Proceeding of the IEEE
International Conference on Robotics and Biomimetics
Dali, China, December 2019

A Cascaded Deep Learning Framework for Real-time and Robust Grasp Planning*

Hongxiang Yu, Qianen Lai, Yuwei Liang, Yue Wang and Rong Xiong
College of Control Science and Engineering
Zhejiang University
Hangzhou, Zhejiang Province, China
hongxiangyu@zju.edu.cn

***Abstract*— Robotic grasping has been regarded as an essential way for robots to interact with the environment, which makes it a research hotspot. To our knowledge, current work on grasp planning could rarely take both robustness and time efficiency into account. To solve this problem, we propose a cascaded deep learning framework, which consists of a regression CNN and a refined CNN. Taking depth image as input, the first network regresses an optimal grasp region. The second network samples and sorts grasp candidates in the region and finally outputs a series of grasp poses with high quality. This architecture ensures real-time performance and acquisition of adequate candidates synchronously, which is especially appropriate for unstructured environment manipulation. We used Gazebo simulation and DART physics engines to build a grasp data generation environment. Comparing with manually annotating or robot collecting, our method makes data generation less expensive, more dynamic, and easier to generalize into the real world. The method proposed is evaluated in both simulation and physical experiments, verifying that our method is able to find multiple robuster grasp poses at a faster rate than the state of the art.**

***Keywords*— Robotic Grasping, Deep Learning, Simulation**

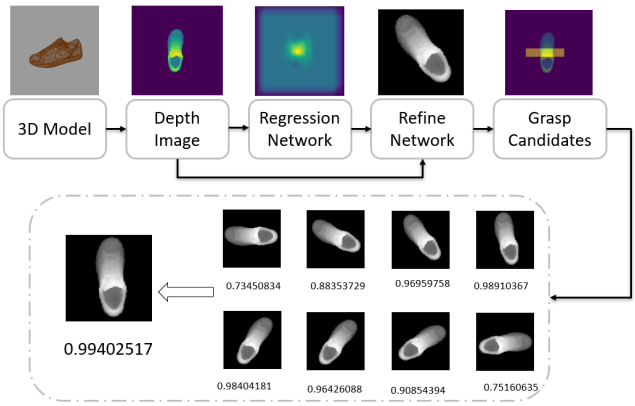

Fig. 1. For a depth image, the Regression Network evaluates each pixel to select a region that have the highest grasp quality. Then we sample from the region with different poses to crop the depth image as the input of the Refined Network. After getting the corresponding scores for each grasp candidates, we finally choose the grasp poses with highest qualities as the planning result.

I. Introduction

Nowadays, intelligent robots are facing increasingly complicated operation requirements, such as passing a tool for human or fruit picking. In these scenarios, robots are asked to interact with objects with various shapes, materials, and poses in unstructured environment and to have the ability to quickly acquire stable grasps through visual input. Whether it is industrial robots on the assembly line or service robots for the family, robots are facing more dynamic and uncertain tasks that makes fast and stable grasp planning algorithms essential.

In recent years, many grasp planning methods based on convolutional neural network have emerged with the development of deep learning [1][2][3][4]. The most popular methods can be classified into two categories. One type utilizes the powerful perceptual ability of deep neural network, directly outputs pixel level estimation of grasp, angle, and width density[5][2], which leads to excellent real-time performance that makes closed loop control realizable. However, the direct estimation of density means the angle and length of the grasp of each pose are fixed. It is especially unreasonable for some objects with rotational symmetry to have only one gripping pose. If the pose is unreachable on account of an obstacle nearby or the arm kinematics, re-planning is needed and the robustness cannot be guaranteed. Another kind of mainstream methods is to sample multiple grasp candidates based on the sliding window. By evaluating each pose and sorting according to its quality, the best pose is found [4][3]. In order to obtain the optimal grasp pose, it is necessary to consider all four degrees of freedom for translation and rotation, which consumes a large amount of computing resources and cannot guarantee real-time performance. Facing the trade-off between robustness and real-time performance, how to balance this two aspects becomes a major challenge for robotic grasp planning.

