micromachines

Article

A Practical Multi-Stage Grasp Detection Method for Kinova Robot in Stacked Environments

Xuefeng Dong, Yang Jiang *, Fengyu Zhao and Jingtao Xia

Faculty of Robot Science and Engineering, Northeastern University, Shenyang 110169, China
* Correspondence: jiangyang@mail.neu.edu.cn; Tel.: +86-024-83656512

Abstract: Grasp detection takes on a critical significance for the robot. However, detecting object positions and corresponding grasp positions in a stacked environment can be quite difficult for a robot. Based on this practical problem, in order to achieve more accurate object position detection and grasp position detection, a new method called MMD (Multi-stage network for multi-object grasp detection algorithm) is proposed in this paper. MMD covers two parts, including the feature extractor and the multi-stage object predictor. The feature extractor refers to a deep convolutional neural network that can generate shared feature layers as well as the initial ROIs (region of interest). A multi-stage refiner serves as the multi-stage object predictor, which continuously regresses the initial ROI to obtain more accurate object detection and grasping detection results. Ablation experiments show that the proposed MMD has better grasp detection performance. The specific performance is that the recognition precision achieves a state-of-the-art 76.71% mAPg on the VMRD dataset. Moreover, test experiments demonstrate the feasibility of our method on the Kinova robot.

Keywords: grasp detection; multi-stage network; multi-task; stack scenarios; VMRD; Kinova robot

Citation: Dong, X.; Jiang, Y.; Zhao, F.; Xia, J. A Practical Multi-Stage Grasp Detection Method for Kinova Robot in Stacked Environments. *Micromachines* **2023**, *14*, 117. https://doi.org/10.3390/mi14010117

Academic Editor: Duc Truong Pham

Received: 23 November 2022
Revised: 27 December 2022
Accepted: 27 December 2022
Published: 31 December 2022

1. Introduction

Grasping is recognized as a basic function for robots, whereas it is quite difficult actually. Existing research [1–6] has focused on grasp detection in single object scenes. However, multi-object scenes [7] are common in practical applications. For a graping task on robots, the position of the object and where the object is grasped are both critical. This knowledge not only allows the robot to understand the types of objects to be grasped, but also to accurately find the grasping position of the object. Thus, the main research idea of this study is to use a multi-stage deep neural network to solve the multi-object grasping problem in stacked scenes. At the same time, it also has very good object detection performance. Therefore, our method has good practical value.

In general, deep learning based object detection schemes are divided into one-stage algorithms and two-stage algorithms, which currently include convolutional neural networks (R-CNN [6]), Fast R-CNN [8] and Faster R-CNN [9]. The initial R-CNN is adopted to detect the class and position of objects in the image. R-CNN applys the selective search method to generate regions of proposal, but it also limits the detection speed. In addition, it follows the method of convolution for the respective proposal of each image, thus significantly increasing the computational effort. Fast R-CNN replaces the method of extracting targets with network training, and the detection speed are significantly improved over before. Fast R-CNN shares convolution features with detection networks, which can greatly reduce the computation time of candidate regions. However, the selective search method is still retained. Faster R-CNN adopts the RPN (Regional Proposal Network) to achieve regional proposal generation, thus notably increasing the speed and quality of proposal box generation and addressing the performance bottleneck problem of Fast R-CNN.

Despite the good effect achieved by Faster-R-CNN, while the IOU (Intersection over Union) threshold is selected high, it will cause model overfitting. Besides, a single threshold will cause mismatch problems. A cascade type of network namely Cascade R-CNN [10] solves the mismatch problem from Faster R-CNN. Cascade R-CNN was implemented based on Faster-R-CNN. Cascade R-CNN is a multi-stage method that generates region proposals from the previous stage and feeds them into the next. It applys a shared feature layer to all stages to avoid a tremendous computational effort. Accordingly, Cascade R-CNN comprises two parts, including a feature extractor and multi-stage refiner. A feature extractor is adopted to generate shared feature layers, and then a multistage refiner is employed with increasing IOU thresholds to refine the region proposals. However, the Cascade R-CNN is designed for object detection and is not suitable for grasp detection. On that basis, MMD adds a grasp detection branch based on Cascade R-CNN to increase grasping accuracy. Several novel modules are proposed to make the network suitable for the grasping task. The experimental results prove that MMD is very effective on the VMRD dataset. The contributions of this study include:

(1) A Cascade R-CNN implementation based on Faster-RCNN for grasp detection is provided. Our model allows for simultaneous grasp detection and target detection.
(2) MMD is an improved multi-stage end-to-end grasp detection model that we proposed. This algorithm performs well on the VMRD dataset in stack scenarios and also shows great environmental adaptability on our homemade test sets.
(3) A FRM (feature refine model) is proposed in MMD, thus allowing the network to improve the quality of the region proposal feature map. Its effectiveness in facilitating the detection of grasping has been proven through experiments.
(4) A box redistribution strategy is proposed in MMD, which avoids filtering the false positive samples to a certain extent and increases the system's fault tolerance for detection. Experimental results also indicate that it can increase the accuracy of grasp detection.
(5) In order to test the practicality of our model, we also carried out experiments on our homemade test sets and our Kinova robot.

2. Related Works

The methods of earliest grasping detection mainly focus on the case of a single object. Jiang [1] et al. proposed a five-dimensional vector representation method for grasping detection on a 2D image plane and an algorithm for predicting the grasping box of a given object from an image. This method transforms the problem of grasping detection into finding five-dimensional vectors in the image and provides a solution for the application of depth learning in grasping detection. In 2014, Lenz et al. [11] of Cornell University proved that the five-dimensional grasp representation of 2D images can be projected into 3D space. They use first-depth learning in conjunction with robot grasping technology to detect objects on the plane. Chen et al. [4] established a new grasping detection model, which can more fairly evaluate the grasping proposals. This method can improve not only detection accuracy but also generalization. Yokota et al. [12] realized a new type of grasping detection network deployed on the picking robot, which can predict the picking position in the case of a small number of data sets. This method performs well in picking tasks. Although these methods have achieved good results, they are not suitable for stacking and multi object situations.

Research on the grasping of stacked objects has emerged in recent years. Guo et al. [13] propose a model-free shared convolutional network to predict grasping points and object detection boxes. The model is capable of performing object detection and grasp detection simultaneously, and the experimental result indicates that the shared model outperforms the single model, thus revealing that the two tasks are associated with each other to a certain extent. Another common method is the sliding window method for grasping detection. However, this method is time-consuming. Later, Redmon et al. [14] proposed a network named Multi-Grasp that directly regressed the grasping coordinates from images. The model can predict the grasping location of multiple objects in images that do not exist occlusion. In fact, the real-time is complex, and occlusion is inevitable. Zhang et al. [5]

studied the problem of grasp detection in stacking scenarios and proposed a novel multi-object robot grasp detection method. The algorithm consists of two parts. One is to detect the position and type of objects in the image; another is to detect the grasping position, for which a fully convolutional neural network is designed and implemented using a directed anchor frame mechanism. Wu et al. [15] proposed a multi-object grasp detection network that can learn object detection and grasp location detection simultaneously using multi-layer features. However, because of the complexity of the application environment, most of the methods do not fully meet the accuracy requirements of practical applications. In this study, a novel grasp model named MMD is proposed, which is capable of achieving a grasping accuracy by multi-stage regression. Compared with the above methods, it not only has higher detection accuracy but can also better meet the actual grasping requirements. Meanwhile, it also demonstrated robust environmental adaptability.

