2022 IEEE International Conference on Robotics and Automation (ICRA)
May 23-27, 2022. Philadelphia, PA, USA

FFHNet: Generating Multi-Fingered Robotic Grasps for Unknown Objects in Real-time

Vincent Mayer[1,2,†], Qian Feng[1,2,†], Jun Deng[1], Yunlei Shi[1],
Zhaopeng Chen[*1], Alois Knoll[2]

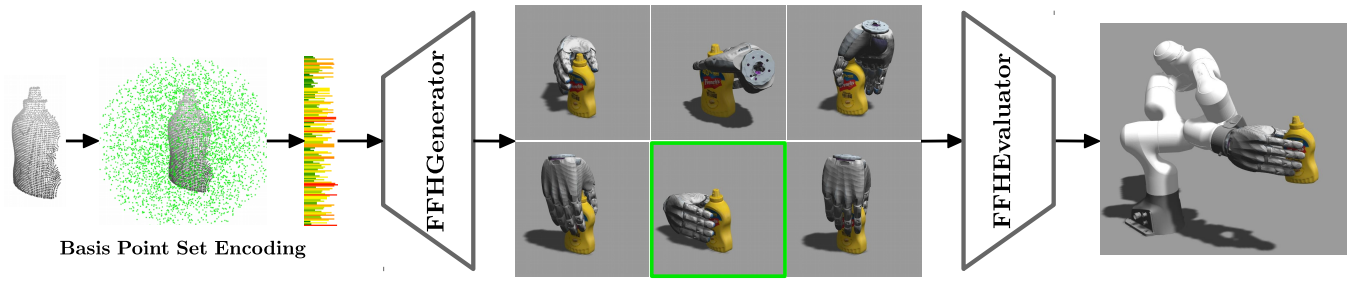


Fig. 1: Our proposed FFHNet consists of the FFHGenerator and the FFHEvaluator. The FFHGenerator receives a BPS-encoded partial object point cloud and generates a diverse set of grasp proposals. The FFHEvaluator ranks the generated proposals, such that the best proposal can be executed.

***Abstract*— Grasping unknown objects with multi-fingered hands at high success rates and in real-time is an unsolved problem. Existing methods are limited in the speed of grasp synthesis or the ability to synthesize a variety of grasps from the same observation. We introduce Five-finger Hand Net (FFHNet), an ML model which can generate a wide variety of high-quality multi-fingered grasps for unseen objects from a single view. Generating and evaluating grasps with FFHNet takes only 30ms on a commodity GPU. To the best of our knowledge, FFHNet is the first ML-based real-time system for multi-fingered grasping with the ability to perform grasp inference at 30 frames per second (FPS). For training, we synthetically generate 180k grasp samples for 129 objects. We are able to achieve 91% grasping success for unknown objects in simulation and we demonstrate the model's capabilities of synthesizing high-quality grasps also for real unseen objects.**

I. Introduction

Grasping is a crucial skill for advanced robotic applications and has huge implications for fields like assistive robotics, manufacturing, or logistics. Nevertheless, grasping largely remains an open problem in robotics. In this work we aim to equip the DLR-HIT Hand II, a 15 degrees of freedom (DOF) anthropomorphic hand [1], with advanced grasping capabilities. Multi-fingered hands like the DLR-HIT Hand II can outperform two-jaw grippers in terms of dexterity and universality. Various works have demonstrated the hands' capabilities on complex grasping and manipulation tasks [2], [3]. However, these works had a human in the loop making the crucial decision of where to place the hand and fingers in relation to the object. Recent years have seen rapid advances in autonomous grasping (i.e. no human in the loop) with two-jaw grippers [4]–[7], but much less in terms of multi-fingered grasping.

Analytical methods for robotic grasping formulate a constrained optimization problem over grasp quality metrics. These methods rely on strong assumptions like accurate object models and simplified contact interactions [8]. In real-world settings, they tend to synthesize fragile grasps and handle partial data and sensing inaccuracies insufficiently [9]–[11]. Data-driven methods can alleviate some of these assumptions. They rely on a pre-calculated database of object models and corresponding grasps. Online object grasping is reduced to object detection, pose estimation, and finding the closest match in the grasp database [12], [13]. Matching against a database tends to work well for known or familiar objects but not partial data of unknown objects. To handle unknown objects and partial data, researchers have capitalized on learning-based methods, especially deep learning. Most state-of-the-art learning methods follow either a grasp sampling and evaluation [14]–[19] or end-to-end framework [20]–[24]. They largely differ in which aspects of the pipelines are learned. One line of work combines heuristics-based grasp sampling with a learning-based grasp evaluation function [4], [15]. The learned function is used to optimize the sampled grasps. This procedure is time-consuming and limited in the variety of sampled grasps. Some end-to-end methods directly predict a grasp from the observed data [20], [25]. These approaches are inherently limited by the one-to-one mapping assumption between object observations and grasps.

In this work, we follow the grasp sampling and evaluation framework (Fig. 1) and present Five-finger Hand Net (FFHNet), a generative and discriminative model capable of synthesizing a variety of multi-fingered grasps from a single partial point cloud in real-time. FFHNet is inspired by [7] and [26] with some key distinctions. In contrast to both [7] and [26] we use no time-consuming post-processing steps, which makes our model 7x faster. Further, we train the model for multi-fingered grasping when [7] uses a two-jaw gripper and [26] generates human-hand meshes for realistic hand-object interactions. Our contributions can be summarized as:

- We introduce a generative model called *FFHGenerator*, a conditional Variational Autoencoder (cVAE) capable of sampling diverse distributions of grasps for unseen

[†] These authors have contributed equally to this work.
[1] Agile Robots AG, {vincent.mayer, qian.feng}@agile-robots.com
[2] Chair for Robotics and Embedded Systems, Technische Universität München

2022 IEEE International Conference on Robotics and Automation (ICRA) | 978-1-7281-9681-7/22/$31.00 ©2022 IEEE | DOI: 10.1109/ICRA46639.2022.9811666

objects in roughly 10 ms, and a discriminative model called *FFHEvaluator* able to successfully predict success for the generated grasps in roughly 20 ms.
- We evaluate the proposed system with unknown objects in simulation and achieve up to 91% success. We also provide insights on the model's performance for real noisy point clouds of YCB [27] objects.
- We generate a new synthetic grasp *dataset* containing 180k grasps for 129 household objects from the BIGBIRD [28] and KIT [29] datasets, along with the automatic data generation and labeling pipeline.

