Frontiers of Information Technology & Electronic Engineering
www.jzus.zju.edu.cn; engineering.cae.cn; www.springerlink.com
ISSN 2095-9184 (print); ISSN 2095-9230 (online)
E-mail: jzus@zju.edu.cn

Attention-based efficient robot grasp detection network*

Xiaofei QIN†[1], Wenkai HU[1], Chen XIAO[2], Changxiang HE[2], Songwen PEI[1,3,4], Xuedian ZHANG†‡[1,3,4,5]

[1]*School of Optical-Electrical and Computer Engineering, University of Shanghai for Science and Technology, Shanghai 200093, China*

[2]*College of Science, University of Shanghai for Science and Technology, Shanghai 200093, China*

[3]*Shanghai Key Laboratory of Modern Optical System, Shanghai 200093, China*

[4]*Key Laboratory of Biomedical Optical Technology and Devices of Ministry of Education, Shanghai 200093, China*

[5]*Shanghai Institute of Intelligent Science and Technology, Tongji University, Shanghai 201210, China*

†E-mail: xiaofei.qin@usst.edu.cn; obmmd_zxd@163.com

Received Oct. 23, 2022; Revision accepted Apr. 9, 2023; Crosschecked May 22, 2023

Abstract: To balance the inference speed and detection accuracy of a grasp detection algorithm, which are both important for robot grasping tasks, we propose an encoder–decoder structured pixel-level grasp detection neural network named the attention-based efficient robot grasp detection network (AE-GDN). Three spatial attention modules are introduced in the encoder stages to enhance the detailed information, and three channel attention modules are introduced in the decoder stages to extract more semantic information. Several lightweight and efficient DenseBlocks are used to connect the encoder and decoder paths to improve the feature modeling capability of AE-GDN. A high intersection over union (IoU) value between the predicted grasp rectangle and the ground truth does not necessarily mean a high-quality grasp configuration, but might cause a collision. This is because traditional IoU loss calculation methods treat the center part of the predicted rectangle as having the same importance as the area around the grippers. We design a new IoU loss calculation method based on an hourglass box matching mechanism, which will create good correspondence between high IoUs and high-quality grasp configurations. AE-GDN achieves the accuracy of 98.9% and 96.6% on the Cornell and Jacquard datasets, respectively. The inference speed reaches 43.5 frames per second with only about 1.2×10^6 parameters. The proposed AE-GDN has also been deployed on a practical robotic arm grasping system and performs grasping well. Codes are available at https://github.com/robvincen/robot_gradet.

Key words: Robot grasp detection; Attention mechanism; Encoder–decoder; Neural network
https://doi.org/10.1631/FITEE.2200502 **CLC number:** TP391.4

1 Introduction

The development of intelligent robots has brought great convenience to people's lives. Intelligent robots can replace humans in labor tasks, such as handling, assembly, palletizing, household service (Wang Q et al., 2022), and logistics sorting. In these applications, robot grasping plays a vital role, in which the position of the object relative to the robot's gripper must be calculated. Traditional methods are based on model analysis, such as form closure and force closure methods, in which the specific parameters of the object must be known and the calculation process is complicated. Therefore, model analysis methods are difficult to use in unstructured scenarios. In recent years, algorithms based on computer vision and deep learning have become mainstream in the field of robot grasping due to the great progress in artificial intelligence. These algorithms

‡ Corresponding author

* Project supported by the National Natural Science Foundation of China (No. 92048205) and the China Scholarship Council (No. 202008310014)

ORCID: Xiaofei QIN, https://orcid.org/0000-0002-0258-5423; Xuedian ZHANG, https://orcid.org/0000-0002-7636-7517

usually use a deep neural network to detect the grasp configuration of the object in a visual image. This process, called grasp detection of the object, can be conducted using system parameters.

Jiang et al. (2011) conducted a pioneering work of grasp detection based on computer vision, giving the definition of correct grasp box, and many subsequent works follow this definition in evaluating performance. Lenz et al. (2015) proposed a two-step grasp detection method based on fully connected neural networks and presented a five-dimensional representation of the grasp configuration. Guo et al. (2017) fused tactile and visual information, and used the sliding window mechanism for grasp detection. Although these early methods achieved good accuracy, they suffered from poor real-time performance, which is important for grasp detection tasks. Therefore, many algorithms optimized for execution speed have emerged subsequently. Redmon and Angelova (2015) proposed a method using the idea of meshing to directly regress multiple grasp configurations of the object. Kumra and Kanan (2017) used ResNet (He et al., 2016) as the feature extraction backbone, extracted features from RGB and depth inputs separately, and fused them in the middle layer. Morrison et al. (2018) proposed a pixel-level grasp detection network with much fewer network parameters and less computation, making it fully meet the real-time requirement of robot grasping. However, although many existing methods have achieved good performance, the balance between the real-time requirement and accuracy still needs improvement.

To meet the real-time requirement of grasp detection and obtain better accuracy, in this paper we propose an attention-based efficient robot grasp detection network (AE-GDN) with an encoder–decoder structure. Inspired by Asif et al. (2019), this network uses DenseBlocks for feature modeling, which is efficient with less computation. In the encoder stages, spatial attention modules are used to improve the detailed information modeling capability. In the decoder stages, channel attention modules are used to improve the semantic information modeling capability.

In the mainstream grasp detection methods, a grasp detection result is regarded as correct if the intersection over union (IoU) between the predicted grasp rectangle and the ground truth (GT) is larger than the threshold. However, as shown in Fig. 1, although the predicted rectangles of the top row seem to meet the IoU threshold, if the grasp is performed, a fingertip on one side of the gripper will collide with the object. For clarity, Fig. 2 presents the top row of Fig. 1. If an IoU threshold of 0.25 is used, as in most methods, the robot will grasp the objects with its fingers at the red bars, which will obviously collide with the objects at the cross points in Fig. 2.

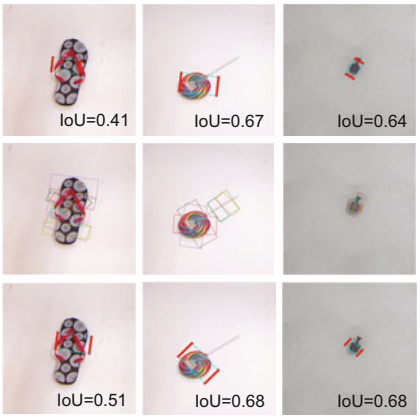

Fig. 1 Visual comparison of grasp detection results and the ground truth

The top row shows grasp rectangles achieved by GG-CNN (Morrison et al., 2018), which has a high IoU but the risk of collision, the middle row shows the ground truth, and the bottom row shows grasp rectangles achieved by this work without collision. IoU: intersection over union

Fig. 2 The collision explanation

IoU: intersection over union. References to color refer to the online version of this figure

The reason is that in the calculation of IoU the central area of the predicted grasp rectangle and the area near the fingertips are treated equally, although the area near the fingertips is obviously more important than the central area. Therefore, when IoU loss is calculated during training, this work parses the predicted grasp configuration into hourglass-shaped boxes, so the impact of the central area of the predicted grasp rectangles is reduced. Note that because the test accuracy given by other methods is based on the IoU calculated by rectangles, for fair comparison, the test accuracy results of this work are also calculated based on rectangles.

The main contributions of this work are summarized as follows:

1. We propose an encoder–decoder structured grasp detection neural network named AE-GDN. The accuracy of this model on the Cornell and Jacquard datasets reaches 98.9% and 96.6%, respectively.

2. The IoU item in the loss function is improved by using an hourglass-shaped predicted grasp box instead of a rectangular one when IoU is calculated, which will reduce the influence of the central area in the predicted grasp boxes and create good correspondence between high IoU values and grasping success rates.

3. We deploy the proposed AE-GDN in a robotic arm grasp system with an antipodal gripper, and apply it to grasp 25 commonly used tools and the necessities with considerable shape diversity and complexity. Real-time grasping can be achieved with a fairly high success rate. The proposed AE-GDN algorithm is open-sourced in our GitHub repository at https://github.com/robvincen/robot_gradet.

2 Related works

Robot grasp detection methods based on deep learning can be divided roughly into two categories according to the fundamental mechanism: object detection (OD) and heatmap matching (HM). OD-based grasp detection algorithms can be divided into two- and one-stage methods. The two-stage method includes a region generation network and a grasp box regressor. First, the regions of interest (RoIs) in the feature map are obtained through the backbone, and the grasp boxes are predicted in the second stage. The one-stage method obtains the grasp boxes by directly regressing the features extracted by the backbone. HM-based grasp detection algorithms output a pixel-level grasp configuration instead of the multiple discrete grasp probabilities that are outputted by OD-based methods, and different preprocessing methods need to be executed according to various output formats.