3. Proposed Method

In this section, we present the design of MMD. The system overview is first established in Section 3.1. Next, the network architecture is elucidated in Section 3.2. Subsequently, a useful model named FRM is presented in Section 3.3. Afterward, a novel algorithm named (BR) box redistribution is illustrated, which is capable of combining the low and high confidence box to obtain a more accurate regression in Section 3.4. Lastly, we will show the definition of the loss function in Section 3.5.

3.1. System Overview

Our architecture comprises two branches: object detection and grasp detection. The object detection branch obtained class and region proposals, and the grasp detection branch is intended to get the grasping position in ROI from object detection. They are not independent, on the contrary, they are combined and connected by sharing features and regions of interest. Here we divide the whole task into three steps, and the paradigm can be summarized as follows:

Step 1: Generate ROIs. Given a single monocular image as input, RPN generated a series of ROIs similar to Faster-RCNN.

Step 2: ROI Refine. A cascade refiner is adopted to generate efficient foreground proposals from ROIs.

Step 3: Grasp detection and object detection. The proposals generated by cascade regression are taken as input, and the grasping detection branch and object detection branch will output the final result.

3.2. Network Architecture

The MMD includs a feature extractor, multi-ROI regression and grasp detection, as shown in Figure 1. ResNet101 is a popular choice for a 2D detector for efficiently converting the RGB image into a feature vector. It is adopted as the backbone of our network to perform feature encoding because of its efficiency and accuracy.

Feature encoding. The ResNet101 [16] have five residual convolution blocks, which are defined as $\{R_1, R_2, R_3, R_{4,} R_5\}$. The network utilizes $\{R_1, R_2, R_3, R_{4,}\}$ to gradually convert the raw image $f \in R^{3\times W\times H}$ into feature vector $S \in R^{1024\times\frac{W}{16}\times\frac{H}{16}}$ with $2\times$, $4\times$, $8\times$, $16\times$ down-sampled size. The feature S will be shared in multi-ROI regress stage.

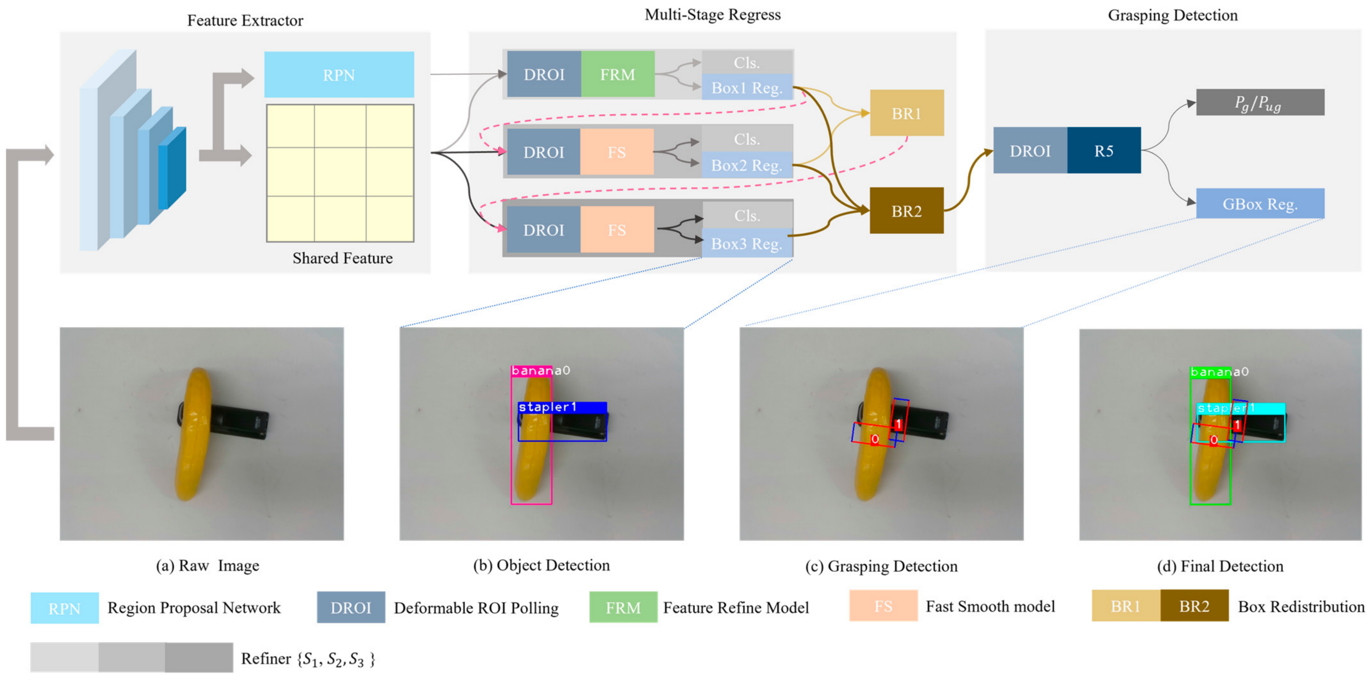

Figure 1. The overall architecture of the proposed MMD. The raw images (**a**) are first fed into 2D convolution network to generate Shared Feature and 2D object proposals. Subsequently, Shared Feature and 2D object proposals are passed into Multi-Stage Regress model to refine proposals Continuously and generate the predicted object Category and Bounding Box (**b**). Lastly, all object box features are aggregated by BR2 from every refiner to grasping detection to learn specific features for grasping box regress (**c**). The final detection result is illustrated as (**d**).

Multi-ROI regress. Here, a series of separate subnetworks with raising IOU thresholds {0.5, 0.6, 0.7} is used to refine region proposals similar as [17]. There are 3 refiners $\{S_1, S_2, S_3\}$ in the network of this study. The $(j-1)$ th region proposals B^{j-1} from prior refiner will input next refiner. Feature extractor $\varphi_j(\bullet)$ process features after deformable ROI pooling operation and output the feature F^j for jth refiner. Then, with F^j, a confidence branch $S(\bullet)$ and a box regression branch $R(\bullet)$ will generate a new object confidence C^j and box B^j, respectively. The process can be formulated as:

$$F^j = \varphi_j\left(B^{j-1}\right),\ C^j = S\left(F^j\right),\ B^j = R\left(F^j\right) \tag{1}$$

where $j = 1,2,3$. To ensure that the network of this study exhibits prominent performance, different modules are selected for different refiners. At the same time, to solve all the inherent problems of multistage networks, that is, errors tend to propagate downstream. We add BR_1 module and BR_2 module after refiner S_2 and refiner S_3, respectively.

Every refiner but refiner S_1 adopt Fs module to as feature extractor. Before this, it is necessary to process feature from ROIs into tensors of same size. So, the deformable ROI pooling operation is adopted from DcNv. For refiner S_2 and refiner S_3, the feature extractor function can be formulated as:

$$\varphi_j(\bullet) = FS\left(DROI\left(B^{j-1}, S\right)\right) \tag{2}$$

where DROI is deformable ROI pooling operation from DcNv and FS is the Fast Smooth module. FS module is made up of two convolution layers with kernel size 7×7 and 1×1, respectively, the former is to ensure that the height and width of the feature map are equal to one after sampling and the latter is to reduce the model complexity.

For refiner S_1, the feature extractor function can be formulated as:

$$\varphi_1(\bullet) = FRM\Big(DROI\Big(\ B^0,\ S\Big)\Big) \tag{3}$$

where FRM is Feature Refine Module. We will intrude the FRM in Section 3.3.

