FCN-Based 6D Robotic Grasping for Arbitrary Placed Objects

Hitoshi Kusano[1], Ayaka Kume[2], Eiichi Matsumoto[2] and Jethro Tan[2]

***Abstract*— We propose a supervised end-to-end learning method that leverages on our grasp configuration network to predict 6 dimensional grasp configurations. Furthermore, we demonstrate a novel way of data collection using a generic teaching tool to obtain high-dimensional annotations for objects in 3D space. We have demonstrated more than 10,000 grasps for 7 types of objects and through our experiments, we show that our method is able to grasp these objects and propose a larger variety of configurations than other state-of-the-art methods.**

I. Introduction

The goal in robotic grasping is to derive a series of configurations of a robot and its end-effector when given data acquired from some sensors in such a way that its end-effector 'grasps' a user-defined target. Being able to grasp arbitrary placed objects would not only enable automation of tasks found in factories and logistics, but also progress the development towards a general purpose service robot. Traditional approaches to grasp an object often rely on object pose estimation [1], [2] or pattern recognition techniques to detect object-specific features in point cloud data [3], [4].

However, such approaches are ill-suited for use in objects placed in unstructured environments, or for use with complex end-effectors with high degrees-of-freedom (DOF), because they provide information about the state of the object, but not the state in which the robot or its end-effector should be in order to grasp the object. Therefore, these approaches make use of heuristics to select a configuration (i.e. joint angles, position, orientation, etc.) for the robot and its end-effector. Furthermore, detection of object-specific features may fail or provide false positives on either deformable objects or objects unknown to the system.

The recent advances and popularity of deep learning have given birth to new methods for grasping that are able to directly generate the configurations of the robot as well as its end-effector. The idea of these new methods is to evaluate the success rate in the configuration space of the end-effector and select the configuration with the highest score.

A. Problem Statement

These new methods, however, have a disadvantage when applied on higher DOF end-effectors or when there is need for increasingly complex grasp strategies as it becomes more and more difficult or even impossible to annotate the data. Therefore, such methods rely on low dimensional annotations making them unable to output 6D grasp configurations. In this work, we therefore pose the following question: *How to enable end-to-end learning to grasp arbitrary placed objects using high dimensional annotations?*

This work was supported by Preferred Networks, Inc.

[1]Kyoto University, Kyoto 606-8501, Japan, kusano@ml.ist.i.kyoto-u.ac.jp

[2]Preferred Networks Inc., Tokyo 100-0004, Japan, {kume, matsumoto, jettan}@preferred.jp

B. Contribution

While current state-of-the-art CNN-based grasping methods rely on annotation on images, to the best of our knowledge there has been no work that utilizes annotations in 3D space for grasping. Therefore, our contributions are as follows:

Contribution 1: A supervised learning method using an end-to-end convolutional neural networks (CNN) model that we call the grasp configuration network (GCN), capable of predicting 6D grasp configurations for a three-finger gripper as well as probability of the configurations to successfully grasp a target object.

Contribution 2: A novel data collection strategy to obtain 6D grasp configurations to use in the training of GCN using a generic teach tool.

The remainder of this paper is organized as follows. Related work is described in Section II, while Section III outlines the GCN. Experimental results using GCN and evaluation are presented in Section IV. Future work is discussed in Section V, while the paper is concluded in Section VI.

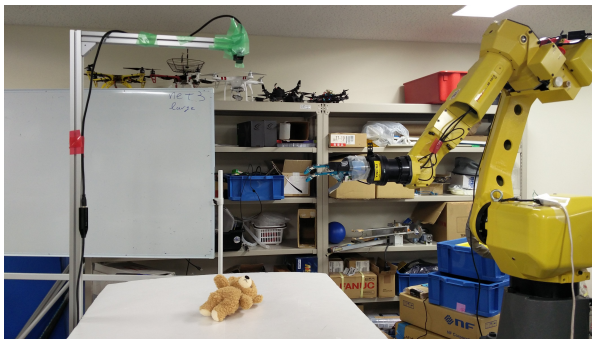
II. Related Work

A. Traditional Methods

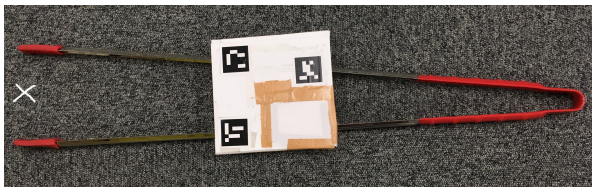
Methods not using machine learning include CAD model matching [5] or template matching. As these methods require careful feature engineering and parameter tuning, data-driven methods have been actively studied to overcome these disadvantages. For surveys on data-driven approaches for grasp synthesis, we refer the reader to Bohg *et al.* [6] and to Li *et al.* [7] for a method based on object pose estimation.

