IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 3 March 2023, accepted 21 March 2023, date of publication 28 March 2023, date of current version 3 April 2023.

*Digital Object Identifier 10.1109/ACCESS.2023.3262560*

RESEARCH ARTICLE

# 3D Point Cloud Semantic Segmentation System Based on Lightweight FPConv

**YU-CHENG FAN[1], (Senior Member, IEEE), KUAN-YU LIAO[1], YOU-SHENG XIAO[2], MIN-HUA LU[3], AND WEI-ZHE YAN[4]**

[1]Department of Electronic Engineering, National Taipei University of Technology, Taipei 10608, Taiwan
[2]Richtek Technology Corporation, Hsinchu 30288, Taiwan
[3]Taiwan Semiconductor Manufacturing Company, Hsinchu 30078, Taiwan
[4]Macronix International Company Ltd., Hsinchu 30078, Taiwan

Corresponding author: Yu-Cheng Fan (skystar@ntut.edu.tw)

This work was supported by the Ministry of Science and Technology of Taiwan under Grant MOST 110-2221-E-027-084-MY3.

**ABSTRACT** In this paper, we proposed a 3D point cloud semantic segmentation system based on lightweight FPConv. In 3D point cloud mapping, data is depicted in a 3D space to represent 3D imagery data. These maps are collected through direct measurements; all points in a 3D point cloud map correspond to a measurement point and, therefore, contain a large amount of data. Data in 3D point cloud maps are stored in point clouds, and they are extracted using 3D image processing or deep learning. However, because of the non-structured and high-dimensional properties of point clouds, the development of 3-D image recognition applications in the field of computer vision warrants further exploration. Large-scale neural networks are highly accurate, but they have the disadvantages of high computation complexity and low portability. Therefore, the present study proposed a 3D point cloud semantic segmentation system based on lightweight FPConv. The proposed network combines depth-wise separate convolution, quantization, and Winograd convolution technology to lighten and accelerate neural network computation. The performance of the presented network was verified using the Stanford 3D Large-Scale Indoor Spaces (S3DIS) large scene database provided by Stanford 3D AI Lab. The results reveal the excellent performance of the proposed model.

**INDEX TERMS** 3D point cloud, FPConv, lightweight, smart cities, semantic segmentation.

## I. INTRODUCTION

3D point cloud maps are generated using laser measurements or RGB-D (red–green–blue depth) camera measurements. 3D LiDAR is a crucial instrument for 3D point cloud scanning. LiDAR is an advanced sensor that employs lasers to scan for spatial data. After the invention of lasers, optical radars were first used during the Apollo 15 mission in 1971 to survey and map the surface of the moon. Optical radars were subsequently applied in fields including those archaeology and agriculture. In 2005, optical radars were integrated into automobiles to serve as ''eyes'' that enable autonomous cars to detect the surrounding environment and avoid obstacles. Point cloud maps generated using 3D LiDAR consist of highly accurate in-depth information, which provides vehicles with sufficient time to respond to potential threats [1], [2].

In addition to its application in autonomous cars, 3D LiDAR has in recent years been applied in consumer electronics products (e.g., the iPhone 12 Pro series and iPhone 12 Pro Max series released in 2020). For the iPhone 12 Pro series, the LiDAR System was applied to support the phone's night portrait camera settings and the focus speed of its night camera mode. Additionally, built-in measurement applications employ LiDAR to measure body height, generate a 3D model of a classroom interior environment, and measure the distance between a user and other individuals. These applications can assist individuals with visual impairments (e.g., individuals with blindness or those who are partially sighted) by employing the information obtained through LiDAR to provide visual support in the daily lives of individuals with visual impairments.

The associate editor coordinating the review of this manuscript and approving it for publication was Yoonsik Choe.

3D point cloud mapping is a common representation format for 3D images [3]. This format retains original geometry data in a 3D space without the use of discretization processing. Therefore, 3D point cloud mapping is employed in various scene understanding applications, of which the most prominent are autonomous vehicles and 3D environmental monitoring applications in smart cities. In recent years, deep learning technology has contributed to the development of various research fields, including visual computing, speech recognition, and natural language processing. However, the challenges for deep learning include the small datasets, high dimensions, and non-structured 3D point clouds used by related applications. Alternatively, large neural networks (NNs) have high accuracy, but they are affected by high computational complexity and low portability. Therefore, researchers must weigh the pros and cons of 3D point cloud mapping or large NNs before implementing these technologies.

This study combined projection-based and point-based methods to perform large-scale 3D point cloud semantic segmentation. Subsequently, quantization and depth-wise separable convolution were conducted to perform the lightweight compression of the network.

## II. LITERATURE REVIEW

### A. 3D POINT CLOUD SEGMENTATION

3D point cloud segmentation requires data on the overall geographical structure and details of each point. The purpose of segmentation is to divide a structure into multiple sets of neighboring points and to divide these points based on segmentation granularity through semantic segmentation (scene level), instance segmentation (target level), and partial segmentation (part level). Guo et al. [4] discussed all available 3D point cloud semantic segmentation methods and divided these methods into projection-based, discretion-based, point-based, and hybrid methods.

#### 1) PROJECTION-BASED METHODS

In projection-based methods, 3D point cloud maps are projected onto a 2D image through methods such as multi-view representation [5], [6], [7], [8] and spherical representation [9], [10], [11], [12]. Multiview representation is sensitive to occlusion and observation angle of view, and it does not thoroughly utilize the structural data at the base layer, which leads to data loss [5], [6], [7], [8]. By contrast, spherical representation retains relatively more data and is more suitable for LiDAR point cloud labelling, but it is vulnerable to errors, discretization, and occlusion [9], [10], [11], [12].

#### 2) POINT-BASED METHODS

The first point-based method was proposed by Qi et al. [13], which developed PointNet, an NN that directly computes point clouds through unstructured and unordered algorithms.

