A Multi-task Convolutional Neural Network for Autonomous Robotic Grasping in Object Stacking Scenes

Hanbo Zhang, Xuguang Lan, Site Bai, Lipeng Wan, Chenjie Yang, and Nanning Zheng

Abstract— **Autonomous robotic grasping plays an important role in intelligent robotics. However, how to help the robot grasp specific objects in object stacking scenes is still an open problem, because there are two main challenges for autonomous robots: (1)it is a comprehensive task to know what and how to grasp; (2)it is hard to deal with the situations in which the target is hidden or covered by other objects. In this paper, we propose a multi-task convolutional neural network for autonomous robotic grasping, which can help the robot find the target, make the plan for grasping and finally grasp the target step by step in object stacking scenes. We integrate vision-based robotic grasping detection and visual manipulation relationship reasoning in one single deep network and build the autonomous robotic grasping system. Experimental results demonstrate that with our model, Baxter robot can autonomously grasp the target with a success rate of 90.6%, 71.9% and 59.4% in object cluttered scenes, familiar stacking scenes and complex stacking scenes respectively.**

I. INTRODUCTION

In the research of intelligent robotics, autonomous robotic grasping is a very challenging task [1]. For use in daily life scenes, autonomous robotic grasping should satisfy the following conditions:

- Grasping should be robust and efficient.
- The desired object can be grasped in a multi-object scene without potential damages to other objects.
- The correct decision can be made when the target is not visible or covered by other things.

For human beings, grasping can be done naturally with high efficiency even if the target is unseen or grotesque. However, robotic grasping involves many difficult steps including perception, planning and control. Moreover, for complex scenes (*e.g.* the target is occluded or covered by other objects), robots also need certain reasoning ability to grasp the target orderly. For example, as shown in Fig. 1, in order to prevent potential damages to other objects, the robot have to plan the grasping order through reasoning, perform multiple grasps in sequence to complete the task and finally get the target. These difficulties make autonomous robotic grasping more challenging in complex scenes.

Therefore, in this paper, we propose a new vision-based multi-task convolutional neural network (CNN) to solve the mentioned problems for autonomous robotic grasping, which can help the robot complete grasping task in complex scenes

Hanbo Zhang and Xuguang Lan are with the Institute of Artificial Intelligence and Robotics, the National Engineering Laboratory for Visual Information Processing and Applications, School of Electronic and Information Engineering, Xi'an Jiaotong University, No.28 Xianning Road, Xi'an, Shaanxi, China. zhanghanbo163@stu.xjtu.edu.cn, xglan@mail.xjtu.edu.cn
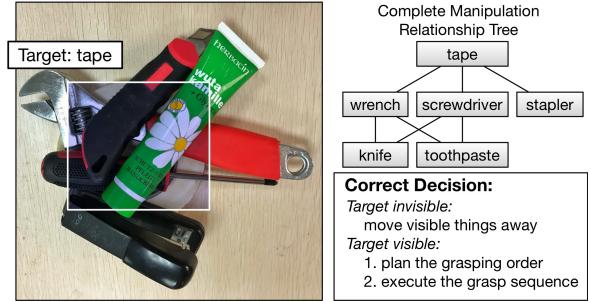
Fig. 1. Grasping task in a complex scene. The target is the tape, which is placed under several things and nearly invisible. Complete manipulation relationship tree indicates the correct grasping order, and grasping in this order will avoid damages to other objects. The correct decision made by the human being should be: if the target is visible, the correct grasping plan should be made and a sequence of grasps should be executed to get the target while if the target is invisible, the visible things should be moved away in a correct order to find the target.

(*e.g.* grasp the occluded or covered target). To achieve this, three functions should be implemented including grasping the desired object in multi-object scenes, reasoning the correct order for grasping and executing grasp sequence to get the target. To help the robot grasp the desired object in multi-object scenes, we design the **Perception** part of our network, which can simultaneously detect objects and their grasp candidates. The grasps are detected in the area of each object instead of the whole scene. In order to deal with situations in which the target is hidden or covered by other objects, we design the **Reasoning** part to get visual manipulation relationships between objects and enable the robot to reason the correct order for grasping, preventing potential damages to other objects. For transferring network outputs to configurations of grasping execution, we design **Grasping** part of our network. For the perception and reasoning process, RGB images are taken as input of the neural network, while for execution of grasping, depth information is needed for approaching vector computation and coordinate transformation.

Though there are some previous works that try to complete grasping in dense clutter [2]–[5], as we know, our proposed algorithm is the first to combine perception, reasoning, and grasp planning simultaneously by using one neural network, and attempts to realize autonomous robotic grasp in complex scenarios. To evaluate our proposed algorithm, we validate the performance of our model in VMRD dataset [6]. For robotic experiments, Baxter robot is used as the executor to complete grasping tasks, in which the robot is required to

find the target, make the plan for grasping and grasp the target step by step.

The rest of this paper is organized as following: Related work is reviewed in Section II; Our proposed algorithm is detailed in Section III; Experimental results including validation on VMRD dataset and robotic experiments are shown in Section IV, and finally the conclusion and discussion are described in Section V.