2.1 Methods based on object detection

2.1.1 Two-stage methods

Lenz et al. (2015) proposed a cascaded network based on a sparse autoencoder structure. At first, the potential grasp boxes are exhaustively generated by a simple network to produce a set of rectangular grasp candidates, and in the second stage a deeper network is used to select the best one. Guo et al. (2017) used the network proposed by Zeiler and Fergus (2014) to fuse visual information and tactile information, and discretized the rotation angle for classification. Zhou et al. (2018) proposed an oriented anchor box matching mechanism with ResNet (He et al., 2016) as the backbone. Zhang et al. (2019) extended the method proposed by Zhou et al. (2018), and used an RoI-based method to solve problems when multiple objects to be grasped are stacked together. Dex-Net 2.0, proposed by Mahler et al. (2017), first generates grasp candidates for the depth image, and then chooses the best grasp box by evaluating the grasp quality of the candidate set. Ainetter and Fraundorfer (2021) improved the second stage of the grasp detection network by introducing a semantic segmentation branch and using it to refine the grasp detection boxes.

The two-stage grasp detection method tends to output pretty accurate grasp boxes. However, this is based on sacrificing computational speed, because generating candidate regions in the first stage is time-consuming and is not suitable for robot grasping tasks with significant real-time requirements.

2.1.2 One-stage methods

One-stage methods directly predict a set of grasp configurations that can be parsed into grasp rectangles in the image. Redmon and Angelova (2015) proposed a regression-based grasp detection method, and AlexNet (Krizhevsky et al., 2012) was adopted as the backbone. This method assumes that each image contains only one grasp object, divides the image into a set of grids, and predicts a grasp configuration for each grid. Kumra and Kanan (2017) proposed a deep convolutional neural network (CNN) to predict robust grasp configurations, demonstrating the potential of multimodal information fusion. A commonly used representation of the grasp configuration includes the grasp rectangle and rotation angle. The rotation angle means the deflection angle of the grasp rectangle relative to the horizontal line. Chu et al. (2018a) discretized the rotation angle into multiple different classes, and used a classification method to predict it. The reason for discretizing the rotation angle is that the angle

is a non-Euclidean value (Hara et al., 2017), so the standard regression loss function is not suitable. Although methods using a discretizing mechanism as that used in Chu et al. (2018a) converge more easily during training, their output angles are limited to the preset classification set. Park et al. (2020) used a spatial transformer network (Jaderberg et al., 2015) and ResNet (He et al., 2016) instead of a sliding window mechanism to meet the real-time requirement.

The one-stage grasp detection method does not need to generate RoIs, so it has high inference speed but limited grasp detection accuracy.

2.2 Methods based on heatmap matching

The heatmap matching method outputs pixel-level grasp configurations. Commonly used grasp configuration primitives include the grasp width, rotation angle, object position, and grasp quality score. The heatmap matching method uses the error between the predicted output of these primitives and their corresponding GT values for supervised training.

The GG-CNN proposed by Morrison et al. (2018) is a pioneering heatmap matching grasp detection method. It is a CNN without any fully connected layer, taking only depth information as input, and the numbers of parameters and calculations have been greatly reduced, so real-time performance is guaranteed. Asif et al. (2019) proposed a densely supervised grasp detector with a multi-level CNN structure. This network generates global, regional, and pixel-level grasp configurations, and chooses the best one as the final output. Kumra et al. (2020) proposed a grasp detection network named GR-ConvNet, which can handle different input modes including RGB, RGB-D, and depth. Based on the GG-CNN of Morrison et al. (2018), GR-ConvNet adds multiple ResBlocks (He et al., 2016) to connect the encoder and decoder, which can provide better mapping capabilities from input to grasp configuration. DD-Net (Wang Y et al., 2021) defines the grasp configuration as a pair of fingertips, and presents a specialized loss function to supervise the training process.

The grasp detection method based on heatmap matching can achieve good balance among robustness, inference speed, detection accuracy, etc., and has aroused extensive research enthusiasm in the robot grasping community in recent years. AE-GDN proposed in this paper is a heatmap matching method. It improves the grasp detection performance mainly through modifications in three aspects: backbone, network architecture, and IoU loss. AE-GDN can achieve fairly high grasp detection accuracy and real-time execution with inference time between 23 and 26 ms.

3 Problem formulation

To better understand the grasp detection algorithm and its role in the overall practical robot grasping task, like other grasp detection papers, we provide a simple definition of the robot grasping problem. In this study, the robot grasping problem is defined as a planar grasping problem, and the image captured by the camera is used to predict the grasp configurations. Then the grasp configurations are converted into grasp commands executed by the robotic arm.

First, the grasp detection algorithm is used to detect the pixel-level grasp configuration, which is noted as $\boldsymbol{G}$. In this study, $\boldsymbol{G}$ is defined as follows:

$$\boldsymbol{G}=(\boldsymbol{W},\boldsymbol{\Theta},\boldsymbol{Q})\in\mathbb{R}^{3\times H\times W}, \tag{1}$$

where $\boldsymbol{W}$, $\boldsymbol{\Theta}$, and $\boldsymbol{Q}$ represent three output images with the same size as the input image. Each pixel in these images can be regarded as the width, rotation angle, and grasp quality score of a grasp rectangle candidate.

Second, an argmax operation is performed on $\boldsymbol{Q}$ of $\boldsymbol{G}$ to obtain the pixel coordinates of the highest grasp quality score, and then the coordinates are used as the index to obtain the width, rotation angle, and grasp quality score of the grasp rectangle in the pixel coordinate system $\boldsymbol{G_p}$. In this study, $\boldsymbol{G_p}$ is defined as follows:

$$\boldsymbol{G_p}=(\boldsymbol{p},w_{\mathrm{p}},\theta_{\mathrm{r}},q), \tag{2}$$

where $\boldsymbol{p}$ denotes the pixel coordinates (u,v) of the grasp center point, w_{p} is the width of the pixel-level grasp rectangle, θ_{r} is the deflection angle relative to the horizontal line of the image in the range of $[\frac{-\pi}{2},\frac{\pi}{2}]$, and q is the pixel-level grasp quality score in the range of $[0,1]$.

Third, $\boldsymbol{G_p}$ is converted into the camera coordinate system through the camera's internal parameters and the depth information captured by the

stereo camera, which is noted as $\boldsymbol{G}_{\mathrm{c}}$. In this study, $\boldsymbol{G}_{\mathrm{c}}$ is defined as follows:

$$\boldsymbol{G}_{\mathrm{c}}=\left(\boldsymbol{P}, w_{\mathrm{c}}, \theta_{\mathrm{r}}, q\right), \tag{3}$$

where $\boldsymbol{P}$ denotes the coordinates (x, y, z) of the grasp center point in the camera coordinate system, w_{c} is the width of the grasp in the camera coordinate system (i.e., the difference of the opening and closing degrees of the gripper), θ_{r} is the same as that in Eq. (2) because in this study the eye-in-hand robot grasping system is used, and q represents the grasp quality score.

Finally, $\boldsymbol{G}_{\mathrm{c}}$ is converted into the robot coordinate system through external parameters of the robot and camera, which is noted as $\boldsymbol{G}_{\mathrm{r}}$. The definition of $\boldsymbol{G}_{\mathrm{r}}$ is similar to that of $\boldsymbol{G}_{\mathrm{c}}$, and will not be given here. The convertion process from $\boldsymbol{G}_{\boldsymbol{p}}$ to $\boldsymbol{G}_{\mathrm{r}}$ can be represented as follows:

$$\boldsymbol{G}_{\mathrm{r}}=\mathrm{Trans}_{\mathrm{rc}}\left(\boldsymbol{G}_{\mathrm{c}}\right)=\mathrm{Trans}_{\mathrm{rc}}\left(\mathrm{Trans}_{\mathrm{cp}}\left(\boldsymbol{G}_{\boldsymbol{p}}\right)\right), \tag{4}$$

$$\mathrm{Trans}_{\mathrm{rc}}(\boldsymbol{G}_{\mathrm{c}})=\begin{bmatrix}\boldsymbol{R} & \boldsymbol{T}\\ \boldsymbol{0} & \boldsymbol{1}\end{bmatrix}\boldsymbol{G}_{\mathrm{c}}=\boldsymbol{G}_{\mathrm{r}}, \tag{5}$$

where $\boldsymbol{R}$ is the rotation matrix from the camera coordinate system to the robot coordinate system, and $\boldsymbol{T}$ is the translation matrix. Similarly, $\boldsymbol{R}$ in $\mathrm{Trans}_{\mathrm{cp}}$ is the rotation matrix from the pixel coordinate system to the camera coordinate system, and $\boldsymbol{T}$ in $\mathrm{Trans}_{\mathrm{cp}}$ is the translation matrix.

