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

Efficient Fully Convolution Neural Network for Generating Pixel Wise Robotic Grasps With High Resolution Images

Shengfan Wang†, Xin Jiang*, Jie Zhao, Xiaoman Wang, Weiguo Zhou and Yunhui Liu

***Abstract*— This paper presents an efficient neural network model to generate robotic grasps with high resolution images. The proposed model uses fully convolution neural network to generate robotic grasps for each pixel using 400 × 400 high resolution RGB-D images. It first down-sample the images to get features and then up-sample those features to the original size of the input as well as combines local and global features from different feature maps. Compared to other regression or classification methods for detecting robotic grasps, our method looks more like the segmentation methods which solves the problem through pixel-wise ways. We use Cornell Grasp Dataset to train and evaluate the model and get high accuracy about 94.42% for image-wise and 91.02% for object-wise and fast prediction time about 8ms. We also demonstrate that without training on the multiple objects dataset, our model can directly output robotic grasps candidates for different objects because of the pixel wise implementation.**

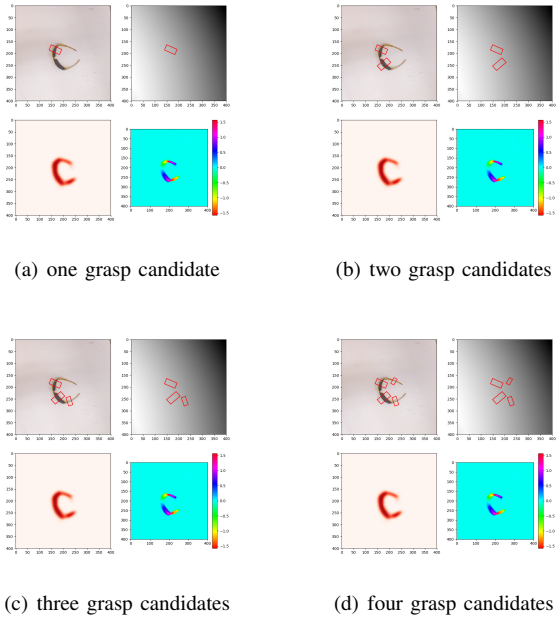
I. Introduction

Researchers have spent large amount of time trying to solve the grasp problem in Robotics. While human beings can easily grasp any object around them in multiple ways, robots still cannot for various reasons related to vision and planning. A robot must know where a object is first and then determine the pose of its gripper to grasp the object. We treat the above problem as a grasp detection problem and try to solve it by using vision, especially through RGB-D cameras.

Previous works treat the grasp detection either as a classification [1] problem or as a regression [2] problem. For classification methods, they usually detect the grasp first by using methods like sliding window to search the potential grasp space [3] and then using neural networks to rank them separately [4], which is time-consuming for the complex procedure. For regression methods, they tend to use neural networks to output the coordinates of the grasps directly [5]. However, for the property of regression, these methods will output the average of the ground truth grasps, which may lead to unreasonable grasps.

Our proposed method tries to solve the problem utilizing some ideas from segmentation tasks [6], [7] and was inspired by the GG-CNN [8]. Instead of evaluating whole grasp candidates to find the best grasps in RGB-D images like classification methods or approximating the 2-D grasp localization and orientation for each object like regression methods, our method predicts pixel-level robotic grasp candidates for objects through one forward propagation. Moreover, recent works [9]–[11] try to output multiple grasp candidates by utilizing skills from object detection like Regions of Interest. To perform traditional detection methods in grasp detection, they have to implement complex procedures. Nevertheless, our method can simply predict multiple grasp candidates without any other difficult steps as shown in Fig. 1. Finally, most previous works used low resolution images like 224 × 224 [2], [5], [11], our model using 400 × 400 images to detect robotic grasps [12].

Our proposed model first down-samples the image to encode features for detecting robotic grasps and then decodes the features to output pixel-level predictions. We train and evaluation our model on Cornell Grasping Dataset and get high accuracy about 94.42% which outperforms the network proposed in GG-CNN [8] by nearly 20%, and run in our personal computer about 8ms for detecting robotic grasps in one image.



(a) one grasp candidate (b) two grasp candidates

(c) three grasp candidates (d) four grasp candidates

Fig. 1. Robotic grasps predicted by our proposed model. Top left is the raw color image. Top right is the raw depth image. Bottom left is the grasp position prediction. Bottom right is the grasp angle prediction.