II. RELATED WORK

A. Robotic Grasp Detection

As the development of deep learning, robotic grasp detection based on convolutional neural network (CNN) achieves state-of-the-art performance on several datasets such as Cornell dataset [7]–[12] and CMU grasp dataset [13]. They are suitable for grasp detection in single-object scenes. There are some works proposed for grasping in dense clutter [2]–[5]. A deep network is used to simultaneously detect the most exposed object and its best grasp by Guo et al. [2], which is trained on a fruit dataset including 352 RGB images. However, their model can only output the grasp affiliated to the most exposed object without perception and understanding of the overall environment and reasoning of the relationship between objects, which limits the use of the algorithm. Algorithms proposed in [3], [4] and [5] only focus on the detection of grasps in scenes where objects are densely cluttered, rather than what the grasped objects are. Therefore, the existing algorithms detect grasps on features of the whole image, and can only be used to grasp an unspecified object instead of a pointed one in stacking scenes.

B. Visual Manipulation Relationship Reasoning

Recent works prove that CNNs achieve advanced performance on visual relationship reasoning [14]–[16]. Different from visual relationship, visual manipulation relationship [6] is proposed to solve the problem of grasping order in object stacking scenes with consideration of the safety and stability of objects. However, when this algorithm is directly combined with the grasp detection network to solve grasping problem in object stacking scenes, there are two main difficulties: 1) it is difficult to correctly match the detected grasps and the detected objects in object stacking scenes; 2) the cascade structure causes a lot of redundant calculations (*e.g.* the extraction of scene features), which makes the speed slow.

Therefore, in this paper, we propose a new CNN architecture to combine object detection, grasp detection and visual manipulation relationship reasoning and build a robotic autonomous grasping system. Different from previous works, the grasps are detected on the object features instead of the whole scene. Visual manipulation relationships are applied to decide which object should be grasped first. The proposed network can help our robot grasp the target following the correct grasping order in complex scenes.
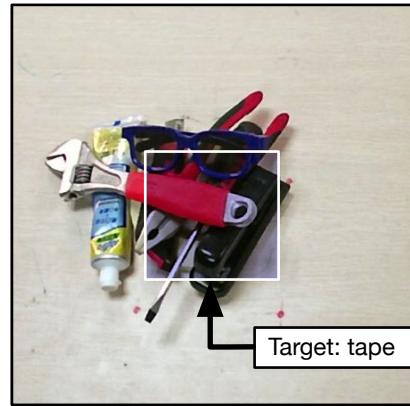

Fig. 2.   Our task is to grasp the target in complex scenes. The target is covered by several other objects and almost invisible. The robot need to find the target, plan the grasping order and execute the grasp sequence to get the target.

III. TASK DEFINITION

In this paper, we focus on grasping task in scenes where the target and several other objects are cluttered or piled up, which means there can be occlusions and overlaps between objects, or the target is hidden under other objects and cannot be observed by the robot. Therefore, we set up an environment with several different objects each time. In each experiment, we test whether the robot can find out and grasp the specific target. The target of the task is input manually. The desired robot behavior is that the final target can be grasped step by step following the correct manipulation relationship predicted by the proposed neural network.

In detail, we focus on grasping task in realistic and challenging scenes as following: each experimental scene includes 6-9 objects, where objects are piled up and there are severe occlusions and overlaps. In the beginning of each experiment, the target is difficult to detect in most cases. Following this setting, it can test whether the robot can make correct decisions to find the target and successfully grasp it.

IV. PROPOSED APPROACH

A. Architecture

The proposed architecture of our approach is shown in Fig. 3. The input of our network is RGB images of working scenes. First, we use CNN (*e.g.* ResNet-101 [17]) to extract image features. As shown in Fig. 3, convolutional features are shared among Region proposal network (RPN, [18]), object detection, grasp detection and visual manipulation relationship reasoning. The shared feature extractor used in our work is ResNet-101 layers before conv4 including 30 ResBlocks. Therefore, the stride of shared features is 16.

