Learning Continuous 3D Reconstructions for Geometrically Aware Grasping

Mark Van der Merwe[1], Qingkai Lu[1], Balakumar Sundaralingam[1], Martin Matak[1], and Tucker Hermans[1,2]

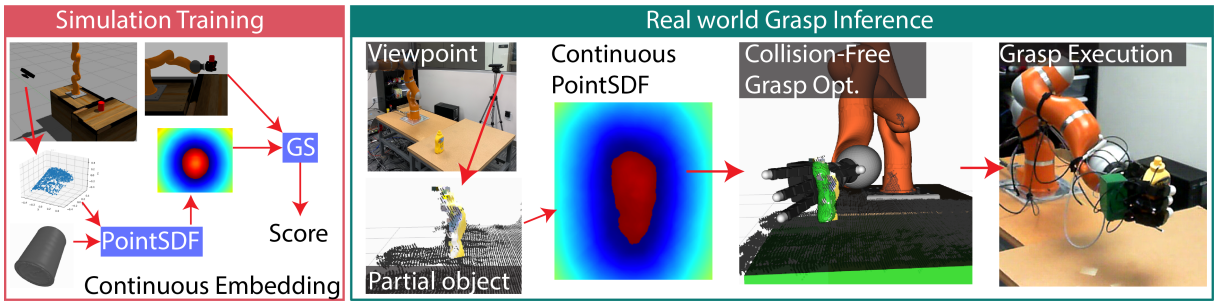

Fig. 1: We train a continuous signed distance function embedding (PointSDF) for the partial object pointcloud, and a grasp success (GS) model in simulation. Leveraging these simulation learned models in a gradient-based grasp optimization, we enable collision-free grasping of novel objects in the real world.

***Abstract*— Deep learning has enabled remarkable improvements in grasp synthesis for previously unseen objects from partial object views. However, existing approaches lack the ability to explicitly reason about the full 3D geometry of the object when selecting a grasp, relying on indirect geometric reasoning derived when learning grasp success networks. This abandons explicit geometric reasoning, such as avoiding undesired robot object collisions. We propose to utilize a novel, learned 3D reconstruction to enable geometric awareness in a grasping system. We leverage the structure of the reconstruction network to learn a grasp success classifier which serves as the objective function for a continuous grasp optimization. We additionally explicitly constrain the optimization to avoid undesired contact, directly using the reconstruction. We examine the role of geometry in grasping both in the training of grasp metrics and through 96 robot grasping trials. Our results can be found on https://sites.google.com/view/reconstruction-grasp/.**

I. Introduction

The ability to reliably grasp previously unseen objects in multiple environments remains an open challenge in robotics [1, 2]. The effects of noisy and partial sensor inputs coupled with the unknown object properties complicate effective grasp synthesis. In this paper, we consider grasping unseen, isolated objects on tabletop environments with multi-fingered dexterous hands.

While analytical robotic grasp synthesis methods can provide desirable guarantees about grasp performance, they rely on metrics that have failed to generalize to the real world and can fail to perform given perceptual uncertainty; as such, much recent work in robotic grasp synthesis has turned to deep-learning based approaches [2]. Most existing deep-learning approaches are trained in an end-to-end fashion [3–13]. That is, the system takes in sensor input, such as an RGB or RGBD image, and outputs a grasp, either via direct regression [12, 13], sampling candidate grasps or motions [5, 6], or solving an optimization problem leveraging the learned network, either in a discrete [3, 4] or continuous [7–9, 11, 14] fashion. As such, there is typically no explicit modeling of the geometry in the scene. Rather, researchers assume the classifier will indirectly learn a geometric understanding of the scene, such that the network will prefer stable grasps that are out of collision. This abandons explicit a priori geometric reasoning, yielding undesirable robot-object collisions [3–6]. We seek to decrease the chance of such collisions by explicitly modeling the 3D environment as part of the grasp planning problem.

The more difficult problem of multi-fingered grasping has similarly followed an end-to-end grasp learning framework [7–12]. One competitive approach to multi-fingered grasping, achieving state-of-the-art results, relies on performing a continuous optimization over the hand configuration to maximize the likelihood of grasp success [8, 9, 11]. In [8, 15], the continuous grasp optimization is shown to generally achieve higher grasp success rates compared sampling-based [16, 17] and regression [12, 13] approaches used for grasping with neural networks. However, similar to other approaches, these optimization-based inference procedures have no explicit understanding of the geometry of the scene and thus may find a solution which causes the hand to be in collision with the environment or even intersect with the object to be grasped. This forces relying on full state knowledge of the environment [11] or abandoning grasp attempts when motion planners fail to find a collision free path to the desired grasp.

The primary obstacle to explicitly incorporating geometric information into grasp synthesis is that these systems have only a single view of the world and thus can only partly understand the object geometry. One approach adopted in the computer vision community is to learn to predict the underlying 3D shape generating the partial view [18–22]. The recent dominant approach has been to learn a voxel-based object reconstruction [19] from the partial view; these reconstructions have been utilized in analytical [23] and learning-based [24] grasp synthesis systems. Recent work has shown that neural networks can effectively learn implicit

[1]School of Computing and Robotics Center, University of Utah, Salt Lake City, UT 84112, USA.
[2]NVIDIA Research, Seattle, WA 98105, USA

shape representations, such as signed distance functions (SDF) or continuous occupancy maps, yielding state-of-the-art 3D reconstruction performance [20–22]. Learning signed distance function reconstructions yields many desirable improvements to voxel based approaches, including arbitrarily high resolution and mesh-free geometric understanding. Indeed, roboticists regularly use signed distance functions to encode collision constraints in trajectory optimization for motion planning [25].