This work was supported by the following projects: Shenzhen Peacock Plan Team grant (KQTD20140630150243062), Shenzhen and Hong Kong Joint Innovation Project (SGLH20161209145252406), Shenzhen Fundamental Research grant (JCYJ20170811155308088).

Shengfan Wang, Xin Jiang, Jie Zhao, Xiaoman Wang and Weiguo Zhou are with the School of Mechanical Engineering and Automation, Harbin Institute of Technology, Shenzhen 518055, China. The author e-mail: 18S053234@stu.hit.edu.cn. The corresponding author email: x.jiang@ieee.org.

Yunhui Liu is with the Department of Mechanical and Automation Engineering, The Chinese University of Hong Kong, Shatin, Hong Kong, China.

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

II. Related Work

For decades, people have been doing researches on robotic grasps [13]–[17]. Most early works [18], [19] used human-designed features to represent grasps in images or required the full 3-D model of objects to generate grasps [20]–[22]. These methods was popular at that time, but they all faced challenges for non-robust features or lacking of the full 3-D models when used in real world applications.

Recent years, learning methods have been proved effective in robotic grasp generation. More and more researchers are trying to use neural network to extract features from images and use these features to detect grasps to be executed by robots. In Amazon Picking Challenge, the learning method proved to be useful when dealing with robust grasps [23].

A. Classification based methods

Jiang et al. [1] first proposed to use a rectangle representation method to estimate the gripper configuration and the rectangle metric to evaluate a grasp. They carefully designed the grasp features from images and tried to search robotic grasps and then rank them to choose the best one. Deep learning methods were first applied by Lenz et al. [3] to grasp detection. While the process of generating and selecting robotic grasps is similar to the one used in [1], Lenz et al used sparse auto-encoder to directly extract features from images and achieved the accuracy of 75.6%. Both methods in [1] and [3] are time-consuming because of the exhaustive search in images to generate potential grasps and are unlikely to be used to real-time jobs. Mahler et al. [4] proposed the Dex-Net robotic grasp dataset and used it to train a neural network called GQ-CNN to classify potential grasps using analytic grasp metrics. They sampled antipodal grasps from depth images and utilized deep learning to select the one which is most likely to be successful when picked by a robot. Chun et al. [24] recently added the spatial transformer network to the pipeline of grasp detection and evaluated on the Cornell Grasp Dataset with the accuracy of 89.60%. With the spatial transformer network, they were able to provide some partial observation for intermediate grasps. Classification based methods tend to be slow, but the procedure is reasonable.

B. Regression based methods

Redmon et al. [2] first proposed using neural network to directly regress the grasp rectangle parameters from images. By removing the steps of searching potential grasps, the regression method is efficient when compared to the classification method. Their accuracy on the Cornell Grasp Dataset was about 88.0% with the prediction time of 76ms. Zhang et al. [25] spent more time on combing rgb features and depth features for accurate grasp detection. They proposed the multi-modal fusion method to regress robotic grasp configurations from RGB-D images and achieved the accuracy of 88.90% for image-wise split and 88.20% for object-wise split and the computation time of 117ms. Kumra et al. [5] used ResNet as a feature extractor to detect robotic grasps from RGD images by replacing the blue channel of a image by the depth channel, which increases the accuracy to about 89%. However, regression based methods tend to output the mean value of ground truth grasps, which may lead to invalid grasps when used.

C. Detection based methods

Chu et al. [9], [10] recently proposed using some key ideas from object detection fields to help generate robotic grasps from images. Their model incorporated a *grasp region proposal network* to generate candidate regions for later grasp detection and was inspired by Faster R-CNN. With the help of *grasp region proposal network* and ResNet feature extractor, their network can generate multiple robotic grasps from one image without any other procedure with high accuracy. They evaluated on the Cornell Grasp Dataset and reported the accuracy of 96.0% for image-wise split and 96.1% for object-wide split with the prediction time of 120ms. Part et al. [12] proposed use high resolution images with 360 $\times$ 360 and Multi-Grasp inspired by YOLO to generate robotic grasps. Their method also used ResNet as the feature extractor and combined the idea of anchor boxes, with which they got the accuracy of 96.6% for image-wise split and 95.4% for object-wise split with the prediction time of 20ms.