RPN follows the feature extractor to output regions of interest (ROI). RPN includes three 3×3 convolutional layers: an intermediate convolutional layer, a ROI regressor and a ROI classifier. The ROI regressor and classifier are both cascaded after the intermediate convolutional layer to output locations of ROIs and the probability of each ROI being a candidate of object bounding boxes.
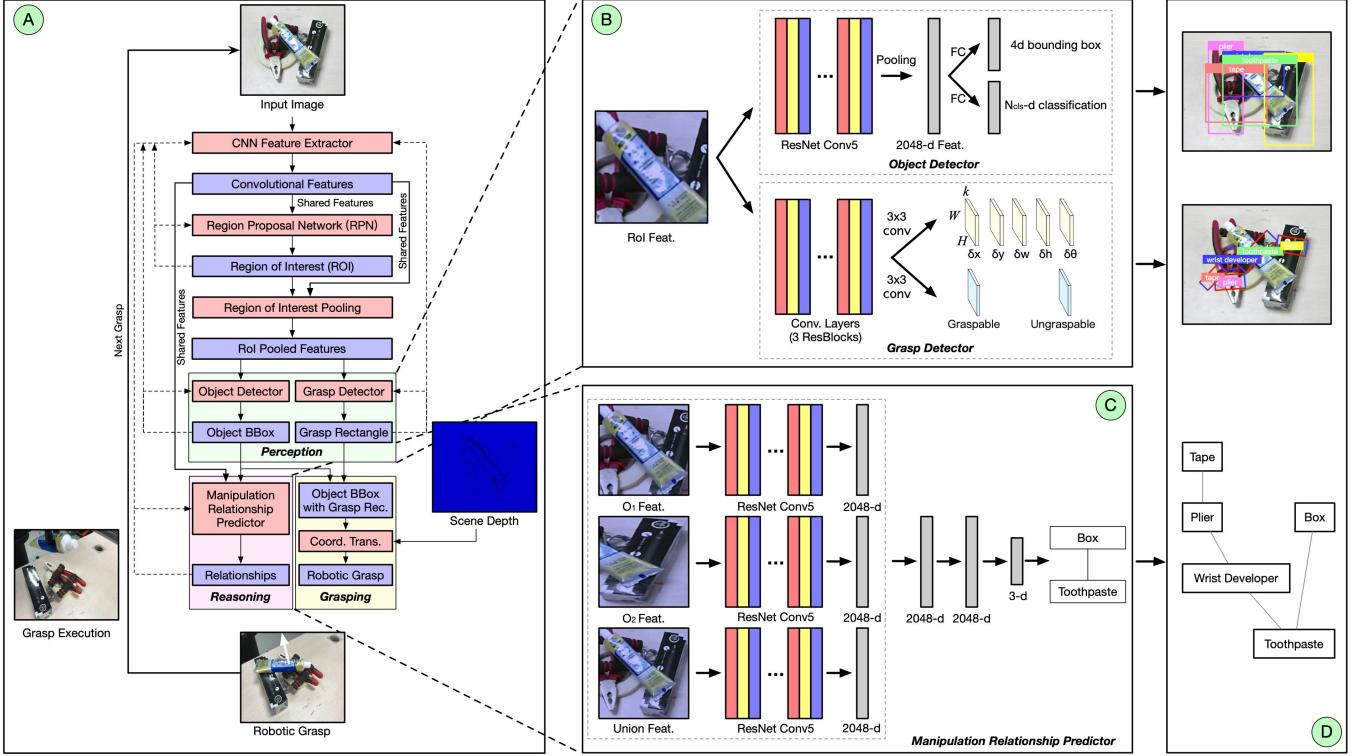
Fig. 3. Architecture of our proposed approach. The input is RGB images of working scenes. The solid arrows indicate forward-propagation while the dotted arrows indicate backward-propagation. In each iteration, the neural network produces one robotic grasp configuration and the robot moves one object. The iteration will not be terminated until the desired target is grasped. (a): Network architecture; (b): Perception part with object detector and grasp detector; (c): Reasoning part with visual manipulation relationship predictor; (d): Expected results.

The mainbody of our approach includes 3 parts: Perception, Reasoning and Grasping. "Perception" is used to obtain the detection results of object and grasp with the affiliation between them. "Reasoning" takes object bounding boxes output by "Perception" and image features as input to predict the manipulation relationship between each pair of objects. "Grasping" uses perception results to transform grasp rectangles into robotic grasp configurations to be executed by the robot. Each detection produces one robotic grasp configuration, and the iteration is terminated when the desired target is grasped.

B. Perception

In "Perception" part, the network simultaneously detects objects and their grasps. The convolutional features and ROIs output by RPN are first fed into ROI pooling layer, where the features are cropped by ROIs and pooled using adaptive pooling into the same size $W \times H$ (in our work, the size is 7×7). The purpose of ROI pooling is to enable the corresponding features of all ROIs to form a batch for network training.

1) Object Detector: Object detector takes a mini-batch of ROI pooled features as input. As in [17], a ResNet' conv5 layer including 9 convolutional layers is adopted as the header for final regression and classification taking ROI pooled features as input. The header's output is then averaged on each feature map. The regressor and classifier are both fully connected layers with 2048-d input and no hidden layer,

outputting locations of refined object bounding boxes and classification results respectively.

2) Grasp Detector: Grasp detector also takes ROI pooled features as input to detect grasps on each ROI. Each ROI is firstly divided into $W \times H$ grid cells. Each grid cell corresponds to one pixel on ROI pooled feature maps. Inspired by our previous work [12], the grasp detector outputs k (in this paper, $k = 4$) grasp candidates on each grid cell with oriented anchor boxes as priors. Different oriented anchor size is explored during experiments including 12×12 and 24×24 pixels. A header including 3 ResBlocks cascades after ROI pooling in order to enlarge receptive field of features used for grasp detection. The reason is that a large receptive field can prevent grasp detector from being confused by grasps that belongs to different ROIs. Then similar to [12], the grasp regressor and grasp classifier follow the grasp header and output $5k$ and $2k$ values for grasp rectangles and graspable confidence scores respectively. Therefore, the output for each grasp candidate is a 7-dimension vector, 5 for the location of the grasp $(\delta x_g, \delta y_g, \delta w_g, \delta h_g, \delta \theta_g)$ and 2 for graspable and ungraspable confidence scores (c_g, c_{ug}).