II. Related Work

A. Grasp Sampling

Geometry-based sampling methods require hand-crafted geometric constraints and heuristics to generate grasp proposals. Grasp poses for two-jaw grippers are sampled from a depth image in [4] and directly from a point cloud in [5], [6]. These sampling schemes do not scale to the extra dimensions required for multi-fingered hands. Lei et al. [30] propose a fast geometry-based sampler by representing their three-finger hand as a C-shape and fitting it to the object point cloud. Lu et al. [15], [16] use a heuristic sampler based on the normals of a fitted 3D bounding box to generate initial grasp configurations, which are later optimized. The mentioned methods tend to generate less diverse grasp proposals. To overcome this, Avigal et al. [31] use 3D reconstruction to render depth images from multiple viewpoints and extend the work from [4] to 6D grasps for two-jaw-grippers. In [32], reconstructed meshes are fed into GraspIt! [33] for sampling grasp poses. Reconstruction can enable more diverse grasp samples but increases computational complexity significantly. Overall, geometry-based sampling methods can perform well but require hand-crafted constraints and are less applicable to unknown objects.

Learning-based sampling methods mainly utilize one of three models: Gaussian Mixture Models (GMMs), cVAEs, or Generative Adversarial Networks (GANs). The methods from [17], [19] both use GMMs. The mixture components are predicted by a neural network based on RGB-D or 3D voxel data, an architecture called Mixture Density Network (MDN) [34]. MDNs suffer from a linear mapping between components and grasp configuration. We represent the mapping from latent space to grasp configuration by a neural network, which can be arbitrarily complex.

The work in [35] trains a GAN to predict a three-fingered grasp given RGB-D data. They predict only a coarse grasp type and require iterative refinement steps. The refinement depends on a slow (8 s) shape completion algorithm. Our model predicts full grasps in real-time. A grasp type and 6D palm pose are also predicted by the GAN in [36], a method that allows only one of four fixed grasps. Both [35] and [36] manually label the grasp type of hundreds of grasps, which is inefficient and not scalable. Our data labeling is automatic.

The model presented by [7] is conceptually similar to ours with some key distinctions: 1.) they use a two-jaw gripper and only predict end-effector poses, our model additionally predicts the full 15 DOF, 2.) they use PointNet++ [37] to process the point cloud which needs multiple days of training while we use a multilayer perceptron (MLP) architecture which can be trained on a low-cost GPU in just 3 h, and 3.) they perform multiple incremental refinement steps taking up to 2 s, while our model takes 30 ms in total. Veres et al. [38] also use a cVAE, but predict contact points and normals. Their method can not rank generated grasps and relies on a subsequent inverse kinematics solver.

B. Grasp Evaluation

Not all grasps generated by grasp sampling methods are equally successful. Therefore, a grasp evaluation model is used to rank or refine generated grasps. Many of the relevant works focus on two-jaw grippers [4], [5], [18], [39]. For multi-fingered hands, the authors in [15] learn a function of grasping success given a grasp and 3D object voxelization. Iterative maximization of the learned success function is used to refine an initial grasp. Such gradient-based optimization is time-consuming (2-3 s). The grasp evaluator model is extended further with a novel reconstruction network in [16]. This increases runtime to more than 30 s. In [19] the grasps sampled from a GMM-based sampler are evaluated through a network that predicts force-closure. However, such grasp metrics have limitations for predicting real-world success [9], [10]. Our evaluator directly predicts the success of lifting an object given a grasp and object point cloud.

C. End-to-end

Direct Regression. A novel model introduced in [22] takes object and end-effector properties as input and directly predicts grasping contact points in 200ms. Their method only works for end-effectors with three or fewer fingers and the predicted contact points might not be kinematically possible. Schmidt et al. in [25] directly output the 6D wrist pose given an RGB-D image. They only predict one grasp per observation and the joint configurations are not predicted. The work from [21] predicts both the palm pose and joint configurations from multi-view depth images. Obtaining multi-view observations is however costly. In general, direct regression methods can achieve fast run time with a direct mapping from observation to a single grasp pose, but the inherent one-to-one mapping assumption severely limits their applicability.

Heatmap. Heatmap approaches output predictions per pixel or voxel that each represents a grasp pose for one observation. Pixel-wise predictions of individual finger contact points from an RGB-D image are presented in [40]. It achieved good performance but relies on GraspIt! from [33] as an external planner. In general, pixel- or voxel-wise ground truth training labels are difficult to obtain.

Reinforcement Learning (RL). is used by [23] and [24] to solve the multi-fingered grasping problem. RL methods require careful hyperparameter-tuning, are difficult to train, and do not scale well to the high-dimensional action spaces.

D. 3D Deep Learning

Typical deep learning architectures for learning on 3D point clouds are PointNet [41] and PointNet++ [37]. Lots of works utilize these architectures for grasp generation or evaluation [7], [42]–[45], but so far only for two-jaw grippers. The main drawback of the PointNet architecture is the high computational demand. Alternatively, there are 3D CNN-based methods for voxelized point cloud data, which suffer from the conflict between accuracy and efficiency.

