

Retraction

Retracted: A Secure and Efficient Multi-Object Grasping Detection Approach for Robotic Arms

Security and Communication Networks

Received 8 January 2024; Accepted 8 January 2024; Published 9 January 2024

Copyright © 2024 Security and Communication Networks. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

This article has been retracted by Hindawi following an investigation undertaken by the publisher [1]. This investigation has uncovered evidence of one or more of the following indicators of systematic manipulation of the publication process:

- (1) Discrepancies in scope
- (2) Discrepancies in the description of the research reported
- (3) Discrepancies between the availability of data and the research described
- (4) Inappropriate citations
- (5) Incoherent, meaningless and/or irrelevant content included in the article
- (6) Manipulated or compromised peer review

The presence of these indicators undermines our confidence in the integrity of the article's content and we cannot, therefore, vouch for its reliability. Please note that this notice is intended solely to alert readers that the content of this article is unreliable. We have not investigated whether authors were aware of or involved in the systematic manipulation of the publication process.

In addition, our investigation has also shown that one or more of the following human-subject reporting requirements has not been met in this article: ethical approval by an Institutional Review Board (IRB) committee or equivalent, patient/participant consent to participate, and/or agreement to publish patient/participant details (where relevant).

Wiley and Hindawi regrets that the usual quality checks did not identify these issues before publication and have since put additional measures in place to safeguard research integrity.

We wish to credit our own Research Integrity and Research Publishing teams and anonymous and named external researchers and research integrity experts for contributing to this investigation.

The corresponding author, as the representative of all authors, has been given the opportunity to register their agreement or disagreement to this retraction. We have kept a record of any response received.

References

- [1] H. Wang, J. Cheng, Y. Xu, S. Ni, Z. Yang, and J. Li, "A Secure and Efficient Multi-Object Grasping Detection Approach for Robotic Arms," *Security and Communication Networks*, vol. 2023, Article ID 7723164, 16 pages, 2023.

Research Article

A Secure and Efficient Multi-Object Grasping Detection Approach for Robotic Arms

Hui Wang ^{1,2}, Jieren Cheng ¹, Yichen Xu ^{1,2}, Sirui Ni ¹, Zaijia Yang ¹,
and Jiangpeng Li ¹

¹School of Computer Science and Technology, School of Information and Communication Engineering, Hainan University, Haikou, China

²RobAI-Lab, Hainan University, Haikou, China

Correspondence should be addressed to Jieren Cheng; cjr22@163.com

Received 18 August 2022; Accepted 10 October 2022; Published 1 June 2023

Academic Editor: Fang Liu

Copyright © 2023 Hui Wang et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Robot grasping is one of the most important abilities of modern intelligent robots, especially industrial robots. However, most of the existing robot arm's grasp detection work is highly dependent on their edge computing ability, and the safety problems in the process of grasp detection are not considered enough. In this paper, we propose a new robotic arm grasping detection model with an edge-cloud collaboration method. With the scheme of multi-object multi-grasp, our model improves the mission success ratio of grasping. The model can not only complete the compression of full-resolution images but also achieve image compression at a limited bit rate. The image compression ratio reaches 2.03%; the structural difference value is higher than 0.91, and our average detection speed reaches 13.62 fps. Furthermore, we have packaged our model as a functional package of the ROS operating system, which can be easily used in actual robotic arm operations. Our solution can be fully applied to other work of robots to promote the development of the field of robotics.

1. Introduction

Grasping ability is one of the most important abilities of modern intelligent robots, especially for industrial robots, which will bring great power to society [1]. As the most common basic action of robots in work, robotic autonomous grasping has great application prospects. Because of its significance, robotic autonomous grasping has been studied for a long time. Recently, robot grasping has made rapid progress due to the rapid development of deep learning. There are many tasks in robot grasping, including object localization, pose estimation, grasp detection, and motion planning. Among these tasks, grasp detection is a key task in the computer vision and robotics disciplines and has been the subject of considerable research.

However, there are still numerous challenges to this task. On the one hand, the algorithm requires hardware computing power. With the widespread use of deep learning algorithms in grasp detection, deep learning models are deployed directly at

the edge (robotic arms). And the hardware computing power is often not well executed, leading to delays and errors in data processing and grasp configuration. At present, most of the robotic arm's grasp detection work is calculated directly at the edge, only with the help of local computing power. This leads to the low efficiency of image detection, and cannot meet the requirements of automatic grasp. On the other hand, security issues in the process of grasp detection are often ignored, leading to the leakage of critical information. In recent years, there are also some studies that try to use cloud computing to solve the problem of insufficient local computing power. They upload the image data directly to the cloud (or fog), and with the help of the cloud's powerful computing power, this way greatly improves the efficiency of grasping. However, the direct transmission of data may lead to the problem of privacy leakage, while the transmission of real-time RGB images is often a major challenge for network bandwidth.

In this work, we propose a robotic arm grasping detection model with an edge-cloud collaboration method. Figure 1

shows the execution flow of our technology model. We use an encoder to compress the images grasped by the camera locally and upload them to the cloud. The uploaded encoded information does not occupy local computing resources, and since it occupies less bandwidth and requires less network configuration, it is more suitable for real scenarios' deployment. In the cloud, our model reconstructs the image by a corresponding decoder, after which it performs a two-stage multi-object grasp detection and returns the obtained grasp configuration to the local side.

The encoding and decoding network of our model is implemented by a GAN (Generative Adversarial Network), which consists of a generator and a discriminator. The generator continuously learns the real image distribution and generates a more realistic image to fool the discriminator. At the same time, the discriminator needs to discriminate the authenticity of the received images. Through the constant confrontation between the generator and the discriminator, they form a min-max game; both sides continuously optimize themselves during the training process until they reach equilibrium. Compared with other methods, GAN can achieve compression for full-resolution images and compression for images with extreme code ratios, which has wide applicability. Also, the reconstructed images have sharper textures and get better picture results. In our model, the decoder is used as the generator and is trained together with the encoder. The customization of the model is very flexible. Besides, it can set the compression ratio by adjusting the feature map size and the number of channels before and after compression. When working, the encoder will be reserved locally, and RGB images will be extracted as feature maps for compression and upload. In the cloud, the images will be reconstructed by the decoder.

The main contribution of this paper is to propose a safe and efficient multi-object grasping detection scheme for robotic arms. This scheme has three advantages:

- (1) High fidelity: We have achieved good results on DIV2K, flickr30k, Cornell, and OCID datasets. The compression ratio can be achieved, and the structural loss of the reconstructed image after the transmission is less than 7%, and there is almost no difference in the result of grasp detection before and after compression.
- (2) Strong security: Transmitting the compressed tensor to the server instead of the original image. This method avoids the leakage of production information or privacy. Compared with traditional image compression algorithms such as JPEG and JP2000, the uploaded data is difficult to be decrypted and is highly reliable. Theoretically, without the corresponding decoder parameters, there is no way to reconstruct the picture even if the transmission information is intercepted.
- (3) High execution efficiency: First, the local side of the operation is offloaded in the cloud, and the limited local arithmetic power is complemented by the arithmetic power provided by the cloud. Second, the compressed information occupies less bandwidth and is transmitted faster. Third, the lightweight neural network fits the actual application scenario.

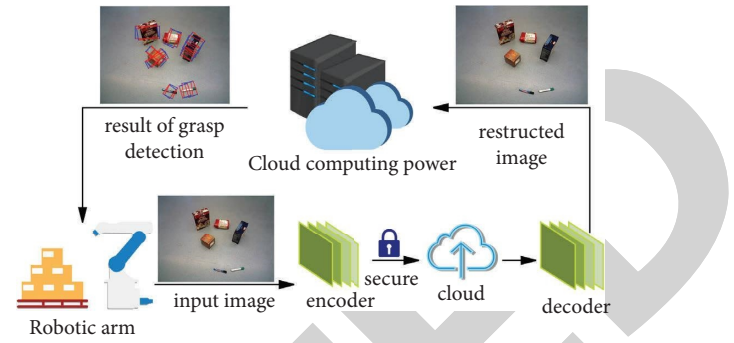


FIGURE 1: The figure shows how the robot arm unloads the local grasp detection task to the cloud. Our model realizes secure and high-fidelity transmission through this encoder-decoder structure. The image is collected locally, and transmitted to the cloud after being compressed. The reconstructed image will be obtained by the decoder in the cloud. Use cloud computing capabilities to assist in grasp detection and return the results to the robot arm.

2. Related Work

The method of achieving automatic grasping of the robotic arm has been improving over the course of long-term research. The traditional methods of perception-based grasping, reconstructing 3D models of objects, and analyzing the geometric features and forces of models, it has gradually expanded to the use of deep learning network models for image object detection and pose estimation [2].