Hua et al. [14] proposed a pointwise CNN in which nearest neighbors are queried and binned into kernel cells and subsequently convoluted with kernel weights. Wang et al. [15] proposed parametric continuous convolutional networks, which perform parametric continuous convolution by using the kernel function of each convolution layer to formulate continuous vector spaces.

The recurrent neural network (RNN), which captures the contextual features of a 3D point cloud, is commonly applied in 3D point cloud semantic segmentation models. Engelmann et al. [16] expanded PointNet and converted a set of points into multiscale and multigrid models to obtain the contextual features of an input end [17]. To overcome the problems caused by static and rigid pooling, Zhao et al. [18] proposed a dynamic aggregation network that considers global scene complexity and local geometric features.

Landrieu and Simonovsky [19] used interconnected simple shapes and Superpoints to represent 3D point clouds and attributedirected graphs (i.e., Superpoint graphs) to acquire structural and contextual information.

### B. PointNet++

PointNet [13] exhibits local feature processing disadvantages. Based on this concept, Qi proposed the PointNet++ [20] framework. PointNet++ employs a hierarchical NN structure to resolve the aforementioned problems. However, PointNet++ is also affected by the sampling density problem [20]. Because sampling in low-density positions may result in the loss of local data, the sampling scale must be increased.

### C. RandLA-NET

RandLA-Net is a lightweight and high-efficiency novel algorithm that is applicable for large-scale 3D point cloud scenes. Hu et al. [21] performed a comprehensive analysis of current sampling methods and employed random point sampling in point clouds to significantly reduce computing load and memory consumption.

### D. KPConv

Kernel point convolution (KPConv) [22] is different from the aforementioned point cloud processing methods. The convolution of the features of point cloud x is defined as:

$$(\mathrm{F} * \mathrm{g})(x) = \sum_{x_i \in N_x} g(x_i - x) fi \tag{1}$$

where $x_i$ and $f_i$ represent the point cloud in point set $\mathcal{P} \in \mathbb{R}^{\mathrm{N}\times 3}$ and the corresponding feature (i.e., the aggregation of $N$ points) in feature set $\mathcal{F} \in \mathbb{R}^{\mathrm{N}\times \mathrm{D}}$, respectively. The kernel function $\mathrm{g}(y_i) = \sum_{k<K} h(y_i, \tilde{x}_k) W_k$ is defined as a regular 2D matrix, which is pointwise multiplied by convolution layers and added to obtain a total. $g$ takes the neighboring positions centered on $x$ as input and is represented as $y_i = x_i - x, y_i \in \beta_\gamma^3$. The domain of the definition of $g$ is $\beta_\gamma^3 = \left\{ y \in R^3 \,|\, \|y\| \leq r \right\}$, providing different weights to g at different layers. Let $\{\tilde{x}_k \,|k < K\} \subset \beta_\gamma^3$ be the kernel points

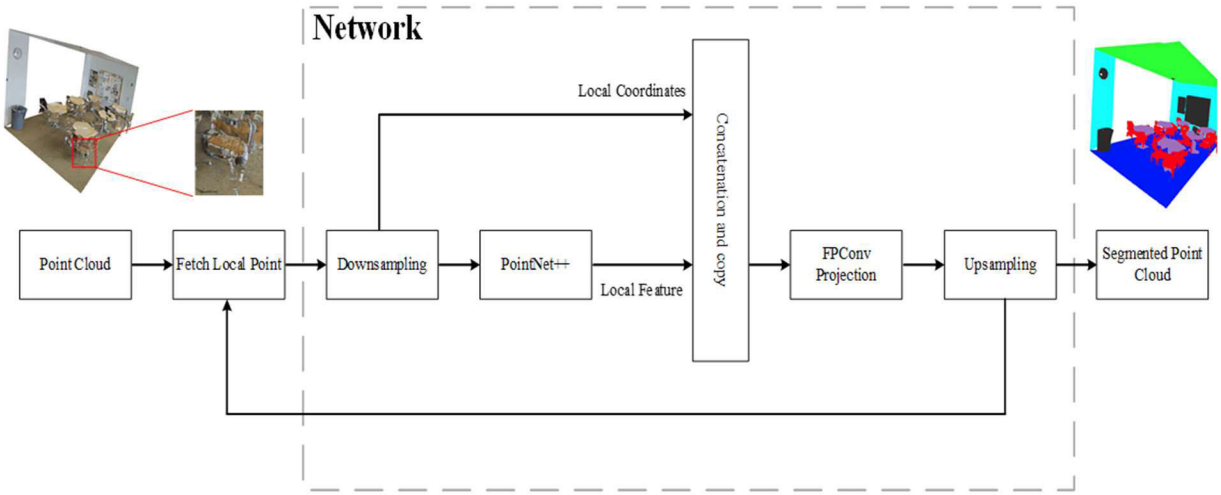


**FIGURE 1.** System architecture diagram.

and $\{W_k \,|k < K\} \subset \mathbb{R}^{D_{in} \times D_{out}}$ be the associated weight matrices [22].

$h$ represents the linear correlation between $\tilde{x}_k$ and $y_i$, and it is used to evaluate the degree of influence of each point cloud; $\sigma$ represents the influence distance of the kernel points, and it is selected according to input density. The linear correlation is thus presented using the following equation [22]:

$$\mathrm{h}\left(y_i, \tilde{x}_k\right) = max\left(0, 1 - \frac{\|y_i - \tilde{x}_k\|}{\sigma}\right). \tag{2}$$

Because the number of kernel points (K) is not constrained, the design of KPConv is highly flexible and adaptable for various domains. The radius of the sphere is set as $1.5\sigma$ to ensure a certain degree of overlap between each kernel point space. Because each convolution layer consists of a set of points ($\tilde{x}_k$), the researchers employed $\{\tilde{x}_k\}$ to represent the global settings for each convolution layer. The researchers established a set of x shifts ($\Delta(x)$) for every convolution location and defined deformable KPConv using the following equation [22]:

$$(\mathrm{F} * \mathrm{g})(x) = \sum_{x_i \in N_x} g_{deform}\left(x_i - x, \Delta(x)\right) fi, \tag{3}$$

where

$$g_{deform}\left(y_i, \Delta(x)\right) = \sum\nolimits_{k<K} h(y_i, \tilde{x}_k + \Delta_k(x)) W_k.$$

### E. FPConv

FPConv [23] differs from convolution performed through projection on a tangent plane [24], [25], [26], [27], [28]. Points in the local region are projected onto a 2D grid, and 2D convolution is performed on the grid. This method is similar to the projection–interpolation method, but it is more versatile. In FPConv, projection and interpolation are simplified into a single-weight map learning process. This general and robust process can be effectively integrated with various model frameworks for point cloud classification, semantic segmentation, and 3D analysis tasks.