We propose utilizing 3D object reconstruction to enable *geometrically-aware grasp synthesis* in a continuous optimization framework [8, 9]. At the core of our approach lies a novel implicit surface reconstruction algorithm, *PointSDF*, which directly regresses signed distance functions from point clouds, providing geometrically rich input and output. We leverage these reconstructions to make our grasping system geometrically aware both implicitly and explicitly.

We implicitly encode geometry by introducing the point cloud embedding from PointSDF into a grasp success prediction network [8]. We enable explicit geometric reasoning by constraining the optimization of our grasp success prediction network to be collision free. We achieve this by extending previous approaches to learning-based grasp optimization [8, 9, 11] to include the full robot arm configuration, instead of only a 6DOF wrist pose, and add SDF collision constraints between the reconstructed object and all links of the robot. By formulating the optimization in the robot joint space, we ensure not only kinematic feasibility of all synthesized grasps but also Euclidean updates of the gradients in the optimization by propagating learned gradients through the kinematic Jacobian. We examine the efficacy of our approach through real robot grasping experiments on a KUKA LBR4 robot with an Allegro multi-fingered hand.

To summarize, our primary contributions are listed below,

- **3D Reconstruction:** a novel single-view reconstruction learning architecture, PointSDF, that learns a signed distance function implicit surface for a partially viewed object.
- **Grasp Success Prediction:** a novel grasp success prediction learning architecture, that implicitly learns geometrically aware point cloud encodings.
- **Grasp Synthesis:** an extended formulation of learning-based grasp synthesis as a constrained optimization problem in the full robot configuration space, ensuring kinematic feasibility and explicit collision avoidance via our learned continuous signed distance function (PointSDF).

We illustrate our key contributions in Fig. 1. We organize the remainder of the paper as follows. In Sec II we present PointSDF for geometric reconstruction. In Sec. III we apply PointSDF to grasp success prediction and define the full grasp synthesis optimization in Sec. IV. We discuss the implementations, experiments, and implications of our results in Sec. V. We then briefly conclude and discuss directions for future work in Sec. VI.

II. 3D Reconstruction via Learned Signed Distance Function

We present a new architecture for predicting a 3D reconstruction of an object from a single view point cloud. Motivated for its use in grasp planning, we desire that our reconstruction approach seamlessly handles seen and unseen objects alike from arbitrary viewpoints and accurately encodes geometric concepts, while efficiently performing inference in terms of both time and space. As such, we propose learning to directly predict the signed distance function, which implicitly represents the object surface as the zero level set of the function. The signed distance function defines the shortest distance between a query point in 3D space and the surface of the object, where distances are negative for points inside the object and positive when outside. We call our architecture PointSDF. Unlike previous iterations of similar design [21], we enable single-pass evaluation using a simple encoder-decoder structure.

Given a point cloud view of the object, $\boldsymbol{o}$, and a query point in 3D space, x, the PointSDF function, f_{SDF} predicts the continuous-valued signed distance from that point to the surface of the fully reconstructed object:

$$f_{SDF}(\boldsymbol{o}, x; \theta) = s; \ \boldsymbol{o} \in \mathbb{R}^{P\times 3}, x \in \mathbb{R}^3, s \in \mathbb{R} \qquad (1)$$

where θ represents the parameters of the network. We train our network using the standard mean-squared error loss for regression between the distance predictions from the network and the true SDF values for query points relative to the training objects.

Directly regressing SDF values from the point cloud holds several key advantages that make our PointSDF representation advantageous for robotics applications, especially grasp planning: first, PointSDF can be evaluated at arbitrary resolution without transforming the prediction into a mesh, maintaining accuracy as compared to discretization inherent in earlier voxel-based approaches in robotics [23, 24, 26]. Nevertheless, a mesh can be extracted if desired by determining the zero isosurface of the SDF, which we can compute via sampling the network throughout the space as shown by [20, 21].

Another benefit of using a SDF representation via a neural network is that the network implicitly learns geometric gradients. Given a query point x and observation $\boldsymbol{o}$, we can derive a vector pointing towards or away from the true reconstructed object by finding the gradient at the point x, $\partial f_{SDF}(\boldsymbol{o}, x; \theta)/\partial x$. We can efficiently compute such gradients using the backpropagation algorithm.

Our architecture builds on recent work deriving convolution operations directly on point clouds [27, 28]. Unlike typical convolutions which require well structured inputs, these approaches work directly on unstructured point clouds. We embed the point cloud using four PointConv layers [28]. This embedding, along with the query point, is passed through several fully-connected layers, leading to a single prediction passed through a tanh activation to get an SDF estimate between -1 and 1. The top half of Figure 2 shows our network design.