The work uses the CNN (FAST R-CNN VGG16) network model to complete the pose estimation after image detection. This work proves the practicality of the object in the case of obscuration through experiments [3]. Another work proposes a multimodal model method for image detection using ResNet for RGB, which has better performance than VGG16 [4]. Others use deep learning networks to calibrate and control the behaviour of robotic arms.

Leoni's work is based on the RNN network model; through the sensor data to learn and train the robot's grasping behaviour, thus making sure the system can achieve the goals [5].

Several works use RL technology to optimize and train a robot's gripping ability. After a lot of training, these methods have achieved good experimental results in limited scenes. However, in more complex and practical scenarios, the scalability of RL is still unknown [6].

It is worth noting that the work of Chu et al. [7] on multi-object grasping detection has achieved good results in recent years. Our work is based on the model they proposed.

Due to the demand for computing power in deep learning, the use of cloud edge fog computing is also more applied in robot-related fields. For example, in the work of Sarker et al. [8] the use of offload cloud computing work reduces the energy consumption and hardware requirements of the robot. This treatment reduces a lot of pressure on the hardware part of the robot and the robotic arm.

Kumar et al. [9] builds a cloud computing framework. Through this framework, any robot can call the infinite computing power of the cloud to calculate. Deng et al. [10] proposed a set of invocation algorithms for fog computation. This method can allocate resources more reasonably and efficiently in a limited computing power environment that is closer to the actual situation.

The processing of cloud edge fog often relies on the stability of the connection and relatively high bandwidth. And in practical application scenarios, the compression of images is an essential part.

Some traditional image compression algorithms can achieve certain results in conventional scenes. Dhawan's summary had already analyzed the advantages and disadvantages of methods such as JPEG. However, this method does not present a good direction for further improvement [11].

Compared with traditional algorithms, the direction of image compression using deep learning has yielded many results.

Johannes et al. use the CNN network as a decoder to deal with image compression problems and obtain good theoretical data. This method is processed by the convolutional neural network, which reduces the amount of both computation and image compression data [12]. But in the case of practical applications, end-to-end joint optimization is often difficult to complete high-effect compression and high-quality reconstruction of the image at the same time.

In addition to this, the limited sensory field of the convolutional kernel makes the training often fail to achieve the expectation. This is because the achievement of full-resolution compression tends to increase the difficulty of training network structures.

Toderici et al. use the LSTM network model and the CNN + RNN network model for image compression [13, 14]. And the network model built by using the LSTM network framework is more robust for different pictures. However, experiments have shown that the training of the model is quite complex. Besides, the image correlation relationship cannot be well grasped, and it can only be limited to small-size pictures.

On the other hand, we studied the application of VAE networks in image compression. By increasing the mass ratio factor of VAE, linear proportion and other methods achieved a fairly good compression effect [15, 16]. However, since VAE networks learn the general and original picture by calculating the mean squared error, the resulting image is more likely to have the edge blur problem.

Rippel et al. was the first to propose the application of GAN networks to image compression [17]. The decoded data is processed and generated by using a GAN network, and it is opposed to the discriminator supported by the real data. The model can not only complete the compression of full-resolution images but also achieve image compression at a limited bit rate. This results in a reconstructed image with a clear texture for better visual sensory effects.

A large number of applications of cloud, edge, and fog computing systems have spawned a very urgent information security problem. In [18], for example, the authors analyze

the data security issues posed by cloud computing. In addition, the review of Randeep and Jagroo [19] points to the security issues that cloud computing can bring. They summarize techniques for overcoming data privacy issues and define pixel key patterns and image steganography techniques for overcoming data security issues.

Some work [20–22] discussed the security of medical information in cloud storage and data sharing environments and gave some feasible solutions. Overall, these studies highlight the security of information (communications) under the cloud computing system.

To sum up, most of the existing robot arm's grasp detection work is highly dependent on their edge computing ability, and the safety problems in the process of grasp detection are not considered enough.

3. Methodology

The RGB image is grasped by the local camera and sent out by the edge side after encoding and compression. The cloud side receives the data, and then the decoder reconstructs the image for grasp detection. The parameters of the encoder and decoder are obtained using generative adversarial networks for training. Two tasks are completed in the grasp detection phase: grasp proposals and grasp configuration, the former determines the location of the object and the latter configures the grasp angle. The system flowchart is shown in Figure 2 and comprises a number of components, which we will be introduced below.

3.1. Image Compression Part. In this section, we will focus on feature extraction, network architecture design, and customized loss function.

3.1.1. Feature Extraction and Compression. Our model uses global generative compression for image compression. Before encoding and decoding, the input image is first passed through two layers of convolution to achieve feature extraction and image compression. We found that by adjusting the number of feature channels and feature map size output here, we could not only balance the processing speed and image compression quality, but also easily change its compression ratio.

We preprocess the image so that the input image is an RGB image with a height of 210 and a width of 150. The encoded image obtained by the encoder is a feature map of 52×37 of 2, 4, 8, and 16 channels; the corresponding compression ratios are 32.58%, 16.29%, 8.14%, and 4.07%, respectively. The calculation of the compression ratio is given by equation (1). It represents the ratio of the parametric quantities of the output tensor $R^{C_c \times H_c \times W_c}$ to the input image $R^{C_i \times H_i \times W_i}$. The reconstructed images are similar to the original images, whose structural similarity index is greater than 0.93.

$$\text{Compression ratio} = \frac{C_c \times H_c \times W_c}{C_i \times H_i \times W_i} \times 100\%. \quad (1)$$

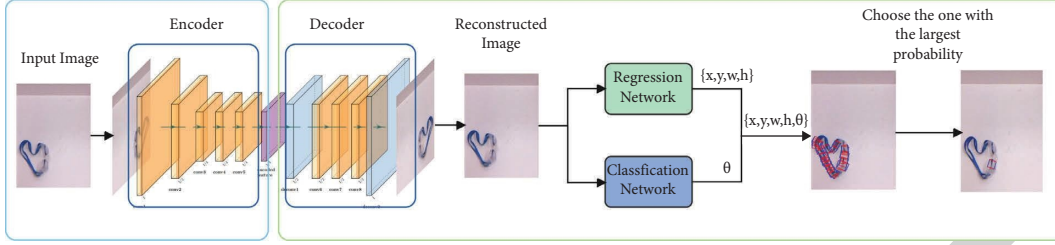


FIGURE 2: The figure shows the general technical flowchart of our approach. The input image is collected at the edge, compressed by the encoder, and then uploaded to the cloud. The image will be reconstructed by the decoder in the cloud. Then grasp parameters are obtained through classification and regression networks to get the bounding box. According to the probability, the most likely bounding box is selected as the final result of grasp detection. The blue box and green box in the figure represent the edge end and cloud end, respectively.

The number of parameters for different compression ratios is shown in Table 1, and the detailed results under different compression ratios will be given in the experimental section. In Figure 3, we show the reconstructed image results under different compression ratios.

3.1.2. Network Architectures. In order to make the network structure as simple as possible, here we have built a lightweight generator advertising network that is similar to DCGAN [23]. The network consists of a generator and a discriminator. It uses a decoder as a generator and trains the encoder and decoder by using the same loss function in training. During training, the goal of the generator is to try to generate real images to deceive the discriminator. And the goal of the discriminator is to try to separate the images generated by the generator from the real images and then paste the 0 and 1 labels, respectively.

After the encoding stage, we only upload the tensor generated by the encoding network to the cloud without anything else. On the cloud, we use the decoder to restore the tensor to a reconstructed image. In the encoder (compressor network), we used three consecutive layers of simple residual layers (ResNet [24]) for encoding. Correspondingly, in the decoder (decompressor network), the two upsample and three layers of residual layers are crossed, and eventually received a reconstructed image. We implemented upsample with transposed convolution and restored the dimensions of the output picture. In the encoding and decoding network, we use LeakyReLU as the activation function and use Tanh in the last layer. In the convolution block during the encoding and decoding phases, we keep the size of the feature map constant by setting the stride and padding, which reduces the loss of information. For the discriminator, we built a simple model based on a combination of convolution and dropout layers.

3.1.3. Loss Function. Generally, in GAN, we tend to use L1 loss (MAE) and L2 loss (MSE) to train discriminators for binary classification problems. However, it cannot be ignored that the simple use of L1 loss for judgment often fails to accurately reflect the level of detail of the image compression and restoration. Structural loss is also a structural loss consideration in image compression tasks. To consider both, we divide the loss function into two parts, namely, the

TABLE 1: Number of parameters before and after image compression.

