Domain Randomization and Generative Models for Robotic Grasping

Josh Tobin[12], Lukas Biewald[3*], Rocky Duan[4*], Marcin Andrychowicz[1], Ankur Handa[5*], Vikash Kumar[2*],
Bob McGrew[1], Alex Ray[1], Jonas Schneider[1], Peter Welinder[1], Wojciech Zaremba[1†], Pieter Abbeel[24†]
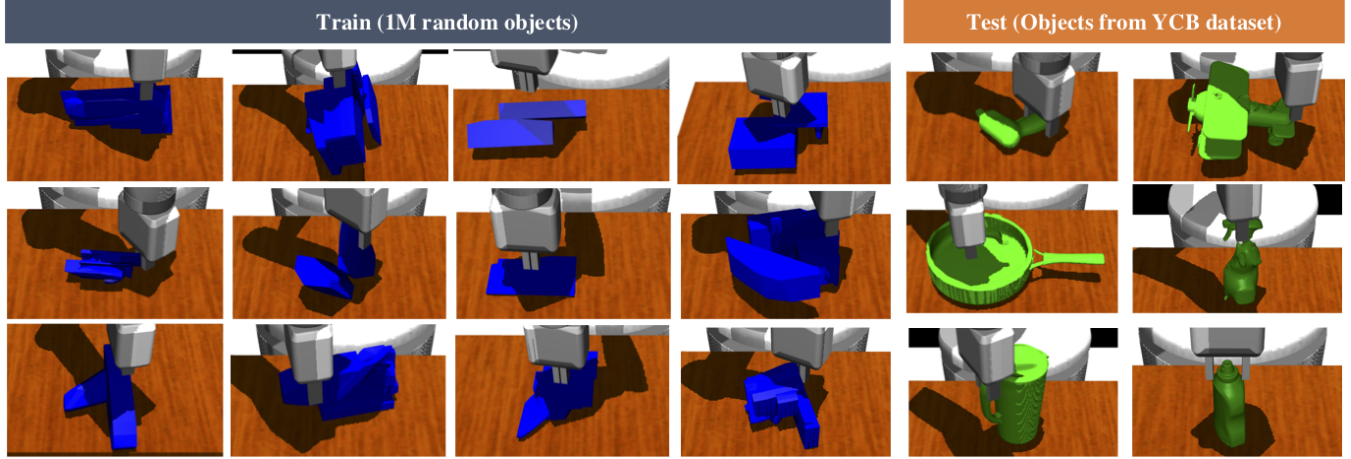
Fig. 1. An overview of our approach. Since creating large numbers of realistic object models is challenging, we train our deep autoregressive model architecture on millions of unrealistic procedurally generated objects (indicated in blue above) and billions of unique grasp attempts. At test time, our model generalizes to realistic objects from the YCB dataset (indicated in green above) [4] with 92% success rate.

Abstract— **Deep learning-based robotic grasping has made significant progress thanks to algorithmic improvements and increased data availability. However, state-of-the-art models are often trained on as few as hundreds or thousands of unique object instances, and as a result generalization can be a challenge.**

In this work, we explore a novel data generation pipeline for training a deep neural network to perform grasp planning that applies the idea of *domain randomization* to object synthesis. We generate millions of unique, unrealistic procedurally generated objects, and train a deep neural network to perform grasp planning on these objects.

Since the distribution of successful grasps for a given object can be highly multimodal, we propose an autoregressive grasp planning model that maps sensor inputs of a scene to a probability distribution over possible grasps. This model allows us to sample grasps efficiently at test time (or avoid sampling entirely).

We evaluate our model architecture and data generation pipeline in simulation and the real world. We find we can achieve a $>90\%$ success rate on previously unseen realistic objects at test time in simulation despite having only been trained on random objects. We also demonstrate an 80% success rate on real-world grasp attempts despite having only been trained on random simulated objects.

I. INTRODUCTION

Robotic grasping remains one of the core unsolved problems in manipulation. The earliest robotic grasping methods used analytical knowledge of a scene to compute an optimal

[1] OpenAI [2] UC Berkeley [3] Weights and Biases, Inc
[4] Embodied Intelligence [5] NVIDIA
* Work done while at OpenAI † Equal advising
Correspondence to josh@openai.com

grasp for an object [1], [31], [32], [38], [39], [45]. Assuming a contact model and a heuristic for the likelihood of success of a grasp, analytical methods can provide guarantees about grasp quality, but they often fail in the real world due to inconsistencies in the simplified object and contact models, the need for accurate 3D models of the objects in question, and sensor inaccuracies [2].

As a result, significant research attention has been given to data-driven grasp synthesis methods [2], [8], [27]–[29], [35], [43]. These algorithms avoid some of the challenges of analytic methods by sampling potential grasps and ranking them according to a learned function that maps sensor inputs to an estimate of a chosen heuristic.

Recently, several works have explored using deep neural networks to approximate the grasp heuristic function [14], [23], [24], [36]. The promise of deep neural networks for learning grasp heuristics is that with diverse training data, deep models can learn features that deal with the edge cases that make real-world grasping challenging.

A core challenge for deep learning grasp quality heuristics is data availability. Due to the difficulty and expense of collecting real-world data and due to the limited availability of high-quality 3D object meshes, current approaches use as few as hundreds or thousands of unique object instances, which may limit generalization. In contrast, ImageNet [18], the standard benchmark for image classification, has about 15M unique images from 22K categories.

In order to increase the availability of training data in simulation, we explore applying the idea of *domain randomization* [41], [46] to the creation of 3D object meshes.