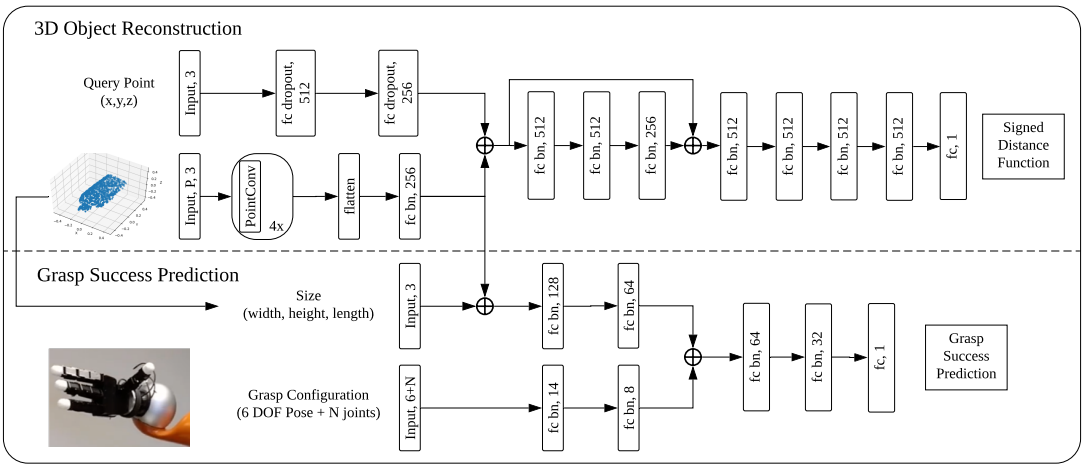

Fig. 2: The top network is a 3D reconstruction network that takes in a point cloud and a query point and regresses to signed distance function values for each query point to the surface of the reconstructed object. The bottom network utilizes the point cloud embedding subnetwork, as well as grasp configuration and point cloud size information to predict whether the given grasp configuration will succeed.

III. Learning Grasp Success Prediction via 3D Reconstruction

Our primary goal is to synthesize high quality grasps for partially viewed objects, which we perform as a continuous optimization over the robot's arm-hand joint configuration. Here we present our design of a learned grasp success metric, that serves as the objective to maximize during grasp synthesis. Following recent, high-performing approaches to multi-fingered grasp planning [8, 9], we model this planning problem as probabilistic inference.

We seek to maximize the posterior probability of a robot arm-hand joint configuration, $\boldsymbol{q}$, generating a successful grasp (i.e., $Y = 1$) given a point cloud observation, $\boldsymbol{o}$, of the target object. For our objective we replace the full configuration, $\boldsymbol{q} = [\boldsymbol{q}_h, \boldsymbol{q}_a]$, with $\boldsymbol{q}_g = [\boldsymbol{q}_h, \texttt{FK}_p(\boldsymbol{q}_a)]$. Here $\boldsymbol{q}_h$ represents the N joint positions of the hand, while $\texttt{FK}_p(\boldsymbol{q}_a)$ computes 6-DOF palm pose of the robot hand as a function of the robot arm joint state $\boldsymbol{q}_a$. Formally we have the following objective:

$$p(\boldsymbol{q}|Y=1,\boldsymbol{o}) \propto p(Y=1|\boldsymbol{q},\boldsymbol{o};\phi)\cdot p(\boldsymbol{q};\psi) \quad (2)$$

$$\propto h(\boldsymbol{q},\boldsymbol{o};\phi)\cdot g(\boldsymbol{q};\psi) \quad (3)$$

where ϕ and ψ parameterize each respective probability distribution. Following Lu et al. [8], we parameterize the grasp success probability distribution $h(\cdot)$ as a neural network and the grasp prior $g(\cdot)$ as a Gaussian mixture model (with 2 components) fit to all grasp configurations seen during training.

In order to make our grasp success prediction network geometrically aware, we utilize the same point cloud encoder architecture used in PointSDF to embed the point cloud of the target object. Along with the grasp configuration, the embedding is passed through several fully-connected layers, leading to a single prediction passed through a sigmoid activation to get our grasp success probability estimate. Our network design is shown in the bottom half of Figure 2.

IV. Reconstruction-Aware Grasp Synthesis via Constrained Optimization

Given the robot joint configuration $\boldsymbol{q}$ encoding the arm and hand joint vectors respectively, we define the grasp synthesis problem as finding a grasp preshape joint configuration for the arm and the hand that enables a collision-free grasp on the object that maximizes the probability of grasp success. This amounts to solving the following constrained, non-convex optimization problem,

$$\underset{\boldsymbol{q}}{\text{argmin}} \quad \max(\beta - s, 0)^2 \quad (4)$$

$$\text{s.t.} \quad \boldsymbol{q}_{min} \preceq \boldsymbol{q} \preceq \boldsymbol{q}_{max} \quad (5)$$

$$a_{SDF}(M_e, \texttt{FK}_l(\boldsymbol{q})) > 0 \quad \forall l \in L; e \in E \quad (6)$$

$$f_{SDF}(\boldsymbol{o}, \texttt{FK}_l(\boldsymbol{q})) > 0 \quad \forall l \in L \quad (7)$$

where $s = -\log(h(\boldsymbol{q}_g, \boldsymbol{o})) - \alpha \log(g(\boldsymbol{q}_g)$ is the cost term derived from the learned grasp success prediction network and prior function. In Eq. 4, we turn Eq. 3 into a least-squares cost to be minimized. As we are combining the log likelihood of a discrete and continuous probability distribution, the range of our objective function in Eq. 4 is unbounded below; as such, we empirically select some value β as a sufficient minimum and square the difference; for our experiments, we use $\beta = -2$. We can derive objective function gradients via backpropagation through our grasp success network.