Input image	Compressed tensor	Compression ratio (%)
94500	30784	32.58
94500	15392	16.29
94500	7296	8.14
94500	3648	4.07

adversarial loss and the structural loss weighting add up to the final loss function.

There are many types of loss functions based on deep learning image algorithms, such as L1 loss and L2 loss. However, for image compression and restoration work, these two loss functions are not easy to recover for the detailed structure of the image and are not enough to intuitively express people's cognitive feelings. In addition, there is also PSNR (peak signal-to-noise ratio) as a common evaluation criterion, but it has a common problem with L1 and L2: their principle is based on pixel-by-pixel comparison differences, without considering human visual perception, so the PSNR index is high, not necessarily representing image quality.

So here we use MSSIM [25] as a structural loss, which is based on SSIM. SSIM is a commonly used image quality evaluation index, which is based on the assumption that the human eye will extract the structural similarity variables when viewing the image. Its final loss value is obtained by comprehensively considering the brightness, contrast, and structural similarity variables. For images x and y , their SSIM is calculated as follows:

$$l(x, y) = \frac{2\mu_x\mu_y + C_1}{\mu_x^2 + \mu_y^2 + C_1}, \quad (2)$$

$$c(x, y) = \frac{2\sigma_x\sigma_y + C_2}{\sigma_x^2 + \sigma_y^2 + C_2}, \quad (3)$$

$$s(x, y) = \frac{\sigma_{xy} + C_3}{\sigma_x\sigma_y + C_3}. \quad (4)$$

In equation (2)–(4), $l(x, y)$ is used to estimate luminance by mean, $c(x, y)$ is used to estimate contrast with variance, and $s(x, y)$ is used to estimate structural similarity with

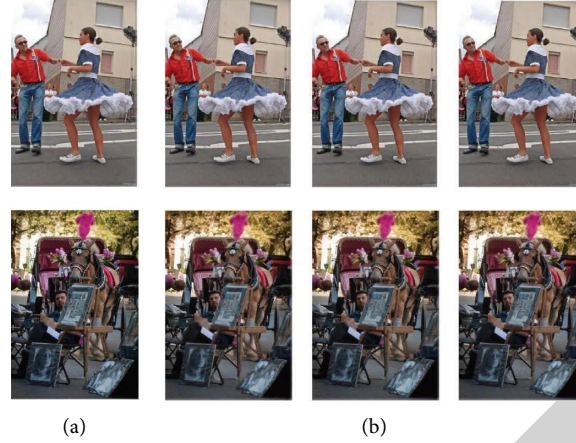


FIGURE 3: Results of reconstructed images: (a) the column is the original image and (b) the three columns are the reconstructed images with 32.58%, 16.29%, and 8.14% compression, respectively.

covariance. The SSIM definition is shown in equation (5), where α , β , and γ are used to adjust the weights of each portion. By default, we set all three of them to 1, and then, we can get equation (6)

$$\text{SSIM}(x, y) = [l(x, y)]^\alpha \cdot [c(x, y)]^\beta \cdot [s(x, y)]^\gamma, \quad (5)$$

$$\text{SSIM}(x, y) = \frac{(2\mu_x\mu_y + C_1)(2\sigma_{xy} + C_2)}{(\mu_x^2 + \mu_y^2 + C_1)(\sigma_x^2 + \sigma_y^2 + C_2)}. \quad (6)$$

MSSIM takes the reference image and the distortion image as input and divides the image into N blocks by sliding window. Then it will weight the mean, variance, and covariance of each window, and the weight w_{ij} meets the $\sum_{i=0}^n \sum_{j=0}^n w_{ij}$. We usually use the Gaussian kernel to calculate the structural similarity SSIM of the corresponding block and use the average value as a final structural similarity measure of the two images. Let's suppose the original image is scale 1, and the highest scale is scale M obtained by the $M-1$ iteration. For the j^{th} scale, only the contrast $c(x, y)$ and the structural similarity $s(x, y)$ will be calculated. Brightness similarity $l(x, y)$ is calculated only in Scale M . The final result is to link the results of the various scales:

$$\text{MSSIM}(x, y) = [l_M(x, y)]^{\alpha_M} \cdot \prod_{j=1}^M [c_j(x, y)]^{\beta_j} \cdot [s_j(x, y)]^{\gamma_j}. \quad (7)$$

Hang et al. [26] demonstrates the quality of these loss functions through three experiments. It shows that MSSIM is more appropriate in comparison. In order to make the output image gain higher quality and easier training, here we use the loss function combined with MSSIM and L1 loss.

3.2. Grasp Detection Part. The entire grasp detection task is divided into two tasks: grasp proposals and grasp configuration. The former determines the location of the object, and the latter configures the angle of the grasp.

3.2.1. Grasp Proposals. Grasp proposals are implemented by using the two-stage detection algorithm and consist of two branches: regression and classification. The model chose the ResNet-50 network as the backbone of model. First, the location of the bounding box is determined by regression, which generates the region proposals, avoiding the time-consuming sliding window method and directly predicting the region proposals on the entire image.

These region proposals will make feature extraction through RPN (Region Proposal Network) [27], and the region frame proposals classification is completed when the region frame proposals extraction is performed. The classification process classifies region features into background and object.

When the RPN network generates a region proposal, the position of the object is preliminarily predicted. During this time, the two links of regional classification and location refinement are completed. As soon as it obtains the region proposals, the ROI pooling layer will accurately refine and regress the position of the region proposals.

After the region target corresponds to the features on the feature map, the characteristics of the region proposals will be further represented through a fully connected layer. Later, the category of the region target and the refinement of the region target position will be completed by classification and regression, so the real category of the object will be obtained. While the regression will get the specific coordinate position of the current target, which is represented as a rectangular box represented by four parameters.

3.2.2. Grasp Configuration. The determination of the grasp configuration is achieved through classification. Grasp orientation coordinate θ divides the direction of the grasp into 20 classes and chooses the class directly with the highest confidence level to grasp.

There is a non-grasp direction class in classes. If the confidence level of the output is lower than that of the non-grasp direction class, this grasp recommendation is considered to be ungraspable in that direction. Setting non-

grasp classes instead of setting specific thresholds will be a better way to handle multi-object and multi-grasp component tasks.

The final output is shown in Figure 4. In the figure's output bounding box, the red line represents the open length of a two-fingered gripper, while the blue line represents the parallel plates of the gripper.

In this scheme, the loss function is designed to be two parts: the grasp proposal loss L_{gpn} and the grasp configuration loss $L_{\text{gcr_reg}}$. As shown in equation (7), $L_{\text{gp_cls}}$ is the cross-entropy loss of the grasp direction classification, and the $L_{\text{gp_eg}}$ and weight λ are the L1 regression loss of the grasp recommendation. In the case of no grasping, $p_i^* = 0$. Correspondingly, the $p_i^* = 1$ when it can be grasped. The parameter t_i^* , and p_i^* are corresponding to the ground truth.

$$L_{\text{gpn}}(\{(p_i, t_i)_{i=1}^I\}) = \sum_i L_{\text{gp_cls}}(p_i, p_i^*) + \lambda \sum_i p_i^* L_{\text{gp_reg}}(t_i, t_i^*). \quad (8)$$

Equation (8) defines the loss function that executes the fetch configuration prediction. In this equation, $L_{\text{gp_cls}}$ is the cross-entropy loss of the grasp orientation classification, and ρ_l is the confidence level of each classification. $L_{\text{gcr_reg}}$ is the regression loss of the bounding box, and the β_c records the corresponding prediction of the grasp bounding box. β_c^* is the correct bounding box. λ_2 is the relative weight.

$$L_{\text{gpn}}(\{(\rho_l, \beta_l)_{l=1}^C\}) = \sum_l L_{\text{gcr_cls}}(\rho_l) + \lambda_2 \sum_c 1_{c \neq 0}(c) L_{\text{gcr_reg}}(\beta_c, \beta_c^*). \quad (9)$$

The total loss consists of the addition of L_{gpn} and $L_{\text{gcr_reg}}$, as shown in equation (9):

$$L_{\text{total}} = L_{\text{gpn}} + L_{\text{gcr}}. \quad (10)$$

4. Experimental

4.1. Experimental Environment. The training environment of the model is an Intel (R) Xeon (R) platinum 8255c, 47 GB memory, 12 cores computer equipped with 24 g video memory GeForce RTX™ 3090 graphics card. The computer system environment is the Ubuntu 20.04 operating system. Later, the test experiment was conducted on another GeForce RTX™ 2080 Ti graphics card.