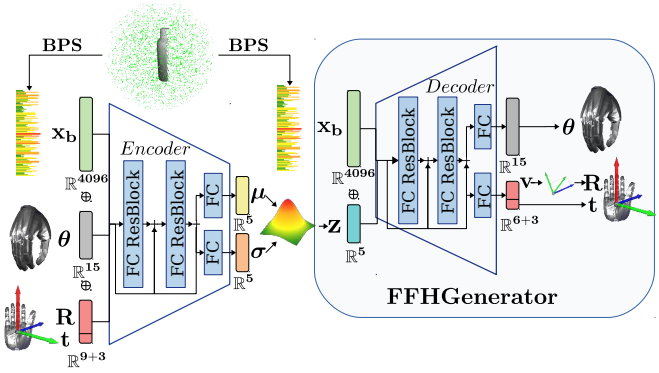


Fig. 2: Given an encoded object observation $\boldsymbol{x_b} \in \mathbb{R}^{4096}$, joint configuration $\boldsymbol{\theta} \in \mathbb{R}^{15}$, palm rotation $\mathbf{R} \in \mathbb{R}^{3\times3}$ and translation $\mathbf{t} \in \mathbb{R}^3$, the encoder maps the distribution of grasps for an object observation into a latent space following a univariate Gaussian distribution. During inference, samples from the latent space can be decoded into grasps.

A method is proposed in [46] to encode 3D point clouds as distances to a fixed set of randomly sampled 3D basis points (Basis Point Set or BPS). The BPS is shown in green in Fig.1. The BPS encoding achieves similar performance to PointNet++ while using three orders of magnitude less computation. We employ the BPS method to encode the input point clouds of FFHNet.

III. Method

A. Problem Statement

The goal of this work is to generate a wide variety of successful multi-fingered grasps $\mathbf{g}$ from a partial point cloud observation $\{P_i^O\}^N \in \mathbb{R}^{N\times3}$ of an object. We assume a successful segmentation of the object's point cloud from the depth data. Grasping success is defined as the ability for the DLR-HIT Hand II to lift the object 20 cm above its resting position without slippage. A grasp $\mathbf{g}$ is represented by the 15-DOF hand joint configuration $\boldsymbol{\theta} \in \mathbb{R}^{15}$ and the 6D pose $(\mathbf{R}, \mathbf{t}) \in SE(3)$ of the palm. In the following subsections, we present our generative and discriminative model Five-finger Hand Net (FFHNet) inspired by the work in [26] on generating realistically-looking hand-object interactions.

B. Grasp Generation

Our grasp generation model, the FFHGenerator shown in Fig. 2, follows the computational framework of a cVAE [47]. The model is trained to maximize the likelihood $p(\mathbf{g}|\mathbf{x_b})$ of a successful grasp $\mathbf{g}$ conditioned on a BPS-encoded object point cloud observation $\mathbf{x_b}$. The generative probabilistic model can be expressed as:

$$p(\mathbf{g}|\mathbf{x_b}) = \int p(\mathbf{z})p(\mathbf{g}|\mathbf{x_b}, \mathbf{z})d\mathbf{z} \tag{1}$$

Given a set of latent samples $\mathbf{z}$, the model in Eq.1 allows associating the same object observation with a variety of grasps. This variety of grasps can be subsequently exploited to fulfill environmental or reachability constraints. Grasping is inherently a one-to-many mapping between observation and grasp as objects can be grasped in many different, equally valid ways. Solving Eq. 1, however, requires integrating over all possible values of the latent variable $\mathbf{z}$ which is intractable.

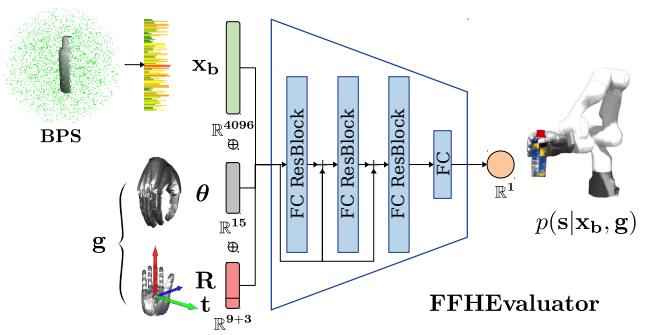


Fig. 3: The FFHEvaluator predicts the probability that a candidate grasp $\mathbf{g}$ for a given observation of an object $\mathbf{x_b}$ will result in success s.

The *encoder* substitutes the intractable posterior $p(\mathbf{z}|\mathbf{x_b}, \mathbf{g})$ with a tractable distribution $q_\phi(\mathbf{z}|\mathbf{x_b}, \mathbf{g})$ mapping different grasps for the same observation into different regions of the latent space. For the probability density of the latent space $p(\mathbf{z})$ we assume a zero-mean, univariate Gaussian $\mathcal{N}(\mathbf{0}, \mathbf{I})$. The *encoder* receives as inputs the encoded object observation $\boldsymbol{x_b} \in \mathbb{R}^{4096}$, the hand joint configuration $\boldsymbol{\theta} \in \mathbb{R}^{15}$, the rotation $\mathbf{R} \in \mathbb{R}^{3\times3}$ and translation $\mathbf{t} \in \mathbb{R}^3$ of the palm relative to the object centroid frame. The *encoder* outputs the mean $\boldsymbol{\mu}$ and variance $\boldsymbol{\sigma}$ for each latent dimension's univariate Gaussian. Through the reparameterization trick introduced in [48] a latent variable sample $\mathbf{z}$ can be obtained from the *encoder's* output via:

$$\mathbf{z} = \boldsymbol{\mu} + \boldsymbol{\sigma} \odot \boldsymbol{\epsilon}, \ \boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}) \tag{2}$$