## III. PROPOSED METHOD

The present study proposed an NN for direct point cloud computing that integrates RandLA-Net; MobileNet, which is fast and lightweight; and FPConv, which provides high accuracy. The proposed NN was designed to achieve high-speed semantic segmentation without substantially compromising accuracy. The performance of the proposed network was verified using the Stanford 3D Large-Scale Indoor Spaces (S3DIS) large scene database [29] provided by Stanford 3D AI Lab.

### A. SYSTEM FRAMEWORK

The system framework is presented in Fig. 1. First, segments of a large scene were extracted and input into the proposed NN. The input segments were then randomly sampled, and their point cloud coordinates were collected. The PointNet++ framework was then employed to obtain the features of the segments and concatenate the features with the corresponding coordinates. After the features were projected on a 2D plane, 2D convolution was performed to extract point features. Finally, nearest-neighbor interpolation was conducted for up-sampling to complete the semantic segmentation process of the large scene.

### B. TRAINING DATASET

The acquisition of point cloud data was initially a challenge. However, in recent years, the use of a large number of training datasets has become essential to improving the accuracy of artificial intelligence (AI). This has prompted the public release of point cloud datasets for use in relevant research and applications. This study employed the S3DIS database to train the proposed network framework. The dataset that was used is further detailed in the following sections.

#### 1) STANFORD 3D LARGE-SCALE INDOOR SPACES

The S3DIS dataset was developed by the Stanford Vision and Learning Lab. The dataset was collected using a Matterport camera (a scanner that combines three types of spacing) from six large-scale indoor areas in three buildings. Data (including 3D coordinate and RGB-D data), were reconstructed after scanning, and point clouds were generated based on samples by using their coordinates. In particular, the data on Areas 1, 3, and 6; Areas 2 and 4; and Area 5 were collected from Buildings 1, 2, and 3, respectively. The buildings contained 271 rooms total, spanned a land area of 6000 $m^2$, and provided more than 215 million points. Each of these points contained one instance-level semantic segmentation label selected from 13 categories, which included *chair*, *table*, *floor*, and *wall*.

### C. NETWORK LIGHTENING

During the network training of mainstream deep-learning frameworks (including Tensorflow and Pytorch), weight, deviation, and activation functions are generally recorded using 32-bit full floating (i.e., full precision 32 [FP32]). However, for large-scale and deep networks, the training process involves substantial variables, which require a computing load that exceeds the capacity of portal devices. Therefore, this paper employed depth-wise separable convolution to reduce the number of variables and adopted symmetric quantization to obtain low-precision weights and deviations. Finally, the base layer, which has low influence, is eliminated, and unnecessary activation layers and pipelines are removed to reduce the computing load.

#### 1) DEPTHWISE SEPARABLE CONVOLUTION

Under traditional convolution, a substantial computing loading is required. The computing process of depth-wise separable convolution differs from that of conventional convolution. Because depth-wise separable convolution involves two steps, namely depth-wise convolution, and pointwise convolution. Under depth-wise separable convolution, a substantially lower computing load is required relative to conventional convolution. The computing load was further reduced when the depth of the output channel and the size of the convolution kernel increased. Therefore, depth-wise separable convolution can be applied to solve the large number of variables involved in large-scale NNs.

In depth-wise separable convolution, each depth-wise convolution computes one channel; therefore, convolution computing is conducted in each channel using a k × k size convolution kernel. The convolution computation is performed independently in each channel of the input layer; thus, the signal features located at a given position in each channel are not thoroughly used. Consequently, pointwise convolution is necessary to concatenate a feature map output obtained through depth-wise convolution.

The computation process of depth-wise separable convolution can be explained using input channels. First, an input is considered as an input feature map consisting of $M$ input channels with a convolution kernel size of k × k. Because the number of convolution kernels is equal to the number of input channels, the input feature map consists of M convolution kernels. After depth-wise convolution computation is completed, an $M$ number of feature maps (equal to the number of input channels) is output. Because the expected output is an output feature map consisting of two output channels, two 1 × 1 × M convolution kernels are employed to perform pointwise convolution with the feature map output obtained through depth-wise convolution. The practical computation process is presented in Fig. 2.

#### 2) QUANTIZATION

Symmetric quantization was employed to transform a 32-bit floating point number into an 8-bit fixed-point number for computation, thereby reducing computing load by 75%. In Eq. (4), $\boldsymbol{s}$ represents the scale factor, and $F_{32}$ represents pre-quantized values. Because the data type exhibits 8-bit accuracy, $D = 256$. Finally, the dequantization process, in which the quantized $y_Q$ value is divided by the scale factor $\boldsymbol{s}$ to obtain the dequantization results, is conducted (Eq. (4)). Fig. 3 illustrates the quantization and dequantization process.

$$y_Q = \text{round}\left(\text{clamp}(-D/2, D/2-1, \text{round}\left(F_{32} \times \text{s}\right))/\text{s}\right) \tag{4}$$

### D. FEATURE AGGREGATION MODULE

RandLA-Net [21] is a high-efficiency and lightweight network for large-scale point cloud segmentation. The network employs random sampling to achieve highly efficient memory access and computation. The present study adopted a local feature aggregation module to collect and preserve geometric features. The design framework of the local feature aggregation module is presented in Fig. 4. First, input point cloud 3D coordinate signals were encoded. The local features of the unit output were subsequently aggregated using attentive pooling. Finally, FPConv was allowed to learn the matrix parameters of the weight projection, and random sampling was employed to increase computation speed.