4.2. Dataset and Data Preprocessing. We used the Flickr30k [28] dataset alone for training the image compression reconstruction, and then validated on all four datasets, Flickr30k, DIV2k [29], Cornell [30], and OCID [31]. The image reconstruction achieved good results on both PSNR and SSIM values. The grasping training and validation were then performed using the OCID dataset with 92% accuracy in general.



FIGURE 4: Results of grasp detection output. The red line represents the open length of a two-fingered gripper, while the blue line represents the parallel plates of the gripper.

Flickr30k: Flickr30k is the first image description dataset that contains 158,915 descriptions and 31,783 images. This dataset is based on the previous Flickr8k dataset and focuses on describing everyday human activities. Of these, 25,426 images were used for training, and 6,357 images each were used for validation and testing.

Div2k: The DIV2K dataset is a commonly used dataset for superresolution image reconstruction. The dataset contains 1000 2K resolution images, including 800 training images, 100 validation images, and 100 test images. And the low-resolution images with 2, 3, 4, and 8 reduction factors are provided.

Cornell: The Cornell grasping dataset is a required dataset for robotic autonomous grasping tasks. The dataset contains 885 RGB-D images of 640×480 px size with 240 graspable objects. The correct grasping candidate is given by a manually annotated rectangular box. Each object corresponds to multiple images with different orientations or poses, and each image is labelled with multiple ground truth grasps, corresponding to the many possible ways of grasping the object.

OCID: We use the OCID_grasp dataset part, which is composed of 1763 selected RGB-D images, of which there are more than 75,000 hand-annotated grasp candidates.

4.3. Training Schedule. We train the whole network for 10 epochs on a single GeForce RTX™ 3090. The initial learning ratio is set to 0.0002. The batch size is set to 30, and the log is output every 50 batches. The input image is first cropped to 210×160 sizes.

4.4. Evaluation Metric

4.4.1. Compressed Image Quality Metrics. PSNR (peak-signal-to-noise ratio) PSNR is defined as equation (11), MAX_I^2 which is the maximum possible pixel value of the image, and MSE is the mean square error of each pixel point of the two images. The minimum value of PSNR is 0, and the larger the PSNR, the smaller the difference between the two images. We test 100 images and finally take the average as the final value.

$$\text{PSNR} = 10 \cdot \log_{10} \left(\frac{\text{MAX}_I^2}{\text{MSE}} \right). \quad (11)$$

SSIM (Structure Similarity Index Measure) equation (6) is the definition of SSIM. SSIM is based on the assumption that the human eye extracts structured information from an image and integrates the differences between two images in terms of luminance, contrast, and structure. $SSIM \leq 1$, the larger the SSIM, the more similar the two images are. We test 100 images and finally take the average as the final value.

4.4.2. Grasping Accuracy Metrics. The accuracy of the grasping parameters is evaluated by comparing the closeness of the grasp candidate to ground truth.

A grasp candidate is considered as successful grasp detection after satisfying the following two metrics:

- (1) The difference between the angle of predicted grasp g_p and ground truth g_t does not exceed 30°
- (2) Intersection over Union (IoU) of g_p and g_t is greater than 25%, which means

$$IoU = \frac{|g_p \cap g_t|}{|g_p \cup g_t|} > 0.25. \quad (12)$$

4.5. Comparative Experiment

4.5.1. Image Compression Quality Experiment. We conducted compression encoding experiments on the pictures of the Flickr30k, Cornell, and DIV2K datasets, respectively. The encoding tensor sizes obtained under different datasets and different compression ratios are shown in Table 2. The data in Table 2 shows that the magnitude of the compression tensor is proportional to the compression ratio and satisfies the previously derived formula to present a linear relationship.

We select 200 images from each of the three datasets of the Flickr30k, Cornell, and DIV2K, and divide them into 2 : 3 batches according to the complexity of the images. The reconstructed image is compared with the original image at different compression ratios. The reconstructed image is compared with the original image at different compression ratios. We get their PSNR and SSIM values and average them to get Tables 3 and 4. The data in Tables 3 and 4 show that our model has achieved good results under picture inputs of different complexity. The average values of PSNR and SSIM reached 35.576 and 0.948, respectively.

In Tables 3 and 4, we use eleven compression ratios to test the image compression and reconstruction effects under different compression ratios. The results show that when the compression ratio is above 4.07%, the accuracy will not decrease too much with the decrease of the compression ratio. When the compression ratio is 2.03% or less, the loss gradually manifests. The results show that our model has strong feature extraction ability and a large range of customizable compression ratios.

With the increase of the compression ratio, the image reconstruction quality increases sublinearly and finally tends to a higher value. From the comparison of eleven groups of values, it can be seen that by weighing the compression ratio

TABLE 2: Output tensor size of the encoder with different compression ratios.

Compression ratio (%)	Flickr30k (kB)	Cornell (kB)	DIV2K (kB)
2.03	15.5	74.8	74.8
4.07	30.3	148	148
8.14	60	297	297
16.29	119	593	593
32.58	237	1187	1187

and image quality, from the image reconstruction quality index, 8.14% and 16.29% are the best compression ratio settings of the network. The data in the table shows that the SSIM value of image reconstruction on the three datasets is greater than 0.82 under these two ratios. The Cornell dataset in the actual grasping environment has the highest score, with a PSNR of 31.768 and an average SSIM of 0.948, which is sufficient to meet the needs of grasping. However, in the actual process of grasping and detecting, the requirements for images are not the same as those for human eyes. We will conduct further experiments in combination with grasping in the two experiments of the grasp detection accuracy experiment and the network architecture experiment.

4.5.2. Grasp Detection Accuracy Experiment. In order to evaluate the effect of encoding and decoding on grasp detection, we compared the results of grasping detection using the original image and the reconstructed image. The results are shown in Figure 5.

We can see from Figure 5 that under the compression ratio of 8.14%, the accuracy does not decrease too much after being compressed. At the same time, the processing speed of our grasp detection algorithm can reach 13.62 fps in the implementation environment.

The Cornell dataset provides images and grasp labels from multiple angles of each object. We carry out the grasp detection experiment based on the same object from multiple angles. Figure 6 shows the effect. Our model can accurately mark the bounding box at different angles.

We detect the accuracy of the multi-object grasp task in the environment of a single object, less than ten objects, and more than ten objects. Count the number of successful grasp detections and calculate the grasp accuracy. The results are shown in Table 4. The results show that when the number of objects is less than 5, our model can basically achieve 100% error-free detection on the OCID dataset. Figure 7 shows the performance of our model on the OCID dataset.

4.5.3. Network Architectures Experiment. In order to reasonably design the parameters of the neural networks, we carried out parameter optimization experiments from the two dimensions of network depth and the number of channels.

We designed different models with two, three, and five convolution blocks of network architectures respectively for image reconstruction experiments. The results are shown in Figure 8 and Table 5, to compare the effect of the number of

TABLE 3: PSNR of different dataset.

Compression ratio (%)	DIV2K			Flickr30K			Cornell		
	High complexity	Low complexity	Average	High complexity	Low complexity	Average	High complexity	Low complexity	Average
32.58	24.48	30.31	27.98	21.31	29.78	26.392	34.785	36.10	35.5
16.29	24.885	30.26	28.108	22.29	29.56	26.652	31.41	32.01	31.768
8.14	24.16	29.31	27.248	21.82	28.31	25.714	29.93	30.41	30.216
4.07	22.265	27.98	25.692	19.7	25.85	23.39	31.57	32.66	32.226
2.03	19.515	24.44	22.472	19.52	24.44	22.472	26.875	27.91	27.496
0.99	18.255	22.67	20.9	17.31	22.34	20.328	29.535	30.44	30.078
0.50	15.55	20.51	18.528	15.91	19.02	17.776	24.38	26.86	25.866
0.13	15.175	18.92	17.422	14.90	18.45	17.03	22.49	24.26	23.552
0.06	16.225	21.21	19.218	15.46	18.49	17.276	19.475	19.11	19.254
0.04	13.09	16.41	15.084	14.26	17.91	16.446	11.915	11.43	11.626
0.03	10.875	14.99	13.344	12.24	13.59	13.05	11.915	11.43	11.626

TABLE 4: SSIM of different dataset.