B. Machine Learning Methods

In the past few years, CNN have become the state-of-the-art for performing general object recognition tasks [8]. In our method, we leverage a fully convolutional network (FCN), which is a type of CNN meant for semantic segmentation [9], [10]. FCN has shown to be compatible with RGB-D input images, which can be exploited for to predict pickability [9]. Nguyen *et al.* [11] uses a CNN model to predict affordances for an object. Unlike our method, however, their CNN model does not predict configurations for the grasp itself. In [12], Lenz *et al.* deploy a deep neural network model with two stages to propose and score grasp configurations. Similarly, Araki *et al.* [13] proposed the use of a single-stage model which was able to simultaneously predict the grasp



(a) System setup



(b) Teach tool

Fig. 1. System setup used for data collection and testing: (a) (from left to right) (i) mount with Intel Realsense SR300 RGB-D camera, (ii) abritrary placed object, (iii) THK TRX-S 3-finger gripper on a (iv) FANUC M-10iA 6 DOF industrial robot arm and (b) Teach tool to demonstrate grasp and record its 6D gripper configuration. Grasp position marked with white 'X'.

probability and the configuration of the end-effector. Guo *et al.* [14] make use of a model inspired by Faster R-CNN [15] and were able to output 4 DOF grasp candidates at multiple locations in the image with a single forward computation of the neural network.

Pinto *et al.* [16] and Levine *et al.* [17] have proposed self-supervising methods by automating the data collection process. An alternative approach to collect data for grasping is from learning by demonstration. Gupta *et al.* [18] have trained a multi-fingered hand to manipulate by demonstrating the desired manipulation motions for objects with their own hands. The robot then tries to imitate these demonstrations using reinforcement learning.

III. The Grasp Configuration Network

A. Data Collection

We propose a method for manually annotated data collection. As teach tool, we created a gripper with three augmented reality (AR) markers, see Figure 1. This teach tool allows us to demonstrate grasps by capturing a 400×400 image from our RGB-D camera, after which the RGB and depth channels are registered and resized to 200×200 for training and inference of GCN. Although the shape of the gripper used in our system is not represented by our teach tool, the teach tool allows capturing of 6-dimensional configurations, containing the position (XYZ), as well as orientation (RPY) in the camera coordinate system, which can in turn be used as grasp configurations. When multiple configurations in one location are demonstrated for an object, all of them are utilized to train GCN.

Because we want to record the actual position of the demonstrated grasp instead of the AR markers, the offset between the AR markers and the center between the pinch tips are added to determine the true position of the demonstrated grasp. Additionally, to increase the accuracy of the observed rotation, two orthogonal vectors of the gripper (in the longitudinal and lateral direction) are used to calculate the rotation.

B. Model Overview

To predict end-effector configurations for grasping an object, we introduce an extension for fully convolutional networks (FCN) that we call the *grasp configuration network (GCN)*, as shown in Figure 2. As input, we provide an RGB-D image obtained from an Intel Realsense SR300 camera to our model, which in turn outputs two maps with scores: (1) a predicted location map (PLM) for graspability per pixel, and (2) a predicted configuration map (PCM) providing end-effector configurations per pixel.

C. Data Discretization

Despite the multimodal nature of grasping, prior works have tried to use regression to model grasp configurations [14]. However, since the optimal value in a regression model is calculated by minimizing a mean of distances between outputted and training data, an assumption is needed that a configuration between multiple grasp configurations must also be valid.

