IEEE Robotics and Automation Letters (RAL) paper presented at
2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
October 14-18, 2024. Abu Dhabi, UAE

# FGDSNet: A Lightweight Hand Gesture Recognition Network for Human Robot Interaction & Symposia*

Guoyu Zhou, Zhenchao Cui✉ and Jing Qi

*Abstract*— **Computer vision-based gesture recognition methods play a significant role in robot visual gesture interaction. since of low accuracy leading by insuffcient feature representation and fusion, the existing gesture segmentation and recognition methods fail to meet the requirements of practical applications. To address these issues, a lightweight two-stage end-to-end gesture recognition network called Fusing Gate Dual Stages Network (FGDSNet) is proposed. This network adopts a dual-branch network structure in the segmentation stage. Existing dual-branch network models often directly fuse detailed features and semantic features, which leads to detailed information being obscured by blurry semantic information. Additionally, there are redundant issues in the feature maps at different levels during the network inference process. Therefore, we embed Cosine Similarity-KL Divergence Attention Module (CoSKLAM) and Gate Filtering Module (GFM) between the local detail branch and the contextual semantic branch. The role of these two modules is to facilitate the fusion of local and global features during the feature extraction process and filter out redundant information. Finally, the segmentation result and original gesture image are used as inputs for the recognition network to predict gesture categories. The relevant experiments show that the proposed network performs well in both gesture segmentation and gesture recognition, while also having real-time inference speed and a smaller parameter size.**

## I. INTRODUCTION

Gesture recognition and human-computer interaction are important areas of focus in today's artificial intelligence and robotics technology. As the demand for intelligent robots continues to increase, the ways in which humans interact with machines are constantly evolving. Gesture recognition, as an intuitive and natural form of interaction, has become a key technology for seamless communication between robots and humans. Gesture interaction is commonly categorized into methods based on wearable devices and methods based on machine vision. Due to the limited range of gestures and expensive sensor prices associated with wearable devices, machine vision-based methods have become the mainstream approach.

Based on deep learning, most gesture recognition methods use a two-stage CNN model, such as gesture segmentation and recognition, and gesture detection and recognition. However, when faced with complex backgrounds, the more background content is eliminated, the fewer noise features there are, resulting in better extraction and recognition of gesture features. Therefore, we first perform gesture segmentation and then proceed with recognition. By utilizing the segmented gesture mask to eliminate background noise interference, we can maximize the preservation of gesture features for accurate identification.

Based on traditional machine learning, gesture segmentation algorithms include those based on skin color models and contour-based methods, among others. These methods typically rely on single, predefined operators to extract features from gestures. This often proves ineffective in segmenting or detecting complex gesture images with diverse hand shapes and complex backgrounds.

With the development and application of deep learning in many fields, methods based on deep learning have become mainstream in gesture segmentation. For example, Al-Hammadi et al. [1] used multiple deep-learning architectures to segment hand regions. Dadashzadeh et al.[2] used a residual network structure and dilated spatial pyramid pooling for gesture segmentation, but the network structure is simple and has weak representation capability for gesture features, resulting in low accuracy of gesture segmentation. Wang et al. [3] enhanced the multi-scale feature extraction capability of the network using MSF modules and LWMS modules, but ignored the issue of parameter size in gesture segmentation networks. Dayananda[5] proposed a new hybrid approach based on RGB-D gesture images. However, compared with RGB images, it is limited to the image dataset. Das et al.[6] performed real-time pixel-level semantic segmentation using an encoder-decoder architecture, but due to simple features, the segmentation accuracy of this method cannot be considered.

Therefore, currently most of the gesture segmentation algorithms cannot meet the dual requirements of accuracy and lightweight. Our aim is to design a lightweight gesture segmentation algorithm that can satisfy both accuracy and lightweight requirements for gesture interaction.

Traditional machine learning algorithms use manually designed features by researchers to recognize gestures. This kind of algorithm requires manual feature design and is difficult to adapt to the diversity of gestures in complex backgrounds. Based on the excellent performance of deep learning in computer vision tasks, it has been widely applied in gesture recognition. Tan et al.[7] used the CNN-SPP network for gesture recognition, while Mohanty et al.[8] proposed a gesture recognition method that does not require segmentation or other preprocessing operations. Dang et al. [4] proposed a new network method based on DeeplabV3+ and customized UNet, which reduced the total

* This work is supported in part by Scientic Research Foundation of Hebei University for Distinguished Young Scholars (Grant No.521100221081),in part by Science and Technology Program of Hebei Province (Grant No.22370301D).

The authors are with the School of Cyber Security and Computer, Hebei University, Baoding, China zhouguoyu2021@163.com; ✉cuizhenchao@gmail.com; qijingalice@163.com

parameter volume by replacing backbone networks, but still had lower overall segmentation and recognition accuracy. Barbhuiya.et.al[15]used self-attention-based VGG networks for gesture recognition. However, none of these methods have conducted experimental analysis on model parameter quantity and computational complexity. Karsh[25] used the Inception V3 architecture to build the network model, but ignored the issue of parameter size and inference speed. Bhaumik[26] employed a hybrid feature attention method to capture detailed edge information of gestures, utilizing multi-scale attention feature fusion and interleaved modules to extract rich spatial information.

Gesture recognition also has broad applications in the field of robotics. For instance, Chavez[28] proposed a large-scale underwater gesture recognition dataset, and in reference[**?**], [27], work on gesture recognition for human-robot interaction underwater was presented. Pamungkas [9] used the Myo armband to acquire electromyographic signals for gesture recognition and robot control. Solly [10] utilized a glove-like controller to remotely operate a mobile robot. Both methods require additional physical devices, which impose limitations on gesture actions and are not conducive to practical applications. Sahoo [11] employed a deep neural network based on attention mechanisms for gesture recognition and subsequent robot control. However, its use of RGB-D as input images restricts its application scope. Zhou [12] proposed an SSD model for remote gesture recognition and conducts relevant experiments on a self-built dataset. Nevertheless, this method still lacks lightweight parameters in terms of quantity. Peral [13] recognized gestures based on joint positions and performs actual experiments with the IVO robot, but it still exhibits lower operating speed and accuracy in recognition.