Compression ratio (%)	DIV2K			Flickr30K			Cornell		
	High complexity	Low complexity	Average	High complexity	Low complexity	Average	High complexity	Low complexity	Average
32.58	0.83	0.86	0.86	0.75	0.85	0.81	0.945	0.95	0.948
16.29	0.835	0.89	0.868	0.78	0.86	0.828	0.945	0.95	0.948
8.14	0.805	0.87	0.846	0.75	0.82	0.792	0.945	0.95	0.948
4.07	0.66	0.81	0.752	0.64	0.87	0.778	0.935	0.94	0.938
2.03	0.48	0.7	0.612	0.51	0.78	0.672	0.905	0.92	0.914
0.99	0.545	0.61	0.586	0.47	0.59	0.542	0.885	0.89	0.888
0.50	0.365	0.46	0.422	0.49	0.6	0.556	0.845	0.87	0.858
0.13	0.33	0.43	0.39	0.35	0.48	0.428	0.825	0.83	0.83
0.06	0.405	0.49	0.458	0.37	0.53	0.466	0.77	0.80	0.79
0.04	0.27	0.34	0.312	0.305	0.478	0.408	0.555	0.56	0.556
0.03	0.225	0.29	0.262	0.23	0.35	0.302	0.555	0.56	0.556

encoder-decoder network layers on the model effect. By comparing these figures and tables, it can be concluded that the reconstructed image of the three-layer convolution block model is better than that of the two-layer coding block network. However, for the five-layer network, considering the operation speed and guarantee ratio, we think that three layers are the better network layers.

Comparison of reconstructed image quality under different channel numbers is shown in the appendix. The image is blurred at a low compression ratio, but it can still be detected and judged. With the increase in compression ratio, the result of grasp detection is close to the original image. The three lines from top to bottom are the input image, the reconstructed image, and the result with the label. It can be concluded from these five groups of pictures that the compression ratio of 0.13% and above is similar to the original picture, which can ensure the accuracy of grasping.

Cornell datasets are all composed of a single object target, which is less difficult to grasp. To further refine our choice of compression ratio, we performed the same experiment on the multi-object grasp dataset, OCID. In the case of fewer objects and fewer stacking occlusions, the grasping detection accuracy does not change too much with the reduction of the compression ratio. However, when the

number of objects increases and numerous stacks appear, the influence of different compression ratios on the results gradually appears. As shown in Figure 9 the first line is the result of the grasp detection of the original image, and the eight lines below are the results under the compression ratios of 16.29%, 4.07%, 0.99%, 0.5%, 0.13%, 0.05%, 0.04%, and 0.03% (additional images are shown in figure X in the appendix). When the compression ratio is relatively large, the reduction of the compression ratio does not mean the reduction of the accuracy. In some cases, the interference of impurities may even be eliminated to improve the accuracy of grasping detection. However, we can clearly see that when the compression ratio is reduced to 0.5%, it is difficult to distinguish the stacked occluded objects. When the compression ratio reaches 0.03% of the limit, it is impossible to perform grasping detection. Therefore, we think that the compression ratio of at least 0.5% should be selected for multi-object grasp detection.

In conclusion, we can draw a conclusion. In the case of a single object or objects without stack occlusion, the compression ratio of 0.13% or above has high accuracy. In a complex scene where multiple objects or objects are stacked and blocked, a compression ratio of 0.5% or above is required.

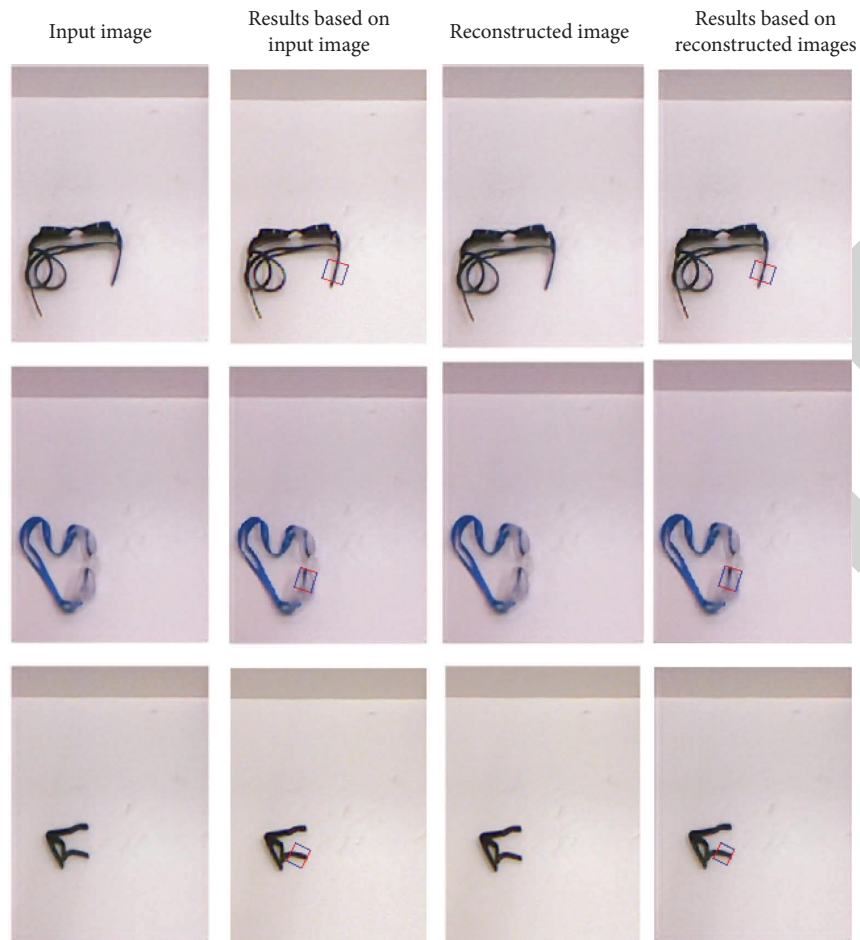


FIGURE 5: The first column is the original input pictures. The second column is the result of grasp detection based on the original image. The third column shows the compressed and reconstructed image. The fourth column is the result of the grasp detection based on the compressed and reconstructed image. It can be seen from the figure that the loss of compression accuracy is little, and grasping accuracy has not been greatly affected.

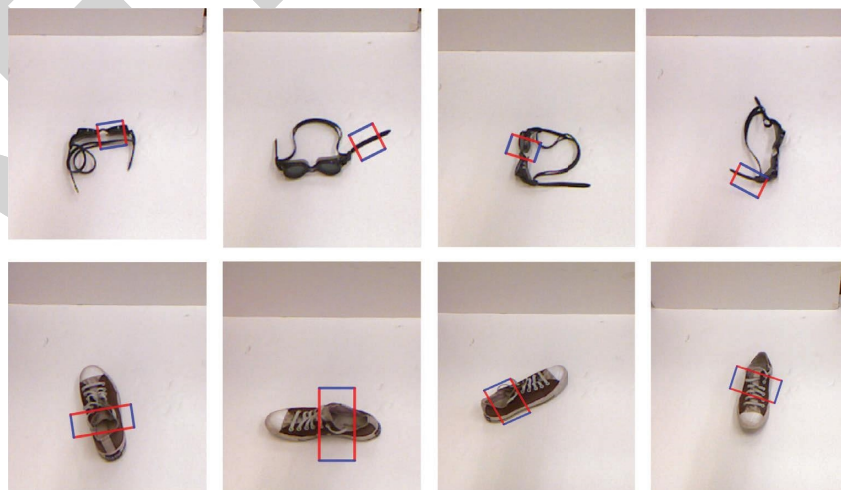


FIGURE 6: Grasp detection experiment based on the same object from multiple angles. The figure shows that our model can accurately mark the bounding box in different directions.



FIGURE 7: Performance of our model on the OCID dataset. In the output bounding box, the red line represents the open length of a two-fingered gripper, while the blue line represents the parallel plates of the gripper.

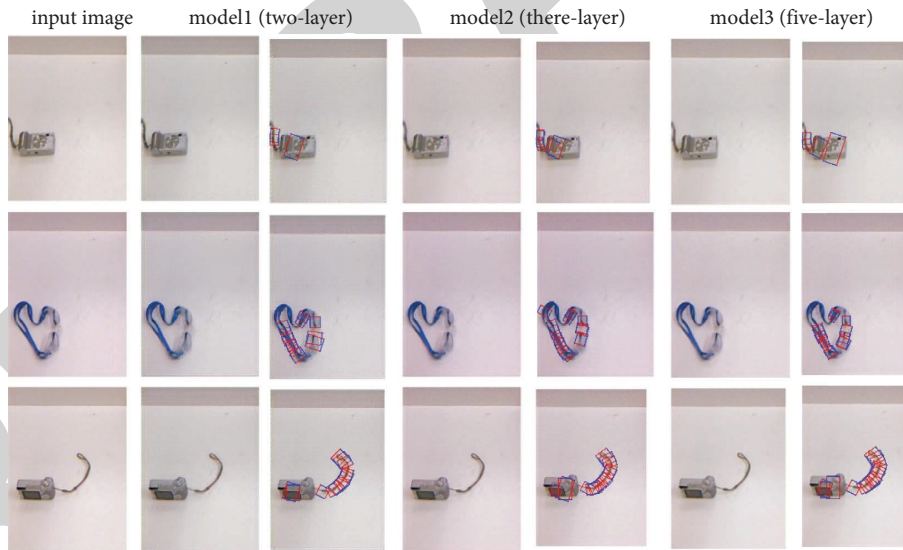


FIGURE 8: Reconstruction and grasp detection result of different models. As the number of convolution layers increases, feature extraction becomes more sufficient, and the quality of reconstructed pictures becomes better.