4 Method

We propose an end-to-end grasp detection CNN named AE-GDN as shown in Fig. 3, in which DenseBlocks (Huang et al., 2017) are used to connect the encoder and decoder. Due to the efficiency of DenseBlock, AE-GDN possesses powerful feature modeling capability and high inference speed. Inspired by CBAM (Woo et al., 2018), spatial and channel attention modules are added to the encoder and decoder stages of AE-GDN, respectively. Because the inputs of the spatial and channel attention modules are different, coming from shallow and deep layers of the network, the capability of AE-GDN to extract detailed and semantic information can be enhanced. During the calculation of traditional IoU loss, the central area of the predicted grasp rectangle and the area near the fingertips are treated equally, leading to bad correspondence between high IoUs and high grasp success rates as shown in Fig. 1. In this study,

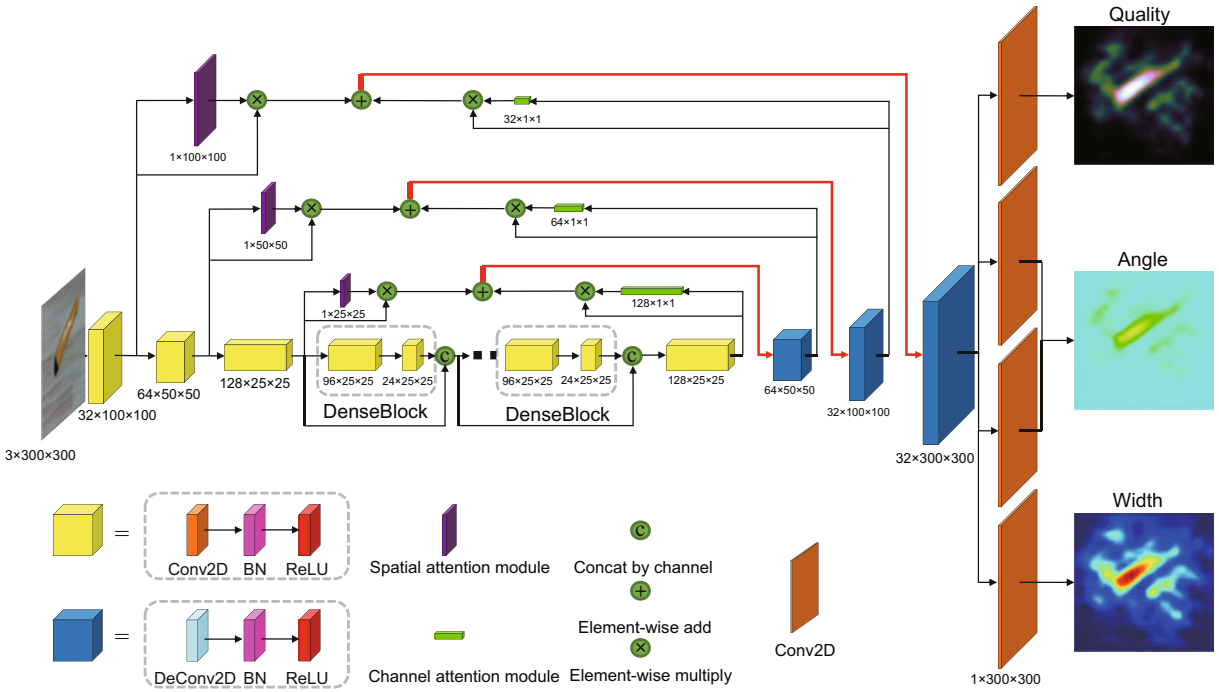

Fig. 3 Model architecture

BN: batch normalization; ReLU: rectified linear unit. In the figure, the red lines represent the fused feature inputted into the decoder stages. References to color refer to the online version of this figure

new IoU loss based on an hourglass-shaped grasp box is designed to reduce the impact of the central area in the predicted grasp box.

4.1 Model architecture

Fig. 3 shows the model architecture of the proposed AE-GDN, which is an encoder–decoder structured pixel-level robot grasp detection network. The output images, which are the grasp quality score image, rotation angle image, and width image, have the same resolution as the input. Specifically, the three output images have the same resolution, so given the pixel with the highest grasp quality, the width and angle can be known. Note that in this study, we follow the approach of GG-CNN (Morrison et al., 2018) to predict the rotation angle. Here, the rotation angle is not directly outputted; instead, $\sin(2\theta)$ and $\cos(2\theta)$ are first calculated and then parsed into the rotation angle image. The reason is that the L_2 loss function cannot be used directly in the non-Euclidean space of the angle value (Hara et al., 2017).

AE-GDN has three encoder and decoder stages. At the bottom of the encoder–decoder structure, some DenseBlocks are used to enhance the feature modeling capability. DenseBlock is an outstanding computer vision feature extractor; it can facilitate model training, strengthen feature propagation, and encourage feature reuse. Importantly, the number of its parameters and the amount of calculations are small due to its narrow layers; it can meet the real-time requirement of robot grasping. According to its output channel number, DenseBlock has many variants, and this study adopts the variant with 24 output channels. As the number of DenseBlocks increases, the performance of the model will increase, but it will also lead to an increase in the inference time and the number of parameters, which is not conducive to the real-time robot grasping. Therefore, after balancing the pros and cons between the grasp detection performance and the inference time, the number of DenseBlocks selected in this study is 18.

In the downsampling path, the output of each encoder stage is inputted into a spatial attention module to enhance the detail information. A 1×1 convolution layer is used for the output of the last DenseBlock to reduce the channel number. In the upsampling path, three channel attention modules are used to enhance the semantic information. The outputs of the spatial and channel attention modules are fused together, and fed into each decoder stage. The output of the last decoder stage is inputted into the grasp prediction layer. The design of the spatial and channel attention modules is inspired by convolutional block attention module (CBAM) (Woo et al., 2018), but the difference is that AE-GDN uses different inputs for spatial and channel attention modules, as shown in Fig. 4. To make the network pay more attention to the object in the image instead of the background, this study feeds the features of the encoder into the spatial attention module. To make the network pay more attention to the semantic information in the features, this study feeds the features of the decoder into the channel attention module. Because spatial information is more abundant in the shallower layer and semantic information is richer in the deeper layer (Fang et al., 2022), the spatial attention module is used only in the downsampling path and the channel attention module is used only in the upsampling path.

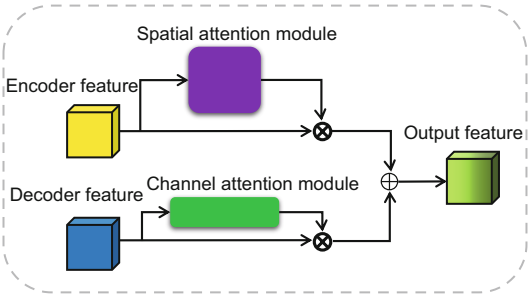

Fig. 4 The improved attention module for feature fusion

The parameter number of AE-GDN will change slightly according to different input modes. When the input mode is depth, RGB, or RGB-D image, there are 1.373×10^6, 1.378×10^6, or 1.380×10^6 parameters, respectively. AE-GDN is smaller in size compared with many works such as GR-ConvNet (Kumra et al., 2020) and DD-Net (Wang Y et al., 2021), and its inference time is between 23 and 26 ms on our graphics processing unit (GPU) workstation, which can meet the real-time requirement of the robot grasping task.