The *decoder* receives a latent sample $\mathbf{z}$ and the encoded point cloud $\boldsymbol{x_b}$ as input and maps these to three outputs: the reconstructed joint configuration $\boldsymbol{\theta} \in \mathbb{R}^{15}$, two 3D vectors $\mathbf{v} \in \mathbb{R}^6$ representing the palm rotation and the reconstructed palm translation $\mathbf{t} \in \mathbb{R}^3$. The continuous, sufficiently compact representation $\mathbf{v}$ is used instead of the full rotation matrix $\mathbf{R}$ because rotation matrices, as well as other representations such as Euler angles and quaternions, suffer from discontinuities or redundancies detrimental to neural network-based learning [49]. The method introduced in [49] is similar to the well-known Gram-Schmidt procedure for orthogonalization and used to construct the full rotation matrix $\mathbf{R}$. The optimization objective of the FFHGenerator is as follows:

$$\mathcal{L}_{Gen} = \sum_{\mathbf{z}\sim q_\phi,\, \mathbf{g}\sim G^*} \mathcal{L}(\hat{\mathbf{g}}, \mathbf{g}) - \alpha \mathcal{D}_{KL}[q_\phi(\mathbf{z}|\mathbf{x_b}, \mathbf{g}), \mathcal{N}(\mathbf{0}, \mathbf{I})] \tag{3}$$

During training the model learns to minimize the distance between a given grasp g from the data and the reconstructed grasp $\hat{g}$. This is represented by the first part of Eq. 3 $\mathcal{L}(\hat{\mathbf{g}}, \mathbf{g})$. We use the L2-distance between reconstructed and ground truth grasp as follows:

$$\mathcal{L}(\hat{\mathbf{g}}, \mathbf{g}) = \frac{1}{n}\sum (w_R\|\hat{\mathbf{R}} - \mathbf{R}\|^2 + w_t\|\hat{\mathbf{t}} - \mathbf{t}\|^2 + w_\theta\|\hat{\boldsymbol{\theta}} - \boldsymbol{\theta}\|^2) \tag{4}$$

Setting $w_R = 1$, $w_t = 100$ and $w_\theta = 10$ results in equal magnitudes of all components of the reconstruction loss. The second part of Eq. 3, the KL-divergence, weighted by α, conceptually ensures that different grasps for the same object observation are encoded into different regions of the latent

space while ensuring that the latent space encoding follows a univariate Gaussian distribution.

C. Grasp evaluation

A high-capacity cVAE like the FFHGenerator with a continuous latent space however interpolates between these modes generating unsuccessful grasps. This raises the necessity for our grasp evaluation model, the FFHEvaluator depicted in Fig. 3, which is able to distinguish between successful and unsuccessful grasps. In contrast to the FFHGenerator, the FFHEvaluator sees successful and unsuccessful pairs of object observation and grasp during training.

The FFHEvaluator takes a grasp $\mathbf{g}$ and an encoded partial point cloud $\mathbf{x_b}$ as input and predicts the probability $p(s|\mathbf{x_b}, \mathbf{g})$ that a given grasp for a given object point cloud results in a successful lifting motion of the object. The goal of the FFHEvaluator is to approximate the binary classification problem of grasping success by correctly predicting the probability of success for a given grasp candidate:

$$\mathcal{L}_{Eva} = -\left(y \log(s) + (1-y)\log(1-s)\right) \tag{5}$$

In Eq. 5, s represents the success probability as predicted by the FFHEvaluator, while y is a binary variable indicating the true success label.

D. Model Training

For both networks, we use a learning rate of $1{\times}10^{-4}$ and the Adam optimizer [50]. Each batch contains 5012 different encoded point cloud observations of a random object in a random position. The object position is constrained to be reachable by the robot and observable by the camera (compare Fig. 4). For the FFHGenerator, each observation is associated with one random successful grasp from the ground truth distribution of grasps for the object. Each batch for the FFHEvaluator consists of 30% successful grasps, 30% unsuccessful grasps and 40% hard negative grasps. The successful and unsuccessful grasping examples are part of the generated dataset. The hard-negative examples are generated by sufficiently perturbing positive examples. For this, we draw inspiration from [7] and add $\pm$3cm to the position of the palm pose and $\pm$0.6rad to its Euler-angle orientation. The coefficient of the KL-loss term in Eq. 3 is set to 0 for the first 2 epochs to prevent early posterior collapse and afterwards set to $5{\times}10^{-3}$. The evaluator is trained for a total of 40 epochs and the generator for 30 epochs.

E. Implementation Details

The purely fully-connected architecture of FFHNet is significantly faster to train than PointNet++ [37] while being proven equally capable when used with the BPS encoding [46]. We use skip connections [51] from each input to each fully-connected residual block (FC ResBlock) or fully-connected (FC) layer. The connections are detailed in Figure 2. The core building block of both models is the FC ResBlock, which consists of two parallel paths from input to output. One path consists of a single FC layer, the other path has two FC layers. Each is followed by a layer of batch norm (BN). The inputs of the encoder $(\mathbf{x_b}, \boldsymbol{\theta}, \mathbf{R}, \mathbf{t})$ and the conditional input $\mathbf{x_b}$ are also pre-processed by BN.

(a) Simulation setup (b) Segmentation (c) Pose sampling

Fig. 4: We show the simulation setup in (a) with a Realsense camera on the left, a Panda robot arm equipped with a DLR-HIT Hand II. (b) shows point cloud segmentation via RANSAC. In (c) a sampled 6D grasp pose is visualized as a coordinate frame.

The leaky rectified linear unit (LeakyRELU) is used as an activation function for each layer. We found two FC Resblocks for the FFHGenerator and three FC Resblocks for the FFHEvaluator achieved the best performance. The output of the FFHEvaluator's final layer is fed through a sigmoid activation function.

IV. Grasp Data Generation

For generating automatically labeled high-quality synthetic data, we use Gazebo9 with the DART engine.