To achieve simultaneous real-time and robust grasp planning, we propose a cascaded deep learning framework which consists of a regression CNN and a evaluation CNN. Figure 1

*This work is supported by National Nature Science Foundation of China (U1609210), Science and Technology Project of Zhejiang Province (2019C01043), and Fundamental Research Funds for the Central Universities.

978-1-7281-6321-5/19//$31.00 ©2019 IEEE

gives illustration of the framework. We train these two deep neural networks separately, while they work in sequence to accelerate planning process. The first network aims at rough planning, is called the Regression Network. It regresses the global grasp quality map and feeds an optimal grasp region to the second network. While the second network is called the Refined Network. It provides a precise evaluation score for grasp candidates and is where multiple poses are sampled and ranked. That is to say, our model has the ability to acquire multiple grasp candidates in a very short time. In order to meet the requirements of network training, we also use the Gazebo simulation platform together with the DART physics engine to build a data generation environment. Compared with the manual labeling or real world acquisition, this data generation method has the advantages of easier replicability, low time cost and great dynamic property.

To summarize, our main contributions are:

- An efficient cascaded deep learning framework, containing two networks, that both the real-time performance and grasp robustness can be guaranteed for robotic manipulation in unstructured environment.
- The Regression Network, enlightened by GGCNN[5] while the angle and width regression layers are removed, can quickly locate the optimal region with the highest overall grasp quality, greatly reducing unnecessary calculations.
- The Refined Network, a classifier based on ResNet18 for refined planning, has the ability of high-level evaluation, which is particularly suitable for differentiating poses with different angles.

II. Related Works

Grasp synthesis problem has been studied for nearly three decades and can be achieved by analytical or empirical approaches[6].

Analytical approaches assume a multi-fingered hand and objects' 3D models are both available, then an objective function based on geometric, kinematic, or dynamic formulations[7] will be optimized to select the finger positions and hand configurations. Ferrari et al. [8] proposed a widely used metric in which grasp wrench space is computed based on the convex hull over the wrenches. Nguyen et al. [9] detected dependent contact regions for the fingertips, that totally constrained the motion of the grasped object. The early research in grasp synthesis often uses the analytical method, which is generally based on the CAD model of the known object. A robust grasp would be directly planned on the CAD model of the object[10][11]. Therefore, analytical grasp synthesis dataset is constructed as the mapping relationship between the grasp position and the object CAD model[12][13] which requires the object point cloud to match the CAD model in the database[14] that leads to large amounts of computation time. Also, depending too much on the pre-build database, these methods cannot well generalize to novel objects.

Empirical, also called data-driven approaches rely on sampling and ranking the grasp candidates[4][3] according to past grasp experience or some pre-determined criterion. Early works like GraspIt![15] develop a simulation environment in which objects can be loaded and evaluated through several quality metrics. Due to the excellent performance of deep learning[16] algorithms in the field of computer vision, the recent excellent grasp synthesis algorithms are based on convolutional neural networks[17], which extract features from the image space. The corresponding datasets are generated as pose and grasp quality pairs in depth images. Totally, there are mainly four routes to generate training data, manual annotation, physical trials in real world, analytical metric computation and sampling in simulation. Lenz et al. generated the Cornell grasp data set by manual annotation[1]. Levine et al. [18] collect over 800,000 grasp attempts using between 6 and 14 robotic manipulators over 2 months. Jeffrey Mahler et al. generated a synthetic grasp dataset for parallel gripper of 6 million images by analytical metrics in Dex-Net 2.0[4]. As for suction, Dex-Net 3.0[19] computes robust suction grasp's quality in Point Clouds using modified GQ-CNN. Edward Johns et al. used the DART physical simulation to generate a grasp dataset[3]. Amaury Depierre et al. [20] used the pybullet physics engine for simulation grasping and proposed a synthetic grasp dataset called Jacquard.