Domain randomization is a technique for learning models that work in a test domain after only training on low-fidelity simulated data by randomizing all non-essential aspects of the simulator. One of the core hypotheses of this work is that by training on a wide enough variety of unrealistic procedurally generated object meshes, our learned models will generalize to realistic objects.

Previous work in deep learning for grasping has focused on learning a function that estimates the quality of a given grasp given observations of the scene. Choosing grasps on which to perform this estimate has received comparatively little attention. Grasps are typically chosen using random sampling or by solving a small optimization problem online. The second goal of this paper is to propose a deep learning-based method for choosing grasps to evaluate. Our hypothesis is that a learned model for grasp sampling will be more likely to find high-quality grasps for challenging objects and will do so more efficiently.

We use an autoregressive model architecture [20], [33], [34] that maps sensor inputs to a probability distribution over grasps that corresponds to the model's weighted estimate of the likelihood of success of each grasp. After training, highest probability grasp according to the distribution succeeds on 89% of test objects and the 20 highest probability grasps contain a successful grasp for 96% of test objects. In order to determine which grasp to execute on the robot, we collect a second observation in the form of an image from the robot's hand camera and train a second model to choose the most promising grasp among those sampled from the autoregressive model, resulting in a success rate of 92%.

The contributions of this paper can be summarized as follows:

- We explore the effect of training a model for grasping using unrealistic procedurally generated objects and show that such a model can achieve similar success to one trained on a realistic object distribution. (Another paper [3] developed concurrently to this one explored a similar idea and reached similar conclusions.)
- We propose a novel generative model architecture and training methodology for learning a sampling distribution for grasps to evaluate.
- We evaluate our object generation, training, and sampling algorithms in simulated scenes and find that we can achieve an 84% success rate on random objects and 92% success rate on previously unseen real-world objects despite training only on non-realistic randomly generated objects.
- We demonstrate that we can deploy these models in real-world grasping experiments with an 80% success rate despite having been trained entirely in simulation.

II. RELATED WORK

A. Domain Randomization

Domain randomization involves randomizing non-essential aspects of the training distribution in order to better generalize to a difficult-to-model test distribution. This idea has been employed in robotics since at least 1997, when Jakobi proposed the "Radical Envelope of Noise Hypothesis", the idea that evolved controllers can be made more robust by completely randomizing all aspects of the simulator that do not have a basis in reality and slightly randomizing all aspects of the simulator that do have a basis in reality [13]. Recently domain randomization has shown promise in transferring deep neural networks for robotics tasks from simulation to the real world by randomizing physics [30] and appearance properties [41], [46], [54].

In another work developed concurrently with this one [3], the authors reach a similar conclusion about the utility of procedurally generated objects for the purpose of robotic grasping. In contrast to this work, theirs focuses on how to combine simulated data with real grasping data to achieve successful transfer to the real world, but does not focus on achieving a high overall success rate. Our paper instead focuses on how to achieve the best possible generalization to novel objects. Ours also has a comparable real-world success rate despite not using any real-world training data.

B. Autoregressive models

This paper uses an autoregressive architecture to model a distribution over grasps conditioned on observations of an object. Autoregressive models leverage the fact that an N-dimensional probability distribution $p(X)$ can be factored as $\prod_{n=1}^{N} p(x_n \mid x_1, \cdots, x_{n-1})$ for any choice of ordering of 1-dimensional variables x_i. The task of modeling the distribution then consists of modeling each $p(x_n \mid x_1, \cdots, x_{n-1})$ [20]. In contrast to Generative Adversarial Networks [9], another popular form for a deep generative model, autoregressive models can directly compute the likelihood of samples, which is advantageous for tasks like grasping in which finding the highest likelihood samples is important. Autoregressive models have been used for density estimation and generative modeling in image domains [7], [10], [20] and have been shown to perform favorably on challenging image datasets like ImageNet [34], [48]. Autoregressive models have also been successfully applied to other forms of data including in topic modeling [19] and audio generation [33].

C. Robotic grasping

Grasp planning methods fall into one of two categories: *analytical* methods and *empirical* methods [42].

Analytical methods use a contact model and knowledge of an object's 3D shape to find grasps that maximize a chosen metric like the ability of the grasp to resist external wrenches [38] or constrain the object's motion [39]. Some methods attempt to make these estimates more robust to gripper and object pose uncertainty and sensor error by instead maximizing the expected value of a metric under uncertainty [15], [50].

Most approaches use simplified Coulomb friction and rigid body modeling for computational tractability [31], [38], but some have explored more realistic object and friction models [1], [37], [40]. Typically, grasps for an object are selected based on sensor data by registering images to a known

database of 3D models using a traditional computer vision pipeline [6], [12], [51] or a deep neural network [11], [53].

Empirical methods instead attempt to maximize the value of a quality metric through sampling [42]. Many approaches use simulation to evaluate classical grasp quality metrics [8], [26], [27], [35]. Others use human labeling or self-supervised learning to measure success [28], [29], [43].

Most techniques estimate the value of the quality metric in real-world trials using traditional computer vision and learning techniques [21], [35], [43], [44]. More recently, deep learning has been employed to learn a mapping directly from sensor observations to grasp quality or motor torques.

D. Deep learning for robotic grasping

Work in deep learning for grasping can be categorized by how training data is collected and how the model transforms noisy observations into grasp candidates.

Some approaches use hand-annotated real-world grasping trials to provide training labels [22]. However, hand-labeling is challenging to scale to large datasets. To alleviate this problem, some work explores automated large-scale data collection [23], [36]. Others have explored replacing real data with synthetic depth data at training time [14], [24], [25], [49], or combining synthetic RGB images with real images [3]. In many cases, simulated data appears to be effective in replacing or supplementing real-world data in robotic grasping. Unlike our approach, previous work using synthetic data uses small datasets of up to a few thousand of realistic object meshes.

