbation. The algorithm runs for a number of iterations until we observe no further empirical improvement in decreasing graspability. By constraining the perturbations, the algorithm attempts to maintain local similarity of the region of perturbation while decreasing the graspability.

V. Deep Learning Algorithm: CEM + GAN

An alternative approach for the problem of generating adversarial grasp objects is to use a data-driven approach to learn a distribution over objects $\mathcal{X}$ and extract adversarial grasp objects by sampling from it. As opposed to the analytical algorithm, which generates an adversarial version of an existing object, the CEM + GAN algorithm takes as input a set $S \subset \mathcal{X}$ of objects and can output of a set of generated objects similar to those in S.

A. Deep Generative Models

One challenge in performing the optimization in Equation III.3 is that the graspability function $g(\mathbf{x}, \pi)$ is not differentiable; therefore, we need to perform the derivative-free optimization by querying the function with different inputs and adjust the model parameters based on the responses of the function. Let $p_\theta(\mathbf{x})$ be a probability distribution over $\mathcal{X}$ parameterized by some $\theta \in \Theta$. Then, we can formulate a similar objective to Equation III.3, but instead optimizing for a distribution of objects that we want to be similar to some prior subset $S \subset \mathcal{X}$:

$$\theta^*(\pi) = \arg\min_{\theta \in \Theta} \mathbb{E}_{\mathbf{x} \sim p_\theta(\cdot)}[g(\mathbf{x}, \pi)] \text{ subject to } \sigma(\mathcal{X}_\theta, S) = 1, \tag{V.1}$$

where $\mathcal{X}_\theta \subset \mathcal{X}$ is the support of the probability distribution p_θ for some parameter $\theta \in \Theta$.

We propose a deep learning method using the cross-entropy method (CEM) and generative adversarial networks (GANs) to approach this optimization problem. Let P_θ be the distribution over $\mathcal{X}$ induced by the model with parameter θ and P_S be the distribution empirically defined by S. We then define the shape similarity constraint $\sigma(\mathcal{X}_\theta, S)$ in Objective III.3 as $\mathrm{D}_{\mathrm{KL}}(P_S || P_\theta) < \epsilon$, where D_{KL} is the Kullback-Leibler divergence between two distributions, and $\epsilon > 0$ is a hyperparameter that can be controlled through the sampling percentile γ (smaller γ means more similar distributions). The GAN loss function implicitly enforces this shape similarity constraint as it has been shown that at the global optimum, the KL-divergence between the generated distribution and the original distribution is zero [37]. We note that the ϵ in the shape similarity constraint is necessary, since GANs do not usually reduce the loss to 0 in practice.

B. Optimization via Resampling

The cross-entropy method (CEM) [38] is an adaptive derivative-free optimization algorithm that has been widely applied. We are interested in finding the distribution of rare events that minimize a real-valued quality function $q(\mathbf{x})$ over $\mathcal{X}$. To minimize graspability, we choose $q(\mathbf{x}) = g(\mathbf{x}, \pi)$.

As a starting point, the GAN is initialized with a prior distribution of objects $S \subset \mathcal{X}$ so that it generates objects similar in shape. We start by training the GAN on this prior set of objects. Then, in a resampling step, we use the GAN to generate objects and take a subset of the objects with the lowest graspability to use as training data to retrain the GAN. We continue alternating between training and resampling steps for a number of iterations.

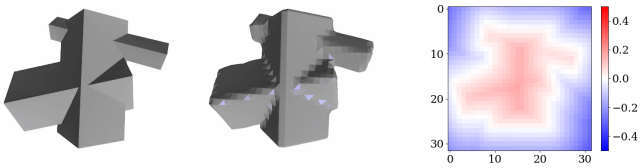

Fig. 2: Example result after converting a mesh to an SDF. The general shape of the mesh is preserved. While the conversion process to an SDF creates some artifacts, the artifacts are much less significant than the ones resulting from conversion to a binary occupancy grid. Left: Original mesh. Middle: SDF after remeshing. Right: A sample cross section of the SDF.

Gupta et al. [39] apply similar techniques to optimize functions over genetic sequences with a GAN by feeding samples with desired properties back into the GAN to generate more sequences with the properties. This suggests that the techniques we use may be general and can potentially be extended to other applications.