To the best of our knowledge, the most existing gesture segmentation and recognition algorithms cannot meet the requirements of mobile robots for algorithm accuracy and lightweight. For instance, traditional dual-branch networks directly fusion of semantic and detail features, which results in pixel coverage issues and consequently low model accuracy. Additionally, there are issues of redundant features across different levels during network inference. Therefore, a real-time two-stage gesture recognition network called Fusion Gate Dual-Stage Network (FGDSNet) is proposed. FGDSNet integrates multiple different gesture features, and its effective feature maps help the network achieve better results in the gesture segmentation part, ultimately combining with a classification network for gesture recognition. The proposed network greatly improves the accuracy of gesture segmentation and recognition, while having a very low number of parameters and real-time inference speed.

The main contributions are summarized as follows:

- A Cosine Similarity-KL Divergence Attention Module (CoSKLAM) is proposed to adjust the weight distribution ratio between advanced semantic features and local detailed features. And a Gate Filtering Module (GFM) is designed to achieve filtering and fusion of different hierarchical features.
- A Simple Aggregation Pyramid Pooling Module (SAPPM) is proposed to aggregate global context. SAPPM promotes segmentation accuracy with minor extra inference time.
- A two-stage end-to-end gesture recognition network is formed by combining the dual-branch hand gesture segmentation network with partial CNN. Compared to other gesture recognition models, the proposed model maintains high accuracy and real-time inference speed on multiple challenging datasets, while having an extremely low parameter count, making it suitable for deployment on mobile robot platforms.

The article is organized as follows. Section 2 provides a detailed introduction to the proposed FGDSNet. To demonstrate the effectiveness of the proposed FGDSNet, corresponding experiments are conducted, and the results are presented in Section 3. The conclusion is drawn in Section 4.

## II. The Proposed Network: FGDSNet

A real-time dual-stage gesture recognition network called Fusing Gate Dual Stages Network (FGDSNet) is proposed. It is divided into two stages: gesture segmentation and gesture recognition. The overview is shown in Fig.1. Specifically, a dual-branch network is applied in the gesture segmentation stage. To address the feature fusion problem, CoSKLAM and GFM are embedded between the two branches. CoSKLAM uses cosine similarity and KL divergence methods to predict the weight distribution ratio of dual-branch feature maps at the pixel level. Based on this ratio, the high-level semantic feature maps and local detail feature maps of the dual-branch output can be effectively fused. GFM applies gate filtering to filter and fuse feature maps at different levels, resulting in a set of gesture feature maps with balanced local details and global semantics. These maps are then used to predict the gesture segmentation results. In addition, CoSKLAM and GFM are often combined together in the dual-branch structure. Subsequently, the segmentation results along with the original gesture images are fed into the gesture recognition stage to predict gesture categories. We will now elaborate on these two stages: gesture segmentation and recognition.

### *A. Hand Gesture Segmentation*

To ensure that the overall model can run in real-time on mobile robots, a lightweight real-time gesture segmentation network is proposed. This network has a typical dual-branch structure, as shown in the upper half of Fig.1. The local detail branch maintains a consistent resolution during sampling to preserve detailed information in high-resolution feature maps. The contextual semantic branch has more channels and an increasing downsampling rate to better analyze long-distance pixel dependencies. CoSKLAM and GFM are used to fuse the feature maps of the dual branches. Additionally, to reduce the overall model parameters and computational complexity, we use Ghost Bottleneck as the building block.

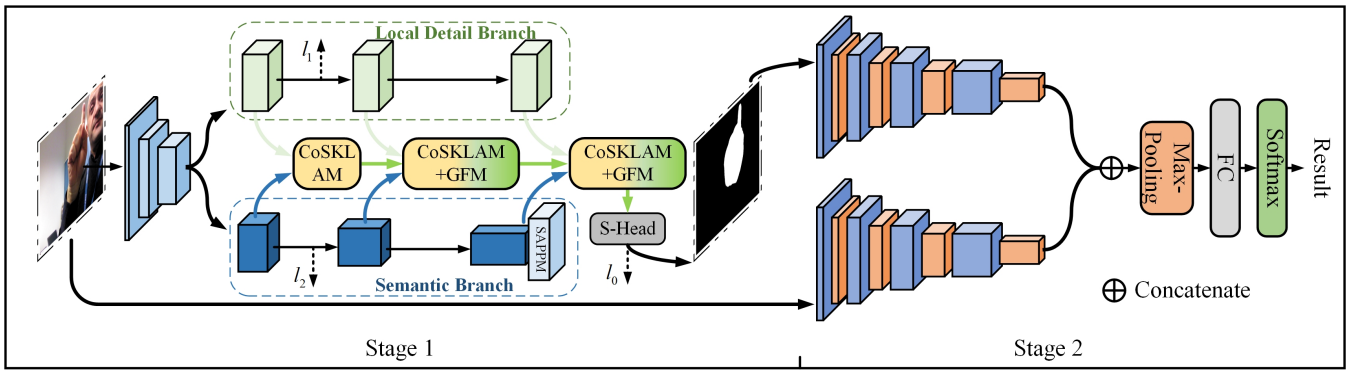



Fig. 1. The overview of the FGDSNet

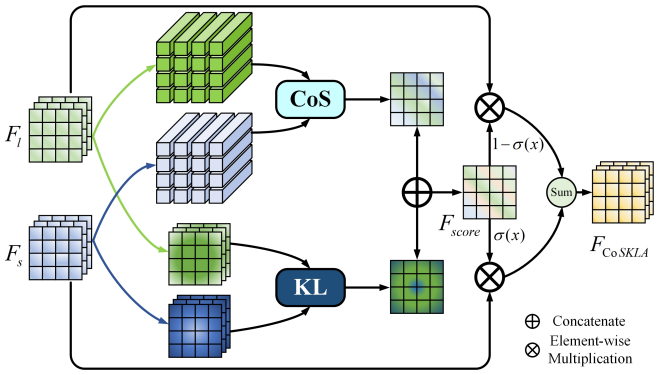



Fig. 2. Cosine Similarity-KL Divergence Attention Module (CoSKLAM)

Two extra loss functions are used to supervise these two branches, namely $l_1$ and $l_2$, as shown in Fig.1. Specifically, the Dice loss [16] is used as $l_1$ to enhance the ability of the local detail branch in extracting gesture boundaries. $l_0$ and $l_2$ represent the CE loss [46]. Therefore, the final loss function for hand gesture segmentation is defined as shown as Equation (1).

$$loss = \lambda_0 l_0 + \lambda_1 l_1 + \lambda_2 l_2, \tag{1}$$

According to experience[17], [18], the parameters is set as $\lambda_0$=1, $\lambda_1$=0.5, and $\lambda_2$=0.4. During the model inference stage, $l_1$ and $l_2$ will not be applied.