One commonly used method for sampling grasps is to learn a visuomotor control policy for the robot that allows it to iteratively refine its grasp target as it takes steps in the environment. Levine and co-authors learn a prediction network $g(I_t, v_t)$ that takes an observation I_t and motor command v_t and outputs a predicted probability of a successful grasp if v_t is executed [23]. The cross-entropy method is used to greedily select the v_t that maximizes g. Viereck and co-authors instead learn a function $d(I_t, v_t)$ that maps the current observation and an action to an estimate of the distance to the nearest successful grasp after performing v_t [49]. Directions are sampled and a constant step size is taken in the direction with the minimum value for d. In contrast to visuomotor control strategies, planning approaches like ours avoid the local optima of greedy execution.

Another strategy to choose a grasp using a deep learning is to sample grasps and score them using a deep neural network of the form $f(I, \boldsymbol{g}) \rightarrow s$, where I are the observation(s) of the scene, $\boldsymbol{g}$ is a selected grasp, and s is the score for the selected grasp [14], [24], [25], [52]. These techniques differ in terms of how they sample grasps to evaluate at test time. Most commonly they directly optimize g using the cross-entropy method [24], [25], [52]. In contrast to these approaches, our approach jointly learns a grasp scoring function and a sampling distribution, allowing for efficient sampling and avoiding exploitation by the optimization procedure of under- or over-fit regions of the grasp score function.

Other approaches take a multi-step approach, starting with a coarse representation of the possible grasps for an object and then exhaustively searching using a learned heuristic [22] or modeling the score function jointly for all possible coarse grasps [14]. Once a coarse grasp is sampled, it is then fine-tuned using a separate network [22] or interpolation [14]. By using an autoregressive model architecture, we are able to directly learn a high- dimensional (20^4 or 20^6-dimensional) multimodal probability distribution.

III. METHOD

Our goal is to learn a mapping that takes one or more observations $I = \{I_j\}$ of a scene and outputs a grasp $\boldsymbol{g}$ to attempt in the scene. The remainder of the section describes the data generation pipeline, model architecture, and training procedure used in our method.
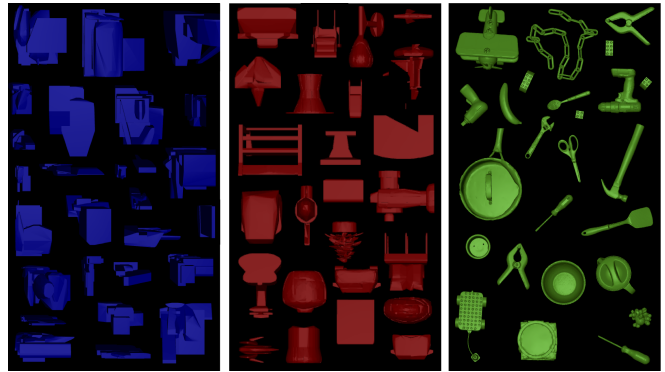
A. Data collection

Fig. 3. Examples of objects used in our experiments. Left: procedurally generated random objects. Middle: objects from the ShapeNet object dataset. Right: objects from the YCB object dataset.

We will describe the process of generating training objects, and then the process of sampling grasps for those objects.

1) Object generation: One of our core hypotheses is that training on a diverse array of procedurally generated objects can produce comparable performance to training on realistic object meshes. Our procedurally generated objects were formed as follows:

1) Sample a random number $n_p \in \{1, \cdots 15\}$
2) Sample n_p primitive meshes from our object primitive dataset
3) Randomly scale each primitive so that all dimensions are between 1 and 15cm
4) Place the meshes sequentially so that each mesh intersects with at least one of the preceding meshes
5) Rescale the final object to approximate the size distribution observed in our real object dataset

To build a diverse object primitive dataset, we took the more than 40,000 object meshes found in the ShapeNet object dataset [5] and decomposed them into more than 400,000 convex parts using V-HACD[1]. Each primitive is one convex part.

[1]https://github.com/kmammou/v-hacd

We compared this object generation procedure against a baseline of training using rescaled ShapeNet objects.

2) Grasp sampling and evaluation: We sample grasps uniformly at random from a discretized 4- or 6-dimensional grasp space (4-dimensional when attention is restricted to upright grasps) corresponding to the (x, y, z) coordinates of the center of the gripper and an orientation of the gripper about that point. We discretize each dimension into 20 buckets. The buckets are the relative location of the grasp point within the bounding box of the object – e.g., an x-value of 0 corresponds to a grasp at the far left side of the object's bounding box and a coordinate of 19 corresponds to a grasp at the far right.

Grasps that penetrate the object or for which the gripper would not contact the object when closed can be instantly rejected. The remainder are evaluated in a physics simulator.