Therefore, the output of "Perception" part for each object contains two parts: object detection result O and grasp detection result G. O is a 5-dimension vector $(x_{min}, y_{min}, x_{max}, y_{max}, cls)$ representing the location and category of an object and G is a 5-dimension vector $(x_g, y_g, w_g, h_g, \theta_g)$ representing the best grasp.
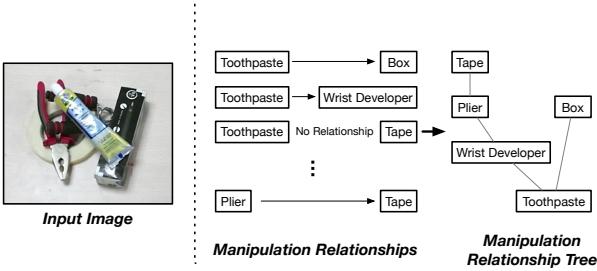
Fig. 4. When all manipulation relationships are obtained, the manipulation relationship tree can be built combining all manipulation relationships. The leaf nodes represent the objects that should be grasped first.

$(x_g, y_g, w_g, h_g, \theta_g)$ is computed by Eq. (1):

$$
\begin{aligned}
x_g &= \delta x_g \times w_a + x_a \\
y_g &= \delta y_g \times h_a + y_a \\
w_g &= exp(\delta w_g) \times w_a \\
h_g &= exp(\delta h_g) \times h_a \\
\theta_g &= \delta \theta_g \times (90/k) + \theta_a
\end{aligned}
\tag{1}
$$

where $(x_a, y_a, w_a, h_a, \theta_a)$ is the corresponding oriented anchor.

C. Reasoning

Inspired by our previous work [6], we combine visual manipulation relationship reasoning in our network to help robot reason the grasping order without potential damages to other objects.

Manipulation Relationship Predictor: To predict manipulation relationships of object pairs, we adopt Object Pairing Pooling Layer (OP^{2}L) to obtain features of object pairs. As shown in Fig. 3.C, the input of manipulation relationship predictor is features of object pairs. The features of each object pair (O_1, O_2) include the features of O_1, O_2 and the union bounding box. Similar to object detector and grasp detector, the features of each object are also adaptively pooled into the same size of $W \times H$. The difference is that the convolutional features are cropped by object bounding boxes instead of ROIs. Note that (O_1, O_2) is different from (O_2, O_1) because manipulation relationship does not conform to the exchange law. If there are n detected objects, the number of object pairs will be $n \times (n - 1)$, and there will be $n \times (n - 1)$ manipulation relationships predicted. In manipulation relationship predictor, the features of the two objects and the union bounding box are first passed through several convolutional layers respectively (in this work, ResNet Conv5 layers are applied), and finally manipulation relationships are classified by a fully connected network containing two 2048-d hidden layers.

After getting all manipulation relationships, we can build a *manipulation relationship tree* to describe the correct grasping order in the whole scene as shown in Fig. 4. Leaf nodes of the manipulation relationship tree should be grasped before the other nodes. Therefore, it is worth noting that the most important part of the manipulation relationship tree is the leaf nodes. In other words, if we can make sure that the

leaf nodes are detected correctly in each step, the grasping order will be correct regardless of the other nodes.

D. Grasping

"Grasping" part is used to complete inference on outputs of the network. In other words, the input of this part is object and grasp detection results, and the output is the corresponding robotic configuration to grasp each object. Note that there is no trainable weights in "Grasping" part.

1) Grasp Selection: As described above, the grasp detector outputs a large set of grasp candidates for each ROI. Therefore, the best grasp candidate should be selected first for each object. According to [11], there are two methods to find the best grasp: (1) choose the grasp with highest graspable score; (2) choose the one closest to the object center in Top-N candidates. The second one is proved to be a better way in [11], which is used to get the grasp of each object in our paper. In our experiments, N is set to 3.

2) Coordinate Transformation: The purpose of the coordinate transformation is to map the detected grasps in the image to the approaching vector and grasp point in the robot coordinate system. In this paper, an affine transformation is used approximately for this mapping. The affine transformation is obtained through four reference points with their coordinates in the image and robot coordinate system. The grasp point is defined as the point in grasp rectangle with minimum depth while the approaching vector is the average surface normal around the grasp point. The grasp point and approaching vector will be mapped into robot coordinate system for location of the robot gripper in grasp execution.

So far, the robot knows which object should be firstly grasped by "Reasoning", where to grasp by "Perception" and how to grasp by "Grasping". By following these steps, objects will be grasped one by one until the target is obtained.