Grasping planning using deep convolutional neural networks is considered as a location detection problem in the image space to find the most robust grasp pose[4][1][21]. The common approach is to pre-sample the candidate grasps in the workspace and uses the neural networks to learn evaluators that filter the feasible grasp poses. Jeffrey Mahler et al. proposed the GQCNN framework evaluated each grasp with the quality metrics and sorted the quality of candidate grasps with high accuracy[4]. The network proposed by Edward Johns et al. [3] and the network proposed by Pinto L et al. [21] both predicted the quality of a set of discrete candidate grasps and directly obtained a single optimal grasp pose. In order to accelerate the detection speed, the recent GGCNN network proposed by Douglas Morrison et al. [5] directly evaluated the grasp quality of consecutive pixels in the image space and simultaneously regressed the grasps angle and width with a single neural network. This architecture has a very fast processing speed, but each pixel has one fixed grasp angle and width. Also, it has lower accuracy than the grasp evaluation methods. In order to overcome the challenging problem of localizing robot grasp configurations directly from the point cloud, Hongzhuo Liang et al. [22] introduced their convolutional neural network called PointNetGPD which can directly process the 3D point cloud.

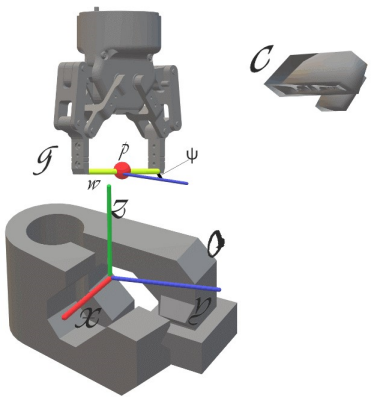

Fig. 2. The object O is represented by a coordinate frame (x, y, z) that fixed it, and a grasp pose g is represented by the grasp center position p, the grasp width w, and the grasp angle ϕ around z-axis.

III. Grasp Definition

We use a specific parallel gripper to grasp objects and, as shown in Figure 2, we define a grasp state in the world coordinate as

$$g = (p, \phi, w) \in R^3 \times S^1 \times R^1 \tag{1}$$

where $p = (x, y, z) \in R^3$ is the position of grasp center point in cartesian space, $w \in R^1$ is the maximal open width of gripper. As the grippers are always perpendicular towards the work plane, we define $\phi \in S^1$ as the rotation around the z axis.

To transform a grasp pose from cartesian space to image space, we have the depth image defined as

$$I = (o,T_o,K_c,T_c) \in R_+^{H \times W} \tag{2}$$

where H,W are the height and width, subscript o represents the property and geometric shape of the object to be grasped, c represents the parameters of the camera, T_o and T_c are the position matrix of the object and the camera in the world coordinate respectively, K_c is the camera's intrinsic parameter matrix. Through the projection transformation of the camera, capturing a depth image i, a grasp g in the workspace turns into g_i in the image space

$$g_i = K_c T_c g = (p_i, \phi_i, w_i, d_i) \in R^2 \times S^1 \times R^1 \times R^1 \tag{3}$$

where $p_i \in R^1$ is the pixel coordinate of the grasp center projected into the image, $\phi_i \in S^1$ is the angle between the grasp axis and the image horizontal axis in pixel coordinates, $w_i \in R^1$ is the pixel width, d_i is the depth of grasp center. As in reference[5], d_i is obtained by the fixed offset of the depth in the grasp center.

IV. Dataset Generation

Training deep neural networks requires large-scale data. At present, there are mainly four methods to acquire grasp training data, manual labeling, real-world robot acquisition, analytical criteria calculation and sampling in simulation. Manual annotation and real-world acquisition can easily generalize to actual execution. However their drawbacks are higher cost, therefore not suitable for generating large-scale datasets. Methods based on analytical criteria or physical simulation can acquire a large amount of data in a short time, but analytical criterion only considers the capture moment, cannot measure the dynamic features after the execution. Therefore, we use the DART physics engine integrated in Gazebo simulation platform for data generation, it can easily verify the dynamic performance of the grasp. Figure 3 shows the results of several grasp attempts. In addition, DART simulation can also obtain the depth information of the object conveniently. Compared with the RGB image, depth information is more stable towards the changes of environment illumination or objects material, which seems appropriate for constructing the map from extracted feature to the grasp quality. Errors can come from everywhere, such as the noise of robotic arm's joints, the inaccuracy of camera and the robot's calibration. So uncertainty is added to improve generalization performance[23].