Object detection and grasp detection. The object confidence C^3 and proposals box B^3 from refiner S_3 is our object detection result that is same as [18]. Since the grasping position also should be obtained, which corresponds to the respective proposal, grasp detection heads are added to the region of interest from refiner S_3. In the training stage, IOU will be calculated between ROI and ground-truth box. Subsequently, matched ROI that have highest IOU will be remained to as final ROI which will be pass to next refiner. The function extractor function $\delta(\bullet)$ can be formulated as:

$$\delta(\bullet) = R_5(DROI(B\ ,\ S)) \tag{4}$$

where B is fused box parameters after BR_2. We will introduce box redistribution in Section 3.4.

Every ROI will be divided into M $\times$ N grid cells after DROI operation (both M and N are set to 7 in the experiments), and in each grid cell, a set of oriented anchors is used, which is a set of rectangles with oriented angles. Each grasping box is represented by a 5D vector $(x,\ y,\ w,\ h,\ \theta)$, as shown in Figure 2. Where the angle θ ranges from 0 to 180 degrees.

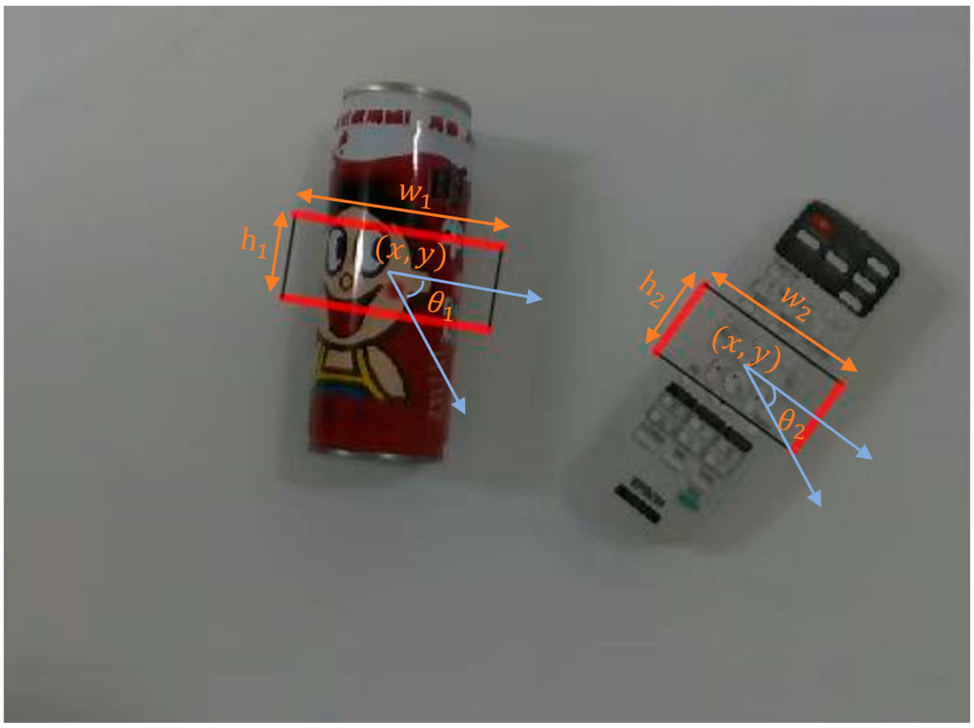

Figure 2. The representation of grasping box. Where $(x,\ y)$ is the box's centre. W and h represent the length and width of the grasping box. θ is the angle formed by the grasping box and the horizontal axis of the camera.

A 5 $\times$ k offset of the grasping rectangle regression variable in each grid cell relative to the anchor box at the location of the grasping rectangle. The respective 5D vector $(\sigma_x, \sigma_y, \sigma_w, \sigma_h, \sigma_\theta)$ is computed with respect to the anchor $(p_x, p_y, p_w, p_h, p_\theta)$ and the predicted grasping rectangle (x, y, w, h, θ) using Equations (5)–(9).

$$\sigma_x = (x - p_x)/\ p_w \tag{5}$$

$$\sigma_y = (y - p_y)/\ p_h \tag{6}$$

$$\sigma_w = log(w/\ p_w) \tag{7}$$

$$\sigma_h = log(h/\ p_h) \tag{8}$$

$$\sigma_\theta = (\theta - p_\theta)/(90/k) \tag{9}$$

where k is equal to 7. The grasping classifier provides $2 \times k$ confidence scores, thus indicating the probability of k anchor frames that are graspable and ungraspable, respectively. Finaly, the grasping detector will predict $M \times N \times k$ grasping proposals. And the appropriate ROI and the best grasp proposals frames are retained using the non-maximum suppression algorithm (NMS).

3.3. Feature Refine Module (FRM)

A feature refine module is introduced in this Section, as illustrated in Figure 3. The pooled features $P_1 = DROI(\ B^0,\ S)$ in the refiner S_1 will be input to FRM first, and feature extraction can be performed to allow the network to focus on the superior region proposal features.

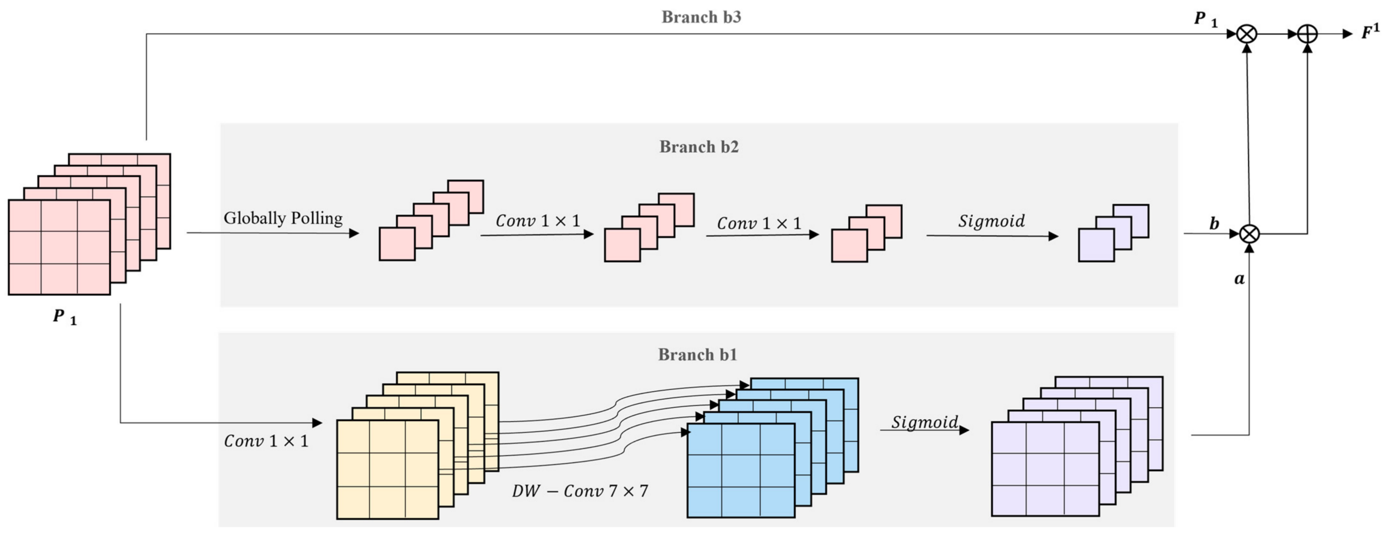

Figure 3. Feature refine model (FRM). The pooled features P_1 is fed into three branchs respectively to aggregate a better region proposal feature map.