4.2 Loss function

The total loss of AE-GDN is defined as follows:

$$\mathcal{L} = \mathcal{L}_{\text{quality}} + \mathcal{L}^{\cos}_{\text{angle}} + \mathcal{L}^{\sin}_{\text{angle}} + \mathcal{L}_{\text{width}} + \mathcal{L}_{\text{GIoU}}, \quad (6)$$

where $\mathcal{L}_{\text{quality}}$ refers to the grasp quality score loss function in GG-CNN (Morrison et al., 2018), and using the mean square error (MSE) loss function, it is defined as follows:

$$\mathcal{L}_{\text{quality}} = \text{MSE}\left(q_{\text{g}}, \hat{q}_{\text{g}}\right), \tag{7}$$

where q_{g} is the predicted grasp score image with size 300×300, and $\hat{q}_{\text{g}}$ is the GT, i.e., the image mask of the center third of the labeled grasp rectangle. $\mathcal{L}^{\cos}_{\text{angle}}$, $\mathcal{L}^{\sin}_{\text{angle}}$, and $\mathcal{L}_{\text{width}}$ can be defined similarly.

Because the ordinary IoU loss function has a weakness, the gradient will vanish when the two boxes do not overlap. The generalized intersection over union (GIoU) (Rezatofighi et al., 2019) solves the above problem; specifically, the larger the distance between the two boxes, the smaller the GIoU. We calculate GIoU as follows:

$$\text{GIoU} = \text{IoU} - \frac{\mathcal{E}\left(P, G\right) - \mathcal{U}\left(P, G\right)}{\mathcal{E}\left(P, G\right)}, \tag{8}$$

where $\mathcal{E}$ is the area of the smallest box that contains both the predicted grasp box P and GT rectangle G. $\mathcal{U}$ is the union of the predicted grasp box P and GT rectangle G.

Fig. 5 shows the examples of GT rectangle G and the predicted grasp box P for calculating GIoU. The left part of Fig. 5 shows an example when G and P do not intersect, and then we have Eq. (9):

$$\text{GIoU} = -\frac{E}{G + P + E} = -\frac{1}{\frac{G+P}{E} + 1}. \tag{9}$$

When G and P get closer to each other, E will get smaller, so GIoU will get larger, which can provide the gradients needed by model convergence.

The right part of Fig. 5 shows an example when G and P have some intersection, and then we have Eq. (10):

$$\begin{aligned} \text{GIoU} &= \frac{I}{G + P - I} - \frac{E}{G + P + E} \\ &= \frac{1}{\frac{G+P}{I} - 1} - \frac{1}{\frac{G+P}{E} + 1}. \end{aligned} \tag{10}$$

When G and P get closer to each other, I will get larger and E will get smaller, so GIoU will get larger, providing converging gradients. Therefore, no matter whether G and P have some intersection or not, the model can always benefit from the supervision of the GIoU loss.

As shown in Fig. 1, when the traditional predicted grasp rectangle (the red rectangle in Fig. 6) is used to calculate IoU, a high IoU value does not correspond to a good grasp configuration, because the center part of the predicted grasp rectangle is not as important as the area around the gripper. We parse the predicted grasp configuration into an hourglass (HG) box (the orange shadow in Fig. 6), and use it as P in Eq. (8). Because the maximum IoU of the HG box and GT rectangle is around 0.5, a small change is made to the loss function, which is defined as follows:

$$\mathcal{L}_{\text{GIoU}} = 0.5 - \text{GIoU}. \tag{11}$$

In summary, the power of AE-GDN comes mainly from two sources. The first source is the delicately designed model architecture, i.e., the efficient DenseBlock backbone and the spatial and channel attention modules. The number of DenseBlocks is chosen to balance the model accuracy and inference speed. The spatial and channel attention modules take different features from the encoder and decoder as input, which can extract both detailed and semantic information from the input image. The second source is the creative idea of the hourglass-shaped predicted grasp box, which makes the model pay more attention to the area around the fingertips.

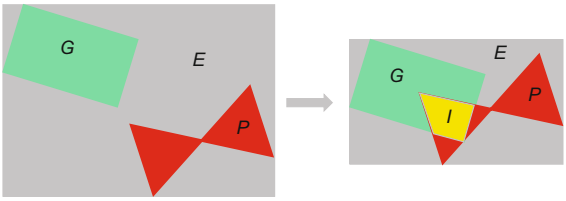

Fig. 5 Examples of the ground truth (GT) rectangle and the predicted grasp box for calculating the generalized intersection over union (GIoU)

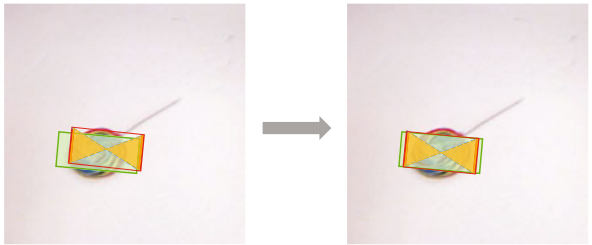

Fig. 6 The effect of the hourglass box, which drives the grasp detection result to change from the left (collision between the object and gripper despite a high IoU value) to the right during training. References to color refer to the online version of this figure

5 Experiments

To evaluate the effectiveness of the AE-GDN proposed in this study, grasp detection and robot grasping experiments are carried out.

5.1 Grasp detection

5.1.1 Datasets and metrics

Grasp detection experiments are carried out on the Cornell and Jacquard datasets.

The Cornell dataset (Jiang et al., 2011) contains a total of 885 pairs of RGB and depth images including 240 objects, so an RGB-D image can also be obtained. The resolution of these images is 640×480, and each pair of images is marked with several grasp rectangles. In our work only 5110 graspable configurations in the Cornell dataset are used.

The Jacquard dataset (Depierre et al., 2018) contains a total of 5.4×10^4 different scenes and 1.1×10^4 different objects, which are all simulation models extracted from a large CAD dataset. The labeled graspable rectangles are generated based on successful grasps in the experimental environment. The Jacquard dataset has more than 1.0×10^6 grasp rectangles.

The current mainstream grasp detection evaluation methods follow the criteria proposed by Jiang et al. (2011), which are also adopted in this work to perform fair comparisons with other methods. The evaluation method judges whether a grasp rectangle is valid based on two conditions. One is that IoU between the predicted rectangle and GT rectangle is ≥ 0.25, and the other is that the angle between the predicted rectangle and the GT rectangle is $\leq 30°$.

5.1.2 Grasp detection implementation details

The network in this stduy is built using PyTorch, the operating system is Ubuntu16.04, the hardware device used for network training and inferencing is an Intel Xeon® ES-1660 v3 eight-core processor clocked at 3.00 GHz, and the graphics card is NVIDIA GeForce GTX1080Ti.

The input data mode can be depth image, RGB image, or RGB-D image. Note that the RGB image and the depth image are aligned, so the depth value of each pixel can truly reflect the depth information. The model is trained using the Adam optimizer, the mini-batch size is 32, the initial learning rate is 0.001, the learning rate decays every 15 epochs, and the decay factor α is 0.5.

Both datasets are divided into two sets with a ratio of 9:1 for training and testing. In the Cornell dataset, there are image-wise (IW) splits and object-wise (OW) splits. The former splits the dataset directly and randomly. The latter divides the dataset according to the type of object. That is, objects in training do not appear in testing.

Because the Cornell dataset is small, data augmentation is used to alleviate overfitting. In the inference stage, we choose the maximum value point of the grasping quality score as the grasp center point, so the corresponding grasp width and angle can be obtained from the two other predicted images.

5.1.3 Evaluation and analysis on the Cornell dataset

For the Cornell dataset, Table 1 shows the performance comparison between AE-GDN and state-of-the-art methods with different input modalities. First, when the modality is depth or RGB-D, the grasp detection accuracy of AE-GDN is better than those of other methods. When the RGB-D input modality is used, AE-GDN can achieve the best accuracy, i.e., 98.9% on the IW splits and 97.9% on the OW splits. Furthermore, Table 1 shows that the highest inference speed of AE-GDN is 44.3 frames per second (FPS), generally between 37.9 and 44.3 FPS. This can meet the real-time requirement of the robot grasping task. Although the accuracy of Zhou et al. (2018) is better than that of AE-GDN when RGB input modality is used, their parameter numbers are much larger. It is worth noting that the inference speed of Kumra et al. (2020) in Table 1 is the reported value from the original paper, which is higher than that of AE-GDN. However, the number of parameters and the amount of calculations of Kumra et al. (2020) calculated with the PyThon package thop are 1.82×10^6 and 9.47 GFlops, which are larger than 1.38×10^6 and 9.19 GFlops of AE-GDN (RGB-D input modality), respectively. Therefore, the theoretical inference speed of Kumra et al. (2020) should be lower than that of AE-GDN. The reason why the reported speed of Kumra et al. (2020) is higher might be that their inference hardware platform is more powerful or their deployment optimization is better. The inference speed of Morrison et al. (2018) is higher than ours because the number of network parameters of this method is only