A. Physical Simulation

The DART physics engine has a good performance in calculating force[24], mesh model collision detection, and dynamic character modeling. It can excellently reflect real-world behaviors, for example, we can obtain model poses variation and stable grasping time after execution, and even the anti-disturbance ability in certain direction is available. We build a parallel gripper in the simulation. This gripper consists of a blue base link, a yellow rod and two fingers. The base link has four degrees of freedom and can move

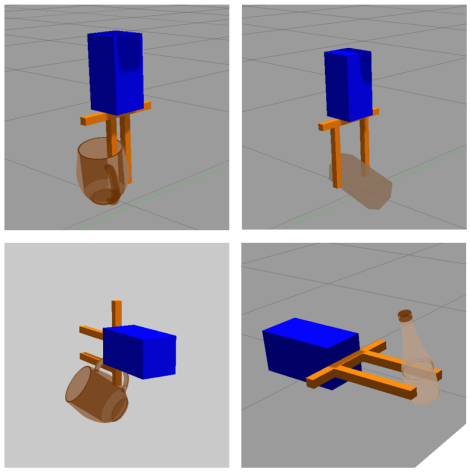

Fig. 3. Samples with DART physics engine in the simulation, which show the grasp effect in both vertical and horizontal directions. We can grasp the same object in various ways.

freely in space or rotate around the z-axis. The length of the yellow rod is 10 cm. The rod and the base link are fixed, and the fingers, controlled by effort controller, move antipodaly on the rod. In order to ensure that the physical simulation has the most realistic behavior, the length of gripper, the surface friction coefficient, the density of object and the force exerted by fingers are all manually determined.

When a grasp is performed, the base link moves and rotates according to the output pose, and then the opposing force is applied to the two fingers by the controller until the two fingers are closed. Next, base link lifts by 30 cm at a speed of 0.3 m/s. In order to make the dataset more generalizable, a small Gaussian noise is added to the grasp pose $g = (p, \phi, w)$ each time we perform the grasp. The actual grasp pose is as follows:

$$\widetilde{g} = \left(\widetilde{p}, \widetilde{\phi}, w\right) = \left(\left(x + \chi_p, y + \chi_p, z + \chi_p\right), \left(\varphi + \chi_\phi\right), w\right) \tag{4}$$

where $\chi_p \sim \mathcal{N}\left(0, {\sigma_p}^2\right)$ and $\chi_\phi \sim \mathcal{N}\left(0, {\sigma_\phi}^2\right)$, σ_p is the standard deviation of pose, σ_ϕ is the standard deviation of angle.

B. Acquisition of Depth Information

We do not use OpenGL to render depth images, alternatively, a kinect simulation in Gazebo capture images directly. We add a Gaussian noise that simulates the uncertainty of perception[23] to the acquired images as an input for the deep neural network.

$$D\left(\widetilde{u}, \widetilde{v}\right) = d\left(u + \chi_u, v + \chi_v\right) + \chi_d \tag{5}$$

where $\chi_u \sim \mathcal{N}\left(0, {\sigma_u}^2\right)$, $\chi_v \sim \mathcal{N}\left(0, {\sigma_v}^2\right)$, $\chi_d \sim \mathcal{N}\left(0, {\sigma_d}^2\right)$, σ_u, σ_v are the standard deviation of the pixel coordinate, σ_d is the standard deviation of depth.

C. Grasp Synthesis

We selected 100 objects from the ModelNet database[25] building our datasets, such as cup, donut, banana, bottle, and other common objects. Each object is set in a number of stable poses and the longest side of the object is resized in the range of 10-20cm to match our gripper, both of these operations enrich the training data. In the process of data generation, model is placed in the center of the camera's field of view. Each pose of objects collects a total of 121 positions, and for each position 6 angles were sampled, for a total of 726 poses. If the object is clamped by the gripper 30 cm from the ground, the grasp attempt is considered to be successful. Each pose is attempted five times. If 5 attempts are all successful, the quality of the pose is labelled as 1. So each pose gets a discrete score between 0 and 1, with an interval of 0.2. This contributes much to the detailed information than simply annotating a pose by 0 or 1. A comfort index has also been recorded to indicate the offset from the object's original pose which is optional in training.