In this paper, we utilize the segmentation method to deal with robust grasp generation, which generally follows Morrison et al [8], and present a simple and efficient network architecture performing well on the Cornell Grasp Dataset. In addition, we also evaluate our method on the Georgia Grasp Dataset to directly generate multiple grasp candidates, which also achieve good results without complex procedures.

III. Problem Description

Given RGB and Depth images of unknown objects, previous works [1], [2], [5], [24] all use different ways to generate antipodal robotic grasps from them. The grasp representation they used was proposed by [1] and then simplified by [3].

$$g = \{x, y, \theta, h, w\} \tag{1}$$

This five dimension rectangle representation includes the center of the rectangle (x, y), the orientation of the rectangle relative to the horizontal axis of the image θ, the height and width of the rectangle (h, w).

Recently, Morrison et al. [8] proposed the *grasp map* presentation of robotic grasps, which is a fully new idea to deal with the 2-D grasp representation and achieved good results when using neural network to predict robotic grasps with RGB-D images. We follow the representation proposed by them and design a new fully convolution neural network which gives improvement. The grasp representation is:

$$g = \{\mathbf{p}, \phi, w, q\} \tag{2}$$

where $\mathbf{p} = (x, y, z)$ is the center position of the gripper, ϕ is the rotation angle relative to the horizontal axis of the image plane, w is the gripper width and q is the grasp quality. The old grasp representation 1 lacks the quality of a grasp, that

is we do not know how good a grasp candidate is. We have to do grasp evaluation if there are multiple grasp candidates. However, with the new representation 2, we can just choose the grasp with highest quality value.

We also assume the 2-D grasp representation can be projected back to 3-D poses executed by robots when we know the camera calibration results.

Robotic grasps can be detected in the depth image $\mathbf{I} = \mathbb{R}^{H \times W}$ with height H and width W. The grasp in image $\mathbf{I}$ is represented by

$$\tilde{g} = \left\{ \mathbf{s}, \tilde{\phi}, \tilde{w}, \tilde{q} \right\} \tag{3}$$

where $\mathbf{s} = (u, v)$ denotes the center point in pixels, $\tilde{\phi}$ denotes the rotation relative to the camera frame, $\tilde{w}$ denotes the gripper width in pixels and $\tilde{q}$ denotes the grasp quality.

The *grasp map* proposed in [8] is

$$\mathbf{G} = \{\Phi, \mathbf{W}, \mathbf{Q}\} \in \mathbb{R}^{3 \times H \times W} \tag{4}$$

where $\Phi, \mathbf{W}, \mathbf{Q}$ are each $\in \mathbb{R}^{1 \times H \times W}$ and each pixel contains the $\tilde{\phi}, \tilde{w}, \tilde{q}$ values respectively.

Like [8], we use neural network to directly generate a grasp $\tilde{\mathbf{g}}$ for each pixel in depth image $\mathbf{I}$, which denotes the pixel-wise grasp representation.

$$M(\mathbf{I}) = \mathbf{G} \tag{5}$$

where the map function M can be approximated by deep neural network and then the best grasp can be found by $\tilde{\mathbf{g}}^* = \max_{\mathbf{Q}} \mathbf{G}$.

IV. Approach

A. Grasp Representation

In order to compare our model to the GG-CNN, we use all the same grasp representation in Section IV of [8].

To build relationships between the rectangle representation of grasps and *grasp maps*, Morrison et al. [8] proposed using the center third of the grasp rectangle as an image mask and then using this mask to set corresponding pixel wise properties of robotic grasps. This process is actually doing segmentation to the grasps. For the fact that the position, width and angle values of one robotic grasp is needed, each ground truth positive rectangle will be converted to three small *grasp maps* in pixels. In each *grasp map*, only the region covered by the mask is taken account of, like shown in Fig. 3. For example, in Cornell Grasp Dataset, the grasp quality values are measured by setting all pixeds inside the regions covered by those positive ground truth grasps to one and the rest pixel to zero.

B. Training and Evaluating dataset

The Cornell Grasp Dataset [3], which shown in Fig. 2, contains 885 RGB-D images of real world objects with thousands of positive and negative grasp rectangles.