C. Signed Distance Generative Adversarial Network

Generative adversarial networks (GANs) [10] are a family of powerful implicit generative models that have demonstrated remarkable capabilities in generating high-quality samples with relatively low inference complexity. Much of the focus of the GAN community has been on generation of realistic images.

A common representation for learning 3D geometry is a binary 3D occupancy grid, in which the value of each cell indicates whether the center of a grid block intersects with the geometry. Due the cells being binary-valued, remeshing occupancy grids may produce blocky artifacts.

Instead, we use the Signed Distance Function (also known as signed distance field or SDF) [40] as an alternative representation for generating 3D geometry. An example is shown in Fig. 2. While SDFs do produce some artifacts due to their finite resolution, they are not as noticeable as the ones created by occupancy grids. SDFs are widely used in the computer vision community for applications such as rendering and segmentation, and in 3D applications for collision checking. The SDF of a closed object $\mathbf{x}$ with a well-defined inside and outside at point v can be given as:

$$f(v) = \begin{cases} d(v, \partial\mathbf{x}), & \text{if } v \in \mathbf{x} \\ -d(v, \partial\mathbf{x}), & \text{if } v \notin \mathbf{x} \end{cases} \tag{V.2}$$

where $\partial\mathbf{x}$ denotes the boundary of $\mathbf{x}$, and d is the Euclidian distance from the closest boundary to a point, and can be defined using the point-to-point version as

$$d(v, \partial\mathbf{x}) := \inf_{w \in \partial\mathbf{x}} d(v, w) \tag{V.3}$$

We draw on techniques used in Spectral-Normalization GAN (SNGAN) [41], which can generate high-fidelity images, and apply them to SDF's. We denote the standard Gaussian noise vector as $\mathbf{z} \in \mathbb{R}^{200}$ drawn from p_z, the empirical distribution defined by training data as p_{data}, the Generator as $G : \mathbb{R}^{200} \to [-1, 1]^{32 \times 32 \times 32}$, and the Discriminator as $D : [-1, 1]^{32 \times 32 \times 32} \to \mathbb{R}$. For the training objective, we use the hinge version of adversarial loss [37]