4.5.4. Changing Uplink Rate Environment Experiment. In practical application situations, the network environment often fluctuates and brings bandwidth changes. With the deterioration of the network environment, the network transmission delay will increase. This makes it necessary for us to choose flexibly among various schemes according to the actual situation. We design experiments to verify how to choose under different network speed conditions.

There are several schemes in the experiment:

- (a) Pure Edge Offloading (EO): All received pictures will be transmitted to the edge server for calculation and then the data will be returned to the local.
- (b) Pure On-device Processing (MO): All the received frames will be directly calculated locally and will not be transmitted to the server.

TABLE 5: Comparison of different models.

Number of convolution layers	PSNR	SSIM
2-layer	28.346	0.9
3-layer	31.768	0.948
5-layer	35.438	0.952

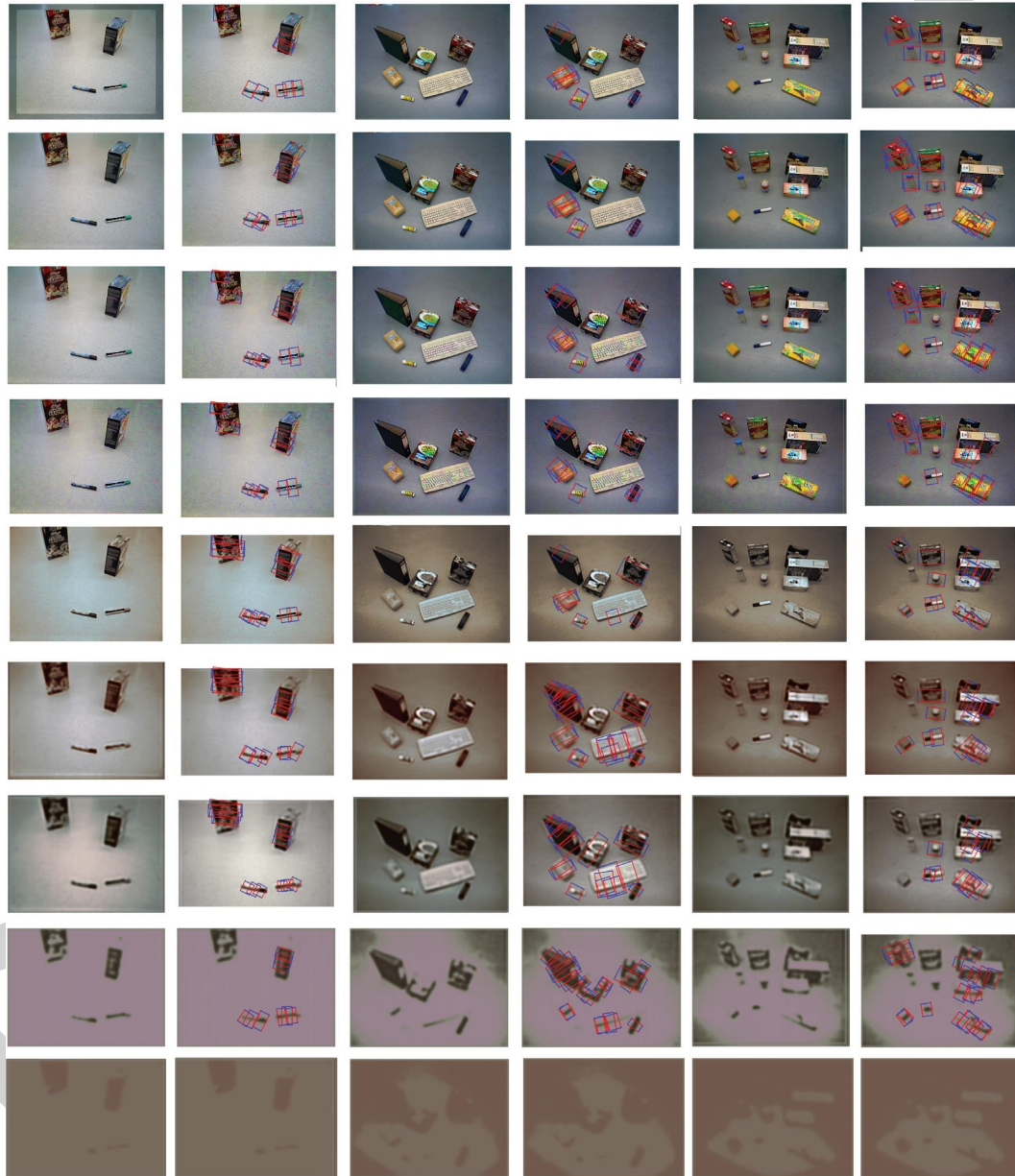


FIGURE 9: The first line is the result of the grasp detection of the original image, and the lines below are the results under the compression ratio of 16.29%, 4.07%, 0.99%, 0.5%, 0.13%, 0.05%, 0.04%, and 0.03%. From top to bottom, the loss of pictures due to compression gradually increases, which slowly affects the bounding box results of grasp detection.

(c) Collaborative scheme (Collaboration): Preprocess the acquired image locally by encoding and compression, then transmit it to the edge for calculation and finally return to the mobile device.

(d) Our model: According to the real-time uplink network speed, improve the grasp scheme and select the optimal scheme under different network speeds.

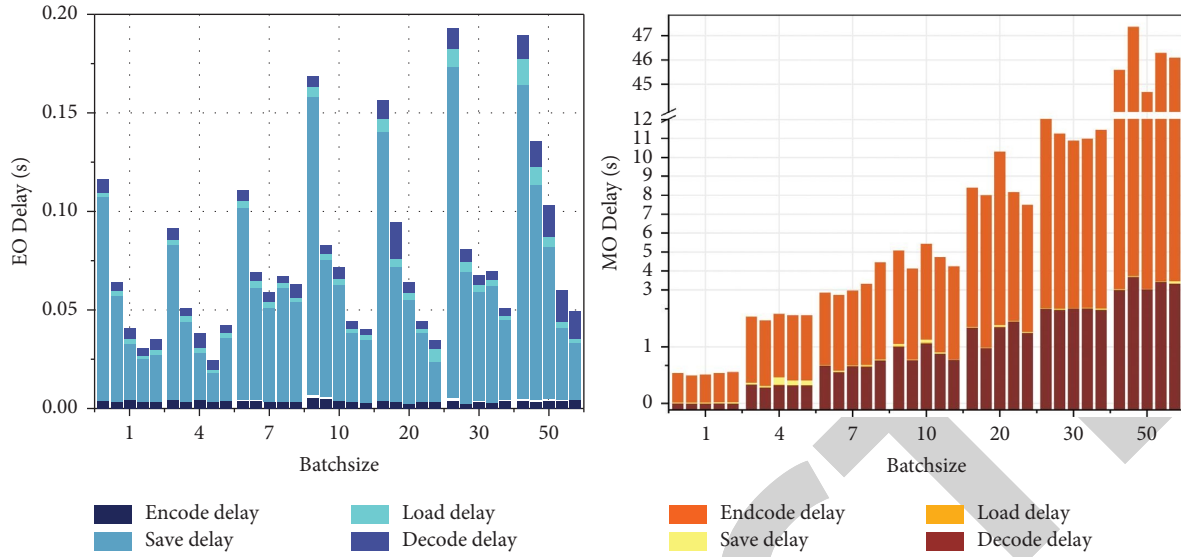


FIGURE 10: Delay under EO and MO. The overall delay consists of four parts: encode delay, save delay, load delay, and decode delay.