We split our collision constraints into two parts. First is a known geometry collision constraint where M_e defines the mesh relating to component e of the environment (e.g., avoid collision with known table geometry). This utilizes an analytical signed distance function constraint in Eq. 6. Second is the unknown target object collision constraint, whose geometry we assume is unknown beyond the single view recieved from a depth sensor $\boldsymbol{o}$. This utilizes our reconstruction network $f_{sdf}(\cdot)$ to enforce the constraint in Eq. 7. Because this network takes in the query point and assigns an SDF, we avoid meshing and can directly utilize the learned network to get SDF values for our robot. We

compute the query points to check for robot-object collision by computing the forward kinematics of each link for the current robot joint configuration $\boldsymbol{q}$ and using each vertex from the link mesh. Each link is assigned the minimum SDF value of all its vertices. As described in Sec. II, we can derive SDF gradients to move a link out of collision directly through the network via backpropagation. The constraints in Eq. 5 encode the kinematic limits of the robot joints.

This formulation allows direct enforcement of geometric constraints that avoids unintentional robot-object contact fo the full robot arm. The formulation also ensures all solution are kinematically feasible, due to the optimization over th full arm-hand joint configuration q.

V. Experiments

Our experiments seek to examine the efficacy of ou reconstruction-based grasping formulation. Specifically, w seek to answer the following questions: 1. How well doe PointSDF perform at single depth view reconstruction? 2 How well does our grasp metric perform at predicting gras success and how does implicit geometric reasoning affec this performance? 3. How well does our grasp formulatio work when deployed on a real robot? See videos of ou experiments at our website: `https://sites.google.com/view/reconstruction-grasp/`.

A. PointSDF Reconstruction

We first describe the training procedure for the PointSDF architecture introduced in Sec. II, then evaluate its performance.

To train PointSDF, we synthetically render, via a simulated camera, the 590 meshes from the Grasp Database [10] and 76 meshes from the YCB Database [29] at 200 random orientations each, adding noise to the depth images to reflect Kinect noise. We backproject the rendered points into a 3D point cloud, which becomes the input to our network. We generate SDF query label pairs by rotating the true mesh to the same random rotation for each view and sampling and labeling points from the surface and space around the mesh.

We employ a simple object frame centered in the point cloud that uses the same orientations as the camera frame. By keeping the camera frame orientation and by applying random rotations during data collection, PointSDF is camera-pose invariant. To simplify learning, we also scale all point clouds to fit in a $1m \times 1m \times 1m$ bounding box. At inference time, SDF estimates can easily be scaled back for use.

To demonstrate how our reconstructed meshes compare to the reconstructions currently used in grasping [23, 24], we reimplement a representative voxel-based reconstruction model based on a 3D CNN encoder-decoder structure [18], which we hereafter refer to as VoxelCNN. We quantify 3D reconstruction performance with three metrics computed between the reconstructed test object and the ground truth mesh: the volumetric IoU, Chamfer-L_1 distance, and a normal consistency score (for description of metrics, we refer the reader to [20]). To recover the mesh from our implicit surface model, we re-implement a hierarchical sampling

TABLE I: Reconstruction performance on different metrics (standard deviation in parentheses).

Metric	Method	Dataset	
		Grasp Database	YCB
IoU	PointSDF	**0.71933 (0.18807)**	**0.56975 (0.28807)**
	VoxelCNN	0.53172 (0.20040)	0.44039 (0.23304)
Chamfer-L_1	PointSDF	**0.00150 (0.00432)**	**0.00256 (0.00237)**
	VoxelCNN	0.00296 (0.00116)	0.00458 (0.00230)
Normal Consistency	PointSDF	0.83611 (0.06540)	0.77919 (0.10078)
	VoxelCNN	**0.83625 (0.06094)**	**0.78665 (0.07273)**

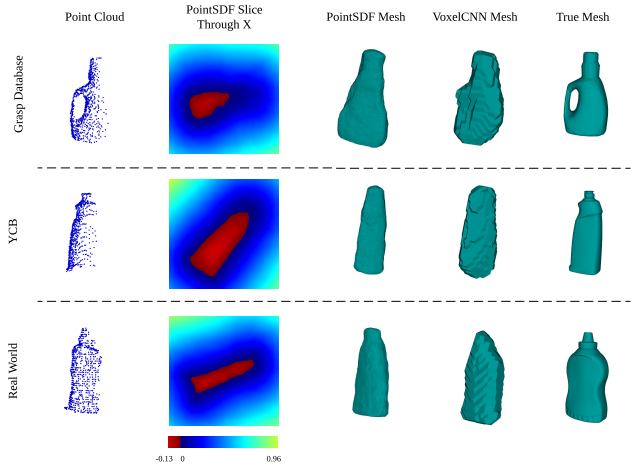

Fig. 3: Qualitative comparison of reconstruction results with objects from the Grasp and YCB dataset, as well as the YCB mustard with a point cloud from the real camera. We also show samples in a plane through the target object.

reconstruction method from [20], and sample to a resolution of 512^3.