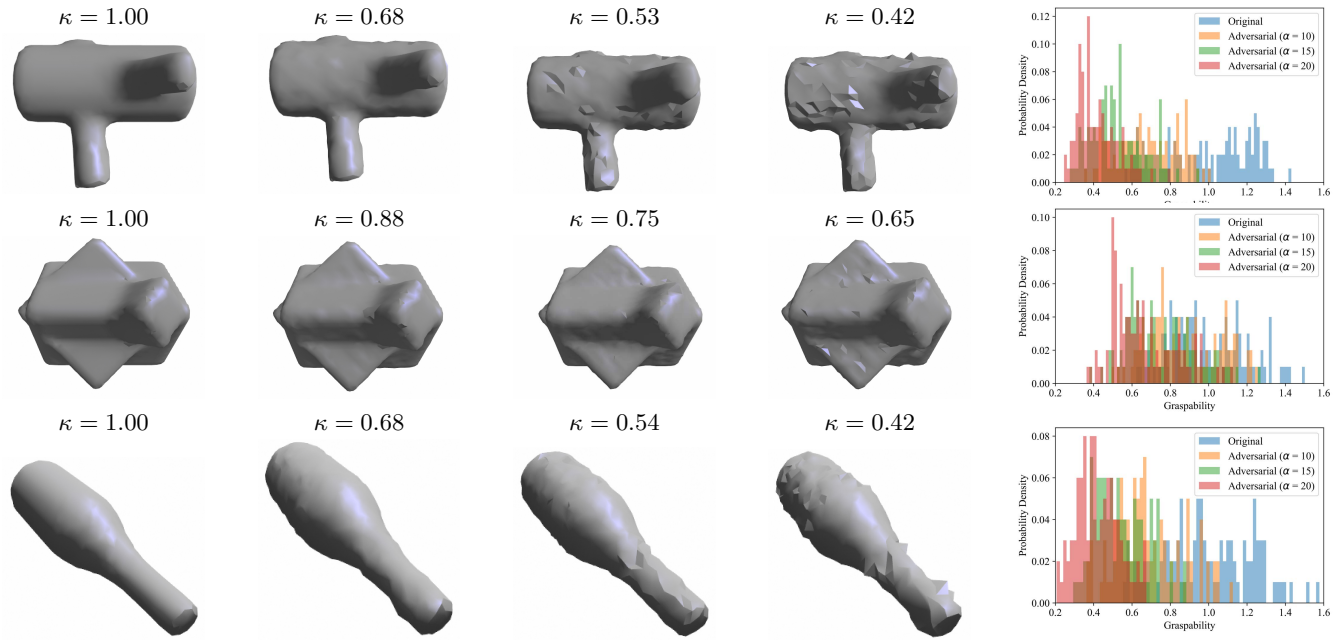

Fig. 3: Analytical Algorithm. We show the progression of an example from each dataset as we increase the surface normal constraint angle α: each row (from left to right) shows the original object and then the perturbed versions using the surface normal constraint with $\alpha = 10$, $\alpha = 15$, and $\alpha = 20$, respectively. The metric κ is the mean noramlized graspability of the generated dataset for the level of α, where the graspability is the empirical 75^{th} percentile of samples from the grasp quality function. The rightmost column shows the histograms of the graspability of all the objects. The analytical algorithm is able to decrease graspability on objects from all three datasets. The objects have been smoothed for visualization purposes with OpenGL smooth shading.

as we empirically found that it stabilizes training. The GAN objective is then

$$\mathcal{L}_D^{data} = -\mathbb{E}_{\mathbf{x}\sim p_{data}(\cdot)}[\min(0, -1 + D(\mathbf{x}))] \tag{V.4}$$

$$\mathcal{L}_D^{gen} = -\mathbb{E}_{\mathbf{z}\sim p_z(\cdot)}[\min(0, -1 - D(G(\mathbf{z})))] \tag{V.5}$$

$$\mathcal{L}_D = \mathcal{L}_D^{data} + \mathcal{L}_D^{gen} \tag{V.6}$$

$$\mathcal{L}_G = -\mathbb{E}_{\mathbf{z}\sim p_z(\cdot)}[D(G(\mathbf{z}))] \tag{V.7}$$

VI. Experiments

We run the two algorithms to minimize overall graspability on two synthetic datasets as well as on the ShapeNet [42] bottles category. To allow a fair comparison between our two algorithms, we converted all three datasets to SDFs. For the synthetic datasets, we used the process presented by Bousmalis et al. [28] where they generated objects to grasp in simulation by randomly attaching rectangular prisms of varying sizes together at varying angles. The intersected cylinders dataset consists of one large central cylinder with two smaller cylinders randomly grafted onto it. To show that the GAN also works on non-cylindrical objects, the intersected prisms dataset is similar to previous dataset but uses prisms instead: it consists of one central rectangular prism with two other rectangular prisms randomly grafted onto it. All three prisms have a wide distribution of sizes. The bottle, cylinder, and prism datasets have averages of 1,391 vertices and 2,783 faces, 1,202 vertices and 2,400 faces, and 2731 vertices and 4739 faces, respectively, and have 479, 1000, and 1000 total objects, respectively. Examples from each of these datasets are shown in Fig. 3.

In the following experiments, we set the angle of the friction cone to be $\arctan(0.5)$. For the graspability metric $g(\mathbf{x}, \pi)$, we chose $\gamma = 75\%$: often, one of the top 25% of grasps is accessible, so we choose to look at the worst case from this set. Consider a set of generated objects $\{\mathbf{x}_1, \mathbf{x}_2, \ldots \mathbf{x}_n\} \subset \mathbf{X}$ from a prior dataset of objects. We define mean normalized graspability as $\kappa = c \cdot \frac{1}{n} \sum_{i=1}^{n} g(\mathbf{x}_i, \pi)$, where c is a normalizing constant. We note that the objects in the figures in the section have been smoothed for visual clarity to demonstrate the behavior of the algorithms, but the metrics represent the results of the objects without smoothing. Meshes in all datasets have large numbers of vertices and faces, and displaying all of them makes it difficult to distinguish differences within and between algorithms.