E. Training

Our networks are trained end-to-end with one multi-task loss function. The loss function includes three parts: object detection loss L_O, grasp detection loss L_G and visual manipulation relationship reasoning loss L_R, where L_O is same as [18] and L_R is a multi-class Negative Log-Likelihood classification loss as shown in Eq. (2):

$$
L_R = - \sum_{(O_i, O_j)} log(p_r^{(O_i, O_j)})
\tag{2}
$$

where $r \in \{0, 1, 2\}$ is the ground truth relationship between O_i and O_j and $p_r^{(O_i, O_j)}$ is the predicted probability that the object pair (O_i, O_j) has the relationship r.

L_G is designed to simultaneously minimize the grasp rectangle regression loss and classification loss. As described above, each oriented anchor $(x_a, y_a, w_a, h_a, \theta_a)$ will be corresponding to 5-dimension offsets $(\delta x_g, \delta y_g, \delta w_g, \delta h_g, \delta \theta_g)$ and 2-dimension of confidence scores (c_g, c_{ug}). Therefore, grasp detection loss is defined as following:
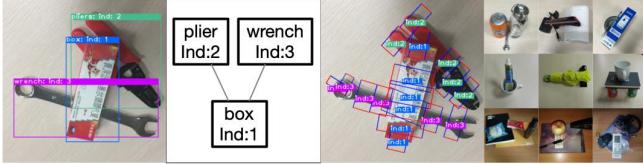
$$
L_G = L_{Greg} + \alpha L_{Gcls}
\tag{3}
$$

Fig. 5. Examples of VMRD dataset with grasps. Each grasp is labeled with an index that indicates the owner of the grasp.

with

$$L_{Greg} = \sum_{i \in Positive} \sum_{m \in \{x,y,w,h,\theta\}} smoothL1(\delta m_g^{(i)} - \delta m_{gt}^{(i)}) \quad (4)$$

$$L_{Gcls} = -\sum_{i \in Positive}^{P} log(c_g^{(i)}) - \sum_{i \in Negative}^{3P} log(c_{ug}^{(i)}) \quad (5)$$

where $\delta m_{gt}, m \in \{x, y, w, h, \theta\}$ represents the ground truth offset and P is the number of oriented anchors that match at least one ground truth. If an oriented anchor is not matched to any ground truth, it will be treated as a negative sample. We select the top-$3P$ negative samples for classification training. According to experience, α is 1 in this paper. The whole loss for our network is defined as follow:

$$loss = L_O + \lambda L_G + \beta L_R \quad (6)$$

where λ and α are all set to 1 in our experiments.

V. EXPERIMENT

A. Dataset

Dataset used to train our models is VMRD dataset with grasps [6]. There are 4683 images with grasps in total, which are divided into training set and testing set with 4233 and 450 images respectively. With consideration of the affiliation between objects and grasps, we define the object to which certain grasps are affiliated as their owner. In VMRD, more than 100k grasps are labeled on these images with object indices that indicate the owner of these grasps. In each image, there are 2-5 objects stacked together with overlaps and occlusions. Manipulation relationships are also labeled as manipulation relationship tree, where the leaf nodes should be grasped before the others. One example is shown in Fig. 5.

During training, we take advantages of online data augmentation including random brightness, random hue, random contrast, color space conversion, vertical rotation and horizontal flip, That is to say, in each iteration, images feed into network are different from all the previous, which can prevent overfitting and enhance the generalization ability of our model in different workspace. Note that the image preprocessing does not change the image size including reshape and crop. Before the images are fed into the network, we approximately subtract 144 on each pixel to mean-center all the inputs.

B. Training Details

Our networks are trained end-to-end on GTX 1080Ti with 11GB memory using PyTorch as the deep learning platform. Learning rate is set to 0.001 according to experience, which is divided by 10 every 20000 iterations. Limited by the memory of GPU, size of mini-batch in this paper is set to 2. We use SGD as the optimizer and set momentum to 0.9. The other settings are same with [18].

C. Metrics

Note that even though our multi-task network is trained and tested end-to-end, we evaluate the performance of each task separately. Therefore, to evaluate our proposed network, metrics of our experiments also include 3 parts: 1) Perception; 2) Reasoning; 3) Grasping.

Perception metrics are used to evaluate the performance of the perception output, including object and grasp detection. In this part, we combine the results of object and grasp detection to see how well our network performs on the test set of VMRD. Similarly, reasoning metrics are used to test the performance of the reasoning output on the test set of VMRD. This part will take our previous work [6] as the baseline. Grasping metrics are used to evaluate how well the proposed network performs in real-world experiments.