Table 1 Comparison using the Cornell dataset

Algorithm	Input	Accuracy (%)		Speed
		Image-wise	Object-wise	(FPS)
Fast search (Jiang et al., 2011)		60.5	58.3	0.02
SAE (Lenz et al., 2015)	RGB-D	73.9	75.6	0.07
Two-stage closed-loop (Wang ZC et al., 2016)	RGB-D	85.3		7.1
STEM-CaRFs (Asif et al., 2017)	RGB-D	88.2	87.5	
GG-CNN (Morrison et al., 2018)	D	73.0	69.0	**52.63**
GraspNet (Asif et al., 2018)	RGB-D	90.6	90.2	41.67
Multiple, M=10 (Ghazaei et al., 2018)	RGB	91.5	90.1	17.86
ResNet-50 model (Chu et al., 2018b)	RGB-D	96.0	96.1	8.33
ResNet-50 RCGN (Zhou et al., 2018)	RGB	97.7	94.9	9.89
GraspPath (Chen et al., 2019)	RGD	86.4	84.7	
GRPN (Karaoguz and Jensfelt, 2019)	RGB	88.7		5
RoI-GD, ResNet-101 (Zhang et al., 2019)	RGB	93.6	93.5	25.16
Multi-scale, multi-grasp (Chen et al., 2020)	RGB	96.5		
GR-ConvNet (Kumra et al., 2020)	D	93.2	94.3	**52.63**
	RGB	96.6	95.5	**52.63**
	RGB-D	97.7	96.6	50
DD-Net, HG-104 (Wang Y et al., 2021)	RGB	97.2	96.1	
AE-GDN w/o Att or HG	D	95.0	94.3	46.1
	RGB	95.4	94.9	41.3
	RGB-D	96.9	95.5	39.8
AE-GDN w/o HG but w Att	D	95.9	95.0	44.3
	RGB	96.0	95.9	40.6
	RGB-D	97.5	96.5	37.9
AE-GDN	D	97.5	96.1	44.3
	RGB	97.2	96.4	40.6
	RGB-D	**98.9**	**97.9**	37.9

Best results are in bold. w/o means without and w means with. HG: hourglass; Att: attention; FPS: frame per second

about 6×10^4. Finally, the results in Table 1 also demonstrate the effectiveness of the attention modules and the HG box modules. At first glance, the spatial and channel attention modules used in this work seem complex, which might influence the speed of the network. However, in fact, the added number of parameters and amount of computations caused by the attention modules are $\leq 3 \times 10^3$ and ≤ 0.006 GFlops (can be calculated by the PyThon package thop), and have little impact on the inference speed of the entire network. From Table 1, it can also be seen that no matter whether the attention module is added or not, the difference in inference speed is no more than 2 FPS. If it is converted into inference time, the difference is no more than 1.3 ms.

The accuracy of the model will definitely increase when the number of DenseBlocks increases, as shown in Table 2 and Fig. 7. However, more DenseBlocks mean more computation and parameters, and also longer inference time. The accuracy reaches a high level when 18 DenseBlocks are used, and the real-time performance is important for practical robot grasping applications. When the number of DenseBlocks exceeds 18, the accuracy gain is not proportional to the increase of the number of DenseBlocks, as shown in Fig. 7. In this work, AE-GDN uses 18 DenseBlocks to balance the accuracy and inference speed.

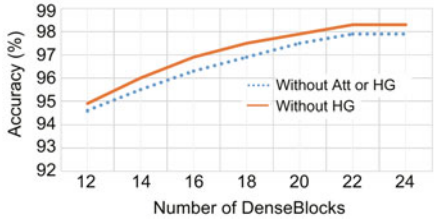

Fig. 7 Image-wise grasp detection accuracy on the Cornell dataset when different numbers of DenseBlocks are used

Note that the solid line represents that the attention module is used but without HG

As described in Section 5.1.1, a predicted grasp rectangle is considered as correct when its IoU value with the GT rectangle is ≥ 0.25. This threshold (0.25) is called the Jaccard index. Obviously, the Jaccard index has a great impact on the performance

of the grasp detection algorithm. The HG box mechanism proposed in this work can enhance the robustness of the grasp detection algorithm to the Jaccard index. As shown in Tables 3 and 4, when the Jaccard index becomes larger, the grasp detection accuracy of AE-GDN with HG does not drop significantly.

5.1.4 Evaluation and analysis on the Jacquard dataset

For the Jacquard dataset, Table 5 shows the performance comparison between AE-GDN and state-of-the-art methods with different input modalities. When the input modality is depth or RGB-D, the grasp detection accuracy of AE-GDN is better than those of other methods. When the RGB-D input modality is used, AE-GDN can achieve the best accuracy, i.e., 96.6%. Although the accuracy of Wang Y et al. (2021)'s method is better than that of AE-GDN when the RGB input modality is used, its number of parameters is much larger. The results in Table 5 also demonstrate the effectiveness of the HG box mechanism. Similar to the Cornell dataset, the impact of the Jaccard index is tested on the Jacquard dataset, as shown in Table 6; when the Jaccard index becomes larger, the grasp detection accuracy of AE-GDN with HG does not drop significantly.

Some visualization results on the Cornell and Jacquard datasets are given in Fig. 8. The column on the right of the two with the same object uses the HG mechanism, which tends to output a more graspable rectangle.

Table 2 Computation amount, parameter number, and image-wise grasp detection accuracy on the Cornell dataset when different numbers of DenseBlocks are used

Computation amount (GFlops)	M ($\times10^6$)	Num	Accuracy (%)	
			w/o Att or HG	w/o HG
8.92	0.96	12	94.6	94.9
9.01	1.09	14	95.5	96.0
9.10	1.23	16	96.3	96.9
9.19	1.38	18	96.9	97.5
9.29	1.54	20	97.5	97.9
9.40	1.71	22	97.9	98.3
9.51	1.89	24	97.9	98.3

M indicates the number of parameters, and Num indicates the number of DenseBlocks. w/o means without and w means with. HG: hourglass; Att: attention

Table 3 Accuracy on the Cornell dataset with different Jaccard indices when the RGB-D input modality is used

Algorithm		Accuracy (%)		
		0.25	0.30	0.35
Guo et al. (2017)'s*	IW	93.2	91.0	85.3
	OW	82.8	79.3	74.1
Guo et al. (2017)'s**	IW	86.4	83.6	76.8
	OW	89.1	85.1	80.5
Chu et al. (2018b)'s	IW	96.0	94.9	92.1
	OW	96.1	92.7	87.6
AE-GDN w/o HG	IW	96.9	94.5	91.4
	OW	96.5	94.7	90.4
AE-GDN	IW	98.9	96.5	94.4
	OW	97.9	96.0	93.5

* means a model trained and evaluated only in the Cornell dataset, and ** means a model trained in the Cornell and THU grasp datasets but evaluated on the Cornell dataset. w/o means without. IW: image-wise; OW: object-wise; HG: hourglass

5.2 Robot grasp

5.2.1 Grasp objects and metrics

The proposed AE-GDN algorithm is deployed on a practical robotic arm grasp system, which is tested to grasp one of the 25 tools and daily necessities in our laboratory as shown in Fig. 9. In the testing process, each object is tested in eight random

Table 4 Accuracy on the Cornell dataset with different Jaccard indices when the RGB input modality is used