A. Analytical Algorithm

We run the analytical algorithm for local perturbations of vertices on antipodal faces on 100 objects from each of the three datasets. We experimented with α values of 10, 15, and 20 degrees for the shape similarity constraint for maximum deviation in surface normals described in Section IV-B. We find that the analytical algorithm decreases the graspability metric for all datasets. With a value of $\alpha = 10$ degrees, the mean normalized graspability is decreased by 32% on the intersected cylinders dataset, 12% on the intersected prisms dataset, and 32% on the ShapeNet bottles datset. At each level of α, we observe that the objects from the prism dataset have the highest graspability; we conjecture that it is difficult to decrease the antipodality of large, flat prism surfaces with only local perturbations. Sample object examples along with their adversarial versions, the associated graspability

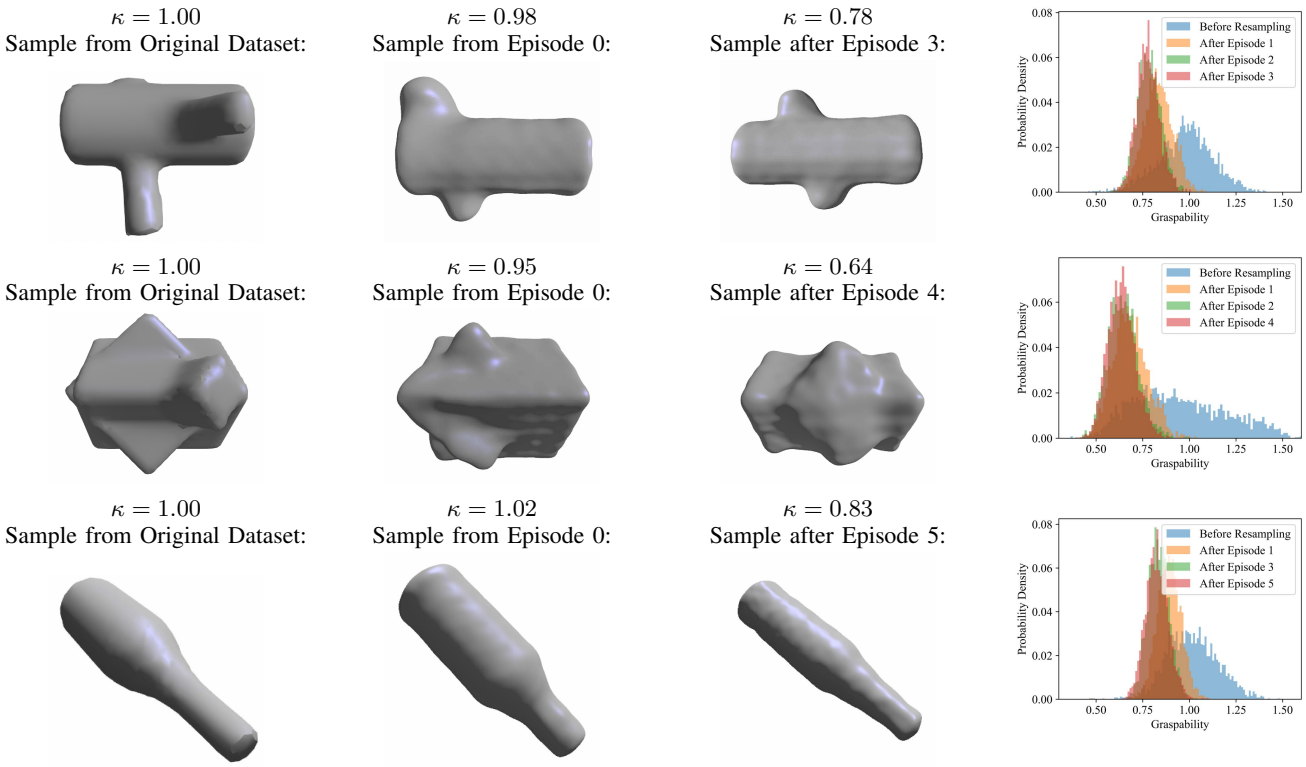

Fig. 4: CEM + GAN Algorithm. The images on the left are example objects from the GAN output distribution as the resampling progresses. "Original" means the original SDF dataset, "Episode 0" denotes the GAN trained on the prior dataset (the first GAN trained, or Episode 0), and "Episode n" denotes the n^{th} GAN trained excluding the first. The κ values are the mean normalized graspabilities over a set of 100 objects generated during the corresponding stage in the training, where the graspability is the empirical 75^{th} percentile of samples from the grasp quality function. The right image is the histogram showing the overall distribution of the graspability metric (normalized to the mean graspability Episode 0) on the GAN output distribution as resampling progresses. As the algorithm progresses through the episodes, the probability mass shifts towards lower graspability. The objects have been smoothed for visualization purposes with OpenGL smooth shading.

metrics, and the distribution of graspability metrics before and after applying the analytical algorithm are shown in Fig. 3. Increasing α decreases the graspability at the cost of similarity to the original object, corresponding to an increasingly relaxed shape similarity constraint.