D. Result

1) Perception: Perception results are shown in Table I, considering detection results of object and grasp simultaneously. An object is thought as a true positive detection when the bounding box O and the best grasp G are all correctly detected. In detail, the detection (O, G) should satisfy the following conditions:

- The bounding box O should have an IoU larger than 0.5 with the ground truth and be classified correctly
- The best grasp G should has a Jaccard Index larger than 0.25 and angle difference less than 30° with at least one ground truth grasp

Formally, a detection (O, G) is a 10-d vector:

$$(x_{min}, y_{min}, x_{max}, y_{max}, cls, x_g, y_g, w_g, h_g, \theta_g) \quad (7)$$

which includes 5 more values $(x_g, y_g, w_g, h_g, \theta_g)$ than only object detection, indicating the best grasp of the corresponding object. Therefore, we can use the same measurement *mAP* in object detection to evaluate the performance of our model, with consideration of whether the best grasp is correct or not. In Table I, *mAP with grasp* combines object

TABLE I

VALIDATION RESULTS OF PERCEPTION

Setting	mAP with grasp (%)	Speed (FPS)
12 × 12 Anchor, k=4	68.0	7.6
24 × 24 Anchor, k=4	65.0	
12 × 12 Anchor, k=4, Hi-res	69.1	6.5
24 × 24 Anchor, k=4, Hi-res	**70.5**	

TABLE II

VALIDATION RESULTS OF REASONING

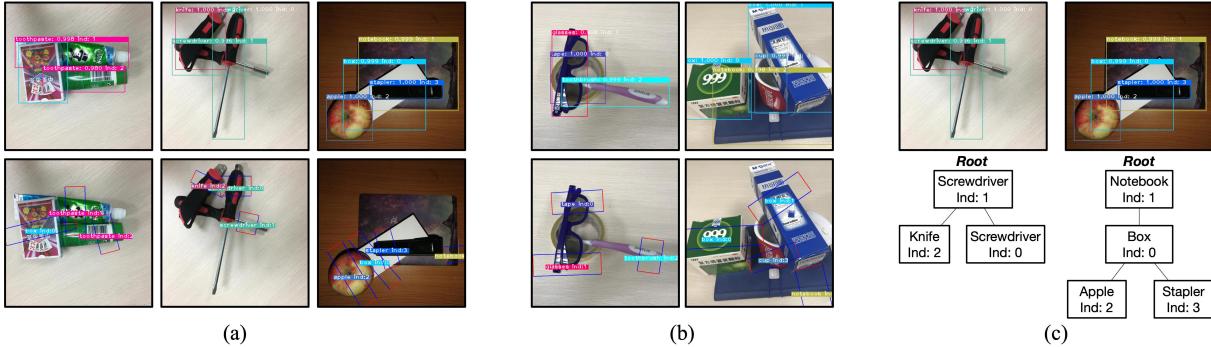| Algorithm	Setting	Obj. Rec.	Obj. Prec.	Image Acc.					
				Total	Object Number per Image				
				(%)	2	3	4	5	
VMRN (baseline) [6]	-	82.3	78.0	63.1	-	-	-	-	
Ours	12×12 Anchor, k=4	85.7	87.2	66.4	61/65	130/209	55/106	53/70	
	24×24 Anchor, k=4	**86.0**	**88.8**	**67.1**	57/65	134/209	60/106	51/70	
	12×12 Anchor, k=4, Hi-res	85.9	85.8	65.3	61/65	132/209	51/106	49/70	
	24×24 Anchor, k=4, Hi-res	84.7	86.5	65.1	60/65	133/209	55/106	45/70	

Fig. 6. Selected results of "Perception" and "Reasoning". (a) True positive examples for detection of object and grasp. (b) Images with incorrect detections. In (a) and (b), the top row is object detection results and the second row is grasp detection results. (c) Examples of manipulation relationship reasoning.

detection O and grasp detection G together to measure the performance of the perception. *Hi-res* means the input image of the neural network is high-resolution (from 600 pixels to 800 pixels in our experiments).

Some examples are shown in Fig. 6. From the true positive examples in Fig. 6(a), we can see that our model can successfully detect objects with their grasps. Failure cases are shown in Fig. 6(b). We can see that the excessive overlap between objects will make it difficult for the model to find the proper grasp for each object. Besides, occlusions and overlaps also make trouble for object detection.

2) Reasoning: Reasoning results are shown in Table II and Fig. 6(c). Following [6], *Obj. Rec.* and *Obj. Prec.* are the recall and precision when the object pair is considered as a sample: it is treated as a positive only when two objects are detected and the relationship between them is reasoned correctly. *Image Acc.* is the image accuracy, in which the image is thought to be a positive only when all objects in it are detected and all relationships are reasoned correctly.

We can see that our model improves the performance of visual manipulation relationship reasoning compared with our previous work [6]. We assume that the improvements are achieved following these changes: (1) Different from our previous work, in this paper, we use ResNet-101 as the feature extractor, or called "backbone" instead of ResNet-50 and VGG-16 in [6]; (2) the object detector is Faster-RCNN from [18] instead of SSD in [19]; (3) the backbone is updated using multi-task loss function including grasp detection loss.