Attentive pooling was conducted for the local feature point set $\hat{F}_i = \left\{\hat{f}_i^1 \ldots \hat{f}_i^k \ldots \hat{f}_i^K\right\}$. First, a common function g (·) was designed to provide an independent attention score for each learning point. Let $s_i^k = g\left(\hat{f}_i^i, W\right)$, and $W$ represent the learnable parameters of a shared MLP. Subsequently, learned attention scores were used to form a soft mask that could automatically select key features. The obtained features are the sum of the neighboring characteristics (Eq. (5)).

$$\hat{f}_i = \sum_{k=1}^{K}\left(\hat{f}_i^K \cdot s_i^k\right) \tag{5}$$

1. FPConv_Base local flattening by learning projection weights: $\pi(\cdot)$ are N (p) dispersion points projected onto the S point. The total signal function S(u) is obtained using

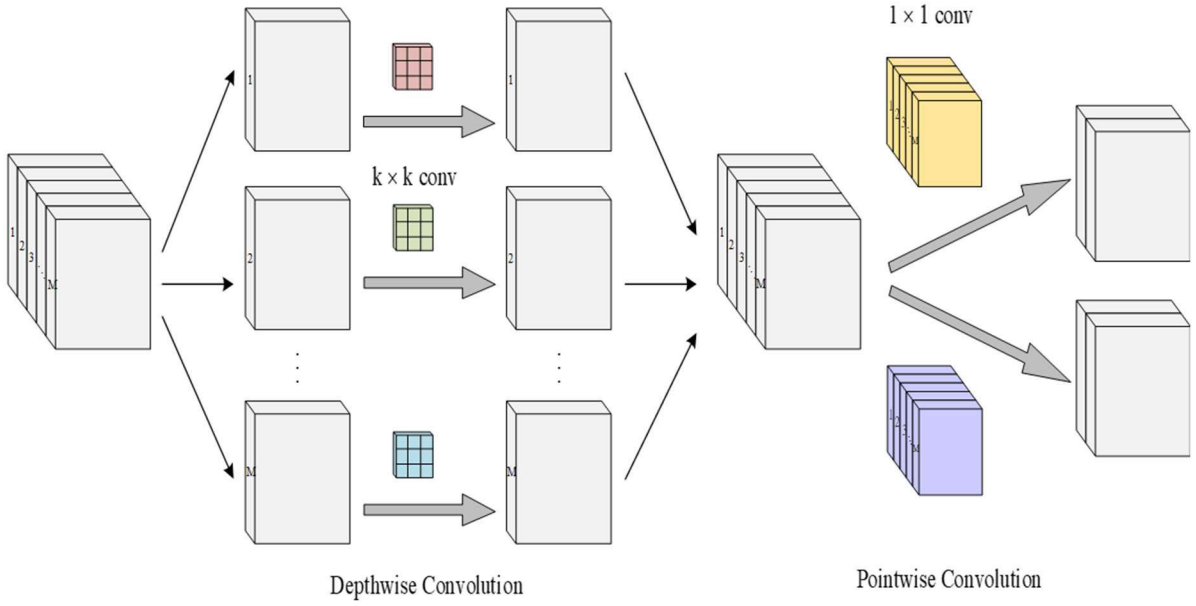


**FIGURE 2.** Depth-wise Separable Convolution Operation.

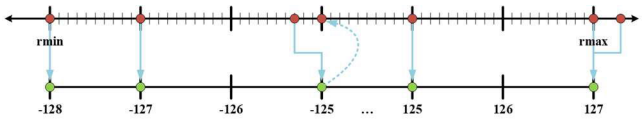


**FIGURE 3.** Symmetric quantization and dequantization.

interpolation (Eq. (6)).

$$S(u) = \sum_i \mathrm{w}(\mathrm{u}, \pi(q_i))\, S(\pi(q_i)) \tag{6}$$

Let S be an $M_w \times M_h$ grid plane. For every grid $S(v_j)$, where $\{1, 2, \ldots, M_w \times M_h\}$, Eq. (7) and (8) are derived as follows:

$$S(v_j) = \sum_{ji} F(q_i) \tag{7}$$

$$w_{ji} = w(v_j, \pi(q_i)) \tag{8}$$

Through the application of approximate discrete equations, the equations are transformed into Eqs. (9), (10), and (11):

$$x(p) = \int_S C(u)\, S(u)\, du = M_c * \left(W_f^T \times F(p)\right) \tag{9}$$

$$L = M_w \times M_h, AW_f \in R^{N \times L}, AW_f(i, \mathrm{j}) = \mathrm{w}(v_j, \pi(q_i)) \tag{10}$$

$$F(p) = (F(q_1), \ldots, F(q_N))^T \in R^{N \times C} \tag{11}$$

2. Farthest point sampling (FPS): Iteration FPS was performed to down-sample point clouds. Relative to random sampling, FPS covers an overall pointset more effectively for a given number of centroids. The operation process of FPConv with FPS is similar to that of merging operations, therefore FPConv was applied to each point in the down-sampled point cloud to search for the neighbours of each point in the point cloud.

$$F_{out}(y_i) = FPConv(F(P_{neb})) \tag{12}$$

### E. NEURAL NETWORK ARCHITECTURE

The NN architecture proposed in the present study is presented in Fig. 5. First, point cloud data was input into the proposed NN. N and D represent the number of points and the feature dimension, respectively. Random sampling, local feature aggregation module, and FPConv weight projection matrix were subsequently combined. The input point cloud was subjected to continuous down-sampling in the FPConv_Res with RS block to conserve computing resources and data storage. Before random down-sampling, FPConv was conducted to learn the corresponding weight projection matrix. Because all modules in the network comprised feed-forward MLPs that were simple and highly efficient, the network provided high computation efficiency. Subsequently, a novel network framework was constructed based on an encoder-decoder structure, and trained feature values were input into the last fully connected layer. This layer computed the category classification probabilities of each point, and the category with the highest probability was determined as the classification result. Finally, nearest neighbour interpolation was performed during decoder up-sampling to further increase computation efficiency.