In order to compare our proposed model with the one in GG-CNN [8], we argument the dataset using similar methods such as random cropping, zooming and rotating images and

Fig. 2. Images from Cornell Grasp Dataset.

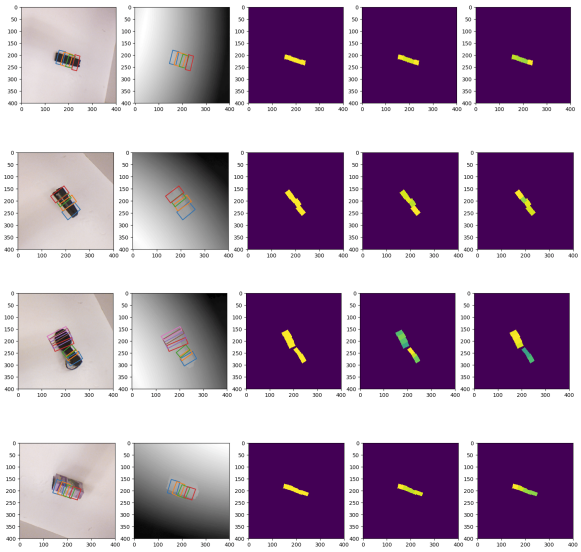


Fig. 3. Examples from the Cornell Grasp Dataset and their *grasp maps*. From left to right: color image, depth image, grasp position map, grasp angle map, grasp width map.

corresponding rectangles. However, we find the zooming range[1] used by Morrison et al. [8] may lead to too small objects to be used to evaluate. As a result, we decrease the zooming range to 0.8 and increase the rotation range to 20 degree in order to increase the diversity of grasps. When generating our dataset, we only use the positive grasps and store each robotic grasp representation separately for later usage.

C. Neural Network Architecture

The fully new model we propose to approximate the function M is shown in Fig. 4.

The pipeline works as one encoder and one decoder. The encoder extracts features for detecting robotic grasps and then the decoder output the pixel-wise grasp parameters. We add Residual Block like connection to the network architecture for the combination of local and global features