For each grasp attempt, we also collect a depth image from the robot's hand camera during the approach to be used to train the grasp evaluation function.
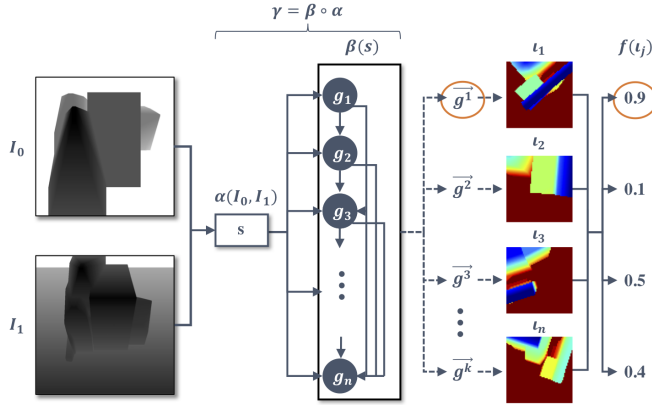
B. Model architecture

Fig. 4.   An overview of sampling from our model architecture. Solid lines represent neural networks, and dotted lines represent sampling operations. The model takes as input one or more observations of the target object in the form of depth images. The images are passed to an image representation module α, which maps the images to an embedding s. The embedding s is the input for the autoregressive module β, which outputs a distribution over possible grasps $\boldsymbol{g}$ for the object by modeling each dimension $\boldsymbol{g}_i$ of the grasp conditioned on the previous dimensions. We sample k high-likelihood grasps $\vec{g}^1, \cdots \vec{g}^k$ from the model using a beam search. For each of those grasps, a second observation is captured that corresponds to an aligned image in the plane of the potential grasp. A grasp scoring model f maps each aligned image to a score. The grasp with the highest score is selected for execution on the robot.

The model architecture used for our experiments is outlined in Figure 4. The model consists of two separate neural networks – a grasp planning module $\gamma(I) = \beta \circ \alpha(I)$ and a grasp evaluation model f. The grasp planning module is used to sample grasps that are likely to be successful. The grasp evaluation model takes advantage of more detailed data in the form of a close-up image from a camera on the gripper of the robot to form a more accurate estimate of the likelihood of each sampled grasp to be successful.

The image representation $s = \alpha(I)$ is formed by passing each image through a separate convolutional neural network.

The flattened output of these convolutional layers are stacked and passed through several dense layers to produce s.

The neural network $\beta(s)$ models a probability distribution $p_\beta(\boldsymbol{g}|s)$ over possible grasps for the object that corresponds to the normalized probability of success of each grasp. The model β consists of n submodules β_i where n is the dimensionality of the grasp. For any grasp $\boldsymbol{g}$, β and $\{\beta_i\}$ are related by

$$\beta(s)(\boldsymbol{g}) = \prod_{i=1}^{n} \beta_i(g_1, \cdots, g_{i-1}, s),$$

where g_i are the dimensions of $\boldsymbol{g}$.

Each β_i is a small neural network taking s and $g_1, \cdots, g_{i-1}$ as input and outputting a softmax over the 20 possible values for g_i, the next dimension of g. We found that sharing weights between the β_i hurts performance at convergence.

The grasp evaluation model f takes as input a single observation from the hand camera of the robot and outputs a single scalar value corresponding to the likelihood of success of that grasp. The model f is parameterized by a convolutional neural network with sigmoid output.

C. Training methodology

Since our method involves capturing depth images from the hand of the robot corresponding to samples from $\gamma(I) = \beta \circ \alpha(I)$, the entire evaluation procedure is not differentiable and we cannot train the model end-to-end using supervised learning. As a result, our training procedure involves independently training γ and f.

Given datasets of objects $D = \{D_1, \cdots D_d\}$, observations $I = \{I^1, \cdots I^d\}$ and successful grasps $G = \{\boldsymbol{g}_1^1, \cdots \boldsymbol{g}_{m_1}^1, \cdots \boldsymbol{g}_1^2, \cdots \boldsymbol{g}_{m_d}^d\}$, γ can be optimized by minimizing the negative log-likelihood of G conditioned on the observations I with respect to the parameters θ of γ, which is given by:

$$J(\theta) = -\frac{1}{d} \sum_{i=1}^{d} \frac{1}{m_i} \sum_{j=1}^{m_i} \log p_\theta(\boldsymbol{g}_j^i \mid I^i).$$

This can be decomposed as

$$-\frac{1}{d} \sum_{i=1}^{d} \frac{1}{m_i} \sum_{j=1}^{m_i} \sum_{k=1}^{n} \log \beta_k \left(\alpha(I^i) \right) \left((\boldsymbol{g}_l^i)_{l<k} \right).$$

This function can be optimized using standard backpropagation and minibatch SGD techniques. [20]

In practice, α is usually a larger model than β and there are often tens or hundreds of successful grasp attempts $\{\boldsymbol{g}_j^i\}$ for a single observation I^i. Therefore it is computationally advantageous to perform the forward pass and gradient calculations for each $\alpha(I^i)$ once for all $\{\boldsymbol{g}_j^i\}$.

This can be achieved in standard neural network and backpropagation libraries by stacking all grasps for a given object so that SGD minibatches consist of pairs $(I_i, \boldsymbol{g}_{\{j\}}^i)$ where $\boldsymbol{g}_{\{j\}}^i$ is the $m_i \times n$ matrix consisting of all successful grasps for object i. To deal with differing values for m_i,

we choose $m = \max(\{m_i\})$ and form an $m \times n$ matrix by padding the $m_i \times n$ matrix with arbitrary values. We can then write the gradient $\nabla_\theta J(\theta)$ of our objective function as follows:

$$-\frac{1}{d}\sum_{i=1}^{d}\frac{1}{m_i}\nabla_\theta\alpha(I^i)\sum_{j=1}^{m}1_j\sum_{k=1}^{n}\nabla_\theta\log\beta_k\left(\alpha(I^i)\right)\left((\boldsymbol{g}_l^i)_{l<k}\right)$$

where 1_j is an indicator function corresponding to whether the entry in j was one of the m_i successful grasps.

This form of the gradient allows us to compute $\nabla_\theta\alpha(I^i)$ once for each I^i, which we found to increase training speed by more than a factor of 10 in our experiments.