### F. WINOGRAD CONVOLUTION ACCELERATOR

Because the proposed NN performs the multiplication accumulation computation of numerous weights and the input feature values, the process requires considerable memory

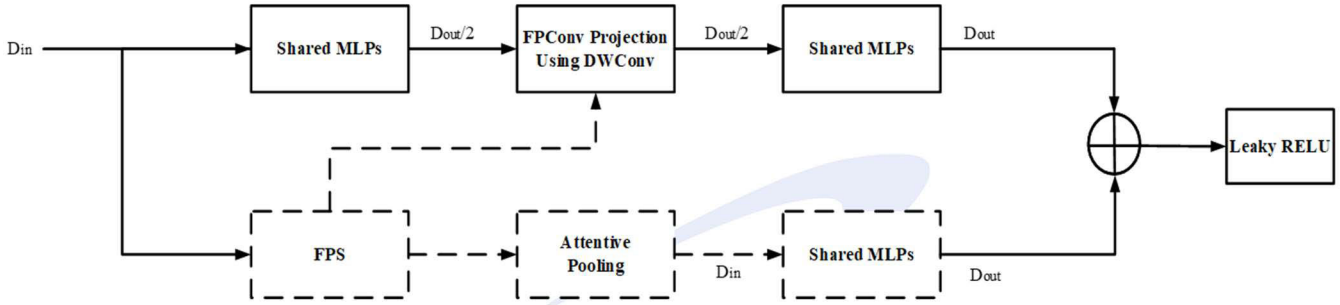


**FIGURE 4.** Feature aggregation module.

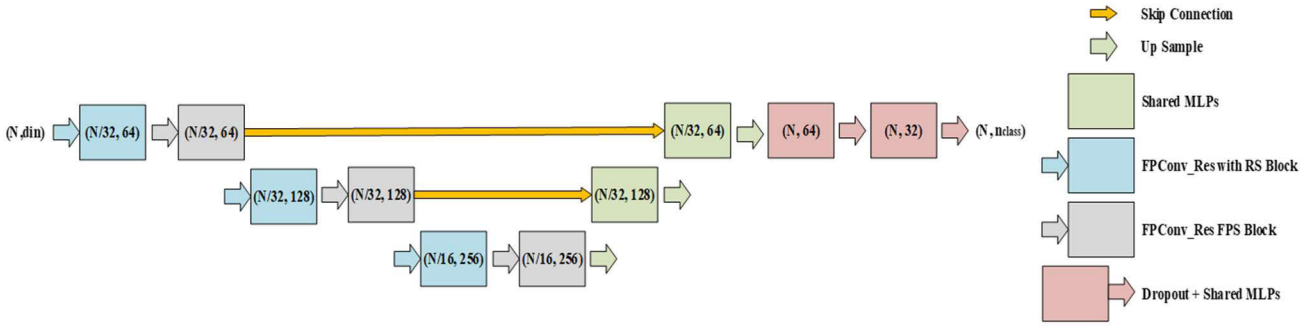


**FIGURE 5.** Network architecture diagram.

and operation time. Therefore, the integration of an acceleration operator into the hardware circuit framework is beneficial. Accordingly, the architecture was designed to include a Winograd convolution circuit that replaced the standard convolution computing process and the separable convolution computing process of FPConv. The Winograd convolution accelerator performs a convolution in batches using tiling as the unit and employs data reuse and pipeline frameworks to increase computation speed. □

This section details the software algorithm design of the 3D point cloud semantic segmentation system and fast convolution design based on lightweight FPConv. Notably, the algorithm framework can be divided into two sections. In the first section, FPConv was transformed into depth-wise separable convolution, and symmetric quantization was performed to lighten the model. In the second section, point clouds were extracted from large-scale scenes, randomly sampled, and input into a feature aggregation module and the NN architecture to complete the semantic segmentation of a large-scale scene. The results are presented in Fig. 6.

## IV. EXPERIMENT METHOD AND RESULTS

This section discusses the testing of the 3D point cloud semantic segmentation system and fast convolution design based on lightweight FPConv and the comparison of relevant data.

**TABLE 1.** Data type.

| Precision | Dynamic Range | Minimum Positive Number |
|---|---|---|
| FP32 | $-3.4 \times 10^{38} \sim 3.4 \times 10^{38}$ | $1.4 \times 10^{-45}$ |
| FP16 | $-65504 \sim 65504$ | $5.96 \times 10^{-8}$ |
| INT8 | $-128 \sim 127$ | 1 |

### A. EXPERIMENT ENVIRONMENT

NN training is highly complex and time-consuming. To effectively increase training speed, GPU and high-capacity memories were employed for parallel processing and computation, respectively. The GPU comprised two NVIDIA GeForce GTX 1080 Ti graphics cards. Through the use of an Ubuntu 18.04 operating system environment, NN training and simulation were conducted using the S3DIS 3-D point cloud data set provided by Stanford. Finally, the 3D point cloud class cation results were presented using Open3D. The experiment simulation environment is presented in Table 2.