A. Object datasets

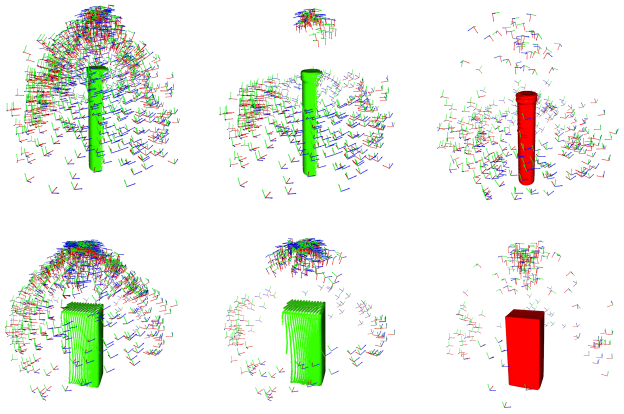
We combine the BIGBIRD [28], KIT [29] and YCB [27] object datasets. Since the YCB objects are physically available, we do not use them for synthetic data generation but experiments on real objects. The BIGBIRD and KIT datasets span around 280 objects. These objects are filtered for their graspability and object type leaving 129 graspable objects.

B. Data generation pipeline

In total, we execute around 180k grasps of which 30k resulted in success. Fig. 4(a) shows our simulation setup. We use a Panda robot model with the DLR-HIT Hand II as end-effector. The data generation pipeline works as follows. First, an object is spawned in front of the robot, and the simulated camera records a point cloud. Fig. 4(b) shows the subsequent segmentation step where the object (grey) is segmented from the ground (red) via RANSAC [52]. The normals for the object point cloud are computed and then used in order to uniformly sample one palm 6D pose $(\mathbf{R}, \mathbf{t})$ per object point. Each 6D pose is associated with one uniformly sampled finger joint configuration θ. The grasps are filtered for reachability and non-collision. The robot and hand are moved to the desired configuration and then a closing primitive is executed. After that, a lifting attempt of the object is made. These steps are repeated for all objects in multiple poses.

The result is a grasping database where each object is associated with a set of successful and failed grasps.

1) Sampling palm poses: In order to sample palm poses, we first choose one point from the object point cloud and its normal as indicated in Fig. 4(c). Then we add between 4.5 and 11.5 cm to the point in its normal direction and random 3D noise with a magnitude of 1 cm. We align the y-axis of the palm pose the longer object side and if necessary flip it such that the thumb points up. We sample uniformly around the obtained position between ± 0.7 rad around the x-direction and ± 0.35 rad around y- and z- directions.



(a) Generated grasps (b) $p(s) > 0.75$ (c) Ground truth

Fig. 5: The first column shows all grasps generated by the FFHGenerator and the second column only grasps for which the FFHEvaluator predicts a success probability of more than 75% (green point clouds). The third column shows the ground truth distribution from the dataset (red object mesh).

2) Sampling joint configurations: Randomly sampling joint angles in the 15-dimensional configuration space is inefficient. Instead, we build upon the idea from Ciocarlie [53] to sample in a meaningful subspace spanned by the so-called *eigengrasps*. We manually define four eigengrasps $\mathbf{e_1}, \mathbf{e_2}, \mathbf{e_3}, \mathbf{e_4} \in \mathbb{R}^{15}$. The full joint configuration:

$$\theta = \sum_{i=1}^{4} \alpha_i \mathbf{e}_i \tag{6}$$

is obtained through sampling the coefficients $\alpha_i \in [0, 1]$.

V. Experiments

This section aims to answer the following four questions:

1) Can the FFHGenerator approximate the distribution of successful grasps and generate high-quality grasps?
2) How much can the FFHEvaluator increase grasping success and how does this influence the generated grasp distribution?
3) Does FFHNet generate good grasps for real inputs?
4) How does FFHNet compare to other multi-fingered grasping methods in terms of speed and success?

A. Experiment Setup

We choose the 12 test objects listed in Table I from the KIT dataset for the experiments in simulation.

Each object is placed in simulation three times in random positions and random yaw-angle orientations. After recording each point cloud we segment the object from the ground plane via RANSAC [52]. We combine the segmented object point cloud with 400 random samples from a univariate Gaussian $\{\mathbf{z}\}^{400} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ to generate 400 different grasps per object. Then we run one forward pass of the FFHEvaluator to obtain the predicted success probability of the grasps for each object. For the experiment in Sec. V-D with real data, we follow a similar procedure but instead use four objects from the YCB dataset.

B. Qualitative Evaluation in Simulation

The qualitative evaluation aims to provide insights into how well the FFHGenerator learns to approximate the distribution of successful grasps for different objects. Fig. 5 shows the raw generated distribution, the filtered distribution where grasps with less than 75% predicted success probability was removed, and the ground truth. We see that the raw distribution nearly covers the entire object surface. Note that FFHNet only observes partial object geometries (green point clouds) but generates grasps also in the occluded regions of the object. Fig. 5(a) reveals that the FFHGenerator interpolates between the top and side modes of the ground truth distribution and generates likely unsuccessful grasps in the region between.

This demonstrates the need for the FFHEvaluator to reject these interpolated grasps. The filtered distribution in Fig. 5(b) follows the ground truth much more closely. We show that the FFHGenerator can approximate the ground truth distribution well and achieves good coverage of the object surface, while the FFHEvaluator is able to correctly reject interpolated grasps.

TABLE I: FFHEVALUATOR INFLUENCE ON GRASPING SUCCESS

Object	Th=0.0	Th=0.5	Th=0.7	Th=0.9	Th=0.95
BakingSoda	75%	86%	88%	97%	-
BathDetergent	51%	57%	70%	86%	83%
BroccoliSoup	82%	82%	90%	82%	-
CoughDrops	73%	72%	77%	89%	99%
Curry	56%	73%	80%	90%	94%
FizzyTablets	43%	63%	77%	89%	85%
InstantSauce	64%	72%	74%	72%	92%
NutCandy	56%	65%	70%	74%	-
PotatoeDump.	70%	77%	93%	80%	90%
Sprayflask	52%	53%	80%	91%	92%
TomatoSoup	63%	64%	66%	70%	99%
YellowCube	48%	60%	64%	69%	85%
Avg. Success	61%	70%	75%	82%	91%
Coverage	99%	87%	83%	28%	5%

TABLE II: PERFORMANCE COMPARISON

Method	Heuristics-based (Sec. IV-B)	FFHGenerator	FFHGenerator + FFHEvaluator
Success	15%	61%	**91%**

C. Quantitative Evaluation in Simulation

This experiment aims to answer quantitatively if the generated grasps from FFHNet result in a successful lifting of the object when the robot reaches the predicted pose $(\mathbf{R}, \mathbf{t})$ and finger configuration θ.