*1) The Cosine Similarity-KL Divergence Attention Module (CoSKLAM) :* The contextual semantic features are often expressed in an abstract manner, which can be very rough when it comes to expressing the details of gestures. This is exactly the opposite of detailed features. However, directly fusing local detailed features and contextual features can bring a new problem. There is a significant semantic gap between them, and other details such as gesture edges are often covered by blurry contextual semantic information.

Therefore, CoSKLAM based on cosine similarity and KL divergence is proposed to predict pixel-level weight scores between high-level semantic features and local detail features. The specific internal structure is shown in Fig.2.

For the cosine similarity calculation part, the local detailed feature map $F_l$ and the contextual semantic feature map $F_s$ is divided into individual pixel-sized feature vectors $\vec{v}_l\ (i,j)$ and $\vec{v}_s\ (i,j)$, while ensuring that $0 \leq i < H, 0 \leq j < W$. Based on this, the cosine similarity is calculated between them at each pixel position. And a cosine similarity weight score matrix$F_{CoS} \in \mathbb{R}^{H\times W\times 1}$ is obtained, as defined in Equation (2).

$$\vec{v}_{CoS}\ (i,j) = f_\sigma\left(\frac{\vec{v}_l\ (i,j)\cdot \vec{v}_s\ (i,j)}{max\left(\left\|\vec{v}_l\ (i,j)\right\|_2 \cdot \left\|\vec{v}_s\ (i,j)\right\|_2, \epsilon\right)}\right), \tag{2}$$

where $\epsilon$ is a small decimal value set to avoid division by zero, and $\vec{v}_{CoS}\ (i,j)$ is the vector corresponding to the pixel position of $F_{CoS}$.

When calculating the KL divergence information, the pixel values of each dimension in $F_l$ are converted into logarithmic probability distributions. We then normalize the calculations on each dimension of $F_s$ and compute the KL divergence between them. After that, element-wise summation and sigmoid operation is performed. Since higher values indicate more significant differences in KL divergence, to have the same numerical meaning as cosine similarity where higher values indicate a higher likelihood of belonging to the same object, we subtract 1 from $KL'$ obtained and finally output the weight score matrix $F_{KL} \in \mathbb{R}^{H\times W\times 1}$. This process is defined by Equation (3), Equation (4) and Equation (5).

$$KL = F_s log\frac{F_s}{F_l}, \tag{3}$$

$$KL' = f_\sigma(sum(KL)), \tag{4}$$

$$F_{KL} = 1 - KL' \tag{5}$$

Two weight score matrices are merged by a 1x1 convolutional layer to obtain $F_{score} \in \mathbb{R}^{H\times W\times 1}$. In $F_{score}$, the larger the weight score $\sigma$ of each pixel, the higher the possibility that this pixel belongs to the same object. Then it is multiplied by the contextual semantic feature map $F_s$ because $F_s$ is more accurate in semantic prediction. Conversely, 1-$\sigma$ is multiplied by $F_l$ to amplify the detailed content at that location for network learning.

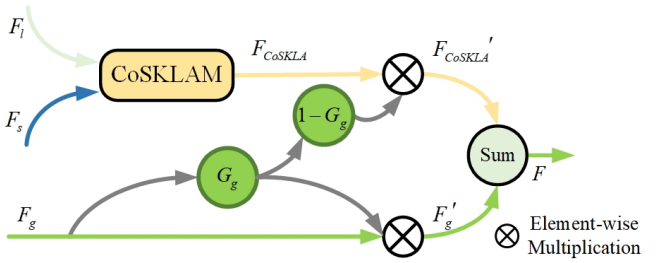



Fig. 3. Gate Filtering Module (GFM)

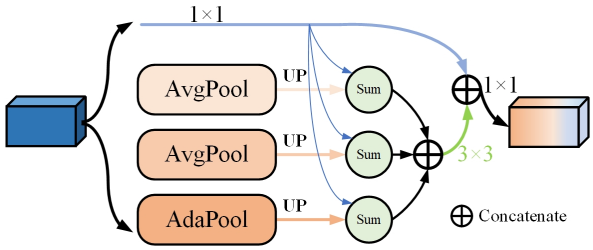



Fig. 4. Simple AggregationPyramid Pooling Module (SAPPM)

The cosine similarity is used to calculate the cosine value of the angle between vectors at corresponding pixel positions to determine their similarity. And the KL divergence is used to measure the similarity of corresponding pixels in a particular dimension. Through these methods, clear weight scores are obtained between local detailed feature maps and high-level semantic feature maps at corresponding pixel positions, and used as a harmonization method between the two.

*2) The Gate Filtering Module (GFM):* The different levels of the dual-branch network generate feature maps with different meanings. We want to preserve the valuable content in these feature maps and filter out redundant and useless information. The gating mechanism is widely used in Long Short-Term Memory(LSTM), Gated Recurrent Unit(GRU), image segmentation and image generation. It adjusts the flow of information through learned gate functions to selectively retain, update, or output important information. Therefore, GFM is embedded between the two branches to control the flow of information across network layers. The specific structure is shown in Fig.3.

