IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received September 30, 2019, accepted October 26, 2019, date of publication October 30, 2019,
date of current version November 13, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2950535

GraspCNN: Real-Time Grasp Detection Using a New Oriented Diameter Circle Representation

YULIN XU[1], LIANG WANG[1], AOLEI YANG[1], AND LING CHEN[2]
[1]School of Mechatronics Engineering and Automation, Shanghai University, Shanghai 200444, China
[2]School of Engineering and Design, Hunan Normal University, Changsha 410081, China

Corresponding author: Aolei Yang (aolei@shu.edu.cn)

This work was supported in part by the Natural Science Foundation of China under Grant 61873158, and in part by the Natural Science Foundation of Shanghai under Grant 18ZR1415100.

ABSTRACT This paper proposes GraspCNN, an approach to grasp detection where a feasible robotic grasp is detected as an oriented diameter circle in RGB image, using a single convolutional neural network. By detecting robotic grasps as oriented diameter circles, grasp representation is thereby simplified. In addition to our novel grasp representation, a grasp pose localization algorithm is proposed to project an oriented diameter circle back to a 6D grasp pose in point cloud. GraspCNN predicts feasible grasping circles and grasp probabilities directly from RGB image. Experiments show that GraspCNN achieves a 96.5% accuracy on the Cornell Grasping Dataset, outperforming existing one-stage detectors for grasp detection. GraspCNN is fast and stable, which can process RGB image at 50 fps and meet the requirements of real-time applications. To detect objects and locate feasible grasps simultaneously, GraspCNN is executed in parallel with YOLO, which achieves outstanding performance on both object detection and grasp detection.

INDEX TERMS Convolutional neural network, grasp detection, grasp pose, oriented diameter circle.

I. INTRODUCTION

The goal of 2D grasp detection is to localize feasible grasps in the images of objects. A camera observes a cluttered scene and finds feasible robotic grasps in the images, as shown in Fig. 1. More specifically, in this work, we aim to predict feasible robotic grasps directly from RGB image.

RGB image has been widely used for 2D object detection. Convolutional neural networks, such as YOLO [1]–[3], SSD [4], Mask RCNN [5] and CornerNet [6], have achieved great success. Currently, deep learning has also been utilized successfully for robotic grasp detection, which has achieved significant improvements over conventional methods. However, some grasp detection methods are a two-stage cascaded system based on deep learning which detect objects in the first stage and then each cropped object region is sent to a second stage network to predict a feasible robotic grasp for this specified object. These complex pipelines are very slow and hard to optimize.

Visual-based grasping is a very simple action for human beings. However, robotic grasping is still a challenging problem in robotics. Robotic arm needs providing an accurate grasp pose. Besides, there should be enough room to accommodate the open gripper without collision in a grasped area.

In this paper, we propose an oriented diameter circle representation for robotic grasp in image space. A single neural network is used to locate feasible grasps by oriented diameter circles directly on RGB image. Extensive experimentation shows that GraspCNN achieves competitive grasp detection accuracy compared to existing one-stage detectors. GraspCNN is executed in parallel with YOLO to perform object grasping detection, which can detect objects and locate feasible grasps simultaneously in a cluttered scene.

To summarize, our main contributions are as follows:

1) A single end-to-end model for grasp detection is presented, which can process RGB image at 50 fps and meet the requirements of real-time applications.

2) An oriented diameter circle representation is introduced and a new metric is proposed to evaluate the predicted grasp. The grasping circle representation is suitable for all kinds of grippers and able to discriminate between good and bad grasps better.

3) A grasp pose localization algorithm is proposed to project an oriented diameter circle back to a 6D grasp pose, which means that the oriented diameter circle is a reliable representation for robotic grasp.

The associate editor coordinating the review of this manuscript and approving it for publication was Hossein Rahmani.

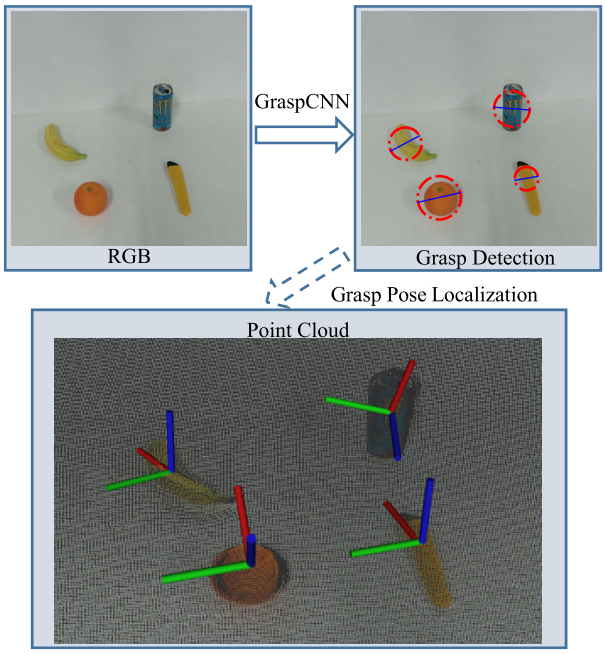


FIGURE 1. GraspCNN directly operates on the raw RGB image and produces the grasp detection results using a single end-to-end trainable network. 6D grasp poses are projected back into point cloud. The y-axis is shown in green and represents the gripper closing direction. The z-axis is shown in blue and represents the grasp approach anti direction.

This paper is organized as follows. Section II contrasts related grasp detection methods, and Section III presents the oriented diameter circle representation and provides a detailed description of computing the grasp pose using the oriented diameter circle representation. Section IV presents the design of GraspCNN, and Section V demonstrates its effectiveness in experiments. Section IV concludes the paper and discusses the future work.

II. METHODS OVERVIEW

Grasp detection considers the grasping task as the problem of predicting the object grasp pose, where the system looks at the scene and chooses the best locations at which to grasp. A typical approach for grasp detection is to use a sliding window to select local image patches, evaluate grasp probabilities, and choose the image patch with the highest grasp probability for grasp. Template-based methods also performed well to detect grasps in ideal scenarios. Currently deep learning has performed state-of-the-art grasp detection results over conventional methods. Real-time robotic grasp detection has achieved remarkable performance improvements.

A. ROBOT GRASPING WITH 3D MODEL OR GRASP TEMPLATES

Most previous work performed stable grasps in ideal scenarios by assuming full knowledge of the object to be grasped. Rosales *et al.* [7] performed optimization of grasps given both a 3D model of the object and the desired contact points for the robot gripper. Pokorny *et al.* [8] defined spaces to discover grasps of graspable objects, then map new objects to these spaces to discover grasps. However, the robot cannot interact with a new environment very well because of the numerous unpredictable objects in our daily life. Other methods follow a template-based approach where grasps that are demonstrated on a set of training objects are generalized to new objects. Herzog *et al.* [9] proposed a template-based grasp selection algorithm operating on depth map which uses demonstrated grasp configurations and generalizes them to grasps for novel objects. Detry *et al.* [10] grasped novel objects by modeling the geometry of local object shapes and fitting these shapes to new objects. Osadchy *et al.* [11] used shape primitives like spheres, cones and boxes to approximate object shape and used the simulation environment GraspIt for grasp stability tests. Template-based methods are useful in detecting textureless objects. However, they cannot handle occlusions between objects very well. Other works use hand-design visual features and hand-code grasping rules, which are difficult to apply in massive objects in real world.