The grasp evaluation function f is trained using supervised learning. Inputs are the hand camera images collected during the data collection process and labels are an indicator function corresponding to whether the grasp was successful.

IV. EXPERIMENTS

The goal of our experiments is to answer the following questions:

1) Can grasping models trained on unrealistic randomly generated objects perform as well on novel realistic objects as those trained on realistic objects?
2) How efficiently can we sample grasps from our proposed autoregressive model architecture?
3) How important is using a large number of unique objects for training grasping models?
4) How well do models trained using our methodology work in the real world?

A. Experimental setup

We evaluated our approach by training grasping models on three datasets:

- ShapeNet-1M, a dataset of 1 Million scenes with a single object from the ShapeNet dataset with randomized orientation and object scale.
- Random-1M, a dataset of 1 Million scenes with a single object generated at random using the procedure above.
- ShapeNet-Random-1M, a dataset with 500,000 scenes from each of the previous datasets.

For each object set, we recorded 2,000 grasp attempts per object. This number of grasps was selected so that more than 95% of objects sampled had at least one successful grasp to avoid biasing the dataset to easier objects[2]. For data generation, we used a disembodied Fetch gripper to improve execution speed.

We trained the models using the Adam optimizer [17] with a learning rate of 10^{-4}. We trained each model using three random seeds, and report the average of the three seeds unless otherwise noted.

We evaluated the model on 300 training and hold-out scenes from ShapeNet-1M and Random-1M, as well as 300 scenes generated from the 75 YCB objects with meshes capable of being graseped by our robot's gripper.

[2]This number is not closer to 100% because a small percentage of the random objects in our training set are un-graspable with the Fetch gripper.

All executions were done using a Fetch robot in the MuJoCo physics simulator [47]. When evaluating the model, we sampled $k = 20$ likely grasps from the model using a beam search with beam width 20. Among these grasps, only the one with the highest score according to f is attempted on the robot. An attempt is considered successful if the robot is able to use this grasp to lift the object by 30cm.

B. Performance using randomly generated training data

Figure 5 describes the overall success rate of the algorithm on previously seen and unseen data. The full version of our algorithm is able to achieve greater than 90% success rate on previously unseen YCB objects even when training entirely on randomly generated objects. Training on 1M random objects performs comparably to training on 1M instances of realistic objects.

Training set	ShapeNet Train	ShapeNet Test	Random Train	Random Test	Ycb
ShapeNet-1M	0.91	**0.91**	0.72	0.71	**0.93**
Random-1M	0.91	0.89	**0.86**	**0.84**	0.92
ShapeNet-Random-1M	**0.92**	0.90	0.84	0.81	0.92

Fig. 5. Performance of the algorithm on different synthetic test sets. The full algorithm is able to achieve at least 90% success on previously unseen objects from the YCB dataset when trained on any of the three training sets.

Note that these results were achieved by limiting the robot to grasps in which the gripper is upright. The success rate is around 10% lower across experiments when using full 6-dimensional grasps. Further experimentation could look into whether significantly scaling the amount of training data or using a combination of the 4-dimensional training data and 6-dimensional training data could improve performance.

Figure 6 compares the success rate of our algorithm to several baselines. In particular, our full method performs significantly better than sampling the highest likelihood grasp from the autoregressive model alone.

Training set	ShapeNet Train	ShapeNet Test	Random Train	Random Test	Ycb
Full Algorithm	**0.91**	**0.89**	**0.86**	**0.84**	**0.92**
Autoregressive-Only	0.89	0.86	0.80	0.76	0.89
Random	0.22	0.21	0.10	0.11	0.26
Centroid	0.30	0.25	0.10	0.12	0.54

Fig. 6. Performance of the algorithm compared to baseline approaches. The Full Algorithm and Autoregressive-Only numbers reported are using models trained on random data. The Autoregressive-Only baseline uses the model γ to sample a single high-likelihood grasp, and executes that grasp directly without evaluating it with the model f. The Random baseline samples a random grasp. The centroid baseline deterministically attempts to grasps the center of mass of the object, with the approach angle sampled randomly.

We observed the following three main failure cases for the learned model:

1) For objects that are close to the maximum size graspable by the gripper (10cm), the grasp chosen sometimes collides with the object.
2) In some cases, the model chooses to grasp a curved object at a narrower point to avoid wider points that may cause collision, causing the gripper to slip off.
3) The model cannot find reasonable grasp candidates for some highly irregular objects like a chain found in the YCB dataset.
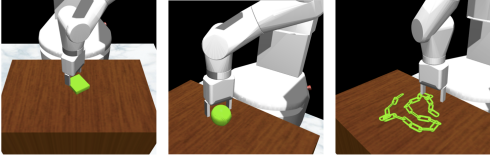
Fig. 7. Examples of three observed failure cases for our learned models. Left: the model chooses a grasp for an object that is close to the maximum size of the gripper that collides with the object. Middle: the model chooses a grasp for a large, highly curved object that causes it to slip off. Right: the model fails to find a promising grasp for a highly irregular object.

The learned models primarily failed on objects for which all features are close in size to the maximum allowed by the gripper. Supplementing the training set with additional objects on the edge of graspability or combining planning with visual servoing could alleviate these cases.

The model also failed for a small number of highly irregular objects like a chain present in the YCB dataset. These failure cases present a larger challenge for the use of random objects in grasping, but additional diversity in the random generation pipeline may mitigate the issue.