For the feature map $F_g$ from the previous level, convolution and sigmoid operations are performed to obtain the gate $G_g$. $G_g$ and $F_g$ are multiplied element-wise to obtain the optimized feature map $F_g'$. Then, the output result $F_{CoSKLA}$ of CoSKLAM is multiplied with 1-$G_g$ to get $F'_{CoSKLA}$. The purpose of this operation is to selectively absorb effective feature information from $F_{CoSKLA}$ that is not present in $F_g$ using 1-$G_g$. Specifically, for a feature vector located at position $(x, y)$ in the feature map, when $G_g(x, y)$ has a small value but $F_{CoSKLA}(x, y)$ is large, multiplying 1-$G_g(x, y)$ by $F_{CoSKLA}(x, y)$ results in a useful feature vector with a larger value. This feature vector will then be fused into $F_g'$ through summation, allowing $F_{CoSKLA}$ to complement $F_g$'s missing effective information. Conversely, redundant information will not pass through the gate, which helps avoid redundancy to some extent. This process is defined as Equation (6):

$$F = G_g \cdot F_g + (1 - G_g) \cdot F_{CoSKLA}, \tag{6}$$

where $\cdot$ represents element-wise multiplication, $G_g = f_\sigma(Conv_{1\times1}(F_g))$. Specifically, GFM is placed to filter and aggregate the feature information of CoSKLAM and each level. After going through GFM, the local detailed features $F_l$, high-level semantic features $F_s$, and output features $F_g$ from the previous layer will be effectively fused.

*3) The Simple Aggregation Pyramid Pooling Module (SAPPM):* Most semantic segmentation models[17], [18] adopt a context semantic module to quickly expand the model's receptive field and build global scene priors. SAPPM is proposed and placed at the output of the last Ghost Bottleneck in the semantic branch, which is used to extract local and global dependency relationships from low-resolution abstract feature maps. The specific internal structure is shown in Fig. 4. Three pooling branches with different sizes of pooling kernels and one 1×1 convolutional branch are applied for multi-scale feature extraction and aggregation. The pooled feature maps are then convolved, upsampled, and added directly to the results of the 1×1 convolutional branch. Finally, we further aggregate features using a 3×3 convolutional operation before concatenating the results of all four branches. A 1×1 convolution is used to fuse multi-scale features.

SAPPM has only 0.03M parameters in the overall network segmentation, and its processing of input feature maps with a resolution of $10 \times 10$ hardly affects the inference speed.

### B. Hand Gesture Recognition

A two-stage network is used to complete gesture recognition, as shown in Fig. 1. In the first stage, the predicted segmentation results can obtain information about the shape and position of gestures. In the second stage, both the segmentation result $F_{seg} \in \mathbb{R}^{H\times W\times 2}$ and original image $F_{ori} \in \mathbb{R}^{H\times W\times 3}$ are inputted into two convolutional neural networks with identical structures. These networks consist of convolutional layers, pooling layers, and fully connected layers. The convolutional layers include 2D convolutions, BatchNorm2d, and ReLU activation functions. The process is defined in Equation (7).

$$Output = f_{cls}(f_1(F_{seg}) \oplus f_2(F_{ori})) \tag{7}$$

where $\oplus$ represents the concatenate, and $f_{cls}$, $f_1$, and $f_2$ respectively represent the feature extraction process. The final recognition result of the gesture is $Output$.

## III. Experiments

To verify the effectiveness of the proposed network, corresponding experiments are conducted on the different datasets. Furthermore, different methods are compared to evaluate the superiority of the proposed network.

### A. Datasets, computation platform, experimental detail and evaluation criteria

The OUHANDS dataset is collected using the Intel RealSense F200 camera and includes 10 hand gestures performed by 23 subjects. The photos in this database present complex background and lighting changes, variations in hand shapes, sizes, and skin color, as well as different levels of occlusion between hands and faces. The HGR1 dataset comprises 899 images in RGB, featuring 25 hand gestures performed by 12 subjects. The dataset contains gesture arms in the GroundTruth, but there is no issue of hand-face occlusion. The EgoHands dataset contains 4800 images, encompassing four activities: puzzles, cards, Jenga, and chess, set against complex backgrounds and involving hand occlusions. We divide the dataset into training and testing sets with a 3:1 ratio. The NUS-II dataset is a 10-class gesture dataset, with complex natural backgrounds, and varying hand shapes and sizes. These gestures are completed by 40 participants from different ethnicities and diverse backgrounds, including both men and women aged between 22 and 56 years old.

The proposed model was trained and tested on a single NVIDIA RTX 3080 using PyTorch 1.11, CUDA 11.3, and cuDNN 8.0. It consists of two stages: the training of a gesture segmentation network followed by its integration with a dual-input CNN to form the FGDSNet. The 320×320 image inputs and fixed parameters from the pre-trained segmentation network are utilized for gesture recognition. The model employs common data augmentation techniques like horizontal flipping and random cropping. Metrics include PixAcc for pixel classification accuracy, mIOU for segmentation performance, FPS and GFLOPs for speed and complexity. Parameters refer to the number of parameters in a neural network model. F1-score evaluates gesture recognition model accuracy by considering precision and recall.

### B. Gesture segmentation experiments

The ablation experiments are conducted on the OUHANDS dataset to evaluate the importance of each module and extra loss function. And comparison experiments with other models are conducted on the OUHANDS, HGR1, and EgoHands datasets.

*1) Effectiveness of CoSKLAM, GFM and SAPPM:* Due to the fact that CoSKLAM and GFM are embedded in the middle part of the dual-branch network and both play a role in feature fusion, the ablation experiments are conducted by combining them. When not using CoSKLAM and GFM, we directly sum up the feature maps for fusion. As shown in Table I, when neither of these two modules is involved in training, the model has the lowest accuracy (only 88.35%). After adding each module separately, the average mIOU of the model improves by 0.5%. However, when both modules are added simultaneously, the model achieves its highest mIOU at 89.57%. Furthermore, we conduct ablation experiments on SAPPM in the presence of both CoSKLAM and GFM. The addition of SAPPM significantly improves the segmentation network accuracy, with an increase of 0.95% in mIOU. Moreover, this module only has a parameter size of 0.03M, so its addition hardly affects the model's lightweight design and real-time performance.