If we let g_1 and g_2 be valid grasp configurations for a target object o in location (x, y), we argue that the mean of g_1 and g_2 might neither be an optimal nor valid grasp. We therefore discretize grasp configurations and use a classification model instead, as proposed in [21]. A naive solution to achieve discretization is to take all possible combination of quantized configurations along each degree of freedom. However, the amount of combinations would increase exponentially as the dimension of a configuration increases. To alleviate this problem, we categorize valid grasp configurations to 300 classes using k-means clustering.

D. Training

1) Pre-training: Similar to FCN, first initialize the weights in the first layer corresponding to RGB of the convolution network using VGG-16 pre-trained on ILSVRC dataset [8] followed by fine-tuning on the NYUDv2 pixel-classification task [22]. We then use these weights as initial weights of GCN.

2) Settings: To train GCN, we employ Adam as optimizer with learning rate $\alpha = 0.0004$ and batch size of 80. To further reduce overfitting during training, we perform data augmentation by applying label-preserving transformations.

Because of the imbalance ratio between positive and negative data (less than $1:10000$), we define L_{PLM}, the loss function for PLM to magnify the gradient for valid grasp as follows. tywhere a denotes the magnification ratio, n the number of training data, W, H the width and the height of the image, t the target PLM, and y the output. We use a value of 200 for magnification ratio a.

For PCM, we calculate the loss for only valid locations. We define L_{PCM} as follows.

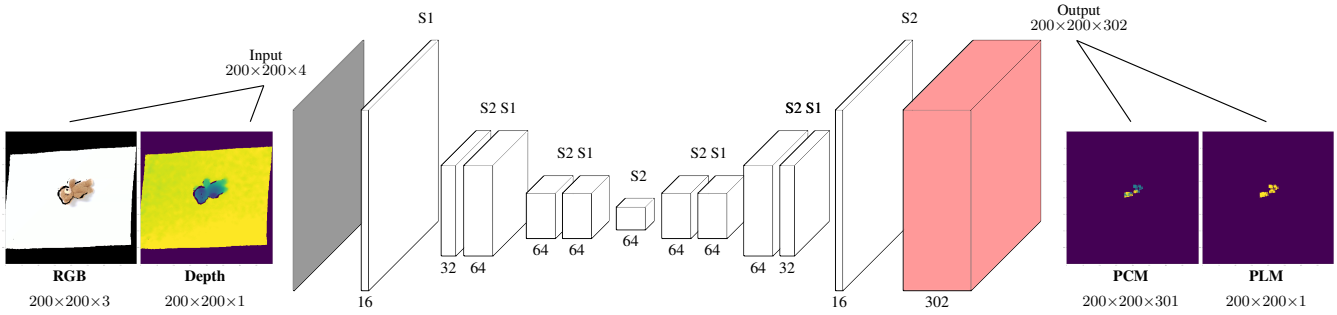


Fig. 2. Network architecture of the grasp configuration network (GCN). The stride of each layer is denoted with S1 (stride 1) or S2 (stride 2). S1 layers use a kernel size of 3×3, and 4×4 for S2 layers. All layers have their padding set to 1. After each layer except for the last layer we apply batch-normalization [19] and use rectified linear units [20] as activation functions. For the last layer, we apply sigmoid function. For training, GCN receives a 64×64 RGB-D image, and a 200×200 RGB-D image as input for inference. Outputs are a predicted location map (PLM) and a predicted configuration map (PCM).

(a) Known objects

(b) Unknown objects

Fig. 3. Objects used in our experiments (from): (A) laundry detergent, (B) plush bear, (C) energy drink, (D) white board eraser, (E) vacuum hose piece, (F) sprinkler, (G) wool, (H) cap, (I) glucose bottle, (J) duct tape, (K) bottle of tea, and (L) sippy cup. Items (A)–(G) were used in the training process, while (H)–(L) were left as 'unknown' objects for experiments.

$$L_{\text{PCM}} = -\frac{1}{nC}\sum_{k=1}^{n}\frac{1}{S_k}\sum_{c=1}^{C}\sum_{i=1,j=1}^{W,H}(t_k^{(i,j)}(Cu_k^{(i,j,c)}\log y_k^{(i,j,c)} + (1-u_k^{(i,j,c)})\log(1-y_k^{(i,j,c)}))),$$

where C denotes the number of classes, S_k the total number of pixels at which $t_k^{(i,j)} = 1$, u the target PCM, and y the output. We use $L = L_{\text{PLM}} + \lambda L_{\text{PCM}}$ as full objective and we set $\lambda = 200$.