[1]https://github.com/dougsm/ggcnn

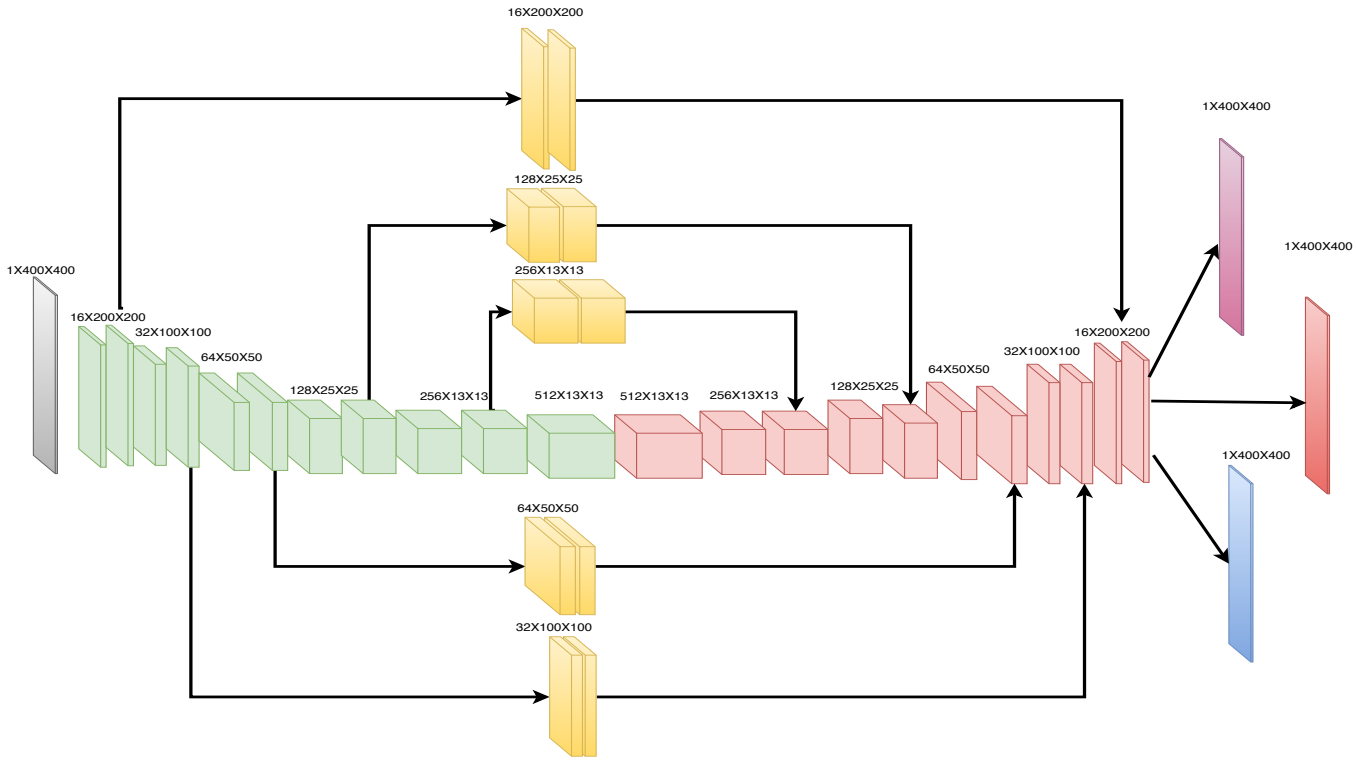


Fig. 4. Proposed fully convolution neural network architecture.

as well as gradient flows. We use relu as the default activation function to all layers and add batch normalization to each convolution and deconvolution layer. For each pair of convolution or deconvolution layer, the kernel size is the same and we use 9-5-3-3-3-3 for down-sampling part and 3-3-3-5-9-5 for up-sampling part. For all the connections, we use the kernel size of 3 by 3. Our network takes one depth image with size of 400 $\times$ 400 as input and through the fully convolution implementation, three 400 $\times$ 400 *grasp maps* will be predicted to generate robotic grasps. This pixel-wise output then can be used to find the best grasp using $\tilde{\mathbf{g}}^* = \max_{\mathbf{Q}} \mathbf{G}$ and the corresponding angle and width value in the other two *grasp maps*. Moreover, the output can also be used to predict multiple grasp candidates directly because of the pixel-wise implementation. The grasp candidate number actually can be set like those pictures shown in Fig. 1, and the only change we need to do is to find the expected number of the local maxima in $\mathbf{G}$. For $\tilde{g} \in \mathbf{G}$, we can also filter some grasps with low $\tilde{q}$ value, which might not be so robotic. This method is simple, efficient and powerful when compared to other classification, regression and detection methods.

V. Experiments

A. Data Pre-processing

For the input to our network is the 400×400 depth image and considering there might be invalid depth values, we do depth in-paint to the image. Before training and evaluating, the grasp width is normalized to be in range [0,1]. All the dimensions are set to the default PyTorch standard [26].

B. Training

Image-wise split and object-wise split are two default training methods on Cornell Grasp Dataset. For both of the training methods, we do five fold cross validation to better evaluate the performance of our model. For each fold of cross validation, we train the neural network from scratch for 100 epochs and use the batch size of 32. The optimizer we use is the Adam and learning rate is set to 0.001. We also use mean square error as the loss function when training. Therefore, the total loss can be calculated by

$$\begin{aligned} L(\hat{q},\hat{\theta},\hat{w}) = \frac{1}{2n}[&\sum_{u=0}^{W}\sum_{v=0}^{H}(\hat{q}_{u,v}-q_{u,v})^2+ \\ &\sum_{u=0}^{W}\sum_{v=0}^{H}(\hat{\phi}_{u,v}-\phi_{u,v})^2+ \\ &\sum_{u=0}^{W}\sum_{v=0}^{H}(\hat{w}_{u,v}-w_{u,v})^2] \end{aligned} \tag{6}$$

where n is the number of training examples; $\hat{q}$, $\hat{\phi}$, $\hat{w}$ are the model predictions, q, ϕ, w are the ground truth labels.

C. Evaluation

The standard rectangle metric is used by us to evaluate our model on Cornell Grasp Dataset. A predicted grasp is considered as a valid grasp if it satisfies both of the two conditions:

1) The grasp angle difference between the predicted grasp and ground truth grasp is less than 30°.