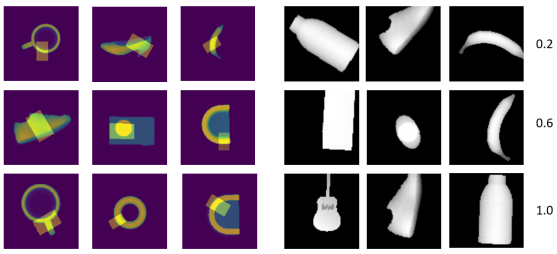

Fig. 4. The final synthetic depth images for training. The left is the training data labeled by rectangles for the Regression Network. The right is the training data in 6 categories for the Refined Network (3 categories samples are shown for illustration).

V. Learning a Grasp Function

A. Cascaded Network

The cascaded framework we proposed is illustrated in Figure 1. The input is a depth image i and the output is several grasp candidates with high quality. In the grasp region detection step, we consider a region with the highest quality in the image space $Region_i$, and all possible grasp positions within its radius R_i to form a possible grasp set.

$$Region_i = (Q_i, P_i) \in R_{0\text{-}1}^{H \times W} \times R^{H \times W} \tag{6}$$

where $Q_i \in R_{0\text{-}1}^{H \times W}$ is the quality set, $P_i \in R^{H \times W}$ is the pose set. We sample m pixels in $Region_i$, and let the Refined Network for pose evaluation to calculate the quality $Q_{k \in i} = f\left(g_k\right) = f\left(p_{k \in i}, \varphi_{k \in i}, w\right)$ corresponding to all possible grasp poses set G_k centered on each pixel k. Then by sorting $Q_k, k = 1, 2, ...m$, we set a threshold t, select n grasp candidates whose qualities exceed t as the final result.

B. Regression Network for Grasp Region Detection

The goal of the Regression Network is to quickly detect the region with the highest quality of the input depth image. For training, a lightweight CNN framework inspired by GG-CNN[5] is used. The network has six layers, as shown in Figure 5, its input is a 300×300 pixels depth image with global information and its output is a fitted map indicates the grasp density distribution. From this distribution, we can find the optimal region with highest quality as the input of the next Refined Network. Since we only care about the high quality areas, those poses with higher quality will be taken into account during the training.

As illustrated in Figure 4, using the camera's internal and external matrix K_c,T_c, we transform the grasp pose into a small rectangle, where the center of the rectangle is the pixel coordinates corresponding to the grasp pose's center, the length is 1/3 of the gripper pixel width, and the width is the length of the gripper pixel width. The rotation angle is the angle between the grasp pose and the horizontal axis of the image. The pixel values within the rectangle are set to be the score of the pose, and the outside pixels are uniformly set

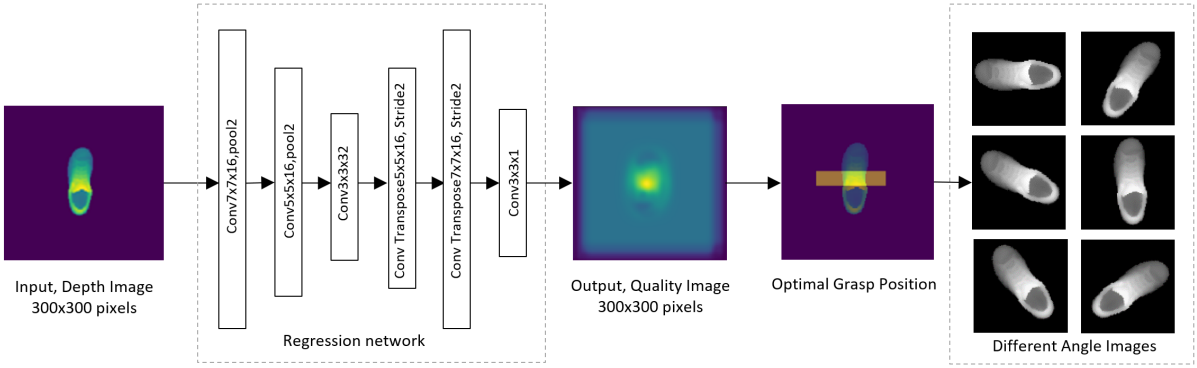

Fig. 5. The structure of the Regression Network. The input is a 300×300 pixels depth image with global information and the output is a density map with 300×300 pixels indicates the grasp quality distribution. From the distribution, the optimal region with highest quality will be found as the input of the next Refined Network.