B. ROBOT GRASPING USING DEEP LEARNING

Recent research in robotic grasp has largely focused on performing grasp detection using deep learning. Lenz *et al.* [12] presented a two-stage system for detecting robotic grasps from RGBD data using a deep learning approach. Redmon and Angelova [13] performed real-time grasp detection for the grasping area of an object using convolutional neural networks. Morrison *et al.* [14] proposed a Generative Grasping Convolutional Neural Network, which predicts the quality and pose of grasps at every pixel. However, there should be an optimal grasp in a neighboring region if it exists. This one-to-one mapping from depth image to find the best grasp is not necessary. Wang *et al.* [15] proposed DenseFusion, a generic network framework for estimating 6D pose of a set of known objects from RGBD images. DenseFusion processes the two data sources individually and uses a novel dense fusion network to extract pixel-wise dense feature embedding, from which the pose is estimated. Xiang *et al.* [16] proposed PoseCNN, a convolutional neural network for 6D object pose estimation. PoseCNN is robust to occlusions, can handle symmetric objects and provide accurate pose estimation using only color images as input. Kalashnikov *et al.* [17] proposed QT-Opt, a scalable self-supervised vision-based reinforcement learning framework for robotic manipulation, which enables dynamic closed-loop control. It automatically learns grasping strategies and probes objects to find the most effective grasps using only RGB vision-based perception from an over-the-shoulder camera. Asif *et al.* [18] proposed a novel CNN architecture termed GraspNet which produces pixel-level labeling of grasping regions using RGB-D images. With squeeze and dilated convolutions, GraspNet achieves competitive grasp detection accuracy and real-time inference speed on embedded GPU hardware. Mousavian *et al.* [19] take 3D point clouds as input and formulate the problem of grasp detection as sampling a set of grasps using a variational autoencoder and assess and refine the sampled grasps using

a grasp evaluator model. Park *et al*. [20] proposed fully convolutional neural network termed FCNN, which can be applied to images with any size for detecting multiple grasps on multiple objects. Chu *et al*. [21] proposed two-stage neural networks combining region proposal network and robotic grasp detection network base on Faster R-CNN.

C. ROBOT GRASPING REPRESENTATION

Several representations have been proposed to give an intuitive description about robotic grasp in image space. Saxena *et al.* [22] represented a grasp as a grasping point, using supervised learning algorithms to detect a grasping point from the image. Le *et al.* [23] proposed a new representation based on a pair of points. However, grasping point representation only indicates where to grasp, which is incomplete to perform a stable grasp. Jiang *et al.* [24] represented a grasp as a 2D oriented rectangle in image space, using support vector machines to select a good grasp from extracted features. The oriented rectangle representation works for two-jaw grippers and it can potentially represent grasps for multi-fingered hands as well. Multiple fingers could be represented using different locations within the rectangle.

Most researchers adopt the rectangle-based method, with two edges corresponding to the gripper plates. However, the oriented rectangle representation is not intuitive and concise enough for multiple fingers. And in order to perform one of these grasps, the gripper must approach the grasp target from a direction roughly orthogonal to the image. Considering this, an oriented diameter circle representation is proposed for robotic grasp, which is suitable for all kinds of grippers, such as a parallel plate gripper, a multi-finger gripper, or a robotic hand.

III. PROBLEM DESCRIPTION

A. MOTIVATION OF ORIENTED DIAMETER CIRCLE REPRESENTATION

The oriented diameter circle representation is inspired from grasp action. The robotic arm would approach an object with accurate position and then the gripper would pick it up in a feasible way with appropriate rotation.

The oriented diameter circle is a 4D representation encoding the gripper configuration, which is represented as:

$$G = \{x, y, d, \theta\} \tag{1}$$

with (x, y) denoting the center of the circle, d denoting the oriented diameter of the circle, and θ denoting the angle of the oriented diameter relative to horizontal diameter. Fig. 2 shows an example of this grasp representation. It is a simplification of the full 7D gripper configuration (the 3D position, 3D orientation and the gripper opening width) and can be projected back to the full 7D gripper configuration. The circle center (x, y) can be used to obtain the 3D grasp position from point cloud; the oriented diameter angle θ and surface normal of grasp position are used to obtain the 3D orientation; the gripper opening width can be calculated using the intrinsic parameters of camera and circle diameter d.

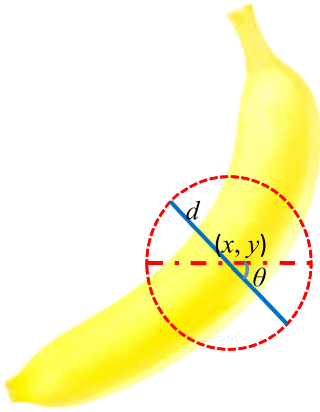


FIGURE 2. A 4D grasp representation based on the oriented diameter circle. A grasp is defined by circle center coordinates (x, y), oriented diameter angle θ relative to the horizontal diameter and circle diameter d corresponding to the gripper opening width before a grasp is performed.

Compared with some previous representations for robotic grasp, the oriented diameter circle representation has two advantages:

First, robotic grasping is presented in a more intuitive way. Grasping circle representation provides the gripper opening width before it closes on an object, which ensures there is enough room to accommodate the open gripper and execute a stable grasp without collision in a grasped area.

Second, the oriented diameter circle representation is suitable for all kinds of grippers without nuisance parameters like the gripper size in the oriented rectangle and it can be associated with grasp behavior of the robotic arm.

B. GRASP EVALUATION METRIC

The oriented diameter circle gives a brief description of robotic grasping in image space. The circle represents a grasped area and the oriented diameter represents the gripper opening width and closing direction. To evaluate the predicted grasp, it is considered to be correct if both:

a. The oriented diameter angle θ is within 30° of the ground-truth grasp.
b. The Intersection Over Union (IOU) of the predicted grasping circle and ground-truth grasping circle is greater than 50 percent.

The area of a grasping circle is $S = \pi * \frac{d^2}{4}$. The IOU reflects how confident the predicted grasping circle represents an appropriate grasped area. The circle metric can discriminate between good and bad grasps better than the rectangle metric since the circle metric weights errors in large grasped areas and small grasped areas unequally. The oriented diameter circle indicates the actual grasp more faithfully.