Two examples in Fig. 6(c) demonstrate the output of visual manipulation relationship reasoning. We can see that our model can efficiently build the manipulation relationship tree
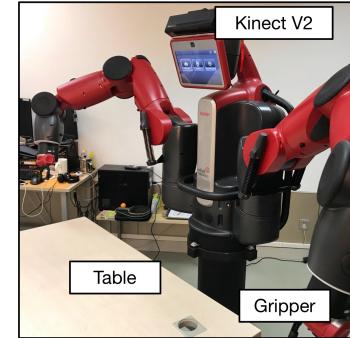

Fig. 7. Robot environment

TABLE III

ROBOTIC EXPERIMENTAL RESULTS

Scene Setting	Success Rate of Each Iteration	
	Baseline	Ours
Object Cluttered Scenes	43.8% (14/32)	**90.6% (29/32)**
Familiar Stacking Scenes	28.1% (9/32)	**71.9% (23/32)**
Complex Stacking Scenes	6.3% (2/32)	**59.4% (19/32)**

for grasping task in object stacking scenes. Note that for scenes where there are more than 1 objects belonging to the same category (like shown in the first example of Fig. 6(c), there are two screwdrivers), our model can also work well by giving each object a unique ID (called "index" in VMRD) to distinguish them.

3) Grasping: In this part, robotic grasping experimental results are explored using the model with 12×12 anchors and Hi-res inputs. Our robotic experiments are conducted
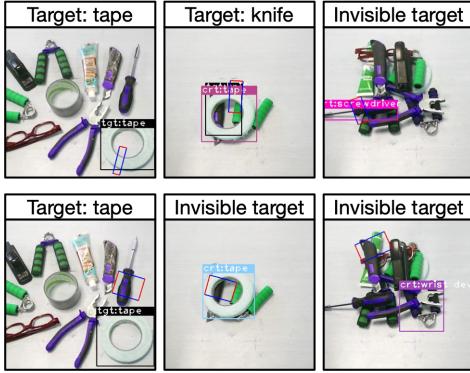
Fig. 8. Comparison between the baseline and our proposed network. The top row is the results of our network and the second row is the results output by the baseline. Final target is abbreviated by "tgt" and the immediate object to be grasped is abbreviated by "crt" (current).

with targets being specified in three types of scenes:

- *Object Cluttered Scenes* where objects are scattered and the target should be directly grasped by the robot.
- *Familiar Stacking Scenes* where there are 2-5 objects stacked like in VMRD dataset. The object number in each scene is chosen uniformly from $\{2, 3, 4, 5\}$.
- *Complex Stacking Scenes* where there are 6-9 objects stacked together. The object number in each scene is chosen uniformly from $\{6, 7, 8, 9\}$.

Among them, the most challenging task is grasping in *Complex Stacking Scenes*. A successful experiment for accomplishing the task is defined as: following the correct grasping order, the specific target is grasped successfully.

We use Baxter robot as the executor and Kinect v2 as the camera. For the specific usage of Kinect v2, RGB images are applied in perception and reasoning, while depth information is added for computing the grasp point and approaching vector in robot coordinate system. Robotic environment is shown in Fig. 7.

Experimental results are shown in Table III. To demonstrate the advantages of our multi-task network, we cascade state-of-the-art model of VMRN in [6] and fully convolutional grasp detection network (FCGN) in [12] as the baseline. Scenes used to test the baseline and our proposed multi-task network are designed as the same. From the table, we can see that our model works well under all three conditions and achieves a success rate of 90.6%, 71.9% and 59.4% respectively, outperforming the baseline by 46.8%, 43.8% and 53.1%. The success rate will decrease with the growth of scene complexity. Some comparisons between the baseline and our proposed network are shown in Fig. 8. Failures of the baseline are mainly caused by: 1) FCGN is designed for grasp detection in single-object scenes and trained on Cornell Grasp Dataset, hence the generalization ability is not satisfactory in multi-object scenes and it cannot detect grasps for all objects; 2) the affiliations between grasps and their owner are likely to be incorrect.

Selected examples of our robotic experiments are shown in Fig. 9 and some failures are demonstrated in Fig. 10. In

(a) Grasping in Object Cluttered Scenes
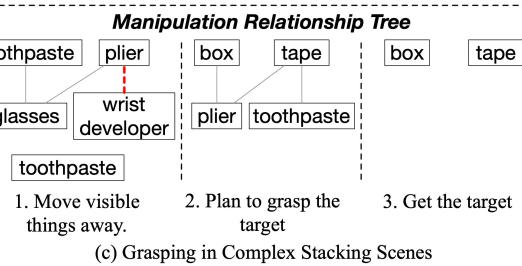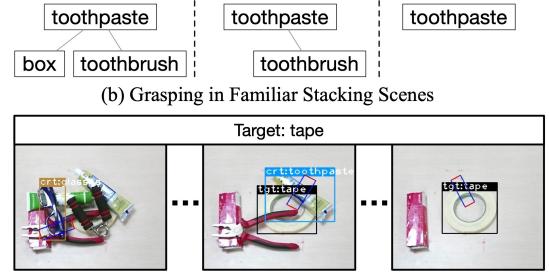