The FRM module comprises three branches $\{b_1, b_2, b_3\}$, and the convolution layers in these branches all use the Mish activation function. For different branchs, we use different scheme to extract feature. For branch b_1, its formula can be presented as:

$$a = sig(AO(P_1)) \tag{10}$$

where AO is composed of a 1×1 convolution and a 7×7 Depthwise Separable Convolution. And $sig()$ is Sigmoid activation function. AO ensures the output features involve spatial features with a larger receptive field, and all values are normalized from 0 to 1.

The second branch is based on the channel attention model. Firstly, the globally pooling operation reduce the input features P_1's height and width to 1. Subsequently, two 1×1 convolutions are to adjust the channel of feature. Lastly, we adopt the sigmoid activation function to normalize the value to $(0,\ 1)$. Its specific formula can be written as:

$$b = sig\{C_{1\times1}((GP(P_1)))\} \tag{11}$$

where GP denotes globally pooling; $C_{1\times1}$ represents 1×1 convolutions.

Afterward, the final feature is obtained as:

$$F^1 = a \bullet b + P_1 \bullet a \bullet b \tag{12}$$

The final feature map covers both channel attention and spatial attention, thus allowing the network to improve the quality of the region proposal feature map.

3.4. Box Redistribution(BR)

In a multi-stage network, the errors tend to propagate along with downstream [17]. To further address this problem, box redistribution is proposed during the testing stage to build more connections between different stages. The idea behind this strategy stems from the fact that each stage produces output with high and low confidence, which can then be combined to produce a more accurate detection box. Notably, a method named "box voting" [17] has been proposed, but it is useless for us. In MMD, features are progressively refined, so the confidence from the refiner S_3 is considered to be more trustworthy. To solve this problem, a box redistribution (BR) mechanism is adopted, which fuses the box weights by directly weighting the detection box confidence and the detection confidence.

$$C = \frac{1}{N_r} \sum_j C^j \tag{13}$$

$$B = \frac{1}{\sum_j C^j} \sum_{j \epsilon \{1,2,3\}} \left(\gamma_j \bullet C^j \bullet B^j \right) \tag{14}$$

where C represents the fused score, B is the fused box parameter, and γ_j is a parameter to adjust the different confidence weights. To be specific, two BRs are set on at the beginning BR_1 and end of the refiner BR_2, respectivately. There $\gamma_{BR_1} = \{0.2,\ 0.8\}$ $\gamma_{BR_2} = \{0.3,\ 0.3, 0.4\}$. And when $\gamma = \{0.5,\ 0.5\}$, Box Redistribution will transforme into box voting [17].

3.5. Loss Function

The network is trained end-to-end with a multi-task loss function. The loss function comprises two components, including object detection loss l_o and grasp detection loss l_g. Since object detection refers to a multi-stage extraction ROI, the loss should be calculated for every ROI, such that the loss function for object detection is defined as:

$$l_o = \sum_{n=1}^{3} \left(l_{cls} + \lambda l_{pos} \right)_n \tag{15}$$

where l_{cls} expresses the categorical cross-entropy loss of the object; l_{pos} represents the Smooth-L1 loss between the predicted position and the ground-truth box; λ is adopted to adjust the weight occupied by the loss; n denotes the refiner number. The specific definition is illustrated as follows.

$$l_o = \sum_{n=1}^{3} \Big(\frac{1}{N_{cls}} \sum_i l_{cls}(p_i, p_i^T) + \lambda \frac{1}{N_{pos}} \sum_i p_i^* l_{pos}(t_i, t_i^T) \Big)_n \tag{16}$$

where N_{cls} and N_{pos} denote number of Classification and sampled region proposals on the training stage, repetitively. The Classification loss component l_{cls} is written as follows:

$$l_{cls}\left(p_i, p_i^T\right) = -\log\left[p_i^T p_i + (1 - p_i^T)(1 - p_i)\right] \tag{17}$$

where p_i denotes the probability that the region is proposed to be predicted as the target region. The object localization lossl_{pos} is written as:

$$l_{pos}\left(t_i, t_i^T\right) = smooth\ L_1\left(t_i - t_i^T\right) \tag{18}$$

where t_i denotes the four parameter coordinates $(x,\ y,\ w,\ h)$ of the rectangular box detected by the network; $(x,\ y)$ represents the center point of the predicted object rectangular box; $(w,\ h)$ expresses the width and height of the measured object rectangular box; t_i^T is the ground truth label.

For the grasping branch, the respective oriented anchor box $(p_x, p_y, p_w, p_h, p_\theta)$ will correspond to a 5D offset $\left(\sigma_x, \sigma_y, \sigma_w, \sigma_h, \sigma_\theta\right)$ and a 2D grasping box confidence score vector (ρ_g, ρ_{ug}). Thus, the grasp detection loss l_g is defined as follows.

$$l_g = l_{Gloc} + \alpha l_{Gcls} \tag{19}$$

$$l_{Gloc} = \sum_{i \in Positive}^{P} \sum_{m \in \{x,y,w,h,\theta\}} smoothL1\left(\sigma m_g^{(i)} - \sigma m_{gt}^{(i)}\right) \tag{20}$$

$$l_{Gcls} = -\sum_{i \in Positive}^{P} \log\left(\rho_g^{(i)}\right) - \sum_{i \in Negative}^{3P} \log\left(\rho_{ug}^{(i)}\right) \tag{21}$$

where σm_{gt},$\sigma m_g \in \{x, y, w, h, \theta\}$ express the grasping ground-truth offset and the predicted proposals offset, respectively. Every positive sample is set as an anchor matching at least one ground truth. An anchor is considered as a negative sample if not matching any ground truth value. Lastly, the top P positive samples and the top 3P negative samples are selected for classification training. Furthermore, the entire loss of the network is defined as follows.

$$loss = l_o + \lambda l_g \tag{22}$$

where λ is all set to 1 in our experiments.

4. Experiment

In this section, we mainly discuss the details of our experiment. The multi-object dataset we use is VMRD. To verify the generalization capability of the model, experiments were also tested on our homemade test sets and the Kinova robot. The specific brief introduction to this section is presented as follows:

First, the dataset applied is presented in Section 4.1. Next, some details will be illustrated during the training stage in Section 4.2. Subsequently, the metric of validation is explained in Section 4.3. Afterward, we show the validation results on the VMRD in Section 4.4. Lastly, we make a series of ablation experiments to prove the effectiveness of our design in Section 4.5. Moreover, in order to ensure the practicality of our model, we have done a series of comparative experiments on the Kinova robot in Section 4.6.

4.1. Dataset

VMRD dataset. The v2 version of the multi-object VMRD [19–21] dataset comprises 4233 training data and 450 test data (RGB images). The dataset covers 31 object classes with 2–5 stacked objects in each image. The dataset is labeled with the true value of the grasping box and the corresponding category for the respective image. The "parent node" and "child node" rules are adopted to annotate each target pair. Actually, this rule specifies the order to follow when grasping different objects. Moreover, VMRD annotates both object kind and position. Since the original intention of the VMRD grasping dataset is to allow multi-objective grasping position detection in the position information of all objects. Furthermore, it is expected that the grasp detection algorithm is capable of acquiring the overall information in the picture rather than the local information.