TABLE I
ABLATION STUDY OF COSKLAM, GFM AND SAPPM

| CoSKLAM | GFM | SAPPM | PixAcc(%) | mIOU(%) |
|---|---|---|---|---|
| | | ✓ | 97.40 | 88.35 |
| | ✓ | ✓ | 97.52 | 88.87 |
| ✓ | | ✓ | 97.50 | 88.82 |
| ✓ | ✓ | ✓ | **97.69** | **89.57** |
| ✓ | ✓ | | 97.48 | 88.62 |

TABLE II
ABLATION STUDY OF EXTRA LOSSES

| $l_1$ | $l_2$ | PixAcc(%) | mIOU(%) |
|---|---|---|---|
| | | 97.17 | 87.42 |
| ✓ | | 97.30 | 87.97 |
| | ✓ | 97.34 | 88.22 |
| ✓ | ✓ | **97.69** | **89.57** |

*2) Effectiveness of extra losses:* To investigate the impact of additional semantic supervision on the performance of network models, the ablation experiments are conducted on $l_1$ and $l_2$, and the results are shown in Table II. Without any extra supervision on both the local detail branch and semantic branch, the network model achieves an mIOU of only 87.42%. However, when additional loss functions $l_1$ and $l_2$ are added separately, there is a significant improvement in mIOU (+0.55% and +0.8% respectively). The highest mIOU of 89.57% is achieved when both $l_1$ and $l_2$ are added simultaneously.

*3) Segmentation performance on the OUHANDS dataset:* The comparison results on the OUHANDS dataset are shown in Table III. It can be seen that our model (FGDSNet-seg) has the highest pixel accuracy (97.69%) and mIOU (89.57%). FGDSNet-seg represents the gesture segmentation part of FGDSNet. The model only has 1M parameters, 0.72M more than HGRNet-seg, but its mIOU is 12.36% higher than HGRNet-seg. DDRNet is a typical dual-branch segmentation network, and we compare it with all models in this series. DDRNet-23-slim has the lowest GFLOPs among all tested networks (1.85), but its performance accuracy is lower, with an mIOU of only 80.14%. DDRNet-23 performs better than DDRNet-23-slim, with a slight increase of 0.81% in mIOU, while DDRNet-39, which has the highest number of parameters, achieves only 80.02% mIOU. SegFormer-b0 achieves an mIOU of only 80.77%, much lower than our method. PP-LiteSeg is often used as a benchmark for lightweight semantic segmentation models. PP-LiteSeg-B performs better with an mIOU of 86.23%, which is 3.34% lower than our model and does not have an advantage in terms of parameter quantity. DeepLabV3+ with MV2(MobileNetV2) has 0.11M more parameters than FGDSNet-seg, but it has a lower mIOU (-6.57%) compared to FGDSNet-seg. Unet with MV2(MobileNetV2) has only 0.37M parameters, but its mIOU is only 79.00%.

TABLE III

SEGMENTATION PERFORMANCES OF DIFFERENT APPROACHES ON OUHANDS, HGR1 AND EGOHANDS DATASETS

| Method | OUHANDS | | HGR1 | | EgoHands | | GFLOPs | Parameters(M) | FPS |
|---|---|---|---|---|---|---|---|---|---|
| | PixAcc(%) | mIOU(%) | PixAcc(%) | mIOU(%) | PixAcc(%) | mIOU(%) | | | |
| HGRNet-seg[2] | 94.64 | 77.21 | 97.55 | 92.97 | 95.95 | 77.46 | 3.07 | **0.28** | 239 |
| DDRNet-23-slim[17] | 95.25 | 80.14 | 97.88 | 93.95 | 97.1 | 83.21 | **1.85** | 5.73 | **251** |
| DDRNet-23[17] | 95.32 | 80.95 | 97.96 | 94.14 | 97.22 | 83.85 | 7.26 | 20.29 | 228 |
| DDRNet-39[17] | 95.1 | 80.02 | 98.25 | 94.98 | 96.93 | 82.31 | 14.25 | 32.65 | 152 |
| Segformer-b0[19] | 95.55 | 80.77 | 98.57 | 95.92 | 97.28 | 84.16 | 2.64 | 3.71 | 160 |
| PP-LiteSeg-T[20] | 96.6 | 85.66 | 98.59 | 95.94 | 97.99 | 88 | 7.15 | 13.52 | 244 |
| PP-LiteSeg-B[20] | 96.83 | 86.23 | 98.62 | 96.02 | 97.98 | 87.92 | 12.32 | 21.58 | 170 |
| DeepLabV3+ with MV2[4] | - | 83.00 | - | 94.00 | - | - | - | 1.11 | - |
| Unet with MV2[4] | - | 79.00 | - | 90.00 | - | - | - | 0.37 | - |
| Rec-Middel[29] | - | - | - | - | - | 82.80 | - | - | - |
| SRU(0)[30] | - | - | - | - | - | 86.20 | - | - | - |
| SRU(3)[30] | - | - | - | - | - | 86.40 | - | - | - |
| DRU(4)[30] | - | - | - | - | - | 87.30 | - | 0.36 | 61 |
| DRU-VGG16[30] | - | - | - | - | - | 89.20 | - | 41.38 | 18 |
| RDU-Net*[31] | - | - | - | - | - | 90.70 | - | - | - |
| RRU-Net[31] | - | - | | - | - | 90.00 | - | - | - |
| RSU-Net[31] | - | - | - | - | - | **92.60** | - | - | - |
| FGDSNet-seg | **97.69** | **89.57** | **98.92** | **96.89** | **98.39** | 90.16 | 2.14 | 1.00 | 179 |

A visual analysis of the prediction results of various methods is conducted on the OUHANDS dataset, as shown in Fig. 5(a). Where there is severe backlighting on the hand position, our model can accurately segment gestures, while other models have issues with overall contour and detail recovery.

*4) Segmentation performance on the HGR1 dataset:* Comparative experiments are conducted on various methods on the HGR1 dataset, and the results are shown in Table III. It can be seen that these methods perform better on the HGR1 dataset because the background of this dataset is relatively simpler. Among them, FGDSNet-seg achieves the highest pixel accuracy (98.92%) and an mIOU of 96.89%. The PP-LiteSeg series still maintains a high mIOU (95.94% and 96.02%). SegFormer-b0 performs slightly lower than PP-LiteSeg-T, while DDRNet's performance improves with increasing model parameters. HGRNet-seg has the lowest pixel accuracy (97.55%) with an mIOU of 92.97%. Although DeepLabV3+ with MV2(MobileNetV2) and Unet with MV2(MobileNetV2) have lower parameter counts, their mIOU is only 94.00% and 90.00% respectively.

We also visualize the prediction results of these methods as shown in Fig. 5(a), where it is evident that FGDSNet-seg outperforms other methods in recovering finger details, obtaining overall contours, and removing backgrounds effectively.