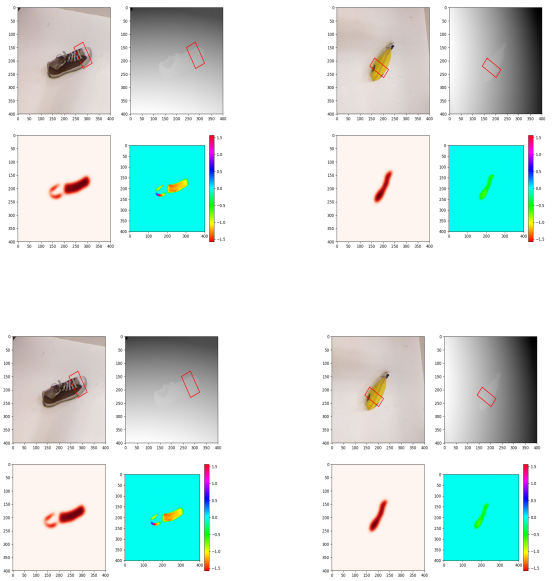


Fig. 5. Some correct predictions on Cornell Grasp Dataset.

2)The Jaccard index calculated by the predicted grasp and ground truth grasp is greater than 0.25, and the Jaccard index is defined as:

$$J(\hat{g}, g) = \frac{|\hat{g} \bigcap g|}{|\hat{g} \bigcup g|} \tag{7}$$

where $\hat{g}$ represents the output prediction of the network and g represents the ground truth label. In fact, the Jaccard index measures how well a prediction matches a ground truth label.

VI. Results

We implement our model and all related code in PyTorch and evaluate the performance of our model on the platform with a single GPU (NVIDIA GeForce GTX 1060), a single CPU (Intel i7-8700K 3.7GHz) and 32 GB memory. After doing five fold cross validation, our model finally get the accuracy about 94.42% for image-wise and 91.02% for object-wise with the prediction time of only 8ms. Fig. 5 shows the image wise training predictions of our model on Cornell Grasp Dataset. For the grasp quality map, the redder the color, the better the quality of a grasp. And for the grasp position map, the different color stands for different grasp angles. We compare our results to others in Table I.

When evaluating our model, we find our model is able to predict the robust grasps even if it does not see the objects in training set but may be judged incorrect for some new grasps not included in the ground truth labels. These false negative grasps are shown in Fig. 6, where the ground truth labels are shown in green and the prediction is shown in red. As a result, the accuracy of our model on Cornell Grasp Dataset might be much higher than the one we report. We also do stricter Jaccard indexes mentioned in [9] in Table II, from

TABLE I
ACCURACY ON CORNELL GRASP DATASET

Model	Input Size	Accuracy		Time
		Image	Object	
Redmon et al. [2]	224 × 224	88.00%	87.10%	76 ms
Zhang et al. [25]	224 × 224	88.90%	88.20%	117 ms
Kumra et al. [5]	224 × 224	89.21%	88.96%	10 ms
Jiang et al. [1]	227 × 227	60.50%	58.30%	-
Lenz et al. [3]	227 × 227	73.90%	75.60%	13.50 sec
Chu et al. [10]	227 × 227	96.00%	96.10%	120 ms
Asif et al. [27]	244 × 244	90.60%	90.20%	24 ms
Morrison et al. [8]	300 × 300	78.56%	-	7 ms
Chun et al. [12]	360 × 360	96.60%	95.40%	20 ms
Chun et al. [24]	400 × 400	89.60%	-	23 ms
*Ours**	400 × 400	94.42%	91.02%	8 ms

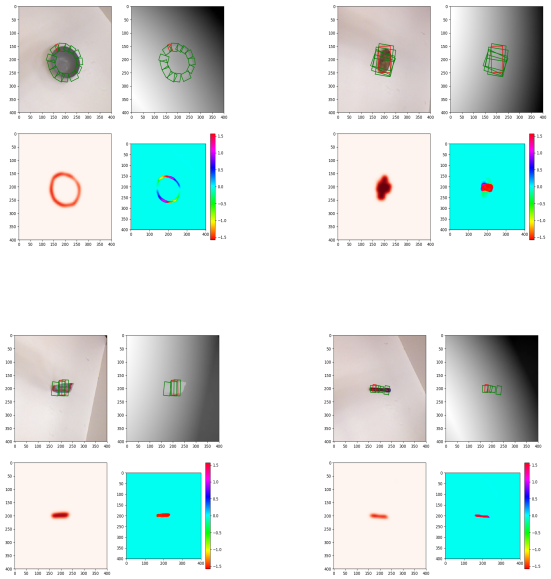


Fig. 6. Some false negative predictions. The green ones are the ground truth labels, the red one is the prediction by our network.

which we can see that our model is also able to achieve high accuracy even if the metric is stricter.