IV. Experiment Results

A. Setup

To evaluate the performance of GCN on grasping, we have conducted experiments on 12 objects with different rigidness and shapes, see Figure 3. Experiments were done in an office environment using the same setup shown in Figure 1. In this environment, the camera provides a view on a 70×50 cm plane in which objects can be placed by us and grasped by the robot. The camera itself is mounted 75 cm above this plane.

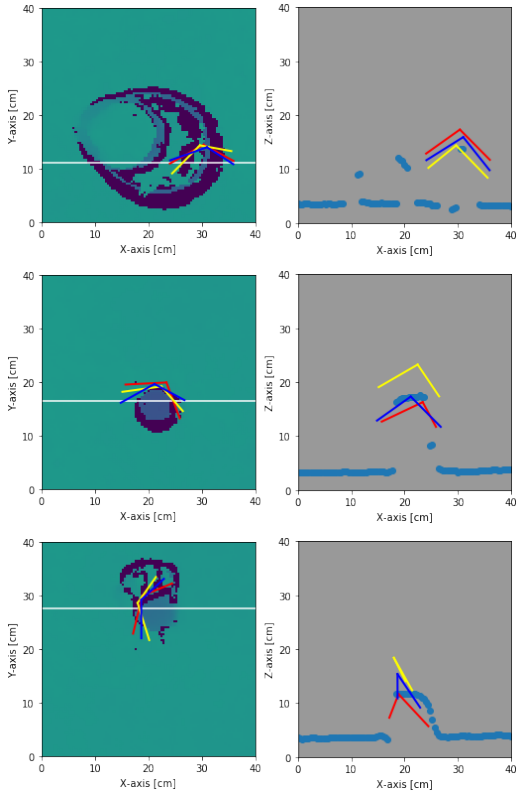


Fig. 4. Example of top scoring grasp configurations on items (H), (I), and (L) in PCM for the top scoring location in PLM. The best scoring configuration is colored red, while yellow and blue shows the second and third best scoring configurations, respectively.

Data collection was performed for only objects (A)–(G). For each of these seven objects, we created 12 cases for arbitrary placement with at least 100 demonstrated grasps per case, totaling to 11320 demonstrated grasps before data augmentation, and 35865 after data augmentation. Data collection per object took about 90 minutes. The remaining objects ((H)–(L)) were used to evaluate the performance of GCN for unknown objects. Training of GCN was performed

TABLE I
RESULTS FOR OBJECT GRASPING EXPERIMENT[1]

Object	Success Rate [%]
(A)	70
(B)	50
(C)	60
(D)	40
(E)	20
(F)	40
(G)	60

[1] Each object has been trialed 10 times.

TABLE II
RESULTS FOR UNKNOWN OBJECT GRASPING EXPERIMENT[1]

Object	Success Rate[%]
(H)	60
(I)	20
(J)	20
(K)	40
(L)	30

[1] Each object has been trialed 10 times.

on a remote server equipped with an Nvidia GeForce GTX Titan X (Maxwell) and took 130 minutes for 2300 epochs, while testing was conducted on a local PC with an Nvidia GeForce GTX 970.

One single inference of GCN on our testing machine took about 0.188 sec. After performing inference on GCN for an object, we select the location with the highest score from PLM followed by the configuration with the highest score for that location in PCM. Additionally, we filter out configurations that causes the robot to collide with its surroundings.

To perform a grasp, we have written a program on the teach pendant which moves the robot from a set home position to an approach position that is 10 cm away from the grasp position proposed by GCN. This is then followed by the grasp motion itself where the hand is actuated to squeeze the object. We consider a grasp to be successful if the target item is still pinched between the fingers of our gripper when the robot returned to its home position afterwards.

B. Results

Example outputs of GCN is shown in Figure 4, displaying the ability of GCN to propose different grasp configurations for the same (X,Y) location for multiple items.

Results of our experiments for both grasping known and unknown objects are listed in Table I and Table II. The low success rate of object (E) can be explained by the camera not being able to capture the differences in depth due to its complex shape.