The oriented diameter circle representation is compared with the oriented rectangle representation in Fig. 3. The ground-truth grasp is shown in red and the predicted grasp is shown in blue. As the Fig. 3 shows, the oriented diameter circle representation can give us a more intuitive impression of differences between the predicted grasp and ground-truth

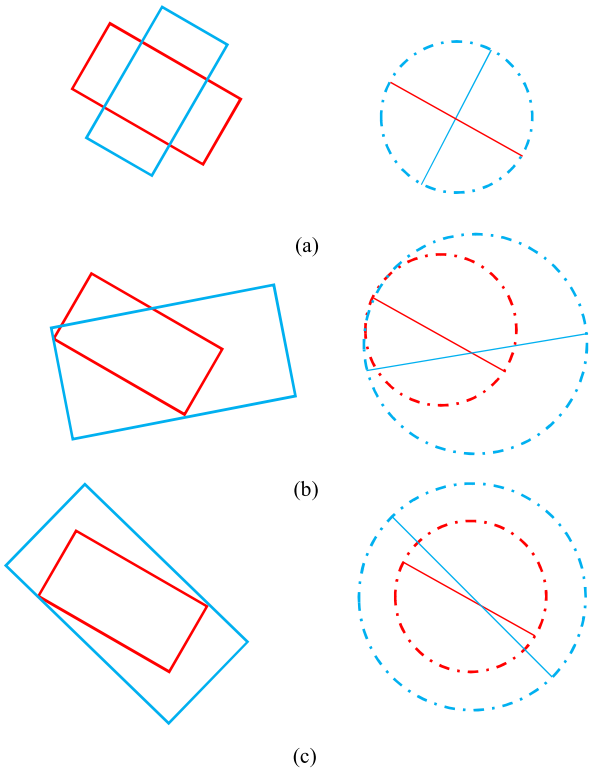


FIGURE 3. Three sets of examples (a), (b), and (c) with grasps represented by the proposed oriented diameter circle and the oriented rectangle. The ground-truth grasp is shown in red and the predicted grasp is shown in blue. The oriented diameter circle representation indicates the actual grasp more faithfully.

grasp. The grasped area is localized perfectly, as shown in Fig. 3(a). The IOU of oriented rectangle cannot indicate that well, while the IOU of oriented diameter circle is 1. Moreover, in contrast to oriented rectangle, it is intuitive to check a good predicted grasp using the oriented diameter circle representation, shown in Fig. 3(b) and Fig. 3(c).

C. GRASP POSE

Recently low-cost RGBD sensors, such as Kinect, have been widely used. With the accessibility of depth image and point cloud, grasp quality and stability have been improved significantly. In this section, a grasp pose localization algorithm is presented to compute the accurate grasp pose using the oriented diameter circle and corresponding point cloud.

Let $P^A = (x_A, y_A, z_A)^T$ denote a random point in coordinate system A. The coordinate system A is rotated around its z-axis counterclockwise through an angle θ, as shown in Fig. 4. After rotation, in new coordinate system B, $P^B = (x_B, y_B, z_B)^T$ describes the same point. Let ${}^B_A R$ denote the rotation matrix from A to B. The transformation is defined as:

$$(x_B, y_B, z_B)^T = {}^B_A R\,(x_A, y_A, z_A)^T \tag{2}$$

The Grasp Reference Frame is defined with its origin at the grasp point P. The unit vector along the z-axis is defined as the surface normal vector of grasp point P. The y-axis

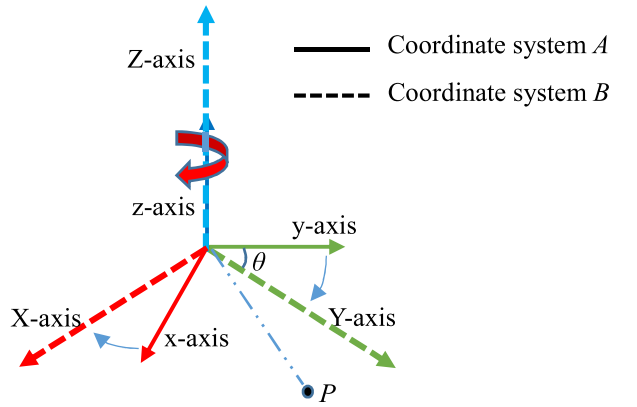


FIGURE 4. Illustration of rotation around z-axis of coordinate system A counterclockwise through an angle θ. P describes the same point in coordinate system A and B.

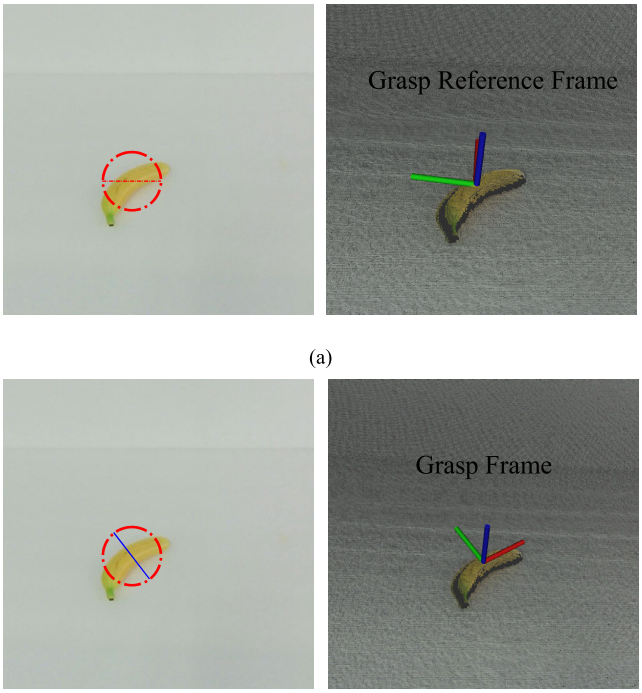


FIGURE 5. Illustration of projection back to a 6D grasp pose in point cloud. The Grasp Reference Frame is constructed using the surface normal of grasp point and horizontal diameter, as shown in (a). Then the Grasp Reference Frame is rotated around its z-axis through an angle θ to get the Grasp Frame, as shown in (b).

parallels the intersection line of the plane y = 0 and the cutting plane of grasp point P in the Camera Frame. Similarly, the Grasp Frame is defined according to the surface normal of grasp point P and the corresponding oriented diameter, as shown in Fig. 5.

Algorithm 1 describes the process of computing the grasp pose using the oriented diameter circle.