Table I shows the grasping results for the 12 test objects. The columns correspond to different grasping success thresholds **(Th=x.y)**. The set of grasps for which the evaluator predicted a success probability lower than the threshold are removed and only the remaining grasps are executed. Consequently, "**(Th=0.0)**" means that all grasps generated by the FFHGenerator were executed. We see that on average 61% of the generated grasps are successful. The geometry-based sampling described in Sec. IV could only generate around 15% grasping success. Since no grasps are filtered and the FFHGenerator covers the object geometry dense, we define coverage for this case as 99%. The last column shows that a yet higher threshold of 95% can increase the average success to 91%. But on average only 5% of generated grasps fall above this threshold. For some objects, no grasps lie above the 95% mark, which is indicated by a "-". For these objects, success rates of 74% up to 97% could be achieved while leaving about 30% of the grasps for execution. The

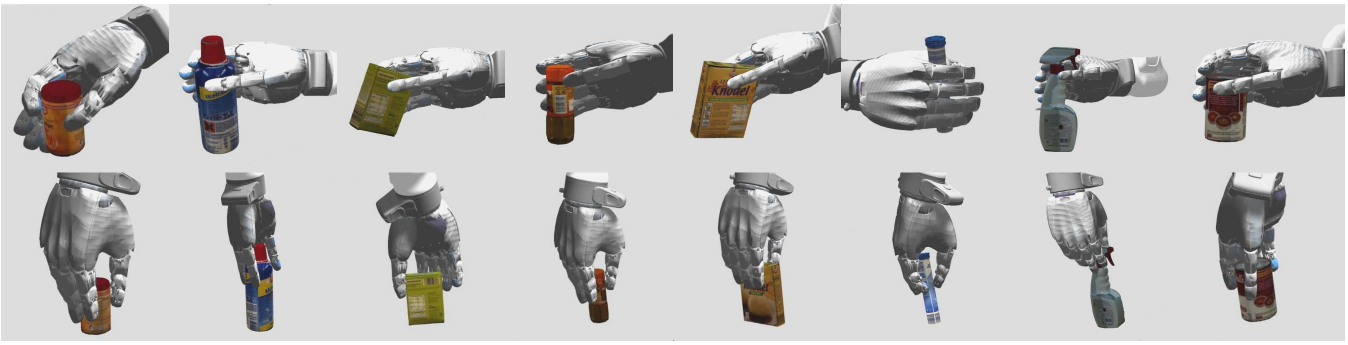
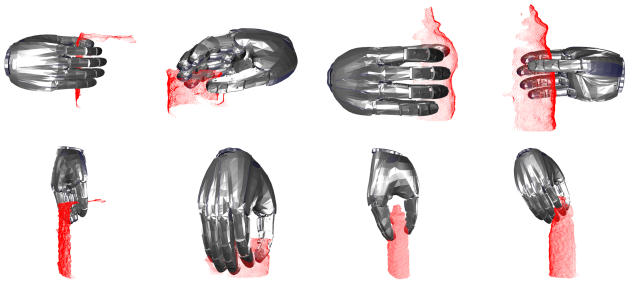


Fig. 6: The successful grasp poses generated from FFHNet for eight test objects in simulation. Each column represents one unknown test object. For each object, our FFHNet model is able to predict both top (first row) and side grasps (bottom row).



(a) Cracker (b) Mug (c) Mustard (d) Bleach

Fig. 7: Side grasps (top row) and top grasps (bottom row) with the highest predicted success probability. The grasps are generated based on real point clouds of four different YCB [27] objects.

results show a tension between grasping success and the variety of poses. We were able to show that average grasping success rates of 70-90% can be achieved while still leaving an acceptable amount of grasps to choose from. To further demonstrate the performance of FFHNet, we compare with other pipelines in TABLE II. Some successful grasp poses from the experiments are also visualized in Fig. 6.

D. Sim-to-real grasping

To gain insights into the performance on real data we capture point clouds of four YCB objects with the Realsense D415. Then, we sample 400 latent vectors from a univariate Gaussian $\mathbf{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ and combine them with each observation to generate 400 grasps per object. Afterwards, we rank the generated grasps with the FFHEvaluator. The first and second rows of Fig. 7 show the highest ranking side and top grasp for each object. All grasps look natural and incorporate large contact areas between the fingertips and object surface.

For the cracker, only a small part of the geometry can be observed, and yet FFHNet is able to generate good grasps. In the region of the mug handle, no grasps were generated. We suspect that this is caused by the BPS representation not being able to capture small and complex geometric details, and our dataset not containing enough grasps in such regions.

E. Comparison

We compare FFHNet to other state-of-the-art methods for multi-fingered grasping in Table III. Our success rate is higher, but direct comparison is not fair because of the differences in objects and environments. Reproduction of other works for our hand was not possible due to the differences in hardware and datasets. We can, however, directly compare the runtime, where FFHNet clearly outperforms other methods, which use time-consuming shape completion methods [14], [35] or gradient-based optimization [17].