In Table I, we see that PointSDF outperforms VoxelCNN on the IoU and Chamfer-L_1 metrics and nearly matches VoxelCNN on normal consistency, indicating increased geometric understanding. In Figure 3, we show representative object reconstructions on previously unseen objects from both datasets. We see that PointSDF reconstructions are smoother and retain finer details as compared to VoxelCNN results. The PointSDF slices also show a desirable gradient in predictions with a clear zero level set.

B. Grasp Success Network

We now describe our specific grasp configuration $\boldsymbol{q}_g$, our data collection and training procedure, and evaluate our grasp success network performance.

We conduct all training and experiments using the four-fingered, 16 DOF Allegro hand mounted on a Kuka LBR4 7 DOF arm. As described in Sec. III, we use the palm pose of the grasp (in the object frame used for reconstruction) and $N = 8$ joint values, representing the two joints of each finger closest to the palm.

We collected simulated grasp data for grasp model training using our robot hand-arm setup inside the Gazebo simulator with the DART physics engine[1]. We use the built-in Gazebo Kinect camera to generate point clouds. We collected training data using a heuristic, geometry-based grasp planner [15] and

[1] `https://dartsim.github.io/`

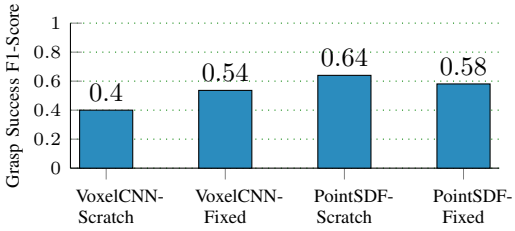

Fig. 4: Grasp success prediction performance across different embedding approaches.

randomly sample joint angles for the first two joints of all fingers within a reasonable range, fixing the last two joints of each finger to be zero. More details of our grasping data collection can be seen from [15]. In total, we train with 7290 grasp examples, and test on a leave-out set of 1821 grasps.

To determine how implicit geometric reasoning effects the performance of grasp success prediction, we train two variations of our network from Sec.III: first, we train the point cloud embedding from scratch, only with regards to the grasp dataset (referred to as "PointSDF-Scratch"). Second, we train the point cloud embedding on reconstruction, then lock the embedding network and only update the grasp metric model (referred to as "PointSDF-Fixed"). The former acts as a baseline which learns geometric reasoning implicitly from the downstream task, whereas the latter seeks to impose geometric reasoning by first training the embedding on the reconstruction tasks.

To compare to voxel-based approaches, we replace the PointSDF embedding with a voxel-based embedding from Sec. V-A, and setup the same two variations ("VoxelCNN-Scratch" and "VoxelCNN-Fixed")

We show in Fig. 4 the F1-score of each classifier (with threshold 0.5 on prediction) against the test set from our simulated grasps. We highlight two key observations from our results. First, we notice that both networks based on the point cloud embedding [28] outperformed the voxel-based networks. Second, locking the embedding network based on the reconstruction slightly decreased the performance of the classifier when using the PointSDF encoder. We speculate that this could be due to, a) the structure of PointConv layers in the encoder, and b) the small size of our training set. We notice that for the VoxelCNN method this role was reversed, indicating that the CNN structure appears to encourage overfitting.

C. Grasp Synthesis

We now combine our grasp success network with our reconstruction network for reconstruction-aware grasp synthesis, as described in Sec. IV.

We implement our optimization in PAGMO [30] and use SLSQP [31] to perform the optimization. Our analytic signed distance function (for the analytical environment collision constraint in Eq. 6) is obtained using GJK [32] and PQP [33]. We seed the optimization by sampling a grasp configuration from the grasp prior g and use Inverse Kinematics (IK) to convert the 6DOF palm pose to a set of arm joints, constraining the IK solution to be above the table. We sample from the component of the prior that represents side grasps.

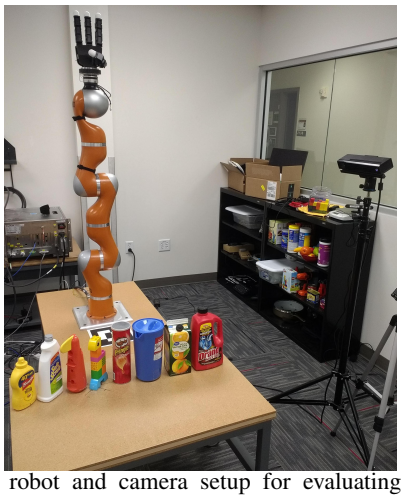

Fig. 5: Our robot and camera setup for evaluating grasping in the real world with a KUKA LBR4 robot with Allegro end-effector. The objects are labeled from left to right as *mustard*, *soft-scrub*, *airplane-toy*, *Lego*, *pringles*, *pitcher*, *juice-box*, and *max-gel* respectively.

We complete our implementation of our formulation by using the PointSDF network for the robot-object collision constraint in Eq. 7 and using our grasp success network "PointSDF-Fixed" for the objective function in Eq. 4. We call this approach "Reconstruction-Grasping." To provide a strong comparison, we introduce another approach that uses the "PointSDF-Scratch" model for the objective function and replaces the learned collision constraint in Eq. 7 with an analytical collision constraint. We provide this analytical collision model with the partial point cloud meshed at 32^3 voxelization. This approach, which we call "Partial-View-Grasping", represents an approach that relies only on the partial geometric information available.