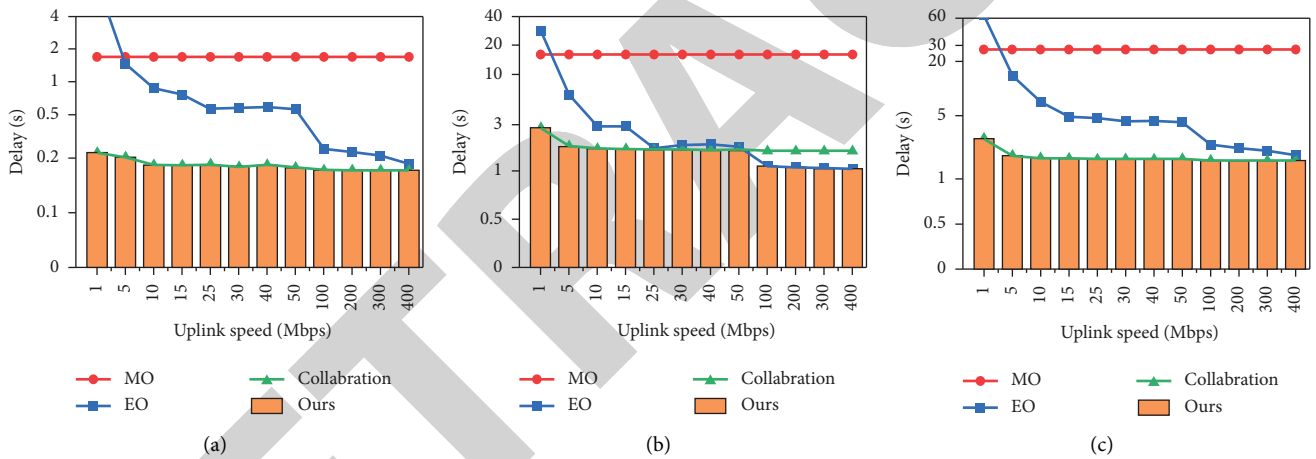


FIGURE 11: Comparison of the overall delay time of four schemes under different network bandwidths. From (a) to (c), the overall delay time for batch sizes of 1, 4, 10 is shown in sequence.

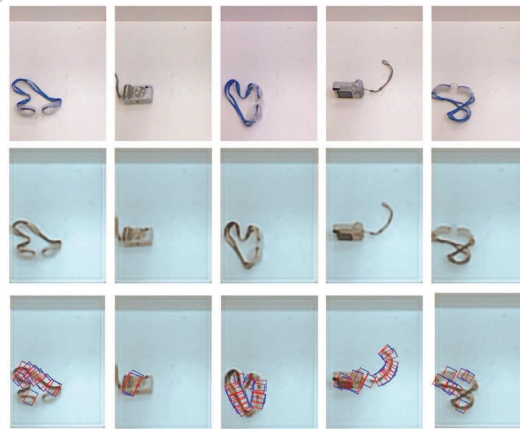


FIGURE 12: Grasp detection result under 0.2% compression ratio. Under this compression ratio, the grasp detection has high accuracy.



FIGURE 13: Grasp detection result under 0.13% compression ratio. The image is blurred, but it can still be used for grasp detection.



FIGURE 14: Grasp detection result under 0.06% compression ratio. The loss of image reconstruction becomes large, and some objects cannot be correctly recognized.

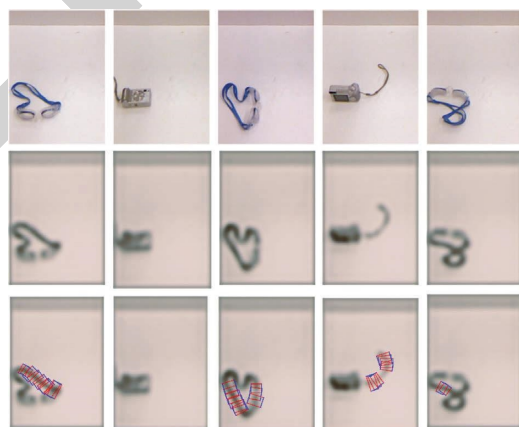


FIGURE 15: Grasp detection result under 0.05% compression ratio. The image is more blurred, but most objects can still be grasped correctly.

We test the delay on different devices and get Figure 10. The x-axis in the figure represents the number of file frames transmitted in a batch, the compression rate in each group decreases in turn, and the y-axis is the delay time. EO in Figure 10 shows that the time for encoding on the CPU is

much less than that for decoding. MO in Figure 10 shows that the decoding speed on the GPU is faster so that the reconstruction takes less time than the loading and saving operation of the model. It can be seen that encoding on the CPU and decoding on the GPU are feasible and can make

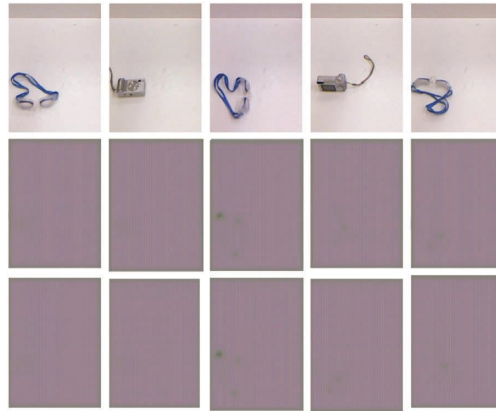


FIGURE 16: Grasp detection result under 0.04% compression ratio. At this compression ratio, single-object grasp detection cannot be performed.



FIGURE 17: The first line is the result of the grasp detection of the original image, and the lines below are the results under the compression ratio of 16.29%, 4.07%, 0.99%, 0.5%, 0.13%, 0.05%, 0.04%, and 0.03%. From top to bottom, the loss of pictures due to compression gradually increases, which slowly affects the bounding box results of grasp detection.

good use of resources. In order to increase efficiency, we explore the impact of the number of frames transmitted at the same time on latency. It is worth noting that the delay of mobile devices fluctuates greatly due to the number of files. The lower compression rate and the smaller tensor volume are beneficial to speed up the loading of the data obtained by the edge server.

We chose a fixed compression rate of 2.03% for transmission experiments under different network bandwidths. Figure 10 shows the delay rates of the four schemes under different network bandwidths. It can be seen that the effects of EO and MO vary greatly in different network conditions. The collaboration method can achieve a balanced result between the two. In most cases, the collaboration method can already achieve a good effect and greatly improve the latency of other solutions. However, in some cases, as shown in Figure 11, when the delay of file transfer does not become the key factor, it is possible that EO and MO will achieve better results than the collaboration scheme. So we established a linear model to switch between these three network models, seeking results that are more suitable for multiple factors and situations.

5. Conclusion

We propose a new grasping detection model and perform grasping detection in RGB images. With the scheme of multi-object multi-grasp, our model improves the mission success ratio of grasping. With the help of edge-cloud collaboration, the computing task is transferred to the cloud with powerful computing power, which greatly improves the speed and accuracy of grasp detection. The encoder and decoder trained by GAN enable the image to be encrypted while compressing, ensuring the security of privacy. The model proves that the combination of autonomous robot grasping and edge-cloud collaboration has great prospects. The model achieves 92% accuracy on the OCID dataset, the image compression ratio reaches 2.03%, the structural difference value is higher than 0.91, and the average detection speed reaches 13.62 fps. Furthermore, we have packaged our model as a functional package of the ROS operating system, which can be easily used in actual robotic arm operations. In the future, we will improve compression, and refine the distribution of tasks between on-premises and the cloud to further improve the efficiency of the model. At the same time, our solution can be fully applied to other work of robots to promote the development of the field of robotics. This work is also potential in some other fields, such as federated learning [32–34], cloud-edge cooperate robotics [35, 36], data collection [37], and smart city.

Appendix

A. Grasp Detection Result under Different Compression Ratios

The comparison of reconstructed image quality under different channel numbers is shown in Figures 12–16. The three lines from top to bottom are the input image, the

reconstructed image, and the labelled result. By comparing these five groups of images, it is proven that in a single object grasping task, a low compression ratio can still achieve good results. Until the compression ratio is as low as 0.06%, it begins to appear that the detected object cannot be recognized and grasped. The results of the multi-object grasp detection task are shown in Figure 17.

Data Availability

All data included in this study are available from the first author or corresponding author upon reasonable request.

Conflicts of Interest

The authors declare that there are no conflicts of interest.

Acknowledgments

This work was supported by Hainan Provincial Natural Science Foundation of China (Grant No. 620MS021), the Key Research and Development Program of Hainan Province (Grant No. ZDYF2021GXJS003, ZDYF2020040), the Major Science and Technology Project of Hainan Province (Grant No. ZDKJ2020012), National Natural Science Foundation of China (NSFC) (Grant No. 62162022, 62162024), the Key Laboratory of PK System Technologies Research of Hainan, Science and Technology Development Center of the Ministry of Education Industry-University-Research Innovation Fund (2021JQR017).