Given an oriented diameter circle in image space, the grasp point P can be obtained directly from point cloud using the

Algorithm 1 Grasp Pose Localization

Input: a point cloud; an oriented diameter circle $G = \{x, y, d, \theta\}$

Output: a 6-DOF grasp pose in the Camera Frame

1: Get the grasp point P using (x, y)
2: Set surface normal vector of P as the unit vector along the z-axis of Grasp Reference Frame
3: Set the intersection line vector of cutting plane of P and the plane $y = 0$ as the unit vector along the y-axis of Grasp Reference Frame
4: Get the unit vector along the x-axis of Grasp Reference Frame according to right hand coordinate system
5: Construct the Grasp Reference Frame
6: Rotate around the z-axis of Grasp Reference Frame through an angle θ to construct the Grasp Frame

circle center (x, y). Let ${}^c_gR$ denote the rotation matrix from the Grasp Frame to the Camera Frame, ${}^c_rR$ denote the rotation matrix from the Grasp Reference Frame to the Camera Frame, and ${}^r_gR$ denote the rotation matrix from the Grasp Frame to the Grasp Reference Frame.

The surface normal vector of P termed $(f_x, f_y, f_z)^T$ is calculated in its neighboring region and set as the unit vector along the z-axis of Grasp Reference Frame. The transformation is represented as:

$$(f_x, f_y, f_z)^T = {}^c_rR(0, 0, 1)^T \tag{3}$$

The intersection line between the plane $y = 0$ and cutting plane of P is calculated and expressed as:

$$\begin{cases} f_x x + f_y y + f_z z = 0 \\ y = 0 \end{cases} \tag{4}$$

The intersection line vector $(\frac{f_z}{\sqrt{f_x f_x + f_z f_z}}, 0, \frac{f_x}{\sqrt{f_x f_x + f_z f_z}})^T$ is set as the unit vector along the y-axis of Grasp Reference Frame. The transformation is given by:

$$(\frac{f_z}{\sqrt{f_x f_x + f_z f_z}}, 0, \frac{f_x}{\sqrt{f_x f_x + f_z f_z}})^T = {}^c_rR\,(0, 1, 0)^T \tag{5}$$

The unit vector along the x-axis of Grasp Reference Frame termed $(n_x, n_y, n_z)^T$ is the cross product of the unit vector along y-axis and the unit vector along z-axis. The transformation is calculated as:

$$(n_x, n_y, n_z)^T = {}^c_rR\,(1, 0, 0)^T \tag{6}$$

The above transformations are combined to get the rotation matrix from the Grasp Reference Frame to the Camera Frame. The rotation matrix is expressed as:

$$\begin{pmatrix} n_x & \frac{f_z}{\sqrt{f_x f_x + f_z f_z}} & f_x \\ n_y & 0 & f_y \\ n_z & \frac{f_x}{\sqrt{f_x f_x + f_z f_z}} & f_z \end{pmatrix} = {}^c_rR \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} = {}^c_rR \tag{7}$$

The Grasp Reference Frame is rotated around its z-axis through an angle θ to construct the Grasp Frame. Similarly, the rotation matrix from the Grasp Frame to the Grasp Reference Frame becomes:

$$\begin{pmatrix} cos\theta & -sin\theta & 0 \\ sin\theta & cos\theta & 0 \\ 0 & 0 & 1 \end{pmatrix} = {}^r_gR \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} = {}^r_gR \tag{8}$$

Two subsequent rotations can be combined into a single rotation. Therefore, the final rotation matrix from the Grasp Frame to the Camera Frame can be written as follows:

$${}^c_gR = \begin{pmatrix} n_x & \frac{f_z}{\sqrt{f_x f_x + f_z f_z}} & f_x \\ n_y & 0 & f_y \\ n_z & \frac{f_x}{\sqrt{f_x f_x + f_z f_z}} & f_z \end{pmatrix} \begin{pmatrix} cos\theta & -sin\theta & 0 \\ sin\theta & cos\theta & 0 \\ 0 & 0 & 1 \end{pmatrix} \tag{9}$$

Let ${}^c_gT$ denote the transformation from the Grasp Frame to the Camera Frame. With the rotation matrix and grasp position, the grasp pose in the Camera Frame is constructed using:

$${}^c_gT = \begin{pmatrix} {}^c_gR & P \\ 0^{3\times1} & 1 \end{pmatrix} \tag{10}$$

An oriented diameter circle can be projected back to a 6D grasp pose, which means that the oriented diameter circle is a faithful representation for robotic grasp.

IV. GRASP DETECTION

GraspCNN takes RGB image as input and predicts feasible grasping circles for every object in a cluttered scene. For training and testing, our model runs on an Intel Core i7-4770K CPU and a NVIDIA GTX1080 GPU.

A. ARCHITECTURE

In this section, we present the design of GraspCNN. Fig. 6 shows the architecture of grasp detection network. GraspCNN is implemented as a fully convolutional neural network, which is all made up of 1×1 and 3×3 convolutional layers. It has 5 initial convolutional layers to extract basic features from RGB image followed by 1×1 and 3×3 convolutional layers to do feature fusion and extract advanced features while the final convolutional layer predicts grasping circles with grasp probabilities. The feature extraction module subsamples the input by a factor of 32 to produce 512 basic feature maps of size 13×13. The feature fusion module consists of three convolutional blocks. The first two blocks termed conv6 and conv7 are 1×1, 3×3, and 1×1 convolutions, which are introduced to compute more abstract features for local patches. The 1×1 convolution layer termed conv8 is responsible for reducing feature dimensions and pursing better abstractions for the final prediction layer. The prediction module produces 6 feature maps of size 13×13 separately, where probability is the grasp confidence prediction, (t_x, t_y) is the grasp point offset prediction,

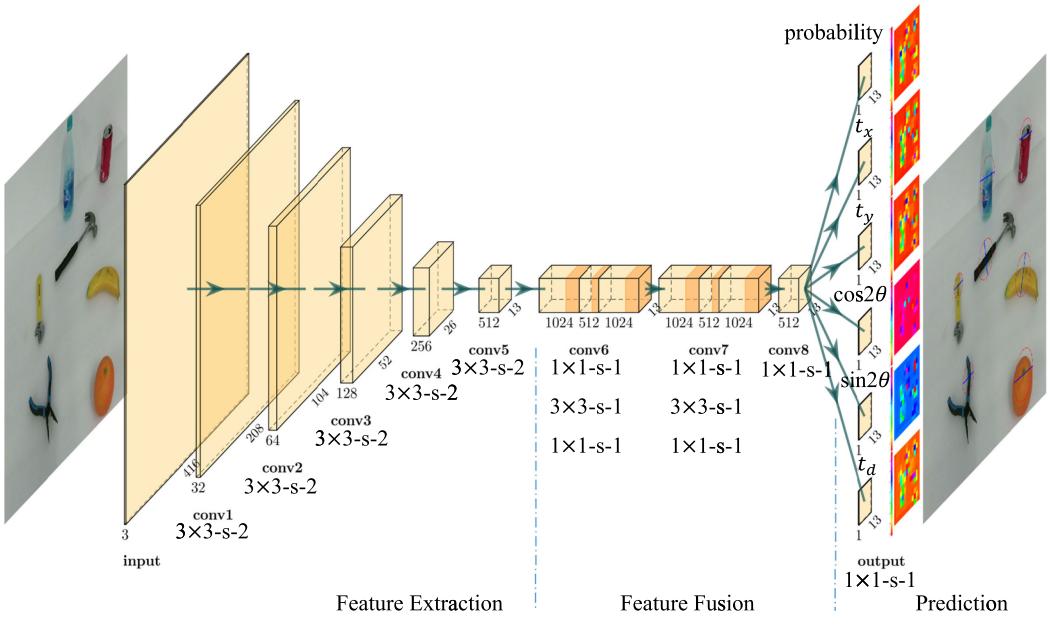


FIGURE 6. Overview of the proposed GraspCNN. Given RGB image, the first five convolutional layers learn basic features of local image regions. Then the next three convolutional blocks extract advanced features. The final layer predicts feasible grasping circles. The convolution layer parameters are denoted as "[kernel size] × [kernel size] –s– [stride]". "s" represents the stride length of the convolution.