### B. EXPERIMENT RESULTS

Currently, two methods can be used to perform NN training and verification using the S3DIS dataset. Under the first method, training is performed using Areas 1, 2, 3, 4, and 5, after which verification is conducted using Area 5. Under the second method, 6-fold cross-validation is performed. In the present study, the first method is employed for training and

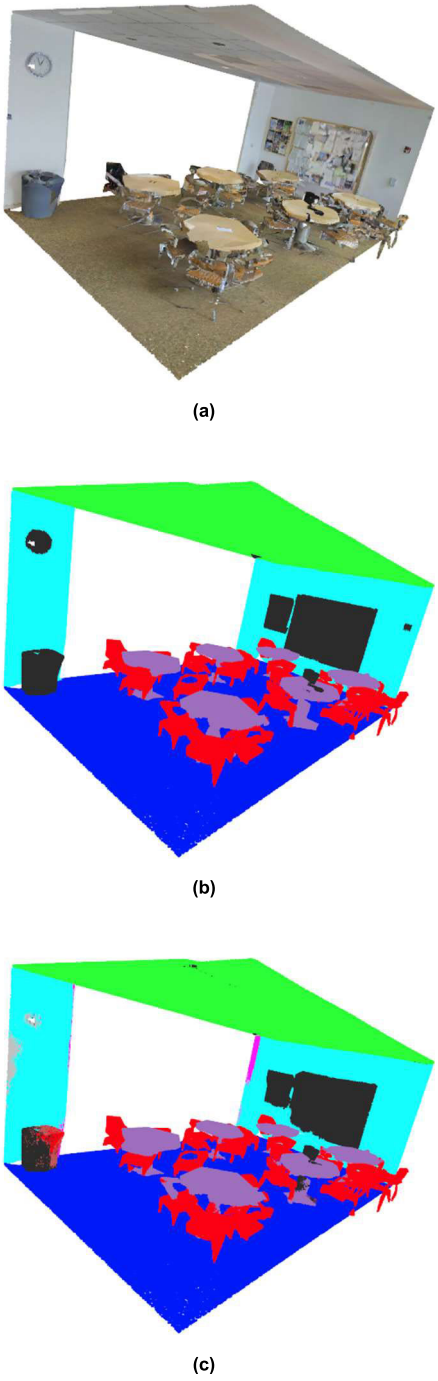


**FIGURE 6.** (a) Original point cloud: Area5_Lobby1; (b) Semantic labeling: Area5_Lobby1; (c) Semantic segmentation: Area5_Lobby1.

verification. The numeric distribution for each scene adopts the list in the S3DIS dataset. The data of Area 5 is input into the trained NN for verification, intending to distinguish each object in the 3-D point cloud map by color (Table 3). Finally, Open3D is adopted to visualize the classification results of the 3-D point cloud map.

Because Area 5 consists of 68 different scenes, 10 different large-scale scenes were randomly selected for verification.

**TABLE 2.** Experimental environment.

| Simulation Environment | |
|---|---|
| OS | Ubuntu 18.04 |
| CPU | Intel®Core™i7-7700K @ 4.20GHz |
| GPU | NVIDIA GeForce GTX 1080 Ti × 2 |
| RAM | 64GBytes |
| Dataset | S3DIS |
| Visualize | Open3D |

After color classification, the visualized results of the 3D point cloud map are classified into points, labels, and predictions for comparison.

### C. EXPERIMENT COMPARISON

To evaluate the accuracy of the semantic segmentation system, numerous standards were employed to evaluate algorithm accuracy. Therefore, before data comparison, a detailed explanation must be provided on the standards used for result evaluation, which are overall accuracy (oA), mean accuracy (mAcc), and mean intersection over union (mIoU).

Let the number of classes be k + 1 (from $L_0 - L_k$, including one void class or background). $p_{ij}$ represents the number of $i$-class pixels determined as a $j$-class pixel. Accordingly, $p_{ii}$, $p_{ij}$, and $p_{ji}$ represent true positives, false positives, and false negatives, respectively.

oA is the simplest evaluation standard, and it is used to evaluate overall model accuracy. In general, oA only calculates the number of points that are correctly classified and divides this number by the total number of pixels (Eq. (13)).

$$\text{oA} = \frac{\sum_{i=0}^{k} p_{ii}}{\sum_{i=0}^{k} \sum_{j=0}^{k} p_{ij}} \tag{13}$$

mAcc is an improved version of oA. This evaluation standard individually calculates classification accuracy and then obtains a mean value (Eq. (14)).

$$\text{mAcc} = \frac{1}{k+1} \sum_{i=0}^{k} \frac{p_{ii}}{\sum_{j=0}^{k} p_{ij}} \tag{14}$$

mIoU is the most commonly used evaluation standard in semantic segmentation. mIoU calculates the intersection-over-union (IoU) ratio of two sets. In semantic segmentation, these two sets represent the ground truth and predicted segmentation. The IoU ratio can be rewritten as the true positives divided by the sum of true positives, false negatives, and false positives. The IoU of each class is first computed, and the mIoU is then determined, (Eq. (15)).

$$\text{mIoU} = \frac{1}{k+1} \sum_{i=0}^{k} \frac{p_{ii}}{\sum_{j=0}^{k} p_{ij} + \sum_{j=0}^{k} p_{ji} - p_{ii}} \tag{15}$$

After the evaluation standards are introduced, a comparison of the proposed method and other point-based methods was conducted. The present study compared the accuracy of the proposed network with that of networks proposed by other studies (Table 4). The x-axis presents the network