Algorithm		Accuracy (%)		
		0.25	0.30	0.35
Song et al. (2020)'s	IW	96.2	94.0	89.5
	OW	95.6	93.4	90.5
AE-GDN w/o HG	IW	96.0	93.9	89.4
	OW	95.9	92.9	88.0
AE-GDN	IW	97.2	95.4	93.4
	OW	96.4	94.5	91.9

w/o means without. IW: image-wise; OW: object-wise; HG: hourglass

Table 5 Comparison on the Jacquard dataset

Algorithm	Input	Accuracy (%)
Depierre et al. (2018)'s	RGB-D	74.2
Zhou et al. (2018)'s	RGB	92.8
Zhang et al. (2019)'s	RGB	93.6
Morrison et al. (2020)'s	D	84.0
Kumra et al. (2020)'s	D	93.7
	RGB	91.8
	RGB-D	94.6
Wang Y et al. (2021)'s	RGB	97.0
AE-GDN w/o HG	D	95.6
	RGB	93.1
	RGB-D	95.8
AE-GDN	D	95.9
	RGB	95.0
	RGB-D	96.6

w/o means without. HG: hourglass

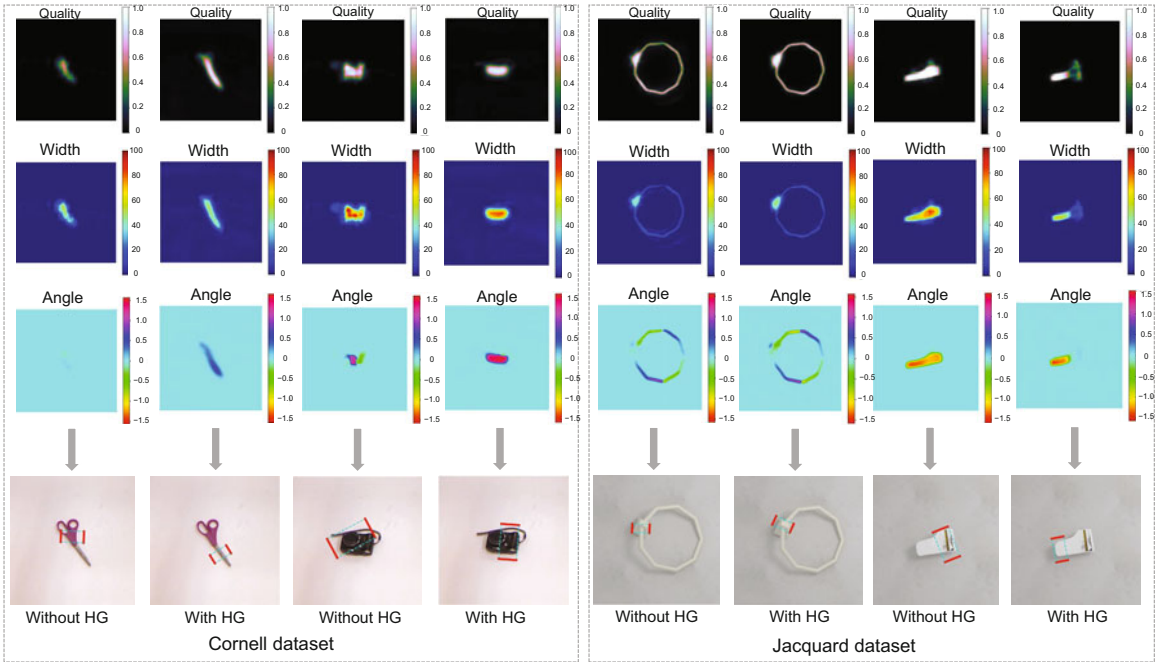

Fig. 8 Visualization results on two datasets

Quality, width, and angle are parsed into the grasping rectangle. In the two columns of the same object, the column on the right uses the hourglass (HG) box mechanism

Table 6 Accuracy on the Jacquard dataset using RGB with different Jaccard indices

Algorithm	Accuracy (%)		
	0.25	0.30	0.35
Song et al. (2020)'s	91.5	89.7	87.3
AE-GDN w/o HG	93.1	90.2	88.4
AE-GDN	95.0	92.3	91.2

w/o means without. HG: hourglass

poses. The evaluation criterion for successful grasping is that according to the grasp configurations detected by AE-GDN, the robotic arm can pick up the object and ensure that the object does not slip off.

5.2.2 System setup and implementation details

The robotic arm used in the grasp test is a UR5 collaborative robot, the end effector is a Backyard P80 gripper, and the vision sensor is an Intel RealSense D435i camera that is fixed on the end of the robotic arm. For data and signals that need to be transmitted, we use ROS (Quigley et al., 2009), which can establish communication between various program nodes. The MoveIt_ROS package is used to establish communication with the robotic arm and control the movement. The motion planning algorithm used is ESTkconfigdefault, which is named Expand the Space Tree.

Fig. 9 Objects for testing the robot grasping system. Each object has eight poses

In the practical object grasp test, the position of the object relative to the end effector of the robotic arm needs to be obtained. In this work, the ROS cv_bridge and tf_packages are used to transform the position of the object from the camera coordinate system to the robot coordinate system. The grasping experiment process is to grasp a bunch of objects from a designed area and move them to another. For the same object, multiple postures and positions are tried. The purpose is to verify the generalizability of the grasp detection algorithm to unknown objects and the stability of the robot grasping system.

5.2.3 Evaluation and analysis

We have carried out 200 robot grasping tests, 8 times for each of 25 objects, 188 of which are successful. Table 7 gives the comparison result of the robot grasping success rate between our approach and mainstream methods, and shows that the method in this work has a high success rate for grasping household objects. The approach proposed by Kumra et al. (2020) has a little higher success rate, because the objects grasped using their approach have lighter weight. Fig. 10 gives some visualization results of the practical robot grasping test, and Fig. 11 gives an example of the robot grasping process. Table 8 shows the numbers of attempted and successful grasps for different objects. It can be seen that the robot grasping system in this work performs well even for irregularly-shaped objects, but some items are too heavy, and the grasping force of the gripper and the friction of the gripper are inadequate, leading to grasping failures. For example, the plush stick has smooth surfaces and is heavy, and tends to slip during the grasping process. Another example is the wire strippers. The algorithm can detect the correct grasp rectangle on its handle; however, due to the heavy head, the wire strippers tend to tilt during the movement and may drop.

Table 7 Comparison of the success rates for grasping household objects among different approaches

Approach	Success rate (%)
Lenz et al. (2015)'s	89
Pinto and Gupta (2016)'s	73
Morrison et al. (2020)'s	92
Chu et al. (2018b)'s	89
Kumra et al. (2020)'s	95.4
Ours	94.0

The robot grasping test is a systematic experiment, and there are many reasons for the 12 grasping failures, such as the object being too large to grasp, the object being too heavy for the end effector, errors in the image acquisition step, low quality of the

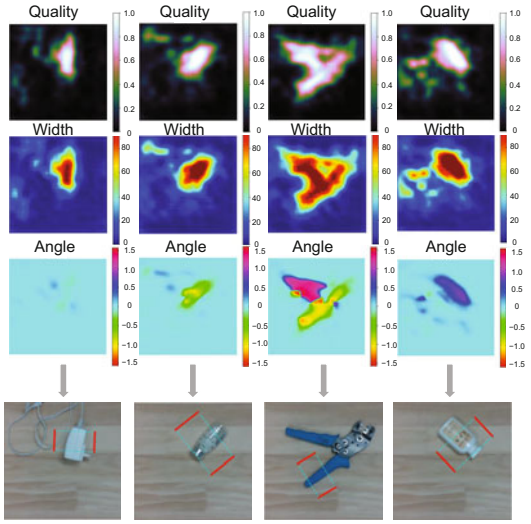

Fig. 10 Visualization results in the practical robot grasping test

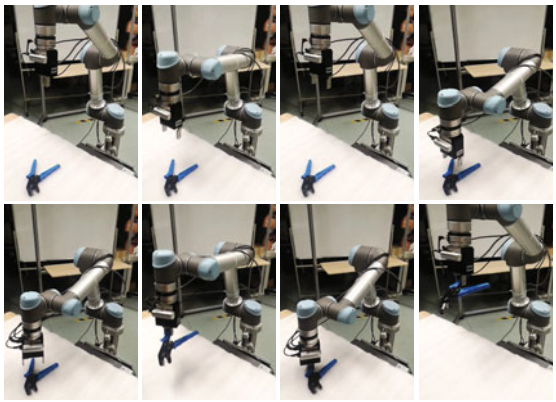

Fig. 11 A robot grasping process example. The process is from left to right, and then from top to bottom

Table 8 Grasp attempt/success table in a single-object scene

Object	Attempt/success	Object	Attempt/success	Object	Attempt/success
Plush stick	8/5	Candy bottle	8/8	Utility knife two	8/8
Plastic syringe one	8/8	Pill bottle	8/8	Charger	8/6
Screwdriver handle one	8/8	Hand cream	8/8	Tape measure	8/8
Plastic cartridge	8/8	Wire strippers	8/5	Vise	8/8
Screwdriver handle two	8/8	Little paper box	8/8	Remote	8/8
Plastic bag roll	8/8	Plastic syringe two	8/8	Hexalobe screwdriver	8/7
Marker pen	8/8	Mouse	8/8	Stapler	8/7
Nail scissors	8/8	Glue	8/8		
Utility knife one	8/8	Tin coil	8/6		

The numbers before and after "/" mean the number of attempts to grasp an object and the number of successful graspings, respectively

predicted grasp rectangle, and possible errors in the hand-eye calibration step. Therefore, it is necessary to conduct independent experimental analysis of each of the above possible reasons to improve the success rate of the robot grasping system.