4.2. Tarining Details

The two Nvidia 3090 graphics cards with 24 GB video memory were employed in this part of the experimental training, and the Pytorch network framework was adopted in the python2 environment, with stochastic gradient descent as the optimizer with an initial learning rate $lr = 0.0001$ and a 10-fold decrease in the learning rate per 12 training rounds. The batch size was set to 3 for 36 training rounds. In the training process, NMS IoU 0.7 was set to keep 2000 proposals with a 1:1 ratio of positive and negative samples. SGD was employed as the optimizer, and the momentum was set to 0.9. Other settings are consistent with those of Cascade RCNN.

4.3. Evalution Metrics

The criteria for correct forecasting are specified as the following two requirements.

(1) The difference in angle between the predicted grasping proposals and the ground-truth box should be less than 30°.

(2) The Jaccard intersection ratio between the predicted grasping proposals and the ground-truth box should not be less than 25%. The specific context is in Figure 4. A is the ground truth box, and B is the predicted grasping box.

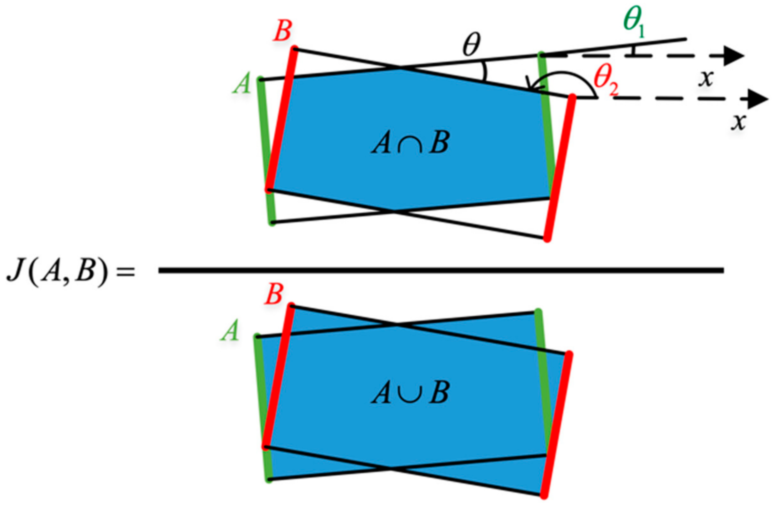

Figure 4. Jaccard coefficient.

The angle from the ground truth box is assumed to be θ_1 and the angle from the prediction box to be θ_2. On that basis, the difference between the two angles can be obtained as $\min\left(|\theta_2 - \theta_1|, 180^{\circ} - |\theta_2 - \theta_1|\right)$. Jaccard intersection ratio measures the correlation between two boxes.

Mean Average Precision (mAP) with grasp is employed as the evaluation metric for multi-target grasp position detection, which is expressed as follows:

$$MAP = \frac{\sum_{j\epsilon\{1,2....N_m\}} Ap^j}{N_m} \tag{23}$$

where N_m denotes the total number of object types; Ap^j expresses the Average Precisio from jth Category.

4.4. Evaluation on VMRD Dataset

This section aims to explore MMD in multi-object scences. Specifically, experiments will be performed on the multi-object grasping dataset VMRD to verify the effectiveness of the MMD proposed in this chapter.

The output of the actual prediction is expressed as follows:

$$f_{out} = \sum_{i=1}^{n} \left(x, y, w, h, cls, x_g, y_g, w_g, h_g, \theta\right)_i \tag{24}$$

The accuracy of grasping detection for various objects is obtained in Figure 5. We found that the shape of the object with the accuracy is random and does not have a special distribution rule, which indicates that the network has indeed learned the corresponding characteristics.

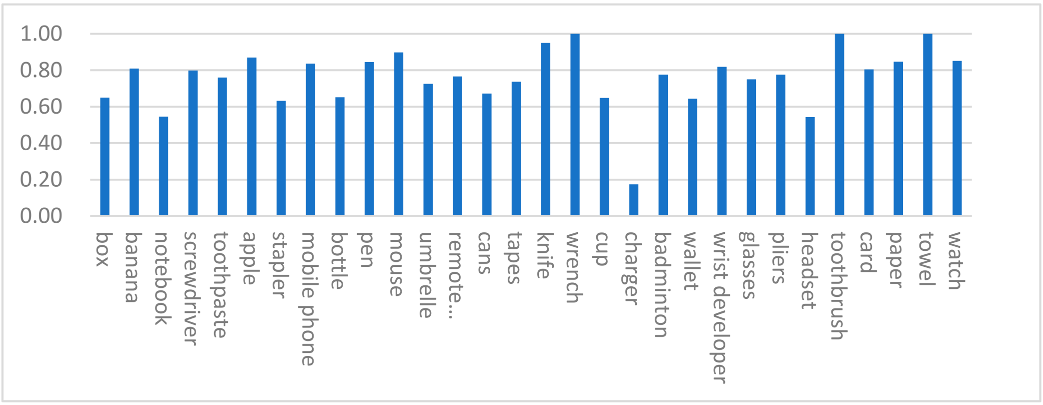

Figure 5. Object grasp position detection accuracy.

The advance of MMD is demonstrated, and our model was trained using all the training sets of the VMRD dataset. Figure 6 presents the sample test results of the model using the algorithm. The results of the VMRD testing set are listed in Table 1. our scheme achieves 76.71% mAPg on the VMRD grasping dataset.

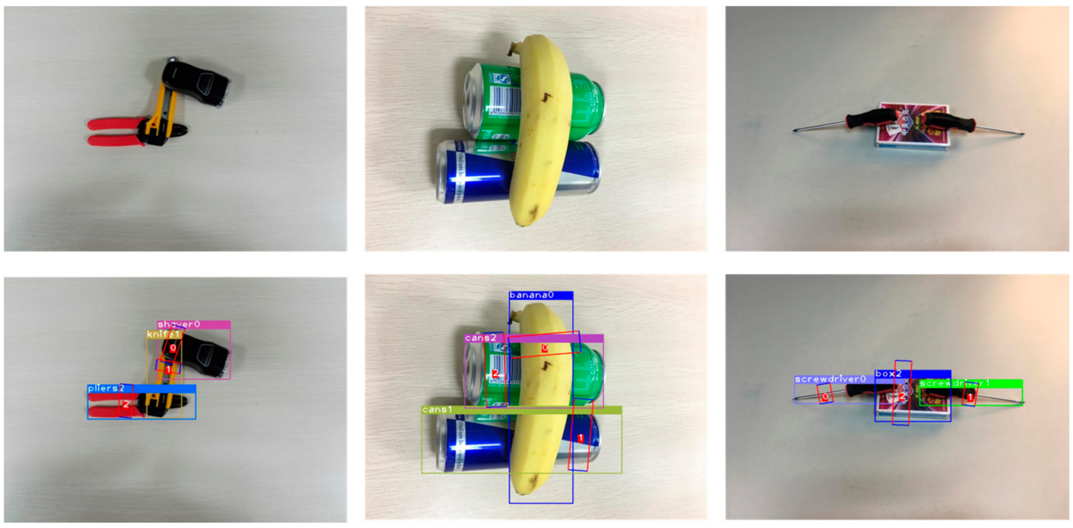

Figure 6. Sample test results of MMD. The three images above are the original images from the VRMD and the following are the results of the test.

Table 1. Performance of different algorithm on VMRD Dataset.