C. Efficiency of the autoregressive model

To test how efficiently we are able to sample grasps from our autoregressive model, we looked at the percentage of objects for which the top k most likely grasps according to γ contain at least one successful grasp. Figure 8 shows that the most likely grasp according to the model succeeds close to 90% of the time on YCB objects, and the incremental likelihood of choosing a valid grasp saturates between 10 and 20 samples, motivating our choice of 20 for k. Note that more objects have successful grasps among the 20 sampled than achieve success using our method, suggesting the performance of the grasp evaluator f could be a bottleneck to the overall performance of the algorithm.
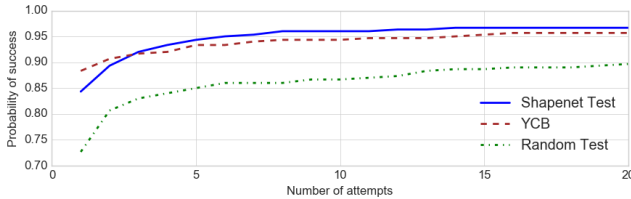

Fig. 8. Percentage of objects that a trained model can grasp successfully as a function of number of the number of samples from the autoregressive model attempted. Here, we sample the 20 grasps from the autoregressive model that have the highest likelihood according to a beam search and count the number of times success occurred on the nth attempt for $n < 20$.

D. Effect of amount of training data

Figure 9 shows the impact of the number of unique objects in the training set on the performance of our models in validation data held out from the same distribution and out-of-sample test data from the YCB dataset. Although with enough data the model trained entirely using randomly generated data performs as well as the models trained using realistic data, with smaller training sets the more realistic object distributions perform significantly better on the test set than does the unrealistic random object distribution.

Note that performance does not appear to have saturated yet in these examples. We conjecture that more training data and more training data diversity could help reduce the effects of the first two failure cases above, but may not allow the model to overcome the third failure case.

Fig. 9. Impact of number of unique training objects used on the performance of the learned model on held out data at test time. Each line represents a different training set. The top chart indicates the performance of the models on held out data from the same distribution (i.e., held out ShapeNet or Random objects depending on the training set), and the bottom chart shows performance on out-of-sample objects from the ShapeNet dataset.

E. Physical robot experiments

We evaluated the ability of models learned on synthetic data using our method to transfer to the real world by attempting to grasp objects from the YCB dataset with a Fetch robot. Figure 10 shows the objects used in our experiments. Models executed in the real-world were the Autoregressive-Only variant of our method trained on the Random-1M dataset with a single depth image from directly above the scene as input. Figure 11 depicts typical grasp executions on the Fetch robot during our experiments.

At test time, the depth input was produced by an Intel RealSense D435 [16]. To model the noise in real-world depth images, we followed the method of Mahler et al. [24], who propose the observation model $I = \alpha \hat{I} + \epsilon$, where $\hat{I}$ is the rendered depth image, α is a Gamma random variable and ϵ is a zero-mean Gaussian random variable.

We tested the learned model on 30 previously unseen objects chosen to capture the diversity of the YCB object dataset. We observed an overall success rate of 80% (24 / 30), which is comparable to the success rate reported on novel objects in other recent papers applying deep learning to parallel jaw robotic grasping [23], [24].

In addition to the failure modes described above, we saw two additional failure modes in real-world experiments. The first was an inability to deal with highly translucent objects like the clear lid in our object dataset. This likely results from the inability of the RealSense to provide accurate readings for

such objects. The second is objects with highly nonuniform densities like the hammer in our dataset. One reason for this failure is likely that our training data pipeline does not generate objects with components of different densities.

A video of our real-world experiments is available on the website for this paper.[3]

Fig. 10. The objects used in our real-world experiments.

Fig. 11. Typical grasp executions on the physical robot.

V. CONCLUSION

We demonstrated that a grasping model trained entirely using non-realistic procedurally generated objects can achieve a high success rate on realistic objects despite no training on a realistic objects. Our grasping model architecture allows for efficient sampling of high-likelihood grasps at evaluation time, with a successful grasp being found for 96% of objects in the first 20 samples. By scoring those samples, we can achieve an overall success rate of 92% on realistic objects on the first attempt. We also demonstrated that models learned using our method can be transferred successfully to the real world.

Future directions that could improve the success rate of the trained models include scaling up to larger training sets, providing the model with feedback from failed grasps to influence further grasp selection, combining our grasp

---

[3]https://sites.google.com/openai.com/domainrandomization-grasping/home

planning module with work on visual servoing for grasping, and incorporating additional sensor modalities like haptic feedback.

Another exciting direction is to explore using domain randomization for generalization in other robotic tasks. If realistic object models are not needed, tasks like pick-and-place, grasping in clutter, and tool use may benefit from the ability to randomly generate hundreds of thousands or millions of 3D scenes.

REFERENCES

[1] Antonio Bicchi and Vijay Kumar. Robotic grasping and contact: A review. In *Robotics and Automation, 2000. Proceedings. ICRA'00. IEEE International Conference on*, volume 1, pages 348–353. IEEE, 2000.

[2] Jeannette Bohg, Antonio Morales, Tamim Asfour, and Danica Kragic. Data-driven grasp synthesisa survey. *IEEE Transactions on Robotics*, 30(2):289–309, 2014.

[3] Konstantinos Bousmalis, Alex Irpan, Paul Wohlhart, Yunfei Bai, Matthew Kelcey, Mrinal Kalakrishnan, Laura Downs, Julian Ibarz, Peter Pastor, Kurt Konolige, et al. Using simulation and domain adaptation to improve efficiency of deep robotic grasping. *arXiv preprint arXiv:1709.07857*, 2017.

[4] Berk Calli, Arjun Singh, Aaron Walsman, Siddhartha Srinivasa, Pieter Abbeel, and Aaron M Dollar. The ycb object and model set: Towards common benchmarks for manipulation research. In *Advanced Robotics (ICAR), 2015 International Conference on*, pages 510–517. IEEE, 2015.