B. CEM + GAN Algorithm

We train the resampling GAN on the previously described intersected prisms and cylinders datasets, as well as the ShapeNet bottles category. All three datasets are preprocessed into signed distance field format with stride 0.03125 after being scaled such that the entire set has bounding boxes of approximately $1 \times 1 \times 1$.

For all three datasets, we sample 2500 new objects and keep 500, and train the GAN for 16000 iterations between resampling steps. Resampling in all experiments rejects output grids that produce non-watertight meshes, as producing meshes with non-orientable faces, gaps, self-intersection, or disjoint pieces is not desirable when generating a distribution of 3D objects. Such outputs are possible because the GAN does not explicitly enforce such constraints, but this rejection rate is very low: for bottles, no grids were rejected in any resampling iteration, and on the intersected sets, rejection rate remained below 10% in all episodes.

Examples of objects from the GAN output distributions and histograms showing the overall distribution of graspability over resampling episodes are shown in Fig. 4. After 3 resampling iterations on the intersected cylinders dataset, the mean normalized graspability is reduced by 22% relative to objects in the original dataset. Similarly, graspability is reduced by 36% on the intersected prisms datset after 4 resampling iterations and by 17% on the ShapeNet bottles dataset after 5 resampling iterations.

C. Shape Similarity

Experiments suggest that both the analytical and the CEM + GAN algorithms decrease the graspability metric, but in different manners. The analytical algorithm maintains local shape similarity through the constraints on surface normal changes, while the GAN introduces geometric changes that decrease graspability while maintaining shape similarity at a more global level (e.g., generates a tapered bottle that resembles a bottle but was not in the original dataset).

To quantify shape similarity, we use 1 iteration of Laplacian smoothing on the generated objects from each of the three datasets by both algorithms to minimize surface roughness and measure the effect of smoothing on object graspability. The objects generated by the analytical algorithm use $\alpha = 10$ degrees for the surface normal deviation constraint. The full results are shown in Table I. Before smoothing, the mean normalized graspability for objects from the analytical algorithm is 10% lower, 30% higher, and 15% lower than objects from the GAN on the intersected cylinders, intersected prisms, and bottles datasets, respectively. After smoothing,

Dataset	Graspability Before Smoothing		Graspability After Smoothing	
	Analytical	CEM + GAN	Analytical	CEM + GAN
Intersected Cylinders	$\mathbf{0.677 \pm 0.031}$	0.783 ± 0.011	0.959 ± 0.048	$\mathbf{0.862 \pm 0.031}$
Intersected Prisms	0.876 ± 0.036	$\mathbf{0.577 \pm 0.012}$	0.961 ± 0.047	$\mathbf{0.777 \pm 0.037}$
ShapeNet Bottles	$\mathbf{0.682 \pm 0.034}$	0.827 ± 0.012	0.980 ± 0.038	$\mathbf{0.899 \pm 0.033}$

TABLE I: Comparison of the normalized mean graspability (reported with 95% confidence intervals) of objects generated by both the analytical algorithm and the GAN algorithm before and after Laplacian smoothing. After smoothing, the objects generated by the GAN have lower graspability metrics than the corresponding objects generated by the analytical algorithm for all three datasets, which suggests that the GAN generates more global adversarial geometries, whereas the analytical algorithm uses local surface roughness to reduce graspability.

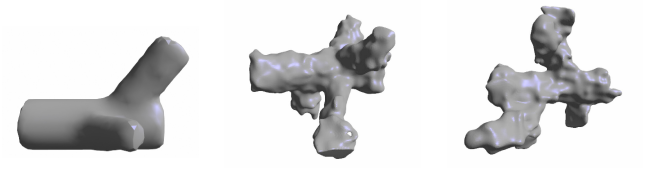

Fig. 5: The left object is from the initial input prior distribution of intersected cylinders, and the others are objects sampled from the GAN's output distribution when it is trained on this prior without spectral normalization. The objects shown have been smoothed via Laplacian smoothing to emphasize that the GAN produces significant structural changes rather than simply adding surface roughness. However, this modified GAN also generates objects that deviate more from the original dataset.

the mean normalized graspability of objects generated by the GAN are lower by 10%, 18%, and 8% on the same datasets. The mean graspability of smoothed objects from the analytical algorithm is at least 95.9% of the original datasets in all cases, suggesting that surface roughness accounts for almost all of the decrease in graspability. Although the GAN also introduces surface roughness (smoothing still increases graspability in all cases), it appears to learn more global geometric changes to decrease graspability.