$(\sin 2\theta, \cos 2\theta)$ is the grasp angle prediction, and t_d is the grasp width prediction.

GraspCNN is trained and evaluated on the Cornell Grasping Dataset [12]. This dataset is specially designed for the parallel plate gripper and adopts the oriented rectangle representation. Therefore, the positive grasping rectangles need converting to the oriented diameter circles at first. Then extensive data augmentation is performed by translating and rotating RGB image randomly. Finally, a center crop of 416 × 416 pixels is used to fit the input layer.

The input RGB image is divided into a 13 × 13 grid. If the grasping circle center falls into a grid cell, that grid cell is responsible for detecting that grasp. The ground-truth grasp in the grid cell is treated as a mask and the corresponding value is set as 1. The final layer predicts both grasp probabilities and grasping circles. The probability reflects how confident that the grid cell contains a feasible grasp. We compute the diameter in pixels (maximum of 160) of each grasping circle in dataset. Therefore, the circle diameter d is normalized by the scaling factor 160 to put it in the range [0, 1]. Grasp is closely related to local information in a neighboring region. The grasping circle center coordinates (x, y) are parameterized to be offsets of corresponding grid cell. Then the offsets are normalized by the subsample factor 32 so they are also bounded between 0 and 1. The oriented diameter angle θ is in the range $[-\pi/2, \pi/2]$, which is two-fold rotationally symmetric. It is parameterized by using $\sin 2\theta$ and $\cos 2\theta$ to keep values in the range $[-1, 1]$ and remove any discontinuities. GraspCNN predicts $(t_x, t_y, t_d, t_{sin}, t_{cos})$ for each grasping circle and its probability p. If that cell is offset from the top left corner of the image by (c_x, c_y) and the oriented diameter circle has circle center (x, y), diameter d and oriented diameter angle θ, the predictions can be represented as:

$$\begin{cases} x = 32\, t_x + c_x \\ y = 32 t_y + c_y \\ d = 160\, t_d \\ t_{sin} = \sin 2\theta \\ t_{cos} = \cos 2\theta \end{cases} \tag{11}$$

Fig. 7 shows the predictions of GraspCNN in image space. The Cornell Grasping Dataset needs converting into the expected output format of the proposed network, which makes grasp distribution easier for the network to learn.

B. TRAINING

GraspCNN is trained from the beginning without pre-training on the ImageNet classification task. 85% images of Cornell Grasping Dataset are randomly selected as training data and remains are test data. It trains on RGB image at the resolution of 416 × 416 and subsamples the input by a factor of 32 to get output feature map of 13 × 13. The RELU activation function is used for all convolutional layers except the last prediction layer, which is designed to use a linear activation function. We use SGD with momentum of 0.9 to optimize for sum-squared error in the output of our model and use a learning rate of 0.0001 and a weight decay of 0.0001 to train GraspCNN for 5 epochs.

At inference time, we first extract the peaks in the probability feature map for each grid cell independently. We detect

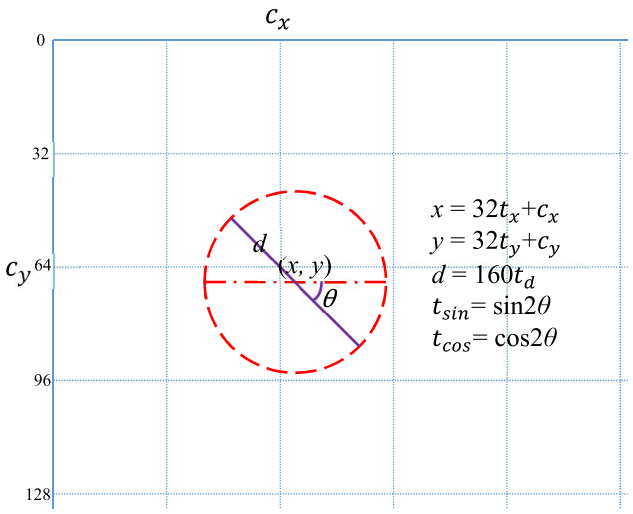


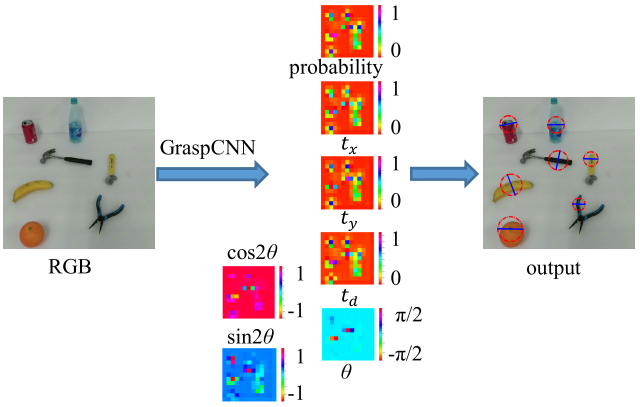
FIGURE 7. Oriented diameter circle predictions. GraspCNN predicts the diameter and center coordinates of the oriented diameter circle. It also predicts sin2θ and cos2θ to remove discontinuities of oriented diameter angle θ.

all grasp probability scores whose value is greater or equal to its 8-connected neighbors and keep the top 20 peaks. The grasp probability score of the top 20 peaks whose value is larger than a certain threshold (e.g., 0.4) will be considered as a feasible grasp detection. GraspCNN predicts (t_x, t_y, t_d, t_{sin}, t_{cos}) and probability score p for each grid cell. If that grid cell is responsible for a feasible grasp detection, the grasping circle can be constructed as shown in Fig. 7. An example is presented to illustrate the accurate predictions of GraspCNN. With the combined network output, optimal grasping circles can be obtained as shown in Fig. 8(a). Fig. 8(b) shows the prediction of probability score feature map. GraspCNN predicts a higher probability score for a graspable region. Fig. 8(c) shows the prediction of diameter map in pixels, which is associated with the probability map. The visualization of network output reflects that GraspCNN is effective to detect grasps from RGB image.