The proposed AE-GDN can also be used in the multi-object scenario. When there are multiple objects in the input image, the output heatmap of grasp quality will have multiple local maxima corresponding to multiple objects. The pixel coordinates of these local maxima can be used as indices to obtain the width and rotation angle of each object. Finally, multiple grasp boxes can be parsed from the combinations of the coordinates, widths, and rotation angles, as shown in Fig. 12. After the grasp boxes are obtained, the actual grasping sequence can be arranged according to certain rules. For example, the robot can always grasp the object with the highest grasp quality score or the object closest to the left upper corner of the image until the last object is grasped. Fig. 13 gives a multi-object grasping process example. It is worth noting that if the distances between multiple objects are relatively small, although AE-GDN can output correct grasp boxes, the robot may fail to grasp some objects because the algorithm calculates the grasp box of each object independently and does not consider the collision between the grasp box and other objects, as shown in Fig. 14. Fig. 15 gives an example of a collision between the gripper and a non-target object in a multi-object scene when the distance between objects is relatively small. In addition, AE-GDN does not give the categories of multiple objects, so sequential grasping according to certain category order cannot be realized. Finally, when occlusion occurs, the overlapped objects may be treated as a single object, so AE-GDN may not output enough grasp boxes for all objects, as shown in Fig. 16.

Robots also have significant requirements for grippers to achieve grasping. Liu et al. (2023) used a soft gripper that can handle various objects of different shapes. In the future, research can be focused on grippers.

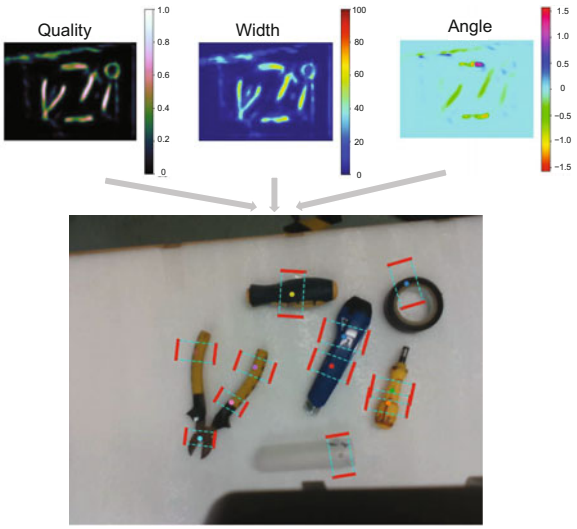

Fig. 12 Visualization results of grasp detection in a multi-object scenario

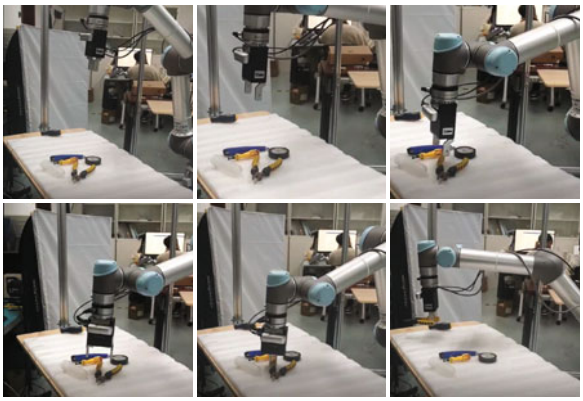

Fig. 13 A multi-object grasp process example. The grasp process is from left to right, and then from top to bottom

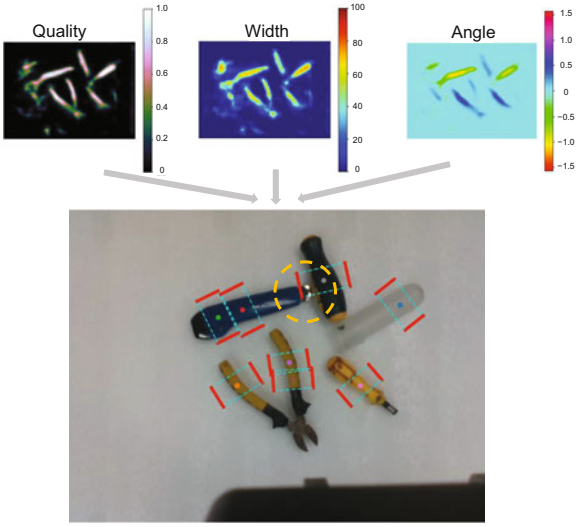

Fig. 14 Example of a collision between the predicted grasp box and another object

6 Conclusions

This paper proposes an efficient pixel-level grasp detection network AE-GDN based on an attention mechanism and an hourglass box matching

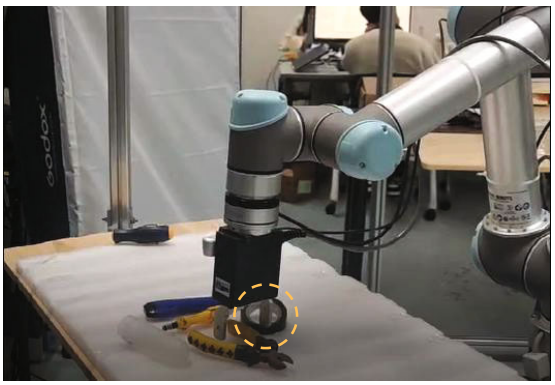

Fig. 15 Example of a collision between the gripper and a non-target object in a multi-object scenario when the distance between objects is relatively small

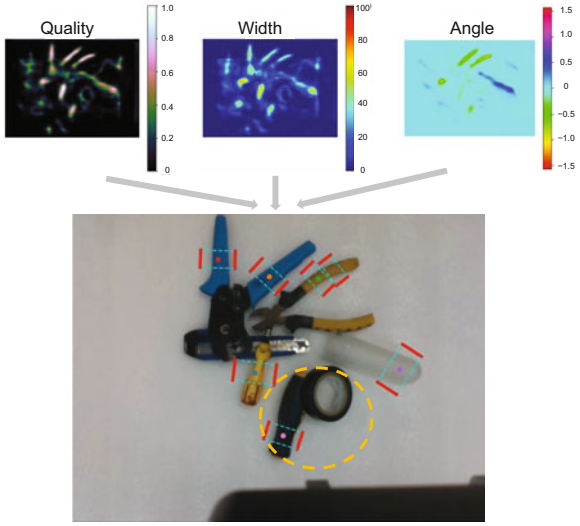

Fig. 16 Example of the algorithm not outputting enough grasp boxes for all objects

mechanism. The attention mechanism can enhance both the detailed and semantic information. The hourglass box matching mechanism creates good correspondence between high IoUs and high-quality grasp rectangles. The accuracy of AE-GDN on the Cornell and Jacquard datasets has reached 98.9% and 96.6%, respectively. The inference speed of the proposed AE-GDN can meet the real-time requirement of robot grasping tasks. The proposed AE-GDN is also deployed on a practical robot grasping system and performs grasping tasks well.

Contributors

Xiaofei QIN and Wenkai HU proposed the design. Xiaofei QIN and Chen XIAO conducted the experiments. Xiaofei QIN and Changxiang HE analyzed the results. Wenkai HU drafted the paper. Xiaofei QIN, Songwen PEI, and Xuedian ZHANG helped organize the paper. All authors revised and finalized the paper.

Compliance with ethics guidelines

Xiaofei QIN, Wenkai HU, Chen XIAO, Changxiang HE, Songwen PEI, and Xuedian ZHANG declare that they have no conflict of interest.

Data availability

The data that support the findings of this study are openly available in the repository at https://github.com/robvincen/robot_gradet.

References

Ainetter S, Fraundorfer F, 2021. End-to-end trainable deep neural network for robotic grasp detection and semantic segmentation from RGB. Proc IEEE Int Conf on Robotics and Automation, p.13452-13458. https://doi.org/10.1109/ICRA48506.2021.9561398

Asif U, Bennamoun M, Sohel FA, 2017. RGB-D object recognition and grasp detection using hierarchical cascaded forests. *IEEE Trans Rob*, 33(3):547-564. https://doi.org/10.1109/TRO.2016.2638453

Asif U, Tang JB, Harrer S, 2018. GraspNet: an efficient convolutional neural network for real-time grasp detection for low-powered devices. Proc 27th Int Joint Conf on Artificial Intelligence, p.4875-4882.