D. Failure Modes

GANs are prone to mode collapse [43], the phenomenon where a GAN can learns to only outputs one distinct object regardless of the input. Furthermore, since resampling decreases diversity of objects in the dataset due to similar generated objects tending to have similar metric scores, complete mode collapse tends to occur after enough resampling episodes. We observed mode collapse by the 9^{th} iteration on all three datasets.

We experimented with several variations of the GAN architecture and observed that removing spectral normalization can lead to more diverse objects on the intersected cylinders dataset. In this experiment, mode collapse does not occur before the metric quality mean stops improving, reaching a decrease of 83% from the original dataset. However, these generated objects deviate quite significantly from the prior dataset. Some examples are shown in Fig. 5.

VII. Discussion and Future Work

We introduce adversarial grasp objects: objects that look visually similar to existing objects, but decrease the predicted graspability given by a robot grasping policy. We present two algorithms that generate adversarial grasp objects.

The analytical algorithm considers perturbations in randomly sampled directions. We are experimenting with variants of the algorithm that choose perturbation directions more

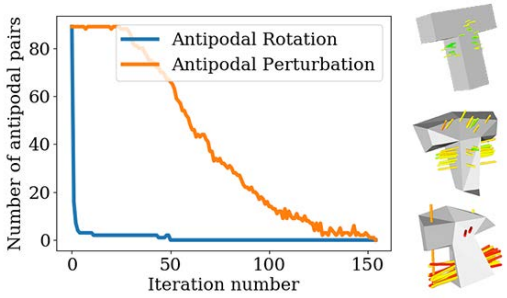

Fig. 6: Results on a T-shaped prism with systematic vertex translation using the directed rotation from the surface normal of one face to another. Left plot: Number of antipodal pairs vs. number of iterations in the algorithm. The antipodal rotation algorithm described in Section VII converges sooner than the antipodal perturbation algorithm described in Section IV-A. The antipodal rotation algorithm takes 0.21 seconds compared to 5.3 seconds for the antipodal perturbation algorithm. Right column: Original object, adversarial grasp object generated by the antipodal perturbation algorithm, and adversarial grasp object generated by the antipodal rotation algorithm.

systematically: for a pair of antipodal faces, we can consider the directed rotation from the surface normal of one face to the other and apply the corresponding rotation matrix to the vertices of each face in a manner that maximizes the antipodality angle. We have some preliminary experiments on simple objects and find that the algorithm is much faster in terms of number of iterations and computation time. In future work, we will extend this algorithm to more complex objects. Preliminary results are shown in Fig. 6.

For the CEM + GAN algorithm, we find that overall, the distributions tend toward thinner objects with fewer parallel surfaces. For example, the resulting distribution of the GAN trained on the bottle prior has primarily thin bottles with both a conical upper portion instead of a cap or stem structure, as well as a tapered main body. Additionally, the metric models point contacts instead of area contacts, which can be disproportionately affected by surface roughness.

In future work, we will explore extensions to different gripper types and to suction grasps and evaluate adversarial objects in physical robot trials.

ACKNOWLEDGEMENT

This research was performed at the AUTOLAB at UC Berkeley in affiliation with the Berkeley AI Research (BAIR) Lab, Berkeley Deep Drive (BDD), the Real-Time Intelligent Secure Execution (RISE) Lab, and the CITRIS "People and Robots" (CPAR) Initiative. The authors were supported in part by donations from Siemens, Google, Toyota Research Institute, Autodesk, Knapp, Honda, Intel, Comcast, Hewlett-Packard and by equipment grants from PhotoNeo, NVidia, and Intuitive Surgical. We thank our colleagues who provided helpful feedback, code, and suggestions, in particular Ajay Tanwani, Michael Laskey, Sanjay Krishnan, Matt Matl, Richard Liaw, Daniel Seita, Mike Danielczuk.