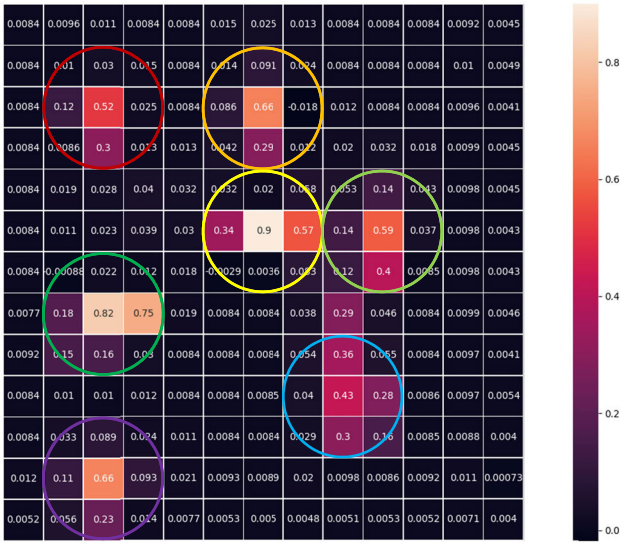
V. RESULTS

A. GRASP DETECTION RESULTS

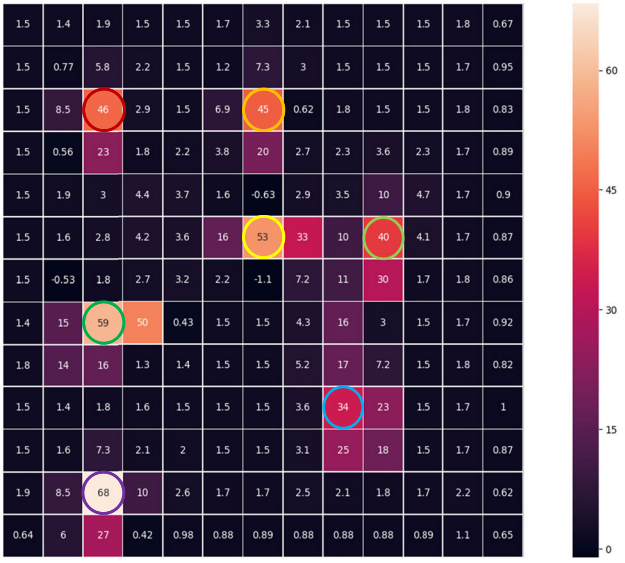
Table 1 shows a comparison of our method with the previous work for grasp detection on the Cornell Grasping Dataset. Fast Search [24], SAE [12], Multiple Grasp [13], GGCNN [14], GraspNet [18], FCNN [20] are existing state-of-the-art grasp detection methods. To evaluate the correctness of our predicted grasping circles, a comparison is performed between the predicted grasping circles and labeled input ground-truth grasping circles. The grasp detection accuracy is evaluated using proposed grasping circle metrics. Across the board, our model outperforms the current state-of-the-art robotic grasp detection algorithms in terms of accuracy. Park *et al*. [20] predict multiple oriented rectangles and confidence scores associated with those oriented rectangles.



(a) Output of GraspCNN



(b) Probability map for each grid cell



(c) Diameter map for each grid cell

FIGURE 8. (a) Illustration of the GraspCNN output. From the combined output, the optimal grasping circles can be obtained. (b) Visualization of the probability map of GraspCNN output. (c) Visualization of the diameter map in pixels of GraspCNN.

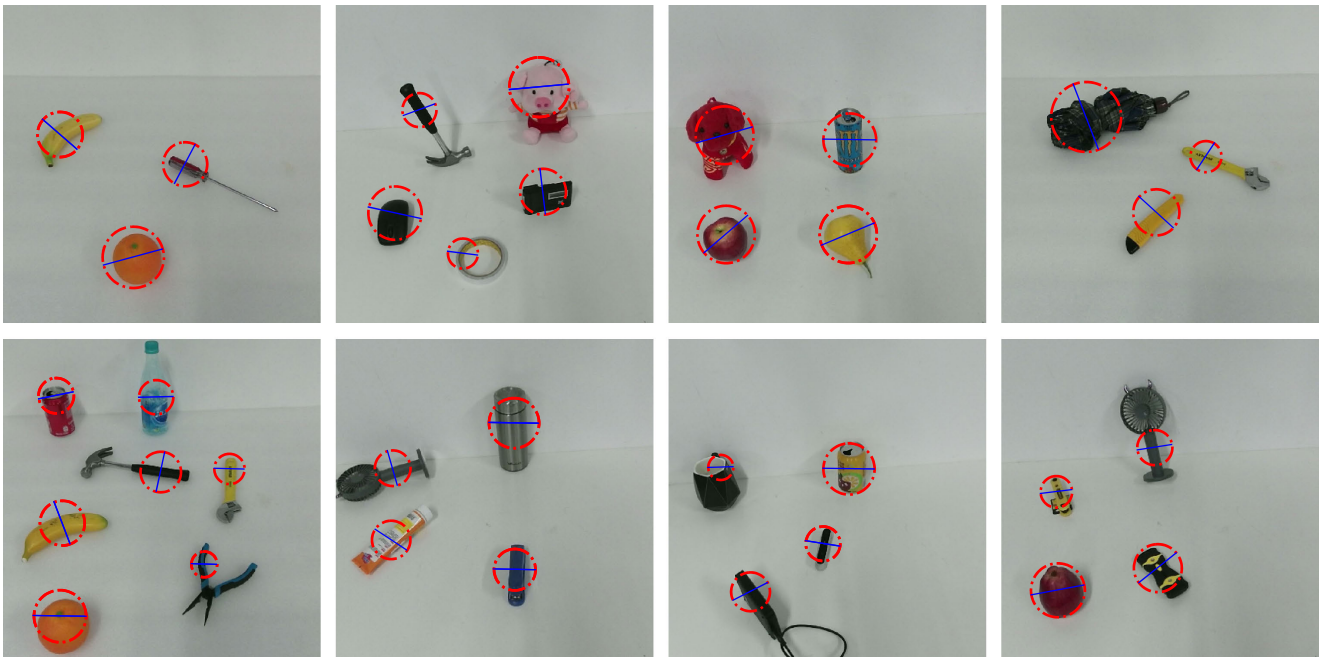


FIGURE 9. Qualitative results of GraspCNN. GraspCNN can detect multiple grasps in a cluttered scene. It performs well to detect grasps for novel objects.