Also we train and evaluate the GG-CNN [8] on the same platform with the Keras implementation and get the accuracy of about 78.56% with the prediction time 7ms when trained in image-wise split.

For the fact that only one object is contained per image in the standard Cornell Grasp Dataset, Chu at el. [9] proposed the Georgia Grasp Dataset which contains multiple objects in an image and all the objects are the same as the ones in Cornell Grasp Dataset to evaluate the model performance on multiple objects. We also would like to evaluate the ability

TABLE II
DIFFERENT JACCARD THRESHOLDS RESULT

Split	0.25	0.30	0.35	0.40
Image-Wise	94.42%	92.83%	90.20%	85.79%
Object-Wise	91.02%	89.15%	83.12%	80.43%

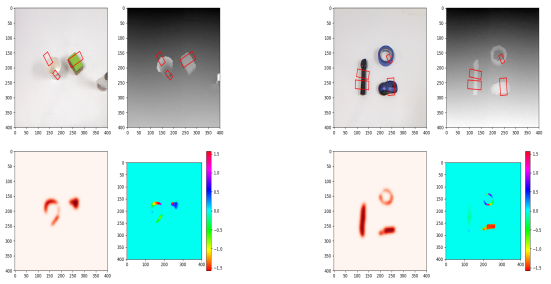


Fig. 7. Predictions on dataset proposed by [9] with multiple objects

TABLE III
DIFFERENT NUMBER OF PREDICTIONS ACCURACY

# of predictions	1	2	3	4	5
Image-Wise	89.52%	87.16%	85.36%	84.79%	83.84%
Object-Wise	90.36%	89.64%	87.60%	83.93%	80.06%

of our model to directly generate multiple grasp candidates in images without any other complex procedures used in the detection methods Fig. 7. When trying to predict multiple robotic grasps, we set the grasp quality threshold to be 0.5 in order to filter some grasps to have more robotic results. For the fact that we can actually control the number of grasp candidates generated in images, we do not calculate the same false positives per image (FPPI)and Miss Rate mentioned in [9]. Instead we use the same metric on Cornell Grasp Dataset and do evaluation to each grasp generated in Georgia Grasp Dataset to calculate the whole accuracy when we pick different number of grasps to be generated. When we trying to evaluate the object-wise trained network on the Georgia Grasp Dataset with multiple grasps, we find that our network is not so confident of its predictions and gives low grasp quality $\tilde{q}$ when compared to the image-wise trained network. If we still set the filter threshold to be 0.5, we cannot find so many maxima in the *grasp quality map*. As a result, when we evaluate the object-wise trained model, we set the threshold to be 0.2 in order to increase the number of predictions. However, there might be more grasps which are not so robotic or even not on the object if we do so. There is a trade-off between the filter threshold and number of predictions to be generated. Our final results is shown in Table III.

VII. Conclusion

In this research, a new fully convolution neural network is presented and it generates robotic grasps by using high resolution depth images. Our propose model encodes the origin input images to features and then decode these features to generate robotic grasp properties for each pixel. It demonstrates well performance on the Cornell Grasp Dataset. Unlike other methods for generating multiple grasp candidates through neural network, the pixel-wise implementation can directly predict multiple grasp candidates through one forward propagation and we can use these pixel-wise results to filter some grasps and even control the number of grasps to be generated. The trained model size is only about 35 MB and the computation time 8ms is fast enough to perform real-time robotic applications with high accuracy.

References

[1] Y. Jiang, S. Moseson, and A. Saxena, "Efficient grasping from rgbd images: Learning using a new rectangle representation," in *2011 IEEE International Conference on Robotics and Automation*, May 2011, pp. 3304–3311.

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

[3] I. Lenz, H. Lee, and A. Saxena, "Deep learning for detecting robotic grasps," *International Journal of Robotics Research*, vol. 34, no. 4-5, pp. 705–724, 2013.