REFERENCES

[1] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. J. Goodfellow, and R. Fergus, "Intriguing properties of neural networks," *CoRR*, vol. abs/1312.6199, 2013. [Online]. Available: http://arxiv.org/abs/1312.6199

[2] N. Papernot, P. D. McDaniel, I. J. Goodfellow, S. Jha, Z. B. Celik, and A. Swami, "Practical black-box attacks against deep learning systems using adversarial examples," *CoRR*, vol. abs/1602.02697, 2016. [Online]. Available: http://arxiv.org/abs/1602.02697

[3] A. Kurakin, I. J. Goodfellow, and S. Bengio, "Adversarial examples in the physical world," *CoRR*, vol. abs/1607.02533, 2016. [Online]. Available: http://arxiv.org/abs/1607.02533

[4] A. Athalye, L. Engstrom, A. Ilyas, and K. Kwok, "Synthesizing robust adversarial examples," *CoRR*, vol. abs/1707.07397, 2017. [Online]. Available: http://arxiv.org/abs/1707.07397

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

[6] 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," *Int. Journal of Robotics Research (IJRR)*, vol. 37, no. 4-5, pp. 421–436, 2018.

[7] L. Pinto and A. Gupta, "Supersizing self-supervision: Learning to grasp from 50k tries and 700 robot hours," in *Proc. IEEE Int. Conf. Robotics and Automation (ICRA)*, 2016.

[8] 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. Robotics: Science and Systems (RSS)*, 2017.

[9] J. Mahler, M. Matl, X. Liu, A. Li, D. Gealy, and K. Goldberg, "Dex-net 3.0: Computing robust vacuum suction grasp targets in point clouds using a new analytic model and deep learning," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2018, pp. 1–8.

[10] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial networks," in *Proc. Advances in Neural Information Processing Systems*, 2014.

[11] J. Mahler, F. T. Pokorny, B. Hou, M. Roderick, M. Laskey, M. Aubry, K. Kohlhoff, T. Kröger, J. Kuffner, and K. Goldberg, "Dex-net 1.0: A cloud-based network of 3d objects for robust grasp planning using a multi-armed bandit model with correlated rewards," in *Proc. IEEE Int. Conf. Robotics and Automation (ICRA)*. IEEE, 2016.

[12] D. Yang, C. Xiao, B. Li, J. Deng, and M. Liu, "Realistic adversarial examples in 3d meshes," *arXiv preprint arXiv:1810.05206*, 2018.

[13] D. Prattichizzo and J. C. Trinkle, "Grasping," in *Springer handbook of robotics*. Springer, 2008, pp. 671–700.

[14] J. Bohg, A. Morales, T. Asfour, and D. Kragic, "Data-driven grasp synthesisa survey," *IEEE Trans. Robotics*, vol. 30, no. 2, pp. 289–309, 2014.

[15] A. Rodriguez, M. T. Mason, and S. Ferry, "From caging to grasping," *Int. Journal of Robotics Research (IJRR)*, p. 0278364912442972, 2012.

[16] C. Goldfeder, M. Ciocarlie, H. Dang, and P. K. Allen, "The columbia grasp database," in *Proc. IEEE Int. Conf. Robotics and Automation (ICRA)*. IEEE, 2009, pp. 1710–1716.

[17] B. León, S. Ulbrich, R. Diankov, G. Puche, M. Przybylski, A. Morales, T. Asfour, S. Moisio, J. Bohg, J. Kuffner, *et al.*, "Opengrasp: a toolkit for robot grasping simulation," in *Proc. IEEE Int. Conf. on Simulation, Modeling, and Programming of Autonomous Robots (SIMPAR)*. Springer, 2010, pp. 109–120.

[18] P. Brook, M. Ciocarlie, and K. Hsiao, "Collaborative grasp planning with multiple object representations," in *Proc. IEEE Int. Conf. Robotics and Automation (ICRA)*. IEEE, 2011, pp. 2851–2858.

[19] M. Ciocarlie, K. Hsiao, E. G. Jones, S. Chitta, R. B. Rusu, and I. A. Şucan, "Towards reliable grasping and manipulation in household environments," in *Experimental Robotics*. Springer, 2014, pp. 241–252.

[20] C. Goldfeder and P. K. Allen, "Data-driven grasping," *Autonomous Robots*, vol. 31, no. 1, pp. 1–20, 2011.

[21] C. Hernandez, M. Bharatheesha, W. Ko, H. Gaiser, J. Tan, K. van Deurzen, M. de Vries, B. Van Mil, J. van Egmond, R. Burger, *et al.*, "Team delft's robot winner of the amazon picking challenge 2016," *arXiv preprint arXiv:1610.05514*, 2016.