Algorithm	mAPg (%)
Faster-RCNN [9] + FCGN [18]	54.5
ROI-GD [22]	68.2
Zhang [5]	70.5
Keypoint-based scheme [7]	74.3
MMD	74.57
MMD + FRM	76.02
MMD + FRM + RR1 + RR2	76.71(+2.41%)

4.5. Ablation Study

Experiments were performed on the VMRD dataset testing set.

(1) Effectiveness of FRM

We have done comparative experiments on FPM in different situations. The results are shown in Table 2. Although grasping detection results with FRM are better than without

FRM, the object detection effect can be damaged. This result illustrates the effectiveness of our module design for grasp detection.

Table 2. Effect of Feature Refine Model. Each colour represents a different comparison experiment.

FRM	RR1	RR2	mAPg (%)	mAPd (%)
			74.57	94.68
√			76.02	93.14
	√		75.73	94.46
√	√		76.68	92.56
		√	74.58	94.73
√		√	76.04	93.17
	√	√	75.76	94.48
√	√	√	76.71	92.86

(2) Position of FRM.

We tested the performance of the model when the FRM was in different positions. The results are shown in Table 3. Our model performs best when FRM is applied to the refiner. Therefore, we adpot the FRM on refiner S_1 in this article. At the same time, we find that the choice to put FRM on refiner S_3 will seriously affect grasping performance. One possible explanation is that the feature map from the refiner S_3 is influenced by the residual convolution block S_5.

Table 3. Effect of Feature Refine Model at different stages.

S_1	S_2	S_3	mAPg (%)	mAPd (%)
			74.57	94.68
√			76.71	92.86
	√		75.10	92.91
		√	1.84	94.60
√	√		75.10	92.91
√		√	1.84	94.60
	√	√	1.69	92.88
√	√	√	1.69	92.88

(3) Effectiveness of Box Redistribution

We also conducted an ablation study on the effectiveness of box redistribution, and the results are shown in Table 4. By adding this design, the accuracy of our model increases by 1.19 percentual points without FRM and by 0.69 percentual points with FRM, which indicates the effectiveness of the box redistribution.

Table 4. Effect of Box Redistribution. Each colour represents a different comparison experiment.

FRM	RR1	RR2	mAPg (%)	mAPd (%)
			74.57	94.68
	√		75.73	94.46
		√	74.58	94.73
	√	√	75.76	94.48
√			76.02	93.14
√	√		76.68	92.56
√		√	76.04	96.17
√	√	√	76.71	92.86

(4) Parameters of Box Redistribution

A series of comparative experiments were set up to explore the effect of different parameters on network performance, and Table 5 lists the experimental results. In the proposed multi-stage network, the detection box from the later refiner has higher reliability. Moreover, the confidence level of the previous stage can avoid over-fitting. Given the result of the above analysis, the parameters in BR2 are empirically selected as 0.3, 0.3, 0.4, respectively. With this design, the proposed model achieves the optimal results, thus verifying the effectiveness of the parametric design.

Table 5. Effect of different parameters on BR1.

γ_1	γ_2	**mAPg (%)**	**mAPd (%)**
0.1	0.9	75.61	93.13
0.2	0.8	76.71	92.86
0.3	0.7	74.35	92.63
0.4	0.6	74.03	92.51
0.5	0.5	72.12	91.9
0.6	0.4	72.05	91.91
0.7	0.3	72.94	91.7
0.8	0.2	72.25	91.2
0.9	0.1	73.29	91.15

(5) Environmental adaptability Research

Due to limited materials, two different testing sets were built in accordance with some of the testing images in VMRD. For the first testing set, 500 RGB images were selected randomly from the VMRD, and existing objects were adopted to mimic their placement and manner as much as possible. The only difference from the original data was that the background plate was changed (Figure 7). The second test set was built with a capacity of 200 using the same items as the VMRD free build, part of which is presented in Figure 8. In terms of the deep learning model, poor adaptability is a common and difficult problem to solve, thus revealing a significant degradation in the model's performance when the environment changes slightly. Although our dataset is small, it is effective in detecting model environmental adaptation. It was confirmed that the proposed model is environmentally robust, and some grasps of that are illustrated as Figures 7 and 8. A plausible explanation is that multi-stage selection and BR design increase system fault tolerance.

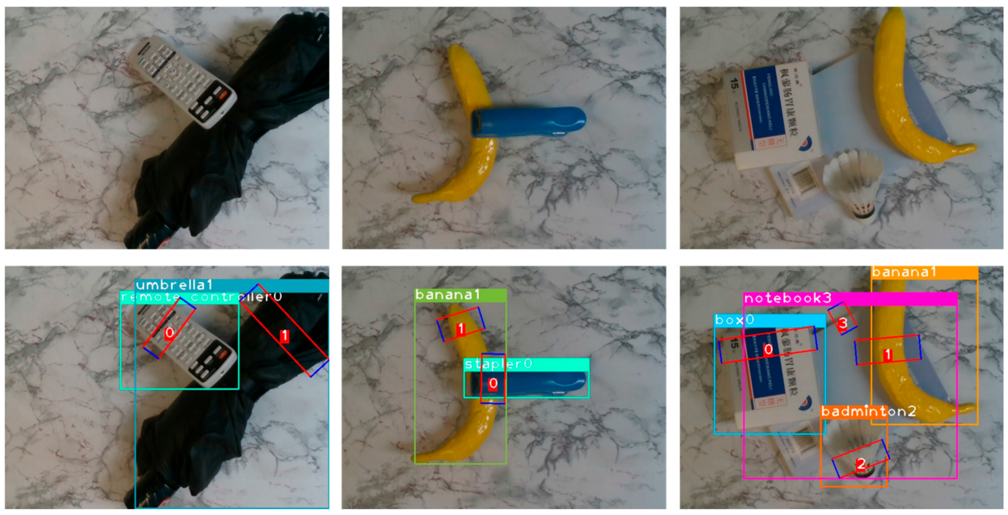

Figure 7. First test set sample. The three images above are the original images from the its and the following are the results of the test.

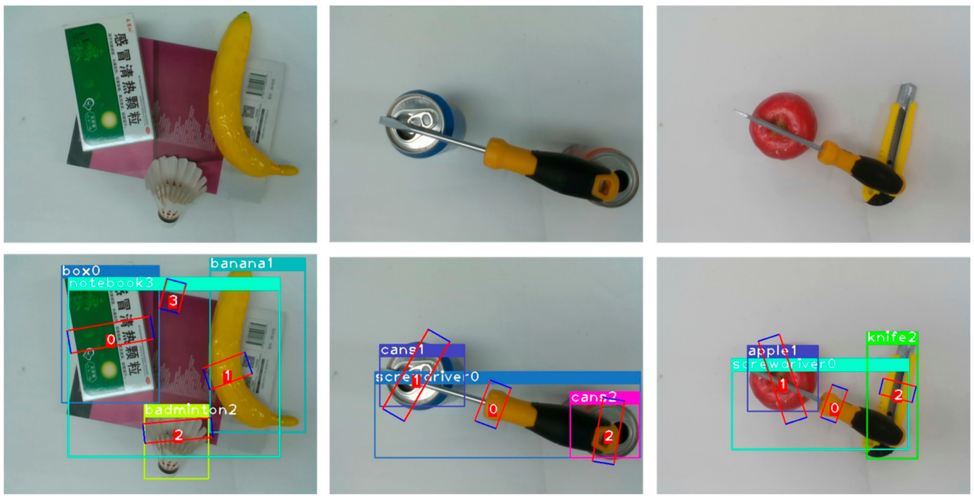

Figure 8. Second test set sample. The three images above are the original images from the its and the following are the results of the test.