References

- [1] J. Sanchez, J.-A. Corrales, B.-C. Bouzgarrou, and Y. Mezouar, "Robotic manipulation and sensing of deformable objects in domestic and industrial applications: a survey," *The International Journal of Robotics Research*, vol. 37, no. 7, pp. 688–716, 2018.
- [2] A. Bicchi and V. Kumar, "Robotic grasping and contact: A review," in *Proceedings 2000 ICRA Millennium conference IEEE international conference on robotics and automation. Symposia proceedings (Cat. No. 00CH37065)*, pp. 348–353, San Francisco, CA, USA, April 2000.
- [3] C. Zhihong, Z. Hebin, W. Yanbo, L. Binyan, and L. Yu, "A vision-based robotic grasping system using deep learning for garbage sorting," in *Proceedings of the 36th Chinese control conference (CCC)*, pp. 11 223–11 226, Dalian, China, July 2017.
- [4] S. Kumra and C. Kanan, "Robotic grasp detection using deep convolutional neural networks," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 769–776, Vancouver, BC, Canada, September 2017.
- [5] F. Leoni, M. Guerrini, C. Laschi, D. Taddeucci, P. Dario, and A. Starita, "Implementing robotic grasping tasks using a biological approach," in *Proceedings. 1998 IEEE International Conference on Robotics and Automation (Cat. No.98CH36146)*, pp. 2274–2280, Leuven, Belgium, May 1998.
- [6] D. Quillen, E. Jang, O. Nachum, C. Finn, J. Ibarz, and S. Levine, "Deep reinforcement learning for vision-based robotic grasping: A simulated comparative evaluation of off-policy methods," in *Proceedings of the IEEE International*

- Conference on Robotics and Automation (ICRA)*, pp. 6284–6291, Brisbane, QLD, Australia, May 2018.
- [7] F. J. Chu, R. Xu, and P. A. Vela, “Real-world multi-object, multi-grasp detection,” *IEEE Robotics and Automation Letters*, vol. 3, no. 4, pp. 3355–3362, 2018.
 - [8] V. K. Sarker, J. P. Queralta, T. N. Gia, H. Tenhunen, and T. Westerlund, “Offloading slam for indoor mobile robots with edge-fog-cloud computing,” in *Proceedings of the 1st international conference on advances in science, engineering and robotics technology (ICASERT)*, pp. 1–6, Dhaka, Bangladesh, May 2019.
 - [9] A. K. Tanwani, R. Anand, J. E. Gonzalez, and K. Goldberg, “Rilaas: Robot inference and learning as a service,” *IEEE Robotics and Automation Letters*, vol. 5, no. 3, pp. 4423–4430, 2020.
 - [10] R. Deng, R. Lu, C. Lai, T. H. Luan, and H. Liang, “Optimal workload allocation in fog-cloud computing toward balanced delay and power consumption,” *IEEE internet of things journal*, vol. 3, no. 6, pp. 1171–1181, 2016.
 - [11] S. Dhawan, “A review of image compression and comparison of its algorithms,” *International Journal of Electronics & Communication Technology, IJECT*, vol. 2, no. 1, pp. 22–26, 2011.
 - [12] J. Ballé, D. Minnen, S. Singh, S. J. Hwang, and N. Johnston, “Variational image compression with a scale hyperprior,” 2018, <https://arxiv.org/abs/1802.01436>.
 - [13] G. Toderici, S. M. O’Malley, S. J. Hwang et al., “Variable rate image compression with recurrent neural networks,” 2016, <https://arxiv.org/abs/1511.06085>.
 - [14] G. Toderici, D. Vincent, N. Johnston et al., “Full resolution image compression with recurrent neural networks,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5435–5443, Honolulu, HI, USA, July 2017.
 - [15] W. Jiang, W. Wang, S. Li, and S. Liu, “Online meta adaptation for variable-rate learned image compression,” 2021, <https://arxiv.org/abs/2111.08256?context=cs>.
 - [16] T. Chen and Z. Ma, “Variable bitrate image compression with quality scaling factors,” in *Proceedings of the ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 2163–2167, Barcelona, Spain, May 2020.
 - [17] O. Rippel and L. D. Bourdev, “Real-time adaptive image compression,” in *Proceedings of the 34th International Conference on Machine Learning*, Sydney NSW Australia, August 2017.
 - [18] M. Joshi, S. K. Budhani, N. Tewari, and S. Prakash, “Analytical review of data security in cloud computing,” in *Proceedings of the 2nd International Conference on Intelligent Engineering and Management (ICIEM)*, pp. 362–366, London, UK, April 2021.
 - [19] K. Randeep and K. Jagroo, “Cloud computing security issues and its solution: A review,” in *Proceedings of the 2nd International Conference on Computing for Sustainable Global Development (INDIACom)*, pp. 1198–1200, New Delhi, India, March 2015.
 - [20] H. Li, Y. Yang, Y.-S. Dai, S. Yu, and Y. Xiang, “Achieving secure and efficient dynamic searchable symmetric encryption over medical cloud data,” *IEEE Transactions on Cloud Computing*, vol. 8, no. 2, pp. 484–494, 2020.
 - [21] S. Shini, T. Thomas, and K. Chithraranjan, “Cloud based medical image exchange-security challenges,” *Procedia Engineering*, vol. 38, pp. 3454–3461, 2012.
 - [22] Q. Yao, X. Han, X. Ma, Y. Xue, Y.-J. Chen, and J. S. Li, “Cloud-based hospital information system as a service for grassroots healthcare institutions,” *Journal of Medical Systems*, vol. 38, no. 9, pp. 104–107, 2014.
 - [23] A. Radford, L. Metz, and S. Chintala, “Unsupervised representation learning with deep convolutional generative adversarial networks,” 2016, <https://arxiv.org/abs/1511.06434>.
 - [24] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 770–778, Las Vegas, NV, USA, June 2016.
 - [25] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, “Image quality assessment: from error visibility to structural similarity,” *IEEE transactions on image processing*, vol. 13, no. 4, pp. 600–612, 2004.
 - [26] H. Zhao, O. Gallo, I. Frosio, and J. Kautz, “Loss functions for image restoration with neural networks,” *IEEE Transactions on Computational Imaging*, vol. 3, no. 1, pp. 47–57, 2017.
 - [27] S. Ren, K. He, R. Girshick, and J. Sun, “Faster r-cnn: Towards real-time object detection with region proposal networks,” *Advances in neural information processing systems*, vol. 28, 2015.
 - [28] X. Jia, E. Gavves, B. Fernando, and T. Tuytelaars, “Guiding long-short term memory for image caption generation,” 2015, <https://arxiv.org/abs/1509.04942>.
 - [29] E. Agustsson and R. Timofte, “Ntire 2017 challenge on single image super-resolution: Dataset and study,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, Honolulu, HI, USA, July 2017.
 - [30] S. Ainetter and F. Fraundorfer, “End-to-end trainable deep neural network for robotic grasp detection and semantic segmentation from rgb,” in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pp. 13 452–13 458, Xi’an, China, May 2021.
 - [31] M. Suchi, T. Patten, D. Fischinger, and M. Vincze, “Easylab: a semi-automatic pixel-wise object annotation tool for creating robotic rgb-d datasets,” in *Proceedings of the International Conference on Robotics and Automation (ICRA)*, pp. 6678–6684, Montreal, QC, Canada, May 2019.
 - [32] B. Liu, L. Wang, and M. Liu, “Lifelong federated reinforcement learning: a learning architecture for navigation in cloud robotic systems,” *IEEE Robotics and Automation Letters*, vol. 4, no. 4, pp. 4555–4562, 2019.
 - [33] B. Liu, L. Wang, M. Liu, and C.-Z. Xu, “Federated imitation learning: A novel framework for cloud robotic systems with heterogeneous sensor data,” *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 3509–3516, 2020.
 - [34] Z. Zheng, Y. Zhou, Y. Sun, Z. Wang, B. Liu, and K. Li, “Applications of federated learning in smart cities: recent advances, taxonomy, and open challenges,” *Connection Science*, vol. 34, pp. 1–28, 2021.
 - [35] B. Liu, L. Wang, and M. Liu, “Elasticros: An elastically collaborative robot operation system for fog and cloud robotics,” pp. 1–19, 2022, https://www.researchgate.net/publication/363032937_ElasticROS_An_Elastically_Collaborative_Robot_Operation_System_for_Fog_and_Cloud_Robotics.
 - [36] B. Liu, L. Wang, X. Chen, L. Huang, D. Han, and C.-Z. Xu, “Peer-assisted robotic learning: a data-driven collaborative learning approach for cloud robotic systems,” in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pp. 4062–4070, Xi’an, China, May 2021.
 - [37] O. Akin, P. Li, S. Agarwal, and S. P. Chinchali, “Data games: A game-theoretic approach to swarm robotic data collection,” in *Proceedings of the 6th Annual Conference on Robot Learning*, Auckland, New Zealand, December 2022.