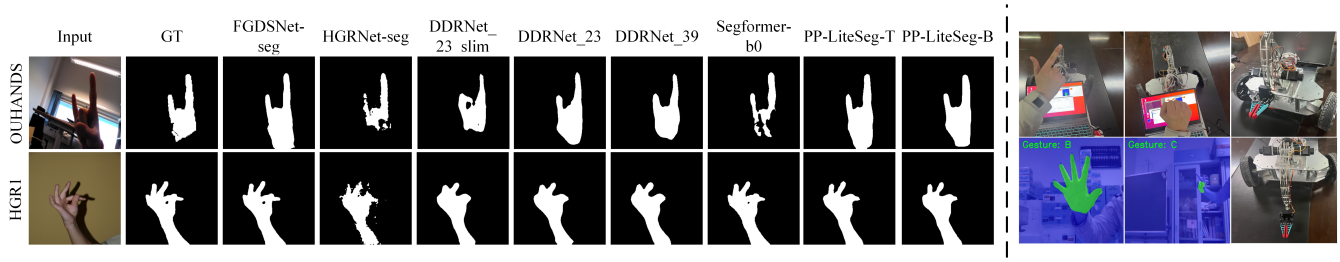
*5) Segmentation performance on the EgoHands dataset:* Table 3 shows the comparison results on the EgoHands dataset. HGRNet-seg has the lowest mIOU (77.46%). The DDRNet series exhibits nearly similar performance(around 83%), but DDRNet-39 has a slightly lower mIOU than DDRNet-23-slim and DDRNet-23. Segformer-b0 has a slightly higher mIOU than the DDRNet series. Rec-Midde's mIOU is only higher than HGRNet-seg. SRU(0), SRU(3), and DRU(4) are lightweight networks with performance slightly lower than the PP-LiteSeg series(around 88%). DRU-VGG16 has a larger parameter count(41.38M), and the mIOU is notably higher than the former three. RDU-Net*, RRU-Net, and RSU-Net perform similarly to the proposed FGDSNet-seg (around 90%), with RSU-Net achieving an mIOU of 92.60%. However, the test set size of these three models is only half of our test set, and they employ a virtual dataset method during training, which improved the model's performance.

According to Table III, the proposed network model maintains a frame rate of 179 frames per second (FPS) in terms of real-time performance. DDR-23-slim, PP-LiteSeg-T, HGRNet-seg, and DDRNet-23 achieve over 200 frames per second (FPS), while DRU(4) and DRU-VGG16 only achieve 61 and 18 frames per second (FPS), respectively. The inference speed of our model remains at an acceptable level, and the parameter count stays relatively low, meeting the requirements for lightweight applications.

### C. Gesture recognition experiments

In this sub-section, we compare the proposed FGDSNet with other models. Table IV shows the accuracy comparison of the proposed FGDSNet with the latest gesture recognition techniques on different hand recognition datasets. The proposed model achieves the highest performance on the OUHANDS and NUS-II datasets, with F1-scores of 97.80% and 99.80% respectively. While mIV3Net demonstrates consistent accuracy with our model on the NUS-II dataset, its parameter count (16.50M) considerably exceeds that of our model (1.20M). On the HGR1 dataset, our model's accuracy falls below that of EfficientNetB0[1] and MobileNetV2 Small, attributed to the fact that the literature [4] divides the HGR1 into training, validation, and test sets in a ratio of 791:54:54, whereas our training and test sets are in the ratio of 647:252.

We compare the parameter count and real-time inference speed of the models, as shown in Table IV. The proposed




(a) The visualization results of different approaches

(b) Gesture-driven robot experiment examples

Fig. 5. Visual segmentation results of different approaches and examples of gesture-driven robot experiments

model exhibits an inference speed second only to HGR-Net, with a parameter count only higher than HGRNet and MobileNetV2 Small, evidently making it sufficiently lightweight.

### *D. Gesture-driven robot experiments*

To validate the applicability of the proposed model in real-world scenarios, we conduct relevant experiments on a mobile robot platform, as shown in Fig. 5(b). It can be seen that our model achieve considerable performance to control robots. (such as turning or manipulating the robotic arm). However, FGDSNet still has some limitations, such as when gestures are made at a distance, the segmentation and recognition results are not significant. This may be due to the low resolution of the input images. In this case, the model's inference speed is fast, but smaller gestures produce fewer pixels in the feature map, which can lead difficulties in segmentation and recognition. Furthermore, between different actions, there are significant variations in gestures. For instance, when an undefined type of gesture occurs, the model often misclassifies this undefined gesture, causing the mobile robot to perform non-ideal actions or lack coherence between actions.

## IV. CONCLUSIONS

Gesture recognition methods based on gesture segmentation usually have high accuracy, but performing gesture segmentation in cluttered backgrounds is a challenging problem. The dual-branch segmentation network processes semantic features and detail features separately, which can capture more effective features. However, it often suffers from low accuracy due to insufficient feature fusion. Inspired by cosine similarity, KL divergence theory, and gate mechanism, we propose the CoSKLAM and GFM. Both are embedded in the middle position of the dual-branch network to effectively fuse semantic features and detail features, and filter features generated at different levels. Based on this, a Fusing Gate Dual Stages Network (FGDSNet) is proposed that uses both the background noise removed gesture segmentation map and the original gesture image as inputs for gesture category prediction. Extensive experiments demonstrate that our network achieves an optimal balance between accuracy and parameter quantity while maintaining real-time inference speed.

It is necessary to establish a new, complex multi-view gesture dataset, while introducing more interfering gestures to prevent gesture misidentification. Additionally, there are plans to modify the model to further enhance its ability to segment and recognize small gestures, as well as to include recognition of dual-hand gestures, meeting diverse practical requirements, and applying it to a wider variety of robotic systems.

## ACKNOWLEDGMENT

We wish to thank compuation supports by Hebei Artificial Intelligence Computing Center.

## REFERENCES

[1] M. Al-Hammadi et al., "Deep Learning-Based Approach for Sign Language Gesture Recognition With Efficient Hand Gesture Representation," IEEE Access, vol. 8, pp. 192527–192542, 2020, doi: 10.1109/ACCESS.2020.3032140.