TABLE III: RUNTIME COMPARISON WITH OTHER METHODS

Authors	Methods	Success Rate	Time
Liu et al. [20]	direct regression	80.0%	3-5s
Lu et al. [17]	gradient-based opt.	75.0%	5-10s
Lundell et al. [35]	GAN-based	60.0%	9.1s
Kiatos et al. [14]	shape completion	86.3%	4.25s
Ours	FFHNet (cVAE)	**91.0%**	**30ms**

VI. Conclusion & Future Work

We present FFHNet, a generative and discriminative model based on the cVAE framework, for synthesizing varied and successful multi-fingered grasps from a single view. Our system is able to perform grasp synthesis at 30 FPS, which, to the best of our knowledge, has not been achieved for multi-fingered grasping before. We showed in the simulation that our method is able to achieve up to 91% grasping success for 12 unseen objects in a table-top grasping scenario. Evaluation of FFHNet on real sensory data from a Realsense D415 showed the high fidelity of generated grasps on real point clouds. This is an indication of successful sim-to-real transfer capabilities. The training data of 180k grasps was collected purely in simulation. For this, we propose a flexible self-supervised data generation pipeline in Gazebo along with a grasp sampling strategy that makes only a few limiting assumptions.

We see as the main drawback of our method the seeming lack of diversity in finger configuration and bias towards pinch grasps. We attribute this to a lack in diversity of the dataset resulting from our grasp sampling method and shortcomings of simulation. One possible solution would be to employ complex grasp planning during data generation.

In future work, we will show the performance on real hardware, which was not available to us during the time of writing. Further, the speed of the system allows closing the loop from visual sensing to actuation. This enables different avenues towards reactive grasping or dynamic human-robot handover scenarios.

Acknowledgement

This research has received funding from the European Unions Horizon 2020 research and innovation programme under the Marie Sklodowska-Curie grant agreement No 778602 ULTRACEPT.

REFERENCES

[1] H. Liu, K. Wu, P. Meusel, N. Seitz, G. Hirzinger, M. Jin, Y. Liu, S. Fan, T. Lan, and Z. Chen, "Multisensory five-finger dexterous hand: The dlr/hit hand ii," in *2008 IEEE/RSJ international conference on intelligent robots and systems*. IEEE, 2008, pp. 3692–3697.

[2] Z. Chen, N. Y. Lii, T. Wimböck, S. Fan, H. Liu, and A. Albu-Schäffer, "Experimental analysis on spatial and cartesian impedance control for the dexterous dlr/hit ii hand," *International Journal of Robotics & Automation*, vol. 29, no. 1, pp. 1–13, 2014.

[3] N. Y. Lii, Z. Chen, B. Pleintinger, C. H. Borst, G. Hirzinger, and A. Schiele, "Toward understanding the effects of visual-and force-feedback on robotic hand grasping performance for space teleoperation," in *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2010, pp. 3745–3752.

[4] 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," *arXiv preprint arXiv:1703.09312*, 2017.

[5] A. ten Pas, M. Gualtieri, K. Saenko, and R. Platt, "Grasp pose detection in point clouds," *The International Journal of Robotics Research*, vol. 36, no. 13-14, pp. 1455–1473, 2017. [Online]. Available: https://doi.org/10.1177/0278364917735594

[6] A. Pas and R. Platt, *Using Geometry to Detect Grasp Poses in 3D Point Clouds*, 01 2018, pp. 307–324.

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

[8] R. M. Murray, Z. Li, S. S. Sastry, and S. S. Sastry, *A mathematical introduction to robotic manipulation*. CRC press, 1994.

[9] C. Rubert, D. Kappler, A. Morales, S. Schaal, and J. Bohg, "On the relevance of grasp metrics for predicting grasp success," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2017, pp. 265–272.

[10] D. Kappler, J. Bohg, and S. Schaal, "Leveraging big data for grasp planning," in *2015 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2015, pp. 4304–4311.

[11] R. Diankov, "Automated construction of robotic manipulation programs," 2010.

[12] K. Huebner, K. Welke, M. Przybylski, N. Vahrenkamp, T. Asfour, D. Kragic, and R. Dillmann, "Grasping known objects with humanoid robots: A box-based approach," in *2009 International Conference on Advanced Robotics*. IEEE, 2009, pp. 1–6.

[13] A. Morales, E. Chinellato, P. Sanz, A. Del Pobil, and A. H. Fagg, "Learning to predict grasp reliability for a multifinger robot hand by using visual features," in *International Conference AI Soft Computing*. Citeseer, 2004.

[14] M. Kiatos, S. Malassiotis, and I. Sarantopoulos, "A geometric approach for grasping unknown objects with multifingered hands," *IEEE Transactions on Robotics*, vol. 37, no. 3, pp. 735–746, 2021.

[15] Q. Lu, K. Chenna, B. Sundaralingam, and T. Hermans, "Planning multi-fingered grasps as probabilistic inference in a learned deep network," in *Int'l Symp. on Robotics Research*, 2017.

[16] M. V. der Merwe, Q. Lu, B. Sundaralingam, M. Matak, and T. Hermans, "Learning continuous 3d reconstructions for geometrically aware grasping," *CoRR*, vol. abs/1910.00983, 2019. [Online]. Available: http://arxiv.org/abs/1910.00983

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

[18] J. Lundell, F. Verdoja, and V. Kyrki, "Beyond top-grasps through scene completion," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2020, pp. 545–551.

[19] Z. Zhao, W. Shang, H. He, and Z. Li, "Grasp prediction and evaluation of multi-fingered dexterous hands using deep learning," *Robotics and Autonomous Systems*, vol. 129, p. 103550, 2020.

[20] M. Liu, Z. Pan, K. Xu, K. Ganguly, and D. Manocha, "Generating grasp poses for a high-dof gripper using neural networks," *arXiv preprint arXiv:1903.00425*, 2019.

[21] ——, "Deep differentiable grasp planner for high-dof grippers," *CoRR*, vol. abs/2002.01530, 2020. [Online]. Available: https://arxiv.org/abs/2002.01530

[22] L. Shao, F. Ferreira, M. Jorda, V. Nambiar, J. Luo, E. Solowjow, J. A. Ojea, O. Khatib, and J. Bohg, "Unigrasp: Learning a unified model to grasp with multifingered robotic hands," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 2286–2293, 2020.