We test our grasp optimization with an Allegro hand mounted on a Kuka LBR4 arm. We use a Kinect 2 camera to get our point clouds, and place the camera such that the right-handed robot plans side-grasps near the occluded area of each object; this provides us with a geometrically difficult grasping problem. We use 6 objects from the YCB dataset [29] and two objects that were used by [9]. All objects as well as the full robot/camera setup are shown in Fig. 5. Only the *chips* object was part of the training dataset and all others are novel objects.

We place a single target object on the table and call the grasp optimization. In order to select good grasps, we accept the grasp plan if the likelihood of success from the learned grasp success network is above 0.6 *and* if the combined grasp score with the prior (i.e., Eq. 4) is below 5.0. We allow up to five chances to satisfy this heuristic, each with a new seed from the prior; failure to find a sufficient plan given five chances is recorded as a failure. Given a successful robot joint configuration, the configuration is sent to MoveIt! to obtain a collision free trajectory to reach the grasp configuration. We allow up to three plans to be sent to MoveIt!, upon which continued lack of a motion plan is labeled a failure. For collision checking during motion planning, we provide the partial mesh of the object to MoveIt! for

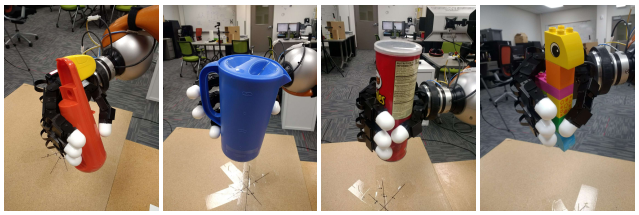

Fig. 6: Representative grasps our approach executed on the physical robot.

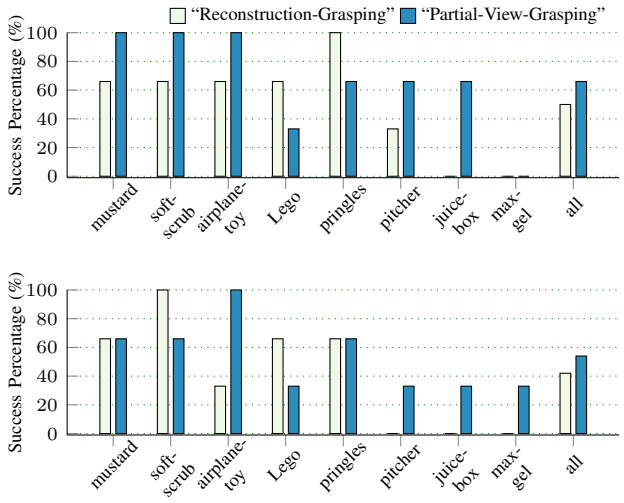

Fig. 7: Success rates on real world grasp task. We plot the "High" camera setup on top and the "Low" camera setup (more occlusion) below.

the "Partial-View-Grasping" method and the reconstructed mesh using PointSDF for the "Reconstruction-Grasping" method (evaluated to 128^3 voxelization). Each successful MoveIt! generated trajectory is executed on the robot and the hand is closed with the same controller used during data collection. Finally, the palm is lifted to 15 cm. We chose a fixed test location on the table for the target object and rotated to 3 different angles, keeping the pose constant across the methods and objects. We also test the camera at two positions, offset by about 0.45 meters vertically (referred to as "high" and "low" scenarios respectively). This is to explore how robust each method is to various levels of occlusion: the lower setting yields heavier occlusion in the input point cloud.

We show some qualitative grasp execution examples from the "Reconstruction-Grasping" method in Fig. 6. We show the per-object and per-camera location performance for each approach in Fig. 7. We ran a total of 96 grasps, split between the two approaches. Across both camera-setups, the

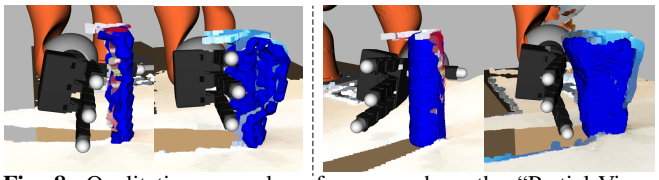

Fig. 8: Qualitative examples of grasps where the "Partial-View-Grasping" (left) planned a grasp in direct collision with occluded space, while "Reconstruction-Grasping" (right) planned a geometry-aware grasp that avoids the occupied space.

"Reconstruction-Grasping" approach succeeded on 48% of its grasp attempts (22 grasps). The "Partial-View-Grasping" approach succeeded on 60% of its grasp attempts (29 grasps). The "Reconstruction-Grasping" approach has an average planning time of 37.81s as compared to "Partial-View-Grasping" approach's slightly shorter 36.55s.

From Fig. 7, we see that the "Reconstruction-Grasping" approach struggles to grasp the larger objects from our object set (*pitcher, juice-box, and max-gel*) and is roughly equivalent in performance across the other objects. We believe the lack of quantitative improvement is due, in part, to the increased difficulty the constraints over the full geometry impose on the optimization. Of the 26 failures, the "Reconstruction-Grasping" optimization failed to find a sufficient solution to our heuristic 11 times, as opposed to only 4 such failures for "Partial-View-Grasping," indicating that the optimization was struggling to both fulfill the constraints and meet the heuristic cutoffs on grasp success likelihood.