[2] A. Dadashzadeh, A. T. Targhi, M. Tahmasbi, and M. Mirmehdi, "HGR-Net: a fusion network for hand gesture segmentation and recognition," IET Comput. Vis., vol. 13, pp. 700–707, 2018, doi: 10.1049/iet-cvi.2018.5796.

[3] S. Wang, S. Zhang, X. Zhang, and Q. Geng, "A two-branch hand gesture recognition approach combining atrous convolution and attention mechanism," Vis Comput, pp. 1–14, Jul. 2022, doi: 10.1007/s00371-022-02602-2.

[4] T. L. Dang, T. H. Pham, Q. M. Dang, and N. Monet, "A lightweight architecture for hand gesture recognition," Multimed Tools Appl, vol. 82, no. 18, pp. 28569–28587, Jul. 2023, doi: 10.1007/s11042-023-14550-7.

[5] N. C. Dayananda Kumar, K. V. Suresh, and R. Dinesh, "Depth Based Static Hand Gesture Segmentation and Recognition," in Cognition and Recognition, D. S. Guru, S. K. Y. H., B. K., R. K. Agrawal, and M. Ichino, Eds., in Communications in Computer and Information Science. Cham: Springer Nature Switzerland, 2022, pp. 125–138. doi: 10.1007/978-3-031-22405-8_11.

[6] S. K. Das, R. Lahkar, K. Antariksha, A. Das, A. Bora, and A. Ganguly, "Lightweight Encoder-Decoder Model for Semantic Segmentation of Hand Postures," in Inventive Computation and Information Technologies. Singapore: Springer Nature Singapore, 2023, pp. 579–591.

[7] Y. S. Tan, K. M. Lim, C. Tee, C. P. Lee, and C. Y. Low, "Convolutional neural network with spatial pyramid pooling for hand gesture recognition," Neural Comput & Applic, vol. 33, no. 10, pp. 5339–5351, May 2021, doi: 10.1007/s00521-020-05337-0.

[8] A. Mohanty, S. S. Rambhatla, and R. R. Sahay, "Deep Gesture: Static Hand Gesture Recognition Using CNN," B. Raman, S. Kumar, P. P. Roy, and D. Sen, Eds., in Advances in Intelligent Systems and Computing, vol. 460. Singapore: Springer Singapore, 2017, pp. 449–461. doi: 10.1007/978-981-10-2107-7_41.

[9] D. S. Pamungkas, I. Simatupang, and S. K. Risandriya, "Comparison Gestures Recognition Using K-NN and Naïve Bayes," in 2020 International Conference on Applied Science and Technology (iCAST), Oct. 2020, pp. 677–681. doi: 10.1109/iCAST51016.2020.9557730.

TABLE IV

RECOGNITION PERFORMANCES OF DIFFERENT APPROACHES ON OUHANDS, HGR1 AND NUS-II DATASETS

| Method | OUHANDS | | HGR1 | | NUS-II | | Parameters(M) | FPS |
|---|---|---|---|---|---|---|---|---|
| | F1-score(%) | Acc(%) | F1-score(%) | Acc(%) | F1-score(%) | Acc(%) | | |
| ExtriDeNet(SD)[21] | 64.56 | 65.10 | 93.36 | 93.56 | 98.50 | 98.75 | 1.30 | 0.77 |
| ResNet 50[22] | 81.38 | - | 86.34 | 86.22 | 97.08 | 97.10 | 25.56 | 40.00 |
| DenseNet-121[23] | 82.81 | - | 80.60 | - | 82.81 | 86.43 | 7.98 | 41.67 |
| MobileNet V3[24] | 86.50 | - | 80.60 | - | - | - | 4.22 | 76.92 |
| AUNet(rec)[14] | 90.90 | - | 83.80 | - | - | - | 31.22 | 2.86 |
| EfficientNetB0[1][4] | - | 89.20 | - | **98.15** | - | - | 5.34 | 32.05 |
| EfficientNetB0[2][4] | - | 88.00 | - | - | - | - | 4.60 | 32.87 |
| MobileNetV2 Small[4] | - | - | - | 96.30 | - | - | 0.71 | 47.24 |
| HGRNet[2] | 88.10 | - | - | - | - | - | **0.50** | **192.08** |
| RBI-2RCNN[35] | - | - | - | - | - | 94.80 | 1.76 | - |
| SpAtNet(PD)[36] | - | - | 93.42 | 93.33 | 96.52 | 96.53 | 3.80 | - |
| DeReFNet[33] | - | - | - | - | - | 96.70 | 5.24 | - |
| HyFiNet[32] | - | - | - | 92.00 | - | 97.78 | 2.30 | - |
| mIV3Net[34] | - | - | - | - | - | **99.80** | 16.50 | - |
| FGDSNet | **97.80** | **97.80** | **94.94** | 94.80 | **99.80** | **99.80** | 1.20 | 146.20 |


[10] E. Solly and A. Aldabbagh, "Gesture Controlled Mobile Robot," in 2023 5th International Congress on Human-Computer Interaction, Optimization and Robotic Applications (HORA), Jun. 2023, pp. 1–6. doi: 10.1109/HORA58378.2023.10156800.

[11] J. P. Sahoo, S. P. Sahoo, S. Ari, and S. K. Patra, "Hand Gesture Recognition Using Densely Connected Deep Residual Network and Channel Attention Module for Mobile Robot Control," IEEE Trans. Instrum. Meas., vol. 72, pp. 1–11, 2023, doi: 10.1109/TIM.2023.3246488.

[12] L. Zhou, C. Du, Z. Sun, T. L. Lam, and Y. Xu, "Long-Range Hand Gesture Recognition via Attention-based SSD Network," in 2021 IEEE International Conference on Robotics and Automation (ICRA), May 2021, pp. 1832–1838. doi: 10.1109/ICRA48506.2021.9561189.

[13] M. Peral, A. Sanfeliu, and A. Garrell, "Efficient Hand Gesture Recognition for Human-Robot Interaction," IEEE Robot. Autom. Lett., vol. 7, no. 4, pp. 10272–10279, Oct. 2022, doi: 10.1109/LRA.2022.3193251.