[23] B. Wu, I. Akinola, A. Gupta, F. Xu, J. Varley, D. Watkins-Valls, and P. K. Allen, "Generative attention learning: a "general" framework for high-performance multi-fingered grasping in clutter," *Autonomous Robots*, pp. 1–20, 2020.

[24] P. Mandikal and K. Grauman, "Dexterous robotic grasping with object-centric visual affordances," *arXiv preprint arXiv:2009.01439*, 2020.

[25] P. Schmidt, N. Vahrenkamp, M. Wächter, and T. Asfour, "Grasping of unknown objects using deep convolutional neural networks based on depth images," in *2018 IEEE international conference on robotics and automation (ICRA)*. IEEE, 2018, pp. 6831–6838.

[26] O. Taheri, N. Ghorbani, M. J. Black, and D. Tzionas, "Grab: A dataset of whole-body human grasping of objects," in *European Conference on Computer Vision*. Springer, 2020, pp. 581–600.

[27] B. Calli, A. Singh, A. Walsman, S. Srinivasa, P. Abbeel, and A. M. Dollar, "The ycb object and model set: Towards common benchmarks for manipulation research," in *2015 international conference on advanced robotics (ICAR)*. IEEE, 2015, pp. 510–517.

[28] A. Singh, J. Sha, K. S. Narayan, T. Achim, and P. Abbeel, "Bigbird: A large-scale 3d database of object instances," in *2014 IEEE international conference on robotics and automation (ICRA)*. IEEE, 2014, pp. 509–516.

[29] A. Kasper, Z. Xue, and R. Dillmann, "The kit object models database: An object model database for object recognition, localization and manipulation in service robotics," *The International Journal of Robotics Research*, vol. 31, no. 8, pp. 927–934, 2012.

[30] Q. Lei, J. Meijer, and M. Wisse, "Fast c-shape grasping for unknown objects," in *2017 IEEE International Conference on Advanced Intelligent Mechatronics (AIM)*, 2017, pp. 509–516.

[31] Y. Avigal, S. Paradis, and H. Zhang, "6-dof grasp planning using fast 3d reconstruction and grasp quality CNN," *CoRR*, vol. abs/2009.08618, 2020. [Online]. Available: https://arxiv.org/abs/2009.08618

[32] J. Varley, C. DeChant, A. Richardson, A. Nair, J. Ruales, and P. K. Allen, "Shape completion enabled robotic grasping," *CoRR*, vol. abs/1609.08546, 2016. [Online]. Available: http://arxiv.org/abs/1609.08546

[33] A. T. Miller and P. K. Allen, "Graspit! a versatile simulator for robotic grasping," *IEEE Robotics & Automation Magazine*, vol. 11, no. 4, pp. 110–122, 2004.

[34] C. M. Bishop, "Mixture density networks," 1994.

[35] J. Lundell, E. Corona, T. N. Le, F. Verdoja, P. Weinzaepfel, G. Rogez, F. Moreno-Noguer, and V. Kyrki, "Multi-fingan: Generative coarse-to-fine sampling of multi-finger grasps," *arXiv preprint arXiv:2012.09696*, 2020.

[36] F. Patzelt, R. Haschke, and H. J. Ritter, "Conditional wgan for grasp generation." in *ESANN*, 2019.

[37] C. R. Qi, L. Yi, H. Su, and L. J. Guibas, "Pointnet++: Deep hierarchical feature learning on point sets in a metric space," *arXiv preprint arXiv:1706.02413*, 2017.

[38] 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.

[39] D. Park and S. Y. Chun, "Classification based grasp detection using spatial transformer network," *CoRR*, vol. abs/1803.01356, 2018. [Online]. Available: http://arxiv.org/abs/1803.01356

[40] J. Varley, J. Weisz, J. Weiss, and P. Allen, "Generating multi-fingered robotic grasps via deep learning," in *2015 IEEE/RSJ international conference on intelligent robots and systems (IROS)*. IEEE, 2015, pp. 4415–4420.

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

[42] P. Ni, W. Zhang, X. Zhu, and Q. Cao, "Pointnet++ grasping: learning an end-to-end spatial grasp generation algorithm from sparse point clouds," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2020, pp. 3619–3625.

[43] Y. Qin, R. Chen, H. Zhu, M. Song, J. Xu, and H. Su, "S4g: Amodal single-view single-shot se (3) grasp detection in cluttered scenes," in *Conference on robot learning*. PMLR, 2020, pp. 53–65.

[44] K.-Y. Jeng, Y.-C. Liu, Z. Y. Liu, J.-W. Wang, Y.-L. Chang, H.-T. Su, and W. Hsu, "A coarse-to-fine (c2f) representation for end-to-end 6-dof grasp detection," *arXiv preprint arXiv:2010.10695*, 2020.

[45] C. Wu, J. Chen, Q. Cao, J. Zhang, Y. Tai, L. Sun, and K. Jia, "Grasp proposal networks: An end-to-end solution for visual learning of robotic grasps," *arXiv preprint arXiv:2009.12606*, 2020.

[46] S. Prokudin, C. Lassner, and J. Romero, "Efficient learning on point clouds with basis point sets," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 4332–4341.

[47] K. Sohn, H. Lee, and X. Yan, "Learning structured output representation using deep conditional generative models," *Advances in neural information processing systems*, vol. 28, pp. 3483–3491, 2015.

[48] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," *arXiv preprint arXiv:1312.6114*, 2013.
[49] Y. Zhou, C. Barnes, J. Lu, J. Yang, and H. Li, "On the continuity of rotation representations in neural networks," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 5745–5753.
[50] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.
[51] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.
[52] M. A. Fischler and R. C. Bolles, "Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography," *Communications of the ACM*, vol. 24, no. 6, pp. 381–395, 1981.
[53] M. T. Ciocarlie and P. K. Allen, "Hand posture subspaces for dexterous robotic grasping," *The International Journal of Robotics Research*, vol. 28, no. 7, pp. 851–867, 2009.