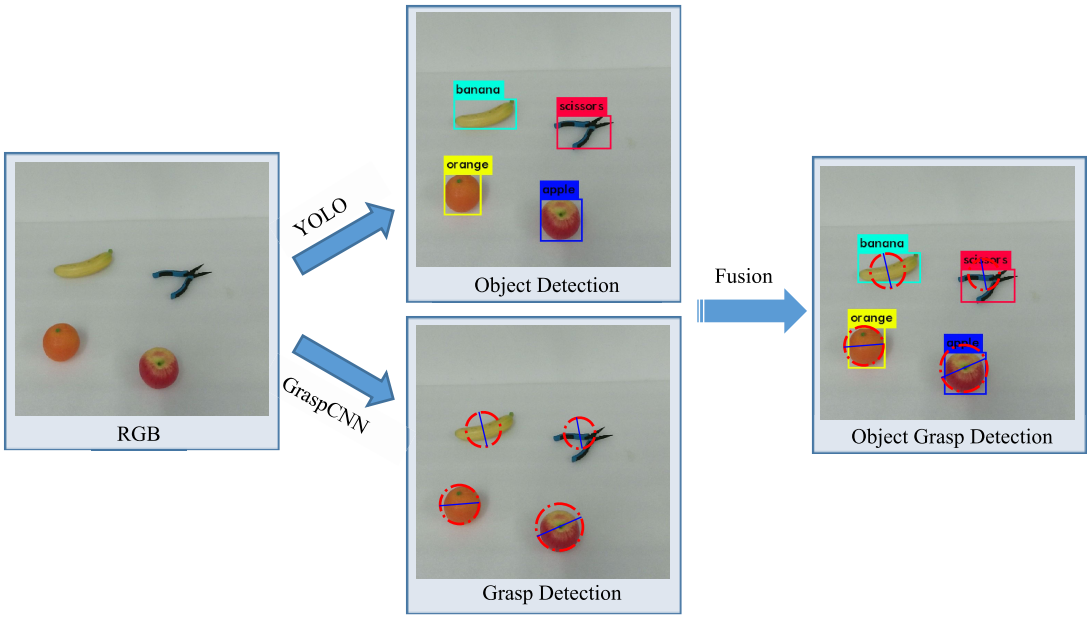


FIGURE 10. Illustration of object grasping detection model. GraspCNN is executed in parallel with YOLO. It can detect objects and locate grasps simultaneously in a cluttered scene.

This approach is most similar to our own, GraspCNN also predicts multiple oriented diameter circles and weights them by a confidence score. The key difference is that GraspCNN maintains a compact design using only the standard convolutions without batch normalization, max pooling and skip connection and only predicts one optimal grasping circle for each cell instead of multiple anchor box candidates. The predictions of GraspCNN are straightforward and closely related to local image regions, which makes it easier to train. After 5 epochs, it performs stable predictions with high accuracy.

Grasp detection is similar to object detection. The goal of 2D object detection is to localize and recognize objects in an image using object bounding boxes, while the goal of 2D grasp detection is just to localize feasible robotic grasps in an image using grasping circles. Compared to object detection, grasp detection is simpler and closely related to local image

TABLE 1. Grasp detection performance on cornell grasping dataset.

Authors	Algorithms	Accuracy	Speed(fps)
Jiang *et al*. [24]	Fast Search	60.5%	-
Lenz *et al*. [12]	SAE	73.9%	0.02
Redmon *et al*. [13]	Multiple Grasp	87.1%	3.31
Douglas *et al*. [14]	GGCNN	90%	50
Asif *et al*. [18]	GraspNet	90.6%	40
Dongwon *et al*. [20]	FCNN	96.1%	50
Chu *et al*. [21]	Based Faster-RCNN	96%	8
Ours	GraspCNN	96.5%	50

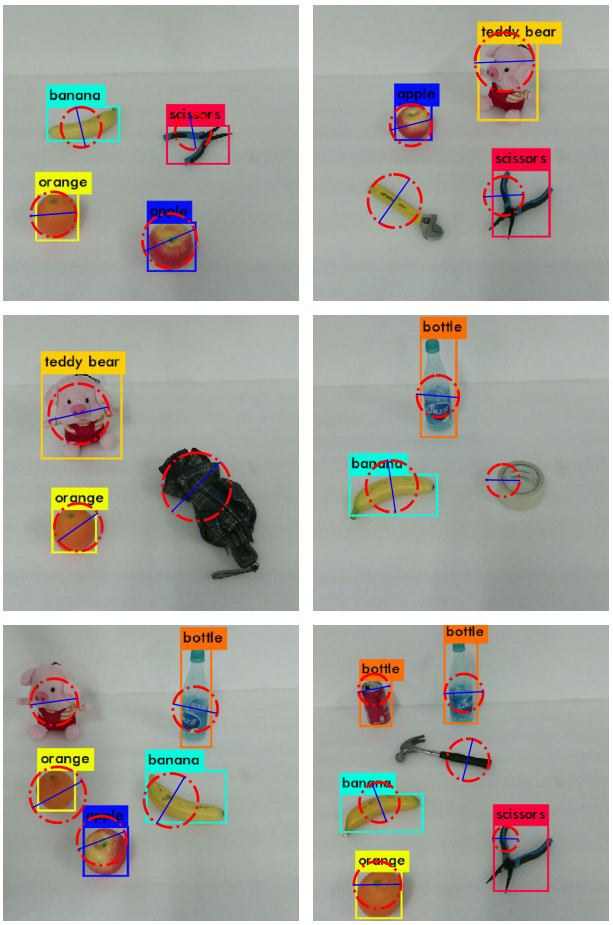


FIGURE 11. Visualization of object grasping detection results. With the combined model GraspCNN + YOLO, it achieves outstanding performance on both object detection and grasp detection.

regions. The output format of GraspCNN is designed to associate with local image regions. It reflects the regular pattern of robotic grasp, which makes GraspCNN more efficient and accurate. The grasp detection results in a real cluttered scene are shown in Fig. 9.

B. OBJECT GRASPING DETECTION RESULTS

In the experiments, KinectV2 publishes RGBD images at a resolution of 960 × 540 pixels with the corresponding point cloud. The RGB image is used as input of YOLO, which utilizes pre-trained weights on COCO dataset to detect objects. Meanwhile, a center crop of 416 × 416 pixels is sent to GraspCNN to locate grasps.

GraspCNN is executed in parallel with YOLO. Fig. 10 shows the framework of object grasping detection model. The oriented diameter circle with the highest probability score in the bounding box of an object is chosen for grasp detection results. The object grasping detection results from real-world testing are shown in Fig. 11.

VI. CONCLUSION AND FUTURE WORK

We present GraspCNN, a single end-to-end network for grasp detection, which is simple to construct and utilizes local region information to predict at least one feasible grasp for every object in a cluttered scene. GraspCNN maintains a compact design using only the standard convolutions and achieves the state-of-the-art performance on Cornell Grasping Dataset. With the combined model GraspCNN + YOLO, it can detect objects and locate grasps simultaneously in clustered environments, while each individual network must be trained separately.

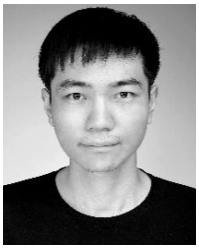
Our future work will focus on using a single unified detection network to perform object grasping detection.