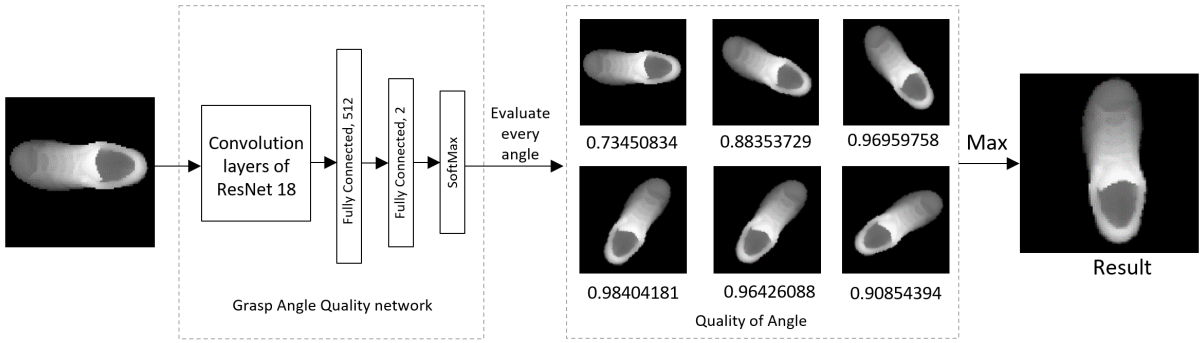

Fig. 6. The structure of the Refined Network. It evaluates grasps with different poses and finds the best grasp angle. The input depth image's length is decided by the pixel length of gripper width. After receiving the optimal region from the Regression Network, the Refined Network will give a consecutive score between 0 and 1 for each input image corresponding to its candidate pose. After sorting these scores, the final result can be induced.

to zero. During training, we select the poses whose labels are greater than 0.6, draw the grasp rectangle as the ground truth, then expand the dataset by rotating, scaling, etc. Since this is a regression problem, the mean square error loss function is introduced.

The evaluation of the model is as follows: according to the output grasp quality distribution, the pixel with the highest grasp quality is found. If we could found a ground truth grasp pose whose score is equal to 1 inside the circular area centered by the pixel with its radius 1/3 of the gripper length, the model is considered to have a successfully regression trial.

C. Refined Network for Grasp Pose Evaluation

The Refined Network establishes a map $q_k = F(g_k)$ of a single grasp pose g_k to the grasp quality q_k. We train the deep neural network whose input is the depth image corresponding to the grasp pose g_k, and output is a consecutive score between 0 and 1, which can be regarded as the grasp probability. The structure of the Refined Network is shown in Figure 6. The input depth image's length is decided by the pixel length of gripper width.

The network is substantially a classifier, consequently we construct the dataset according to the actual meaning of each grasp. The training set and the test set both contain two categories 0 and 1. If a pose g_k has ground true value q_k, then $5q_k$ cropped depth images centered at the grasp poses pixel coordinates (u_k, v_k) with a rotation of $-\varphi_k$ will be thrown into category 1. At the same time, $5(1-q_k)$ depth images are cropped and dropped into the category of 0. We use ResNet18 as the training framework. The network will predict the probability that each small depth image is classified as 0 or 1. After a softmax layer, the probability of category 1 is regarded as the grasp quality of this pose. The network uses the cross entropy loss function to realize a gradient descent on the mini-batches, the loss function is:

$$L = \sum_{i}^{B} \delta(k, y_i) \bullet softmax\left(y_i^k\right) \tag{7}$$

where

$$B : the\ number\ of\ training\ images\ of\ a\ batch$$

$$y_i : the\ ground\ truth\ score\ of\ ith\ image$$

$$\delta(k, y_i) = \begin{cases} 1 \; if \; k = y_i \\ 0 \; if \; k \neq y_i \end{cases}$$