[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," 2017.

[5] S. Kumra and C. Kanan, "Robotic grasp detection using deep convolutional neural networks," 2016.

[6] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 3431–3440.

[7] K. Yang, X. Hu, L. M. Bergasa, E. Romera, and K. Wang, "Pass: Panoramic annular semantic segmentation," *IEEE Transactions on Intelligent Transportation Systems*, 2019.

[8] D. Morrison, P. Corke, and J. Leitner, "Closing the loop for robotic grasping: A real-time, generative grasp synthesis approach," 2018.

[9] F. J. Chu and P. A. Vela, "Deep grasp: Detection and localization of grasps with deep neural networks," 2018.

[10] F. Chu, R. Xu, and P. A. Vela, "Real-world multiobject, multigrasp detection," *IEEE Robotics and Automation Letters*, vol. 3, no. 4, pp. 3355–3362, Oct 2018.

[11] H. Zhang, X. Lan, X. Zhou, and N. Zheng, "Roi-based robotic grasp detection in object overlapping scenes using convolutional neural network," *arXiv preprint arXiv:1808.10313*, 2018.

[12] D. Park, Y. Seo, and S. Y. Chun, "Real-time, highly accurate robotic grasp detection using fully convolutional neural networks with high-resolution images," *arXiv preprint arXiv:1809.05828*, 2018.

[13] A. Bicchi and V. Kumar, "Robotic grasping and contact: A review," in *ICRA*, vol. 348. Citeseer, 2000, p. 353.

[14] K. B. Shimoga, "Robot grasp synthesis algorithms: A survey," *The International Journal of Robotics Research*, vol. 15, no. 3, pp. 230–266, 1996.

[15] 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.

[16] J. Bohg, A. Morales, T. Asfour, and D. Kragic, "Data-driven grasp synthesisa survey," *IEEE Transactions on Robotics*, vol. 30, no. 2, pp. 289–309, 2014.

[17] S. Caldera, A. Rassau, and D. Chai, "Review of deep learning methods in robotic grasp detection," *Multimodal Technologies and Interaction*, vol. 2, no. 3, p. 57, 2018.

[18] A. Saxena, J. Driemeyer, J. Kearns, and A. Y. Ng, "Robotic grasping of novel objects," in *Advances in neural information processing systems*, 2007, pp. 1209–1216.

[19] A. Saxena, J. Driemeyer, and A. Y. Ng, "Robotic grasping of novel objects using vision," *The International Journal of Robotics Research*, vol. 27, no. 2, pp. 157–173, 2008.

[20] A. T. Miller, S. Knoop, H. I. Christensen, and P. K. Allen, "Automatic grasp planning using shape primitives," in *Robotics and Automation, 2003. Proceedings. ICRA'03. IEEE International Conference on*, vol. 2. IEEE, 2003, pp. 1824–1829.

[21] M. Ciocarlie, K. Hsiao, E. G. Jones, S. Chitta, R. B. Rusu, and I. A. Şucan, "Towards reliable grasping and manipulation in household environments," in *Experimental Robotics*. Springer, 2014, pp. 241–252.

[22] A. Herzog, P. Pastor, M. Kalakrishnan, L. Righetti, T. Asfour, and S. Schaal, "Template-based learning of grasp selection." in *ICRA*, 2012, pp. 2379–2384.

[23] A. Zeng, S. Song, K.-T. Yu, E. Donlon, F. R. Hogan, M. Bauza, D. Ma, O. Taylor, M. Liu, E. Romo *et al.*, "Robotic pick-and-place of novel objects in clutter with multi-affordance grasping and cross-domain image matching," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2018, pp. 1–8.

[24] D. Park and S. Y. Chun, "Classification based grasp detection using spatial transformer network," *arXiv preprint arXiv:1803.01356*, 2018.

[25] Q. Zhang, D. Qu, F. Xu, and F. Zou, "Robust robot grasp detection in multimodal fusion," in *MATEC Web of Conferences*, vol. 139. EDP Sciences, 2017, p. 00060.

[26] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer, "Automatic differentiation in pytorch," 2017.

[27] U. Asif, J. Tang, and S. Harrer, "Graspnet: An efficient convolutional neural network for real-time grasp detection for low-powered devices." in *IJCAI*, 2018, pp. 4875–4882.