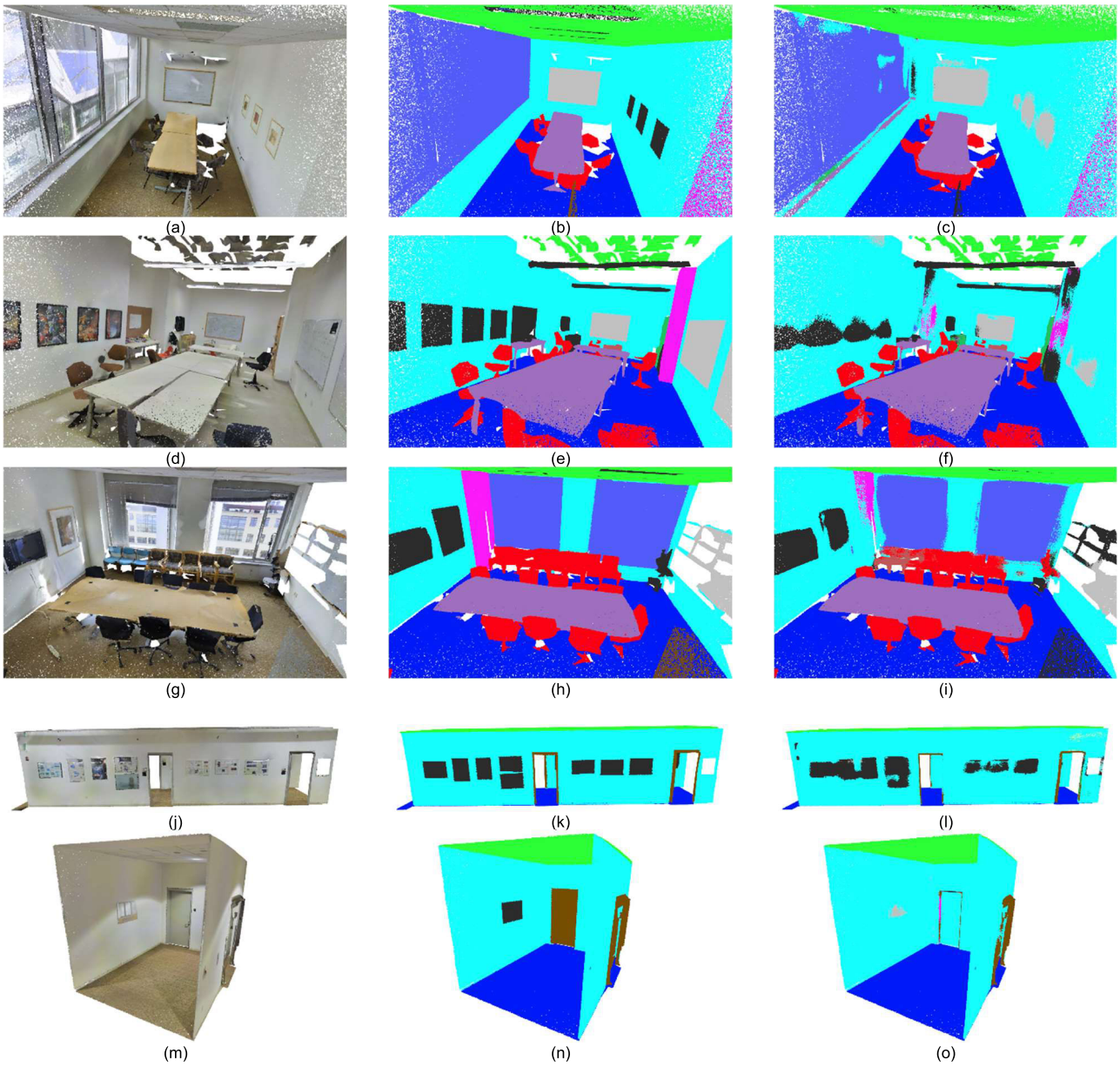


**FIGURE 7.** (a) Original point cloud: Area5_Conference Room1; (b) Semantic labeling: Area5_Conference Room1; (c) Semantic segmentation: Area5_Conference Room1; (d) Original point cloud: Area5_Conference Room2; (e) Semantic labeling: Area5_Conference Room2; (f) Semantic segmentation: Area5_Conference Room2; (g) Original point cloud: Area5_Conference Room3; (h) Semantic labeling: Area5_Conference Room3; (i) Semantic segmentation: Area5_Conference Room3; (j) Original point cloud: Area5_Hallway6; (k) Semantic labeling: Area5_Hallway6; (l) Semantic segmentation: Area5_Hallway6; (m) Original point cloud: Area5_Hallway8; (n) Semantic labeling: Area5_Hallway8; (o) Semantic segmentation: Area5_Hallway8.

models (comprising the proposed model and those proposed by other studies), and the y-axis compares their accuracy. Overall network accuracy was evaluated using mAcc and mIoU. Notably, the oA, mAcc, and mIoU of the proposed network were 86.8%, 67.2%, and 61.9%, respectively. The proposed network exhibited outstanding accuracy relative to PointNet, SegCloud, and SPGraph. In the mIoU comparison (In TABLE 4), our proposed method (FP32) outperforms PointNet [13] by 21.30%, SegCloud [30] by 13.50%, and SPGraph [19] by 4.40%. In the mAcc comparison (In TABLE 4), the proposed method (FP32) is 19.50% better than PointNet [13], 11.10% better than SegCloud [30], and 2.00% better than SPGraph [19]. This experiment proves that our proposed method has good mIoU and mAcc with high accuracy.

Subsequently, the object recognition accuracy level of the proposed network and those of the networks proposed by other studies were compared. Table 5 presents the comparison results; the x-axis lists the compared network models, and the y-axis compares the accuracy of the models in terms of their

**TABLE 3.** Color table for each object category.

| Ceiling | Floor | Wall | Beam | Column | Window | Door | Table | Chair | Sofa | Bookcase | Board | Clutter |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | | | | | | |

**TABLE 4.** Comparison of the accuracy.

| Method / Accuracy | PointNet [13] | SegCloud [30] | SPGraph [19] | Proposed (FP32) | Proposed (INT8) |
|---|---|---|---|---|---|
| **mIoU** | 41.1% | 48.9% | 58.0% | 62.4% | 61.9% |
| **mAcc** | 49.0% | 57.4% | 66.5% | 68.5% | 67.2% |