In Fig. 8 we show an example of a plan generated by the "Partial-View-Grasping" approach that yields a final grasp in contact with the target object. We see that the "Reconstruction-Grasping" approach can avoid this error through its understanding of the full geometry. Interestingly, we found actual grasp execution to be relatively robust to grasps in contact with the object; only one grasp failure for "Partial-View-Grasping" was clearly due to planning in contact with the object. Further work into determining when target geometry matters to the grasp success is needed to understand the extent to this robustness; intuitively, the qualitative results of Fig. 8 could impact performance broadly, even if our results did not reflect this as clearly.

VI. Conclusion and Future Work

We explore how geometry can be leveraged in grasp synthesis. We incorporate a learned signed distance function via a shared embedding space for grasp success prediction and add collision constraints to the grasp optimization to yield geometrically-aware grasp synthesis. Our results indicate that while our approach exhibits desirable qualitative geometric reasoning (e.g., Fig. 8), the difficulty of the optimization hurts our approach's quantitative gains. In future work, we hope to improve the constrained optimization efficacy and better understand when object geometry matters. We also plan to explore incorporating feedback from multiple viewpoints and tactile sensing [34] to improve reconstruction and, in-turn, grasp success predictions. Finally, we hope to explore using the learned grasp success models as constraints for optimization of in-hand manipulation tasks [35].

Acknowledgments

Mark Van der Merwe was supported in part by UROP at the University of Utah. Qingkai Lu and Balakumar Sundaralingam were supported by NSF Awards #1846341 and #1657596. The authors would like to thank M. Wilson, G. Tabor, and F. Ramos for helpful discussions.

REFERENCES

[1] A. Sahbani, S. El-Khoury, and P. Bidaud, "An overview of 3D object grasp synthesis algorithms," *Robotics and Autonomous Systems*, vol. 60, no. 3, pp. 326–336, 2012. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0921889011001485

[2] J. Bohg, A. Morales, T. Asfour, and D. Kragic, "Data-driven grasp synthesis-A survey," *IEEE Transactions on Robotics*, vol. 30, no. 2, pp. 289–309, 2014. [Online]. Available: http://ieeexplore.ieee.org/document/6672028/

[3] A. Zeng, S. Song, K. T. Yu, E. Donlon, F. R. Hogan, M. Bauza, D. Ma, O. Taylor, M. Liu, E. Romo, N. Fazeli, F. Alet, N. C. Dafle, R. Holladay, I. Morena, P. Qu Nair, D. Green, I. Taylor, W. Liu, T. Funkhouser, and A. Rodriguez, "Robotic pick-and-place of novel objects in clutter with multi-affordance grasping and cross-domain image matching," in *IEEE Intl. Conf. on Robotics and Automation*. IEEE, 2018, pp. 3750–3757. [Online]. Available: https://ieeexplore.ieee.org/document/8461044/

[4] L. Pinto and A. Gupta, "Supersizing self-supervision: Learning to grasp from 50K tries and 700 robot hours," in *IEEE Intl. Conf. on Robotics and Automation*, 2016, pp. 3406–3413. [Online]. Available: http://ieeexplore.ieee.org/document/7487517/

[5] 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," *International Journal of Robotics Research*, vol. 37, no. 4-5, pp. 421–436, 2018. [Online]. Available: http://arxiv.org/abs/1603.02199

[6] J. Mahler, M. Danielczuk, B. DeRose, V. Satish, K. Goldberg, M. Matl, and S. McKinley, "Learning ambidextrous robot grasping policies," *Science Robotics*, vol. 4, no. 26, 2019.

[7] J. Varley, J. Weisz, J. Weiss, and P. Allen, "Generating multi-fingered robotic grasps via deep learning," in *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems*, 2015, pp. 4415–4420. [Online]. Available: http://ieeexplore.ieee.org/document/7354004/

[8] Q. Lu, K. Chenna, B. Sundaralingam, and T. Hermans, "Planning Multi-Fingered Grasps as Probabilistic Inference in a Learned Deep Network," in *Intl. Symposium on Robotics Research*, 2017. [Online]. Available: http://arxiv.org/abs/1804.03289

[9] Q. Lu and T. Hermans, "Modeling Grasp Type Improves Learning-Based Grasp Planning," *IEEE Robotics and Automation Letters*, vol. 4, no. 2, pp. 784–791, 2019. [Online]. Available: https://ieeexplore.ieee.org/document/8613819/

[10] D. Kappler, J. Bohg, and S. Schaal, "Leveraging big data for grasp planning," in *IEEE Intl. Conf. on Robotics and Automation*, 2015, pp. 4304–4311. [Online]. Available: http://ieeexplore.ieee.org/document/7139793/

[11] Y. Zhou and K. Hauser, "6DOF Grasp Planning by Optimizing a Deep Learning Scoring Function," in *RSS Workshop on Revisiting Contact - Turning a Problem into a Solution*, 2017.

[12] M. Liu, Z. Pan, K. Xu, K. Ganguly, and D. Manocha, "Generating Grasp Poses for a High-DOF Gripper Using Neural Networks," *arXiv*, 2019. [Online]. Available: https://youtu.be/ULvl{_}vfFQjo/http://arxiv.org/abs/1903.00425

[13] 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.