VI. Experimental results

To validate our method, we conduct experiments in both simulation and real world. In the simulation experiment, we mainly verify that our method is faster than the state of the art methods which based on sampling + evaluation + sorting as GQCNN[4], and to verify that all grasps in the candidate set are feasible. We also conducted a physical experiment based on the ABB grasping platform as shown in Figure 7. The physical experiment is based on the depth information of the actual sampling by a Primesense 1.09, verifying whether the synthetic data can achieve feasible generalization results in the real world.

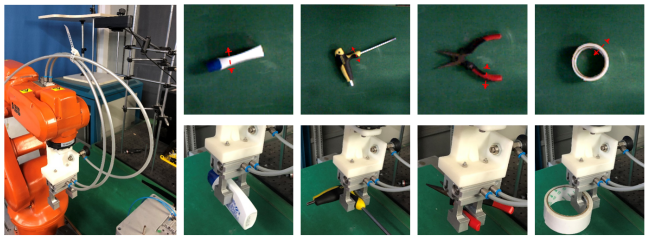

Fig. 7. Our experimental platform is based on the ABB-IRB120 robotic arm and a composite mechanical end-effector. The camera is Primessense 1.09 and is placed 700 mm above the plane. The right are some planning and implementation results in the physical experiment.

A. Simulation Experiments

We re-selected 20 objects from ModelNet to generate a validation set to test the generalization performance for novel objects. The model is placed in the field of camera's view in a random pose. Then we use a virtual depth camera to capture a depth image, and add uncertainy use a noise model. Using our framework, several optimal grasp poses are detected, shown in Figure 8, then those poses are executed in Gazebo to record the success rate. Repeat the above process for other methods, the final process time is 90ms and the success rate is 95% in Gazebo simulation.

B. Physical Experiments

We use 20 common object and 3D prints as validation objects set. As shown in Figure 9, these objects are placed in random pose on the experimental platform. Each object was subjected to 10 grasping experiments to record the results.

We recorded the success rate of the acquisition and the calculation time of planning, then compared it with the method proposed by Lenz[1], Jeffrey[4], Levine[18], Douglas[5] et al. The result is shown in Table 1, indicates that our algorithm, in terms of calculation time and acquisition accuracy, has a better performance. Since we use a fast quality regression network to reduce evaluation candidates, our algorithm has

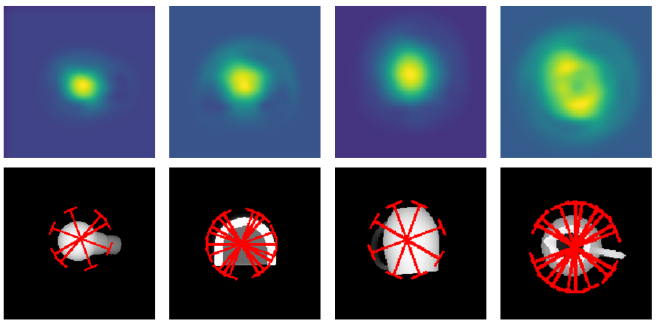

Fig. 8. Some of the grasp poses that our method planned in simulation experiments. For symmetric objects, the framework provides multiple symmetrical grasp poses.

Fig. 9. The objects we use in physical experiments.

obvious advantages in computing time compared with methods that sample large number of grasp candidates. At the same time, due to the high-quality evaluation network, our framework still has a greater success rate of capture.