(b) Grasping in Familiar Stacking Scenes

(c) Grasping in Complex Stacking Scenes

Fig. 9. Examples of successful grasping under three conditions. Final target is abbreviated by "tgt" and the immediate object to be grasped is abbreviated by "crt" (current). The red dotted line represents a detection error: the relationship between the wrist developer and the plier is not detected correctly.
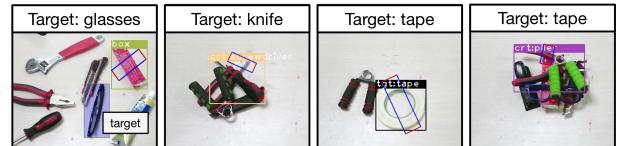

Fig. 10. Failures during robot experiment.

the first picture, target detection failure is caused by the low confidence score of the glasses. As described in the previous section, if the robot cannot see the target, it will move visible things away to find it. Therefore, the box is selected. The second picture demonstrates incorrect grasp detection, which is caused by the confusion of grasps belonging to two different objects due to excessive object overlap. The third picture shows an oversized grasp for our robot. The incorrect order is come across occasionally like shown in the last picture.

VI. Conclusions

In this paper, we propose a multi-task deep network for autonomous robotic grasping in complex scenes, which can help robot find the target, make the plan for grasping and finally grasp the target step by step in object stacking scenes. Experiments demonstrate that with our model, Baxter robot can autonomously grasp the target with a success rate of 90.6%, 71.9% and 59.4% in object cluttered scenes, familiar stacking scenes and complex stacking scenes respectively. In future work, we will try to overcome object and grasp detection difficulty in excessive overlap scenes for better performance of our model.

ACKNOWLEDGMENT

REFERENCES

[1] Jeannette Bohg, Antonio Morales, Tamim Asfour, and Danica Kragic. Data-driven grasp synthesis—a survey. *IEEE Transactions on Robotics*, 30(2):289–309, 2014.

[2] Di Guo, Tao Kong, Fuchun Sun, and Huaping Liu. Object discovery and grasp detection with a shared convolutional neural network. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 2038–2043. IEEE, 2016.

[3] Jeffrey Mahler, Matthew Matl, Xinyu Liu, Albert Li, David Gealy, and Ken Goldberg. Dex-net 3.0: Computing robust robot suction grasp targets in point clouds using a new analytic model and deep learning. *arXiv preprint arXiv:1709.06670*, 2017.

[4] Sergey Levine, Peter Pastor, Alex Krizhevsky, Julian Ibarz, and Deirdre Quillen. Learning hand-eye coordination for robotic grasping with deep learning and large-scale data collection. *The International Journal of Robotics Research (IJRR)*, 2016.

[5] Marcus Gualtieri, Andreas ten Pas, Kate Saenko, and Robert Platt. High precision grasp pose detection in dense clutter. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 598–605. IEEE, 2016.

[6] Hanbo Zhang, Xuguang Lan, Xinwen Zhou, Zhiqiang Tian, and Nanning Zheng. Visual manipulation relationship network for autonomous robotics. In *IEEE International Conference on Humanoid Robots*, pages 118–125. IEEE, 2018.

[7] Ian Lenz, Honglak Lee, and Ashutosh Saxena. Deep learning for detecting robotic grasps. *The International Journal of Robotics Research (IJRR)*, 34(4-5):705–724, 2015.

[8] Joseph Redmon and Anelia Angelova. Real-time grasp detection using convolutional neural networks. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 1316–1322. IEEE, 2015.

[9] Sulabh Kumra and Christopher Kanan. Robotic grasp detection using deep convolutional neural networks. In *Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2017.

[10] Di Guo, Fuchun Sun, Huaping Liu, Tao Kong, Bin Fang, and Ning Xi. A hybrid deep architecture for robotic grasp detection. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 1609–1614. IEEE, 2017.

[11] Fu-Jen Chu and Patricio A Vela. Deep grasp: Detection and localization of grasps with deep neural networks. *arXiv preprint arXiv:1802.00520*, 2018.

[12] Xinwen Zhou, Xuguang Lan, Hanbo Zhang, Zhiqiang Tian, Yang Zhang, and Nanning Zheng. Fully convolutional grasp detection network with oriented anchor box. *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2018.

[13] Lerrel Pinto and Abhinav Gupta. Supersizing self-supervision: Learning to grasp from 50k tries and 700 robot hours. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 3406–3413. IEEE, 2016.

[14] Cewu Lu, Ranjay Krishna, Michael Bernstein, and Li Fei-Fei. Visual relationship detection with language priors. In *European Conference on Computer Vision (ECCV)*, pages 852–869. Springer, 2016.

[15] Xiaodan Liang, Lisa Lee, and Eric P Xing. Deep variation-structured reinforcement learning for visual relationship and attribute detection. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4408–4417. IEEE, 2017.

[16] Bo Dai, Yuqi Zhang, and Dahua Lin. Detecting visual relationships with deep relational networks. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[17] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016.

[18] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in Neural Information Processing Systems (NIPS)*, pages 91–99, 2015.

[19] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C Berg. Ssd: Single shot multibox detector. In *European Conference on Computer Vision (ECCV)*, pages 21–37. Springer, 2016.