Asif U, Tang JB, Harrer S, 2019. Densely supervised grasp detector (DSGD). Proc 33rd AAAI Conf on Artificial Intelligence, p.8085-8093. https://doi.org/10.1609/aaai.v33i01.33018085

Chen L, Huang PF, Meng ZJ, 2019. Convolutional multi-grasp detection using grasp path for RGBD images. *Rob Auton Syst*, 113:94-103. https://doi.org/10.1016/j.robot.2019.01.009

Chen L, Huang PF, Li YH, et al., 2020. Detecting graspable rectangles of objects in robotic grasping. *Int J Contr Autom Syst*, 18(5):1343-1352. https://doi.org/10.1007/s12555-019-0186-2

Chu FJ, Xu RN, Vela PA, 2018a. Deep grasp: detection and localization of grasps with deep neural networks. https://arxiv.org/abs/1802.00520v2

Chu FJ, Xu RN, Vela PA, 2018b. Real-world multiobject, multigrasp detection. *IEEE Rob Autom Lett*, 3(4):3355-3362. https://doi.org/10.1109/LRA.2018.2852777

Depierre A, Dellandréa E, Chen LM, 2018. Jacquard: a large scale dataset for robotic grasp detection. Proc IEEE/RSJ Int Conf on Intelligent Robots and Systems, p.3511-3516. https://doi.org/10.1109/IROS.2018.8593950

Fang B, Long XM, Sun FC, et al., 2022. Tactile-based fabric defect detection using convolutional neural network with attention mechanism. *IEEE Trans Instrum Meas*, 71:5011309. https://doi.org/10.1109/TIM.2022.3165254

Ghazaei G, Laina I, Rupprecht C, et al., 2018. Dealing with ambiguity in robotic grasping via multiple predictions. Proc 14th Asian Conf on Computer Vision, p.38-55. https://doi.org/10.1007/978-3-030-20870-7_3

Guo D, Sun FC, Liu HP, et al., 2017. A hybrid deep architecture for robotic grasp detection. Proc IEEE Int Conf on Robotics and Automation, p.1609-1614. https://doi.org/10.1109/ICRA.2017.7989191

Hara K, Vemulapalli R, Chellappa R, 2017. Designing deep convolutional neural networks for continuous object orientation estimation. https://arxiv.org/abs/1702.01499

He KM, Zhang XY, Ren SQ, et al., 2016. Deep residual learning for image recognition. Proc IEEE Conf on Computer Vision and Pattern Recognition, p.770-778. https://doi.org/10.1109/CVPR.2016.90

Huang G, Liu Z, van der Maaten L, et al., 2017. Densely connected convolutional networks. Proc IEEE Conf on on Computer Vision and Pattern Recognition, p.4700-4708. https://doi.org/10.1109/CVPR.2017.243

Jaderberg M, Simonyan K, Zisserman A, et al., 2015. Spatial transformer networks. Proc 28th Int Conf on Neural Information Processing Systems, p.2017-2025.

Jiang Y, Moseson S, Saxena A, 2011. Efficient grasping from RGBD images: learning using a new rectangle representation. Proc IEEE Int Conf on Robotics and Automation, p.3304-3311. https://doi.org/10.1109/ICRA.2011.5980145

Karaoguz H, Jensfelt P, 2019. Object detection approach for robot grasp detection. Proc Int Conf on Robotics and Automation, p.4953-4959. https://doi.org/10.1109/ICRA.2019.8793751

Krizhevsky A, Sutskever I, Hinton GE, 2012. ImageNet classification with deep convolutional neural networks. Proc 25th Int Conf on Neural Information Processing Systems, p.1097-1105.

Kumra S, Kanan C, 2017. Robotic grasp detection using deep convolutional neural networks. Proc IEEE/RSJ Int Conf on Intelligent Robots and Systems, p.769-776. https://doi.org/10.1109/IROS.2017.8202237

Kumra S, Joshi S, Sahin F, 2020. Antipodal robotic grasping using generative residual convolutional neural network. Proc IEEE/RSJ Int Conf on Intelligent Robots and Systems, p.9626-9633. https://doi.org/10.1109/IROS45743.2020.9340777

Lenz I, Lee H, Saxena A, 2015. Deep learning for detecting robotic grasps. *Int J Rob Res*, 34(4-5):705-724. https://doi.org/10.1177/0278364914549607

Liu FK, Sun F, Fang B, et al., 2023. Hybrid robotic grasping with a soft multimodal gripper and a deep multistage learning scheme. *IEEE Trans Rob*, 39(3):2379-2399. https://doi.org/10.1109/TRO.2023.3238910

Mahler J, Liang J, Niyaz S, et al., 2017. Dex-Net 2.0: deep learning to plan robust grasps with synthetic point clouds and analytic grasp metrics. https://arxiv.org/abs/1703.09312

Morrison D, Corke P, Leitner J, 2018. Closing the loop for robotic grasping: a real-time, generative grasp synthesis approach. https://arxiv.org/abs/1804.05172

Morrison D, Corke P, Leitner J, 2020. Learning robust, real-time, reactive robotic grasping. *Int J Rob Res*, 39(2-3):183-201. https://doi.org/10.1177/0278364919859066

Park D, Seo Y, Chun SY, 2020. Real-time, highly accurate robotic grasp detection using fully convolutional neural network with rotation ensemble module. Proc IEEE Int Conf on Robotics and Automation, p.9397-9403. https://doi.org/10.1109/ICRA40945.2020.9197002

Pinto L, Gupta A, 2016. Supersizing self-supervision: learning to grasp from 50K tries and 700 robot hours. Proc IEEE Int Conf on Robotics and Automation, p.3406-3413. https://doi.org/10.1109/ICRA.2016.7487517

Quigley M, Conley K, Gerkey BP, et al., 2009. ROS: an open-source robot operating system. Proc ICRA Workshop on Open Source Software, p.5.

Redmon J, Angelova A, 2015. Real-time grasp detection using convolutional neural networks. Proc IEEE Int Conf on Robotics and Automation, p.1316-1322. https://doi.org/10.1109/ICRA.2015.7139361

Rezatofighi H, Tsoi N, Gwak J, et al., 2019. Generalized intersection over union: a metric and a loss for bounding box regression. Proc IEEE/CVF Conf on Computer Vision and Pattern Recognition, p.658-666. https://doi.org/10.1109/CVPR.2019.00075

Song YN, Gao L, Li XY, et al., 2020. A novel robotic grasp detection method based on region proposal networks. *Rob Comput Integr Manuf*, 65:101963. https://doi.org/10.1016/j.rcim.2020.101963

Wang Q, Fan Z, Seng WH, et al., 2022. Cloud-assisted cognition adaptation for service robots in changing home environments. *Front Inform Technol Electron Eng*, 23(2):246-257. https://doi.org/10.1631/FITEE.2000431

Wang Y, Zheng YT, Gao BY, et al., 2021. Double-dot network for antipodal grasp detection. Proc IEEE/RSJ Int Conf on Intelligent Robots and Systems, p.4654-4661. https://doi.org/10.1109/IROS51168.2021.9636706

Wang ZC, Li ZQ, Wang B, et al., 2016. Robot grasp detection using multimodal deep convolutional neural networks. *Adv Mech Eng*, 8(9):1687814016668077. https://doi.org/10.1177/1687814016668077

Woo S, Park J, Lee JY, et al., 2018. CBAM: convolutional block attention module. Proc 15th European Conf on Computer Vision, p.3-19. https://doi.org/10.1007/978-3-030-01234-2_1

Zeiler MD, Fergus R, 2014. Visualizing and understanding convolutional networks. Proc 13th European Conf on Computer Vision, p.818-833. https://doi.org/10.1007/978-3-319-10590-1_53

Zhang HB, Lan XG, Bai ST, et al., 2019. RoI-based robotic grasp detection for object overlapping scenes. Proc IEEE/RSJ Int Conf on Intelligent Robots and Systems, p.4768-4775. https://doi.org/10.1109/IROS40897.2019.8967869

Zhou XW, Lan XG, Zhang HB, et al., 2018. Fully convolutional grasp detection network with oriented anchor box. Proc IEEE/RSJ Int Conf on Intelligent Robots and Systems, p.7223-7230.