V. FUTURE WORK

Despite the results of our experiments, our data set used to train GCN does not reflect real world settings in e.g. warehouse logistics. Therefore, we plan to not only extend our data set with more items, but also add cases in which an object is arbitrary placed in cluttered environments. Furthermore, since GCN has shown potential to learn more complex grasp configurations such as pinching, we would like to extend our method using higher DOF end-effectors. Other functionalities that can improve the performance and robustness of GCN, thus requiring consideration includes using data from point clouds instead of only a depth channel, and obtaining multiple views of the grasping scene.

VI. CONCLUSION

In this paper, we proposed a fully convolutional network model, called the grasp configuration network (GCN) that is able to output 6D grasp configurations when given an RGB-D image as input. Furthermore, we introduced a data collection method for use with GCN that leverages on demonstrations of the grasps rather than on-image annotations to enable training on high-dimensional annotated data. Through the use of GCN, a classification-based model, we demonstrated robotic grasps on arbitrary placed objects with different rigidness and shapes.

REFERENCES

[1] N. Hudson, *et al.*, "End-to-end dexterous manipulation with deliberate interactive estimation," in *Proc. IEEE ICRA*, May 2012.

[2] M. Zhu, *et al.*, "Single image 3D object detection and pose estimation for grasping," in *Proc. IEEE ICRA*, 2014.

[3] A. Herzog, *et al.*, "Learning of grasp selection based on shape-templates," *Autonomous Robots*, vol. 36, no. 1-2, 2014.

[4] Y. Domae, *et al.*, "Fast graspability evaluation on single depth maps for bin picking with general grippers," in *Proc. IEEE ICRA*, 2014.

[5] U. Klank, *et al.*, "Real-time cad model matching for mobile manipulation and grasping," in *Proc. IEEE Humanoids*, 2009.

[6] J. Bohg, *et al.*, "Data-driven grasp synthesis - a survey," *IEEE Transactions on Robotics*, vol. 30, no. 2, Apr. 2014.

[7] W. Li, *et al.*, "Recent advances on application of deep learning for recovering object pose," in *Proc. IEEE ROBIO*, Dec 2016.

[8] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[9] J. Long, *et al.*, "Fully convolutional networks for semantic segmentation," in *Proc. IEEE CVPR*, 2015.

[10] P. Pinheiro, *et al.*, "Learning to segment object candidates," in *Proc. NIPS*, 2015.

[11] A. Nguyen, *et al.*, "Detecting object affordances with convolutional neural networks," in *Proc. IEEE IROS*, Oct 2016.

[12] I. Lenz, *et al.*, "Deep learning for detecting robotic grasps," *The International Journal of Robotics Research (IJRR)*, vol. 34, no. 4-5, 2015.

[13] R. Araki, *et al.*, "Graspabilityを導入したDCNNによる物体把持位置検出 (Introducing Graspability for Object Grasp Position Detection by DCNN)," in *The Robotic Society of Japan*, 2016.

[14] D. Guo, *et al.*, "Deep vision networks for real-time robotic grasp detection," *International Journal of Advanced Robotic Systems (IJARS)*, vol. 14, no. 1, 2017.

[15] S. Ren, *et al.*, "Faster R-CNN: Towards real-time object detection with region proposal networks," in *Proc. NIPS*, 2015.

[16] L. Pinto and A. Gupta, "Supersizing self-supervision: Learning to grasp from 50k tries and 700 robot hours," in *Proc. IEEE ICRA*, 2016.

[17] S. Levine, *et al.*, *Learning Hand-Eye Coordination for Robotic Grasping with Large-Scale Data Collection*. Springer International Publishing, 2017.

[18] A. Gupta, *et al.*, "Learning dexterous manipulation for a soft robotic hand from human demonstration," in *Proc. IEEE IROS*, 2016.

[19] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," *Proc. ICML*, 2015.

[20] V. Nair and G. E. Hinton, "Rectified linear units improve restricted boltzmann machines," in *Proc. ICML*, 2010.

[21] A. van den Oord, *et al.*, "Pixel recurrent neural networks," in *Proc. ICML*, 2016.

[22] N. Silberman, *et al.*, "Indoor segmentation and support inference from rgbd images," in *Proc. ECCV*, 2012.