REFERENCES

[1] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2016, pp. 779–788.
[2] J. Redmon and A. Farhadi, "Yolo9000: Better, faster, stronger," in *Proc. IEEE Int. Conf. Comput. Vis. Pattern Recognit.*, Jul. 2017, pp. 7263–7271.
[3] J. Redmon and A. Farhadi, "YOLOv3: An incremental improvement," 2018, *arXiv:1804.02767*. [Online]. Available: https://arxiv.org/abs/1804.02767
[4] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask R-CNN," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Oct. 2017, pp. 2961–2969.
[5] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, "SSD: Single shot multibox detector," in *Proc. Eur. Conf. Comput. Vis.*, Amsterdam, The Netherlands, Oct. 2016, pp. 21–37.
[6] H. Law and J. Deng, "Cornernet: Detecting objects as paired keypoints," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, Jun. 2018, pp. 734–750.
[7] C. Rosales, J. M. Porta, and L. Ros, "Global optimization of robotic grasps," in *Proc. Robot. Sci. Syst.*, Jun. 2011, pp. 289–296.
[8] F. T. Pokorny, K. Hang, and D. Kragic, "Grasp Moduli spaces," in *Proc. Robot. Sci. Syst.*, Jun. 2013.
[9] A. Herzog, P. Pastor, M. Kalakrishnan, L. Righetti, J. Bohg, T. Asfour, and S. Schaal, "Learning of grasp selection based on shape-templates," *Auton. Robots*, vol. 36, nos. 1–2, pp. 51–65, Jan. 2014.
[10] R. Detry E. Baseski, M. Popovic, Y. Touati, N. Kruger, O. Kroemer, J. Peters, and J. Piater, "Learning object-specific grasp affordance densities," in *Proc. IEEE 8th Int. Conf. Develop. Learn.*, Jun. 2009, pp. 1–7,
[11] M. Osadchy, Y. L. Cun, and M. L. Miller, "Synergistic face detection and pose estimation with energy-based models," *J. Mach. Learn. Res.*, vol. 8, pp. 1197–1215, May 2007.
[12] I. Lenz, H. Lee, and A. Saxena, "Deep learning for detecting robotic grasps," *Int. J. Robot. Res.*, vol. 34, nos. 4–5, pp. 705–724, 2015.
[13] J. Redmon and A. Angelova, "Real-time grasp detection using convolutional neural networks," in *Proc. IEEE Int. Conf. Robot. Automat. (ICRA)*, May 2015, pp. 1316–1322.
[14] D. Morrison, P. Corke, and J. Leitner, "Closing the loop for robotic grasping: A real-time, generative grasp synthesis approach," 2018, *arXiv:1804.05172*. [Online]. Available: https://arxiv.org/abs/1804.05172

[15] C. Wang, D. Xu, Y. Zhu, R. Martin-Martin, C. Lu, L. Fei-Fei, and S. Savarese, "DenseFusion: 6D object pose estimation by iterative dense fusion," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 3343–3352.
[16] Y. Xiang, T. Schmidt, V. Narayanan, and D. Fox, "PoseCNN: A convolutional neural network for 6D object pose estimation in cluttered scenes," 2018, *arXiv:1711.00199*. [Online]. Available: https://arxiv.org/abs/1711.00199
[17] D. Kalashnikov, A. Irpan, P. Pastor, J. Ibarz, A. Herzog, E. Jang, D. Quillen, E. Holly, M. Kalakrishnan, V. Vanhoucke, and S. Levine, "Qt-opt: Scalable deep reinforcement learning for vision-based robotic manipulation," in *Proc. CoRL*, Oct. 2018, pp. 1–23. [Online]. Available: https://arxiv.org/abs/1806.10293, 2018.
[18] U. Asif, J. Tang, and S. Harrer, "Graspnet: An efficient convolutional neural network for real-time grasp detection for low-powered devices," in *Proc. IJCAI*, Jul. 2018, pp. 4875–4882.
[19] A. Mousavian, C. Eppner, and D. Fox, "6-DOF GraspNet: Variational grasp generation for object manipulation," 2019, *arXiv:1905.10520*. [Online]. Available: https://arxiv.org/abs/1905.10520
[20] D. Park, Y. Seo, and S. Y. Chun, "Real-time, highly accurate robotic grasp detection using fully convolutional neural networks with high-resolution images," 2018, *arXiv:1809.05828*. [Online]. Available: https://arxiv.org/abs/1809.05828
[21] F.-J. Chu, R. Xu, and P. A. Vela, "Real-world multiobject, multigrasp detection," *IEEE Robot. Autom. Lett.*, vol. 3, no. 4, pp. 3355–3362, Oct. 2018.
[22] A. Saxena, J. Driemeyer, and A. Y. Ng, "Robotic grasping of novel objects using vision," *Int. J. Robot. Res.*, vol. 27, no. 2, pp. 157–173, 2008.
[23] Q. V. Le, D. Kamm, A. F. Kara, A. Y. Ng, "Learning to grasp objects with multiple contact points," in *Proc. IEEE Int. Conf. Robot. Automat.*, May 2010, pp. 5062–5069.
[24] Y. Jiang, S. Moseson, and A. Saxena, "Efficient grasping from RGBD images: Learning using a new rectangle representation," in *Proc. IEEE Int. Conf. Robot. Automat. (ICRA)*, May 2011, pp. 3304–3311.

YULIN XU received the B.Sc. degree in electrical engineering from Donghua University, Shanghai, China, in 1986, the M.Sc. degree in control science and engineering from the Zhengzhou Institute of Technology, Zhengzhou, China, in 1991, and the Ph.D. degree in automation and robotics from the University of Strasbourg, Strasbourg, France, in 2003. She is currently an Associate Professor with the School of Mechatronics Engineering and Automation, Shanghai University, Shanghai. Her research interests include bio-inspired robotics, and robotic grasping and manipulation.



LIANG WANG received the B.Sc. degree in electrical engineering and automation from Shanghai University, Shanghai, China, in 2017, where he is currently pursuing the master's degree in computer vision and robotics. His current research interests include computer vision, and robotic grasping and manipulation.

AOLEI YANG received the B.Sc. degree from the Hubei University of Technology, Wuhan, China, in 2004, the M.Sc. degree from Shanghai University, Shanghai, China, in 2009, and the Ph.D. degree from Queen's University Belfast, Belfast, U.K., in 2012. He is currently an Associate Professor with the School of Mechatronic Engineering and Automation, Shanghai University. His current research interests include robotics, image processing and vision learning, and cooperative control of multiagents.

LING CHEN received the B.Sc. and M.Sc. degrees in control science and engineering from Central South University, Changsha, China, in 2008 and 2010, respectively, and the Ph.D. degree in computing and electronics engineering from the University of Essex, U.K., in 2014. He is currently a Lecturer with the School of Engineering and Design, Hunan Normal University, Changsha. His research interests include simultaneous localization and mapping (SLAM) and robot localization.