[14] S. Wang, S. Zhang, X. Zhang, and Q. Geng, "A two-branch hand gesture recognition approach combining atrous convolution and attention mechanism," Vis. Comput., pp. 1–14, Jul. 2022, doi: 10.1007/s00371-022-02602-2.

[15] A. A. Barbhuiya, R. K. Karsh, and R. Jain, "ASL Hand Gesture Classification and Localization Using Deep Ensemble Neural Network," Arab J Sci Eng, vol. 48, no. 5, pp. 6689–6702, May 2023, doi: 10.1007/s13369-022-07495-w.

[16] R. Deng, C. Shen, S. Liu, H. Wang, and X. Liu, "Learning to predict crisp boundaries." arXiv, Jul. 26, 2018. doi: 10.48550/arXiv.1807.10097.

[17] Y. Hong, H. Pan, W. Sun, and Y. Jia, "Deep Dual-resolution Networks for Real-time and Accurate Semantic Segmentation of Road Scenes." arXiv, Sep. 01, 2021. doi: 10.48550/arXiv.2101.06085.

[18] J. Xu, Z. Xiong, and S. P. Bhattacharyya, "PIDNet: A Real-time Semantic Segmentation Network Inspired by PID Controllers." arXiv, Apr. 06, 2023. doi: 10.48550/arXiv.2206.02066.

[19] E. Xie, W. Wang, Z. Yu, A. Anandkumar, J. M. Alvarez, and P. Luo, "SegFormer: Simple and Efficient Design for Semantic Segmentation with Transformers." arXiv, Oct. 28, 2021. doi: 10.48550/arXiv.2105.15203.

[20] J. Peng et al., "PP-LiteSeg: A Superior Real-Time Semantic Segmentation Model." arXiv, Apr. 06, 2022. doi: 10.48550/arXiv.2204.02681.

[21] G. Bhaumik, M. Verma, M. C. Govil, and S. Vipparthi, "ExtriDeNet: an intensive feature extrication deep network for hand gesture recognition," The Visual Computer, vol. 38, pp. 3853–3866, 2021, doi: 10.1007/s00371-021-02225-z.

[22] K. He, X. Zhang, S. Ren, and J. Sun, "Deep Residual Learning for Image Recognition." arXiv, Dec. 10, 2015. doi: 10.48550/arXiv.1512.03385.

[23] G. Huang, Z. Liu, L. van der Maaten, and K. Q. Weinberger, "Densely Connected Convolutional Networks." arXiv, Jan. 28, 2018. doi: 10.48550/arXiv.1608.06993.

[24] A. G. Howard et al., "MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications." arXiv, Apr. 16, 2017. doi: 10.48550/arXiv.1704.04861.

[25] B. Karsh, R. H. Laskar, and R. K. Karsh, "mIV3Net: modified inception V3 network for hand gesture recognition," Multimed. Tools Appl., Jun. 2023, doi: 10.1007/s11042-023-15865-1.

[26] G. Bhaumik, M. Verma, M. C. Govil, and S. K. Vipparthi, "HyFiNet: Hybrid feature attention network for hand gesture recognition," Multimed. Tools Appl., vol. 82, no. 4, pp. 4863–4882, Feb. 2023, doi: 10.1007/s11042-021-11623-3.

[27] A. Gomez Chavez, A. Ranieri, D. Chiarella, and A. Birk, "Underwater Vision-Based Gesture Recognition: A Robustness Validation for Safe Human–Robot Interaction," IEEE Robot. Autom. Mag., vol. 28, no. 3, pp. 67–78, Sep. 2021, doi: 10.1109/MRA.2021.3075560.

[28] A. Gomez Chavez, A. Ranieri, D. Chiarella, E. Zereik, A. Babić, and A. Birk, "CADDY Underwater Stereo-Vision Dataset for Human–Robot Interaction (HRI) in the Context of Diver Activities," J. Mar. Sci. Eng., vol. 7, no. 1, Art. no. 1, Jan. 2019, doi: 10.3390/jmse7010016.

[29] H. Zhang, H. Zhang, C. Wang, and J. Xie, "Co-Occurrent Features in Semantic Segmentation," in 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Jun. 2019, pp. 548–557. doi: 10.1109/CVPR.2019.00064.

[30] W. Wang, K. Yu, J. Hugonot, P. Fua, and M. Salzmann, "Recurrent U-Net for Resource-Constrained Segmentation," in 2019 IEEE/CVF International Conference on Computer Vision (ICCV), Oct. 2019, pp. 2142–2151. doi: 10.1109/ICCV.2019.00223.

[31] T.-H. Tsai and S.-A. Huang, "Refined U-net: A new semantic technique on hand segmentation," Neurocomputing, vol. 495, pp. 1–10, Jul. 2022, doi: 10.1016/j.neucom.2022.04.079.

[32] G. Bhaumik, M. Verma, M. C. Govil, and S. K. Vipparthi, "HyFiNet: Hybrid feature attention network for hand gesture recognition," Multimed. Tools Appl., vol. 82, no. 4, pp. 4863–4882, Feb. 2023, doi: 10.1007/s11042-021-11623-3.

[33] J. P. Sahoo, S. P. Sahoo, S. Ari, and S. K. Patra, "DeReFNet: Dual-stream Dense Residual Fusion Network for static hand gesture recognition," Displays, vol. 77, p. 102388, Apr. 2023, doi: 10.1016/j.displa.2023.102388.

[34] B. Karsh, R. H. Laskar, and R. K. Karsh, "mIV3Net: modified inception V3 network for hand gesture recognition," Multimed. Tools Appl., Jun. 2023, doi: 10.1007/s11042-023-15865-1.

[35] J. P. Sahoo, S. P. Sahoo, S. Ari, and S. K. Patra, "RBI-2RCNN: Residual Block Intensity Feature using a Two-stage Residual Convolutional Neural Network for Static Hand Gesture Recognition," Signal Image Video Process., vol. 16, no. 8, pp. 2019–2027, Nov. 2022, doi: 10.1007/s11760-022-02163-w.

[36] G. Bhaumik and M. C. Govil, "SpAtNet: a spatial feature attention network for hand gesture recognition," Multimed. Tools Appl., Oct. 2023, doi: 10.1007/s11042-023-16988-1.