[22] S. Hinterstoisser, S. Holzer, C. Cagniart, S. Ilic, K. Konolige, N. Navab, and V. Lepetit, "Multimodal templates for real-time detection of texture-less objects in heavily cluttered scenes," in *Proc. IEEE Int. Conf. on Computer Vision (ICCV)*. IEEE, 2011, pp. 858–865.

[23] B. Kehoe, A. Matsukawa, S. Candido, J. Kuffner, and K. Goldberg, "Cloud-based robot grasping with the google object recognition engine," in *Proc. IEEE Int. Conf. Robotics and Automation (ICRA)*. IEEE, 2013, pp. 4263–4270.

[24] A. Herzog, P. Pastor, M. Kalakrishnan, L. Righetti, J. Bohg, T. Asfour, and S. Schaal, "Learning of grasp selection based on shape-templates," *Autonomous Robots*, vol. 36, no. 1-2, pp. 51–65, 2014.

[25] D. Kappler, J. Bohg, and S. Schaal, "Leveraging big data for grasp planning," in *Proc. IEEE Int. Conf. Robotics and Automation (ICRA)*, 2015.

[26] E. Johns, S. Leutenegger, and A. J. Davison, "Deep learning a grasp function for grasping under gripper pose uncertainty," in *Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*. IEEE, 2016, pp. 4461–4468.

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

[28] K. Bousmalis, A. Irpan, P. Wohlhart, Y. Bai, M. Kelcey, M. Kalakrishnan, L. Downs, J. Ibarz, P. Pastor, K. Konolige, *et al.*, "Using simulation and domain adaptation to improve efficiency of deep robotic grasping," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2018, pp. 4243–4250.

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

[30] A. van den Oord, N. Kalchbrenner, and K. Kavukcuoglu, "Pixel recurrent neural networks," *CoRR*, vol. abs/1601.06759, 2016. [Online]. Available: http://arxiv.org/abs/1601.06759

[31] M. Veres, M. Moussa, and G. W. Taylor, "Modeling grasp motor imagery through deep conditional generative models," *IEEE Robotics and Automation Letters*, vol. 2, no. 2, pp. 757–764, 2017.

[32] Z. Wu, S. Song, A. Khosla, F. Yu, L. Zhang, X. Tang, and J. Xiao, "3d shapenets: A deep representation for volumetric shapes," in *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2015, pp. 1912–1920.

[33] C. Jiang, P. Marcus, *et al.*, "Hierarchical detail enhancing mesh-based shape generation with 3d generative adversarial network," *arXiv preprint arXiv:1709.07581*, 2017.

[34] J. Mahler, B. Hou, S. Niyaz, F. T. Pokorny, R. Chandra, and K. Goldberg, "Privacy-preserving grasp planning in the cloud," in *Proc. IEEE Conf. on Automation Science and Engineering (CASE)*. IEEE, 2016, pp. 468–475.

[35] C. Ferrari and J. Canny, "Planning optimal grasps," in *Proc. IEEE Int. Conf. Robotics and Automation (ICRA)*, 1992, pp. 2290–2295.

[36] J. Friedman, T. Hastie, and R. Tibshirani, *The elements of statistical learning*. Springer series in statistics Springer, Berlin, 2001, vol. 1.

[37] J. H. Lim and J. C. Ye, "Geometric gan," *arXiv preprint arXiv:1705.02894*, 2017.

[38] R. Y. Rubinstein, "Optimization of computer simulation models with rare events," *European Journal of Operational Research*, vol. 99, no. 1, pp. 89 – 112, 1997. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0377221796003852

[39] A. Gupta and J. Zou, "Feedback gan (fbgan) for dna: a novel feedback-loop architecture for optimizing protein functions," *arXiv preprint arXiv:1804.01694*, 2018.

[40] S. Osher and R. Fedkiw, *Level set methods and dynamic implicit surfaces*. Springer Science & Business Media, 2006, vol. 153.

[41] T. Miyato, T. Kataoka, M. Koyama, and Y. Yoshida, "Spectral normalization for generative adversarial networks," *CoRR*, vol. abs/1802.05957, 2018. [Online]. Available: http://arxiv.org/abs/1802.05957

[42] 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.

[43] H. Thanh-Tung, T. Tran, and S. Venkatesh, "On catastrophic forgetting and mode collapse in generative adversarial networks," *arXiv preprint arXiv:1807.04015*, 2018.