[5] Angel X Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, et al. Shapenet: An information-rich 3d model repository. *arXiv preprint arXiv:1512.03012*, 2015.

[6] Matei Ciocarlie, Kaijen Hsiao, Edward Gil Jones, Sachin Chitta, Radu Bogdan Rusu, and Ioan A Şucan. Towards reliable grasping and manipulation in household environments. In *Experimental Robotics*, pages 241–252. Springer, 2014.

[7] Mathieu Germain, Karol Gregor, Iain Murray, and Hugo Larochelle. Made: masked autoencoder for distribution estimation. In *Proceedings of the 32nd International Conference on Machine Learning (ICML-15)*, pages 881–889, 2015.

[8] Corey Goldfeder, Peter K Allen, Claire Lackner, and Raphael Pelossof. Grasp planning via decomposition trees. In *Robotics and Automation, 2007 IEEE International Conference on*, pages 4679–4684. IEEE, 2007.

[9] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 27*, pages 2672–2680. Curran Associates, Inc., 2014.

[10] Karol Gregor, Ivo Danihelka, Andriy Mnih, Charles Blundell, and Daan Wierstra. Deep autoregressive networks. *arXiv preprint arXiv:1310.8499*, 2013.

[11] Saurabh Gupta, Pablo Arbeláez, Ross Girshick, and Jitendra Malik. Aligning 3d models to rgb-d images of cluttered scenes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4731–4740, 2015.

[12] Stefan Hinterstoisser, Stefan Holzer, Cedric Cagniart, Slobodan Ilic, Kurt Konolige, Nassir Navab, and Vincent Lepetit. Multimodal templates for real-time detection of texture-less objects in heavily cluttered scenes. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, pages 858–865. IEEE, 2011.

[13] Nick Jakobi. Evolutionary robotics and the radical envelope-of-noise hypothesis. *Adaptive behavior*, 6(2):325–368, 1997.

[14] Edward Johns, Stefan Leutenegger, and Andrew J Davison. Deep learning a grasp function for grasping under gripper pose uncertainty. In *Intelligent Robots and Systems (IROS), 2016 IEEE/RSJ International Conference on*, pages 4461–4468. IEEE, 2016.

[15] Ben Kehoe, Akihiro Matsukawa, Sal Candido, James Kuffner, and Ken Goldberg. Cloud-based robot grasping with the google object recognition engine. In *Robotics and Automation (ICRA), 2013 IEEE International Conference on*, pages 4263–4270. IEEE, 2013.

[16] Leonid Keselman, John Iselin Woodfill, Anders Grunnet-Jepsen, and Achintya Bhowmik. Intel realsense stereoscopic depth cameras. *arXiv preprint arXiv:1705.05548*, 2017.

[17] Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[18] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[19] Hugo Larochelle and Stanislas Lauly. A neural autoregressive topic model. In *Advances in Neural Information Processing Systems*, pages 2708–2716, 2012.

[20] Hugo Larochelle and Iain Murray. The neural autoregressive distribution estimator. In *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, pages 29–37, 2011.

[21] Quoc V Le, David Kamm, Arda F Kara, and Andrew Y Ng. Learning to grasp objects with multiple contact points. In *Robotics and Automation (ICRA), 2010 IEEE International Conference on*, pages 5062–5069. IEEE, 2010.

[22] Ian Lenz, Honglak Lee, and Ashutosh Saxena. Deep learning for detecting robotic grasps. *The International Journal of Robotics Research*, 34(4-5):705–724, 2015.

[23] Sergey Levine, Peter Pastor, Alex Krizhevsky, Julian Ibarz, and Deirdre Quillen. Learning hand-eye coordination for robotic grasping with deep learning and large-scale data collection. *The International Journal of Robotics Research*, page 0278364917710318, 2016.

[24] Jeffrey Mahler, Jacky Liang, Sherdil Niyaz, Michael Laskey, Richard Doan, Xinyu Liu, Juan Aparicio Ojea, and Ken Goldberg. Dex-net 2.0: Deep learning to plan robust grasps with synthetic point clouds and analytic grasp metrics. *arXiv preprint arXiv:1703.09312*, 2017.

[25] Jeffrey Mahler, Matthew Matl, Xinyu Liu, Albert Li, David Gealy, and Ken Goldberg. Dex-net 3.0: Computing robust robot suction grasp targets in point clouds using a new analytic model and deep learning. *arXiv preprint arXiv:1709.06670*, 2017.

[26] Andrew T Miller and Peter K Allen. Graspit! a versatile simulator for robotic grasping. *IEEE Robotics & Automation Magazine*, 11(4):110–122, 2004.

[27] Andrew T Miller, Steffen Knoop, Henrik I Christensen, and Peter K Allen. Automatic grasp planning using shape primitives. In *Robotics and Automation, 2003. Proceedings. ICRA'03. IEEE International Conference on*, volume 2, pages 1824–1829. IEEE, 2003.

[28] Luis Montesano, Manuel Lopes, Alexandre Bernardino, and José Santos-Victor. Learning object affordances: from sensory–motor coordination to imitation. *IEEE Transactions on Robotics*, 24(1):15–26, 2008.

[29] Antonio Morales, Eris Chinellato, Andrew H Fagg, and Angel P Del Pobil. Using experience for assessing grasp reliability. *International Journal of Humanoid Robotics*, 1(04):671–691, 2004.