TABLE I
Comparison With Other Methods

method	time	sucess rate
Lenz et al.	13.5s	89
Mahler et al.	0.8s	93
Levine et al.	0.2-0.5s	80
Douglas et al.	20ms	84
Our method	90ms	89

VII. Conclusion

In this paper, we propose a new grasp planning method that introduces an efficient cascaded network framework takes care of both the real-time performance and robustness. We use the DART physics engine in Gazebo to generate a large amount of synthetic grasp data that fits the real world, define the comfort and quality of the grasp pose, and guarantee to cover as much as possible poses in the possible workspace. Our proposed cascaded grasp planning framework consists of a fast regression network and a high-precision grasp

evaluation network that returns the grasp quality of each input pose. The regression network considers the entire image space and captures a best quality region which is given to the evaluation network for a more refined assessment, resulting in a fast and robust grasp plan. In order to achieve faster and more reliable planning in the industry, our future work will focus on the improvement of cascaded networks and the expansion of the datasets.

REFERENCES

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

[2] J. Redmon and A. Angelova, "Real-time grasp detection using convolutional neural networks," in *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1316–1322, IEEE, 2015.

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

[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] D. Morrison, P. Corke, and J. Leitner, "Learning robust, real-time, reactive robotic grasping," *The International Journal of Robotics Research*, p. 0278364919859066, 2019.

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

[7] A. Bicchi and V. Kumar, "Robotic grasping and contact: A review," in *Proceedings 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings (Cat. No. 00CH37065)*, vol. 1, pp. 348–353, IEEE, 2000.

[8] C. Ferrari and J. F. Canny, "Planning optimal grasps.," in *ICRA*, vol. 3, pp. 2290–2295, 1992.

[9] V.-D. Nguyen, "Constructing force-closure grasps," *The International Journal of Robotics Research*, vol. 7, no. 3, pp. 3–16, 1988.

[10] A. T. Miller, S. Knoop, H. I. Christensen, and P. K. Allen, "Automatic grasp planning using shape primitives," 2003.

[11] J. Bohg and D. Kragic, "Learning grasping points with shape context," *Robotics and Autonomous Systems*, vol. 58, no. 4, pp. 362–377, 2010.

[12] C. Goldfeder, M. Ciocarlie, H. Dang, and P. K. Allen, "The columbia grasp database," in *2009 IEEE international conference on robotics and automation*, pp. 1710–1716, IEEE, 2009.

[13] K. Huebner and D. Kragic, "Selection of robot pre-grasps using box-based shape approximation," in *2008 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 1765–1770, IEEE, 2008.

[14] A. Herzog, P. Pastor, M. Kalakrishnan, L. Righetti, T. Asfour, and S. Schaal, "Template-based learning of grasp selection," in *2012 IEEE International Conference on Robotics and Automation*, pp. 2379–2384, IEEE, 2012.

[15] A. T. Miller and P. K. Allen, "Graspit! a versatile simulator for robotic grasping," 2004.

[16] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *nature*, vol. 521, no. 7553, p. 436, 2015.

[17] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, pp. 1097–1105, 2012.

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

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

[20] A. Depierre, E. Dellandréa, and L. Chen, "Jacquard: A large scale dataset for robotic grasp detection," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 3511–3516, IEEE, 2018.

[21] L. Pinto and A. Gupta, "Supersizing self-supervision: Learning to grasp from 50k tries and 700 robot hours," in *2016 IEEE international conference on robotics and automation (ICRA)*, pp. 3406–3413, IEEE, 2016.

[22] H. Liang, X. Ma, S. Li, M. Görner, S. Tang, B. Fang, F. Sun, and J. Zhang, "Pointnetgpd: Detecting grasp configurations from point sets," in *2019 International Conference on Robotics and Automation (ICRA)*, pp. 3629–3635, IEEE, 2019.

[23] J. Weisz and P. K. Allen, "Pose error robust grasping from contact wrench space metrics," in *2012 IEEE international conference on robotics and automation*, pp. 557–562, IEEE, 2012.

[24] T. Erez, Y. Tassa, and E. Todorov, "Simulation tools for model-based robotics: Comparison of bullet, havok, mujoco, ode and physx," in *2015 IEEE international conference on robotics and automation (ICRA)*, pp. 4397–4404, IEEE, 2015.

[25] Z. Wu, S. Song, A. Khosla, F. Yu, L. Zhang, X. Tang, and J. Xiao, "3d shapenets: A deep representation for volumetric shapes," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1912–1920, 2015.