[14] M. Liu, Z. Pan, K. Xu, K. Ganguly, and D. Manocha, "Generating grasp poses for a high-dof gripper using neural networks," *CoRR*, vol. abs/1903.00425, 2019. [Online]. Available: http://arxiv.org/abs/1903.00425

[15] Q. Lu, M. Van der Merwe, B. Sundaralingam, and T. Hermans, "Multi-fingered grasp planning via inference in deep neural networks," *IEEE Robotics & Automation Magazine*, 2020.

[16] I. Lenz, H. Lee, and A. Saxena, "Deep Learning for Detecting Robotic Grasps," *Intl. Journal of Robotics Research*, vol. 34, no. 4-5, pp. 705–724, 2015.

[17] 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," *Intl. Journal of Robotics Research*, p. 0278364917710318, 2016.

[18] A. Brock, T. Lim, J. M. Ritchie, and N. Weston, "Generative and Discriminative Voxel Modeling with Convolutional Neural Networks," *arXiv*, 2016. [Online]. Available: http://arxiv.org/abs/1608.04236

[19] C. B. Choy, D. Xu, J. Gwak, K. Chen, and S. Savarese, "3D-R2N2: A Unified Approach for Single and Multi-view 3D Object Reconstruction," in *European Conf. on Computer Vision*, 2016. [Online]. Available: http://arxiv.org/abs/1604.00449

[20] L. Mescheder, M. Oechsle, M. Niemeyer, S. Nowozin, and A. Geiger, "Occupancy Networks: Learning 3D Reconstruction in Function Space," in *IEEE Conf. on Computer Vision and Pattern Recognition*, 2019. [Online]. Available: http://arxiv.org/abs/1812.03828

[21] J. J. Park, P. Florence, J. Straub, R. Newcombe, and S. Lovegrove, "DeepSDF: Learning Continuous Signed Distance Functions for Shape Representation," in *IEEE Conf. on Computer Vision and Pattern Recognition*, 2019. [Online]. Available: http://arxiv.org/abs/1901.05103

[22] Z. Chen and H. Zhang, "Learning Implicit Fields for Generative Shape Modeling," in *IEEE Conf. on Computer Vision and Pattern Recognition*, 2018. [Online]. Available: http://arxiv.org/abs/1812.02822

[23] J. Varley, C. Dechant, A. Richardson, J. Ruales, and P. Allen, "Shape completion enabled robotic grasping," in *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems*, 2017, pp. 2442–2447.

[24] X. Yan, J. Hsu, M. Khansari, Y. Bai, A. Pathak, A. Gupta, J. Davidson, and H. Lee, "Learning 6-DOF Grasping Interaction via Deep Geometry-Aware 3D Representations," in *IEEE Intl. Conf. on Robotics and Automation*, 2018, pp. 3766–3773.

[25] M. Zucker, N. Ratliff, A. D. Dragan, M. Pivtoraiko, M. Klingensmith, C. M. Dellin, J. A. Bagnell, and S. S. Srinivasa, "CHOMP: Covariant Hamiltonian optimization for motion planning," *International Journal of Robotics Research*, vol. 32, no. 9-10, pp. 1164–1193, 2013. [Online]. Available: http://journals.sagepub.com/doi/10.1177/0278364913488805

[26] J. Lundell, F. Verdoja, and V. Kyrki, "Robust Grasp Planning Over Uncertain Shape Completions," *arXiv*, 2019. [Online]. Available: http://arxiv.org/abs/1903.00645

[27] C. R. Qi, L. Yi, H. Su, and L. J. Guibas, "Pointnet++: Deep hierarchical feature learning on point sets in a metric space," in *Advances in Neural Information Processing Systems*, 2017, pp. 5099–5108.

[28] W. Wu, Z. Qi, and L. Fuxin, "Pointconv: Deep convolutional networks on 3d point clouds," in *IEEE Conf. on Computer Vision and Pattern Recognition*, June 2019.

[29] 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 *Intl. Conf. on Advanced Robotics*, 2015, pp. 510–517. [Online]. Available: http://ieeexplore.ieee.org/document/7251504/

[30] F. Biscani, D. Izzo, and C. H. Yam, "A global optimisation toolbox for massively parallel engineering optimisation," *arXiv preprint arXiv:1004.3824*, 2010.

[31] D. Kraft and K. Schnepper, "Slsqpa nonlinear programming method with quadratic programming subproblems," *DLR, Oberpfaffenhofen*, 1989.

[32] C. J. Ong and E. G. Gilbert, "The gilbert-johnson-keerthi distance algorithm: A fast version for incremental motions," in *Proceedings of International Conference on Robotics and Automation*, vol. 2. IEEE, 1997, pp. 1183–1189.

[33] S. Gottschalk, M. C. Lin, and D. Manocha, "Obbtree: A hierarchical structure for rapid interference detection," in *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*. Citeseer, 1996, pp. 171–180.

[34] B. Sundaralingam, A. Lambert, A. Handa, B. Boots, T. Hermans, S. Birchfield, N. Ratliff, and D. Fox, "Robust Learning of Tactile Force Estimation through Robot Interaction," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2019. [Online]. Available: https://arxiv.org/abs/1810.06187

[35] B. Sundaralingam and T. Hermans, "Geometric In-Hand Regrasp Planning: Alternating Optimization of Finger Gaits and In-Grasp Manipulation," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2018.