[30] Igor Mordatch, Kendall Lowrey, and Emanuel Todorov. Ensemble-cio: Full-body dynamic motion planning that transfers to physical humanoids. In *Intelligent Robots and Systems (IROS), 2015 IEEE/RSJ International Conference on*, pages 5307–5314. IEEE, 2015.

[31] Richard M Murray, Zexiang Li, S Shankar Sastry, and S Shankara Sastry. *A mathematical introduction to robotic manipulation*. CRC press, 1994.

[32] Van-Duc Nguyen. Constructing force-closure grasps. *The International Journal of Robotics Research*, 7(3):3–16, 1988.

[33] Aaron van den Oord, Sander Dieleman, Heiga Zen, Karen Simonyan, Oriol Vinyals, Alex Graves, Nal Kalchbrenner, Andrew Senior, and Koray Kavukcuoglu. Wavenet: A generative model for raw audio. *arXiv preprint arXiv:1609.03499*, 2016.

[34] Aaron van den Oord, Nal Kalchbrenner, and Koray Kavukcuoglu. Pixel recurrent neural networks. *arXiv preprint arXiv:1601.06759*, 2016.

[35] Raphael Pelossof, Andrew Miller, Peter Allen, and Tony Jebara. An svm learning approach to robotic grasping. In *Robotics and Automation, 2004. Proceedings. ICRA'04. 2004 IEEE International Conference on*, volume 4, pages 3512–3518. IEEE, 2004.

[36] Lerrel Pinto, James Davidson, and Abhinav Gupta. Supervision via competition: Robot adversaries for learning tasks. *arXiv preprint arXiv:1610.01685*, 2016.

[37] Domenico Prattichizzo, Monica Malvezzi, Marco Gabiccini, and Antonio Bicchi. On the manipulability ellipsoids of underactuated robotic hands with compliance. *Robotics and Autonomous Systems*, 60(3):337–346, 2012.

[38] Domenico Prattichizzo and Jeffrey C Trinkle. Grasping. In *Springer handbook of robotics*, pages 955–988. Springer, 2016.

[39] Alberto Rodriguez, Matthew T Mason, and Steve Ferry. From caging to grasping. *The International Journal of Robotics Research*, 31(7):886–900, 2012.

[40] Carlos Rosales, Raúl Suárez, Marco Gabiccini, and Antonio Bicchi. On the synthesis of feasible and prehensile robotic grasps. In *Robotics and Automation (ICRA), 2012 IEEE International Conference on*, pages 550–556. IEEE, 2012.

[41] Fereshteh Sadeghi and Sergey Levine. (cad) 2 rl: Real single-image flight without a single real image. *arXiv preprint arXiv:1611.04201*, 2016.

[42] Anis Sahbani, Sahar El-Khoury, and Philippe Bidaud. An overview of 3d object grasp synthesis algorithms. *Robotics and Autonomous Systems*, 60(3):326–336, 2012.

[43] Ashutosh Saxena, Justin Driemeyer, and Andrew Y Ng. Robotic grasping of novel objects using vision. *The International Journal of Robotics Research*, 27(2):157–173, 2008.

[44] Ashutosh Saxena, Lawson LS Wong, and Andrew Y Ng. Learning grasp strategies with partial shape information. In *AAAI*, volume 3, pages 1491–1494, 2008.

[45] John D Schulman, Ken Goldberg, and Pieter Abbeel. Grasping and fixturing as submodular coverage problems. In *Robotics Research*, pages 571–583. Springer, 2017.

[46] Josh Tobin, Rachel Fong, Alex Ray, Jonas Schneider, Wojciech Zaremba, and Pieter Abbeel. Domain randomization for transferring deep neural networks from simulation to the real world. *arXiv preprint arXiv:1703.06907*, 2017.

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

[48] Aaron van den Oord, Nal Kalchbrenner, Lasse Espeholt, koray kavukcuoglu, Oriol Vinyals, and Alex Graves. Conditional image generation with pixelcnn decoders. In D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems 29*, pages 4790–4798. Curran Associates, Inc., 2016.

[49] Ulrich Viereck, Andreas ten Pas, Kate Saenko, and Robert Platt. Learning a visuomotor controller for real world robotic grasping using easily simulated depth images. *arXiv preprint arXiv:1706.04652*, 2017.

[50] Jonathan Weisz and Peter K Allen. Pose error robust grasping from contact wrench space metrics. In *Robotics and Automation (ICRA), 2012 IEEE International Conference on*, pages 557–562. IEEE, 2012.

[51] Ziang Xie, Arjun Singh, Justin Uang, Karthik S Narayan, and Pieter Abbeel. Multimodal blending for high-accuracy instance recognition. In *Intelligent Robots and Systems (IROS), 2013 IEEE/RSJ International Conference on*, pages 2214–2221. IEEE, 2013.

[52] Xinchen Yan, Mohi Khansari, Yunfei Bai, Jasmine Hsu, Arkanath Pathak, Arbhinav Gupta, James Davidson, and Honglak Lee. Learning grasping interaction with geometry-aware 3d representations. *arXiv preprint arXiv:1708.07303*, 2017.

[53] Andy Zeng, Kuan-Ting Yu, Shuran Song, Daniel Suo, Ed Walker, Alberto Rodriguez, and Jianxiong Xiao. Multi-view self-supervised deep learning for 6d pose estimation in the amazon picking challenge. In *Robotics and Automation (ICRA), 2017 IEEE International Conference on*, pages 1386–1383. IEEE, 2017.

[54] Fangyi Zhang, Jürgen Leitner, Michael Milford, and Peter Corke. Sim-to-real transfer of visuo-motor policies for reaching in clutter: Domain randomization and adaptation with modular networks. *arXiv preprint arXiv:1709.05746*, 2017.