4.6. Experiment on Kinova Robotic Arm

This section consists of three parts. Our equipment condition is introduced in Section 4.6.1. Then, some details are explained for hand-eye calibration and camera calibration in Sections 4.6.2 and 4.6.3. Finally, we show the experimental results and process of Kinova robot arm in Section 4.6.4.

4.6.1. Details of Equipment

A 16.04 Ubuntu operating system was employed on the Kinova robot. The core processor of the robotic arm is an Intel i5-10400 CPU that made in China. Moreover, the Kinova robot is also equipped with a Kinova Mico2 six-degree-of-freedom robotic arm manufactured by kinavo company from Quebec, Canada, and an Intel Realsense D435 camera made in China.

4.6.2. Hand-Eye Calibration

In this paper, the open source project named easy_hand-eye [1] is used to complete the hand-eye calibration, and the experiment adopts an automatic method without relying on additional hardware for eye-in-hand calibration. The camera is fixed at the end of the robot, and the coordinate system is shown in Figure 9.

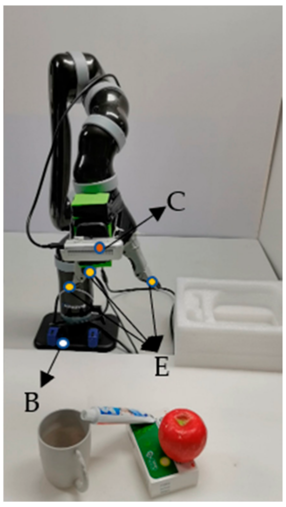

Figure 9. Coordinate System Declaration. Where B is the base coordinate system, E is the End Actuator Coordinate System, C is the camera coordinate system.

We define the coordinate system of the calibration as K. As well as ${}^{M}_{N}T$ is present the conversion matrix of N coordinate system to M coordinate system, where $M, N \in \{B, C, E, K\}$. During the calibration process, the position of the calibration board is kept fixed, and the camera pose is adjusted automatically or manually. Based on the statement above, the transformation of the calibration plate coordinate system relative to the base coordinate system of the manipulator can be defined as:

$$ {}^{B}_{K}T = {}^{B}_{E}T^{(1)} \bullet {}^{E}_{C}T \bullet {}^{C}_{K}T^{(1)} = {}^{B}_{E}T^{(2)} \bullet {}^{E}_{C}T \bullet {}^{C}_{K}T^{(2)} \tag{25} $$

Then we invert both sides simultaneously,

$$ {}^{B}_{E}T^{(2)^{-1}} \bullet {}^{B}_{E}T^{(1)} \bullet {}^{E}_{C}T = {}^{E}_{C}T \bullet {}^{C}_{K}T^{(2)} \bullet {}^{C}_{K}T^{(1)^{-1}} \tag{26} $$

Let

$$ P = {}^{B}_{E}T^{(2)^{-1}} \bullet {}^{B}_{E}T^{(1)} \tag{27} $$

$$ Q = {}^{C}_{K}T^{(2)} \bullet {}^{C}_{K}T^{(1)^{-1}} \tag{28} $$

$$ X = {}^{E}_{C}T \tag{29} $$

Therefore

$$ PX = XQ \tag{30} $$

where X is the parameters obtained through hand-eye calibration.

4.6.3. Camera Calibration

The internal parameters of the camera are calibrated using the Zhang Zhengyou calibration method [23]. Firstly, Complete Realsense D435 installation driver under Ubuntu16.04. Then, it is necessary to start the camera and install Kalibr in the ROS environment. The calibration board is a black and white checkerboard with a size of 6×7 and a resolution of $0.029 \times 0.029m^2$. The calibration data is recorded through the rosbag in the calibration tool. Then the calibration program completes the final calibration. Lastly, we get

$$ K = \begin{bmatrix} f_x & s & x_0 \\ 0 & f_y & y_0 \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 624.1 & 0 & 318.54 \\ 0 & 625.22 & 241.02 \\ 0 & 0 & 1 \end{bmatrix} \tag{31} $$

4.6.4. Grasp Experiment on Knovia

In order to verify the utility of our model, we decided to do some experiments on stacked multi-object scenes. The experimental method is to select 2 to 3 kinds of objects and place them on the table, then randomly stack the objects. The robot needs to grasp sequentially according to the results of the grasping detection. The criterion for judging whether the grasping is successful or not is the same as above, and a total of 150 grasping experiments were carried out. Figure 10 depicts the actual scene multi-target grasp detection results. The grasping process shown in Figure 11.

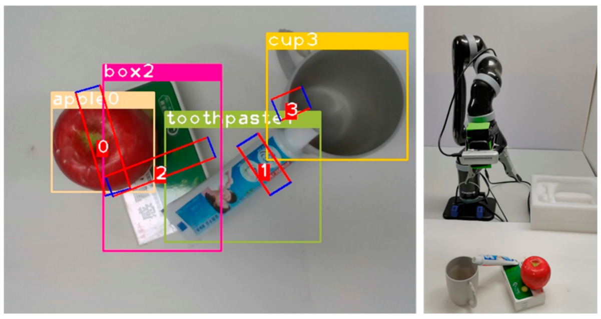

Figure 10. Multi-target grasp detection result.

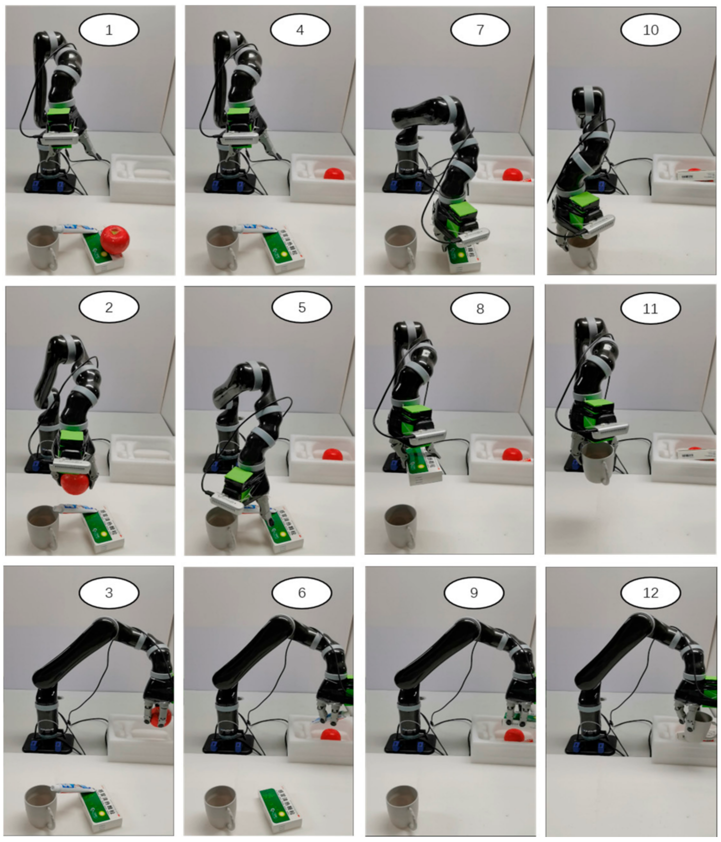

Figure 11. The grasp process in actual scences.

During the experiment, the number of grasps and success for each object were counted. And the final results are shown in Table 6.

Table 6. Comparison of experimental results before and after improvement.

Method	Experiment Number	Number of Success	Grasp Success Rate (%)
Cascade R-CNN [10]	150	142	94.60
MMD + FRM + RR1 + RR2	150	146	97.33

Experiments show that our approach is effective. In the stacking scene, it can complete the grasping task very well. This illustrates the practicality of our method.

5. Conclusions

In this study, a grasping detection algorithm based on multi-stage regress is proposed for the multi-object grasp position detection problem in complex environments. This algorithm is capable of detecting all object grasp positions and all object positions and types in the image. The algorithm is tested on VMRD, a multi object grab dataset. As indicated by the experimental results, the proposed algorithm outperforms existing multi-object grasp detection algorithms on the VMRD dataset. Moreover, to ensure the generalizability and practicality of the model, it was also tested on two test sets of our own making and on a kinova robot arm. The results reveal that MMD exhibits robust environmental adaptability and good practical performance. It also verifys the effectiveness of MMD. These good detection effects are closely related to FRM and BR modules. In fact, although our model has achieved a good grasping effect, it still faces the general problem of deep learning beacause the target category is fixed. At the same time, the impact of the selection of different backbone networks and network parameters has not been explored in this paper.

In addition, our grasping order is still marked manually, which is also an urgent matter to be solved. Therefore, the follow-up research is still meaningful.

Author Contributions: Conceptualization, X.D., Y.J. and J.X.; methodology, F.Z., X.D. and J.X.; software, F.Z., X.D. and J.X.; validation, X.D. and J.X.; formal analysis, X.D.; investigation, X.D.; resources, Y.J.; data curation, X.D.; writing—original draft preparation, X.D.; writing—review and editing, X.D. and F.Z.; visualization, F.Z. and X.D.; supervision, Y.J.; project administration, Y.J. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by Science and Technology Plan Project of Shenfu Innovation Demonstration Zone, liaoning province, China: No.2020JH19 and National Natural Science Foundation of China: No. U20A20197.

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Jiang, Y.; Moseson, S.; Saxena, A. Efficient grasping from rgbd images: Learning using a new rectangle representation. In Proceedings of the International Conference on Robotics and Automation, Shanghai, China, 9–13 May 2011; pp. 3304–3311.
2. Krizhevsky, A.; Sutskever, I.; Hinton, G.E. Imagenet classification with deep convolutional neural networks. *Adv. Neural Inf. Process. Syst.* **2012**, *25*, 3406–3413. [CrossRef]
3. Pinto, L.; Davidson, J.; Gupta, A. Supervision via competition: Robot adversaries for learning tasks. In Proceedings of the International Conference on Robotics and Automation, Singapore, 29 May–3 June 2017; pp. 1601–1608.
4. Chen, L.; Huang, P.; Meng, Z. Convolutional multi-grasp detection using grasp path for RGBD images. *Robot. Auton. Syst.* **2019**, *113*, 94–103. [CrossRef]
5. Zhang, H.; Lan, X.; Bai, S.; Wan, L.; Yang, C.; Zheng, N. A multi-task convolutional neural network for autonomous robotic grasping in object stacking scenes. In Proceedings of the International Conference on Intelligent Robots and Systems, Macau, China, 3–8 November 2019; pp. 6435–6442.
6. Girshick, R.; Donahue, J.; Darrell, T.; Malik, J. Rich feature hierarchies for accurate object detection and semantic segmentation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Columbus, OH, USA, 23–28 June 2014; pp. 580–587.
7. Li, T.; Wang, F.; Ru, C.; Jiang, Y.; Li, J. Keypoint-based robotic grasp detection scheme in multi-object scenes. *Sensors* **2021**, *21*, 2132. [CrossRef] [PubMed]
8. Girshick, R. Fast r-cnn. In Proceedings of the IEEE International Conference on Computer Vision, Santiago, Chile, 7–13 December 2015; pp. 1440–1448.
9. Ren, S.; He, K.; Girshick, R.; Sun, J. Faster R-CNN: Towards real-time object detection with region proposal networks. *IEEE Trans. Pattern Anal. Mach. Intell.* **2016**, *39*, 1137–1149. [CrossRef] [PubMed]
10. Cai, Z.; Vasconcelos, N. Cascade r-cnn: Delving into high quality object detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 6154–6162.
11. Lenz, I.; Lee, H.; Saxena, A. Deep learning for detecting robotic grasps. *Int. J. Robot. Res.* **2015**, *34*, 705–724. [CrossRef]
12. Yokota, Y.; Suzuki, K.; Kanazawa, Y.; Takebayashi, T. A multi-task learning framework for grasping-position detection and few-shot classification. In Proceedings of the International Symposium on System Integration, Honolulu, HI, USA, 12–15 January 2020; pp. 1033–1039.
13. Guo, D.; Kong, T.; Sun, F.; Liu, H. Object discovery and grasp detection with a shared convolutional neural network. In Proceedings of the International Conference on Robotics and Automation, Stockholm, Sweden, 16–21 May 2016; pp. 2038–2043.
14. Redmon, J.; Angelova, A. Real-time grasp detection using convolutional neural networks. In Proceedings of the International Conference on Robotics and Automation, Seattle, WA, USA, 26–30 May 2015; pp. 1316–1322.
15. Wu, G.; Chen, W.; Cheng, H.; Zuo, W.; Zhang, D.; You, J. Multi-object grasping detection with hierarchical feature fusion. *IEEE Access* **2019**, *7*, 43884–43894. [CrossRef]
16. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep residual learning for image recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016.
17. Wu, H.; Deng, J.; Wen, C.; Li, X.; Wang, C.; Li, J. CasA: A Cascade Attention Network for 3-D Object Detection from LiDAR Point Clouds. *IEEE Trans. Geosci. Remote Sens.* **2022**, *60*, 1–11. [CrossRef]
18. Zhou, X.; Lan, X.; Zhang, H.; Tian, Z.; Zhang, Y.; Zheng, N. Fully convolutional grasp detection network with oriented anchor box. In Proceedings of the International Conference on Intelligent Robots and Systems, Madrid, Spain, 1–5 October 2018; pp. 7223–7230.
19. Park, D.; Seo, Y.; Shin, D.; Choi, J.; Chun, S.Y. A single multi-task deep neural network with post-processing for object detection with reasoning and robotic grasp detection. In Proceedings of the International Conference on Robotics and Automation, Paris, France, 31 May–31 August 2020; pp. 7300–7306.

20. Zhang, H.; Yang, D.; Wang, H.; Zhao, B.; Lan, X.; Ding, J.; Zheng, N. Regrad: A large-scale relational grasp dataset for safe and object-specific robotic grasping in clutter. *IEEE Robot. Autom. Lett.* **2022**, *7*, 2929–2936. [CrossRef]
21. Zhang, H.; Lan, X.; Zhou, X.; Tian, Z.; Zhang, Y.; Zheng, N. Visual Manipulation Relationship Network for Autonomous Robotics. In Proceedings of the IEEE-RAS 18th International Conference on Humanoid Robots (HUMANOIDS), Beijing, China, 6–9 November 2018.
22. Zhang, H.; Lan, X.; Bai, S.; Zhou, X.; Tian, Z.; Zheng, N. Roi-based robotic grasp detection for object overlapping scenes. In Proceedings of the International Conference on Intelligent Robots and Systems, Macau, China, 3–8 November 2019; pp. 4768–4775.
23. Zhang, Z. A flexible new technique for camera calibration. *IEEE Trans. Pattern Anal. Mach. Intell.* **2000**, *22*, 1330–1334. [CrossRef]