**TABLE 5.** Comparison of object recognition accuracy.

| Method / Object | PointNet [13] | SegCloud [30] | SPGraph [19] | KPConv [22] | FPConv [23] | Proposed (FP32) | Proposed (INT8) |
|---|---|---|---|---|---|---|---|
| **Ceiling** | 88.0% | 90.1% | 89.4% | 92.6% | 94.6% | 93.1% | 92.8% |
| **Floor** | 97.3% | 96.1% | 96.9% | 97.3% | 98.5% | 98.0% | 97.9% |
| **Wall** | 69.8% | 69.9% | 78.1% | 81.4% | 80.9% | 80.8% | 80.8% |
| **Beam** | 0.1% | 0.0% | 0.0% | 0.0% | 0.0% | 0.0% | 0.0% |
| **Column** | 3.9% | 18.4% | 42.8% | 16.5% | 19.1% | 20.0% | 20.0% |
| **Window** | 46.3% | 38.4% | 48.9% | 54.5% | 60.1% | 57.5% | 56.9% |
| **Door** | 10.8% | 23.1% | 61.6% | 69.5% | 48.9% | 43.3% | 43.0% |
| **Table** | 58.9% | 70.4% | 75.4% | 80.2% | 80.6% | 73.9% | 73.5% |
| **Chair** | 52.6% | 75.9% | 84.7% | 90.1% | 88.0% | 83.4% | 82.8% |
| **Sofa** | 5.9% | 58.4% | 52.6% | 66.4% | 53.2% | 60.0% | 50.8% |
| **Bookcase** | 40.3% | 40.9% | 69.8% | 74.6% | 68.4% | 65.9% | 65.8% |
| **Board** | 26.4% | 13.0% | 2.1% | 63.7% | 68.2% | 64.8% | 64.3% |
| **Clutter** | 33.2% | 41.6% | 52.2% | 58.1% | 54.9% | 51.2% | 50.5% |

ceiling–clutter object recognition performance for each scene of Area 5. The proposed network sampled 4096 and 512 random points, which had radii of 0.1, 0.4, and 1.6; the number of random points sampled by the other networks was 8192, 2048, 512, and 128, and they had radii of 0.1, 0.2, 0.4, 0.8, and 1.6. Table 5 reveals that after lightweight compression (comprising quantization and depth-wise separate convolution) was performed, the proposed NN exhibited outstanding recognition accuracy (relative to the networks proposed by other studies) in terms of ceiling, floor, window, and board recognition. Furthermore, the proposed NN maintained an acceptable level of overall accuracy. In the comparison of object recognition accuracy (In TABLE 5), our proposed method is 19.88% better than PointNet [13], 11.98% better than SegCloud [30], and 2.88% better than SPGraph [19]. This experiment proves that our method has high accuracy in object recognition accuracy.

The recognition speed and portability of an NN are determined by model parameters and module sizes; specifically, a lighter module has a higher network speed and smaller module size but provides reduced overall network accuracy. The proposed model is compared with the networks proposed by other studies, which include PointNet [13], KPConv [22], FPConv [23], and the lightened models that were discussed in another study [31], namely SENet–PointNet, CBAM–PointNet, LAM–PointNet, SENet–PointNet++, CBAM–PointNet++, and LAM–PointNet++. The comparison results are presented in Tables 6.

**TABLE 6.** Compare parameters and model size with other models.

| Method | Parameters | Module Size | Memory Size (Bytes) | Access Time (msec) |
|---|---|---|---|---|
| PointNet [13] | 3.5M | 40.7MB | 14,000,000 | 23.3 |
| KPConv [22] | 14.5M | 59.1MB | 58,000,000 | 96.7 |
| FPConv [23] | 17.43M | 66.7MB | 69,720,000 | 116.2 |
| SENet-PointNet [31] | 4.02M | None | 16,080,000 | 26.8 |
| CBAM-PointNet [31] | 4.05M | | 16,200,000 | 27.0 |
| LAM-PointNet [31] | 3.45M | | 13,800,000 | 23.0 |
| SENet-PointNet++ [31] | 1.98M | | 7,920,000 | 13.2 |
| CBAM-PointNet++ [31] | 2.03M | | 8,120,000 | 13.5 |
| LAM-PointNet++ [31] | 1.68M | | 6,720,000 | 11.2 |
| Proposed | 1.47M | 24.6MB | 5,880,000 | 9.8 |

In the comparison of the number of parameters, our proposed method is only 42% of PointNet [13], 10.1% of KPConv [22], 8.4% of FPConv [23], 36.6% of SENet-PointNet [31], 36.3% of CBAM-PointNet [31], 42.6% of LAM-PointNet [31], 74.2% of SENet-PointNet++ [31], 72.4% of CBAM-PointNet++ [31], and 87.5% of LAM-PointNet++ [31]. In addition, in the comparison of module size, our proposed method is only 60.44% of PointNet [13], 41.62% of KPConv [22], and 36.88% of FPConv [23]. The above experiments can prove that the proposed Lightweight FPConv algorithm can simplify the number of parameters and module size, and achieve higher accuracy at the same time.

In TABLE 6, the number of parameters of the proposed lightweight FPConv algorithm is 1.47M and the module size is 24.6MB. However, the parameters of KPConv [22] are 14.5M and the module size is 59.1MB; the parameters of FPConv [23] are 17.43M and the module size is 66.7MB. It can be known from experiments that, in the comparison of the number of parameters, the proposed method is only 10.1% of KPConv [22] and only 8.4% of FPConv [23]. In the module size experiment, the proposed method is only 41.6% of KPConv [22] and only 36.9% of FPConv [23].

To verify the performance of our proposed method, we analyze the memory storage space and access speed required by the parameters of our proposed network. Our system uses Samsung double data rate fifth-generation synchronous dynamic random-access memory (DDR5 SDRAM, K4RAH086VB-BCQK) and operates at 4800 Mbps with a supply voltage of 1.1V.

The memory size of the PointNet [13], KPConv [22], and FPConv [23] methods require 14,000,000 bytes, 58,000,000 bytes, 69,720,000 bytes, and the access time is 23.3 msec, 96.7 msec, and 116.2 msec respectively. In addition, the memory size of SENet-PointNet [31], CBAM-PointNet [31], LAM-PointNet [31], SENet-PointNet++ [31], CBAM-PointNet++ [31], and LAM-PointNet++ [31] schemes requires 16,080,000 bytes, 16,200,000 bytes, 13,800,000 bytes, 7,920,000 bytes, 8,120,000 bytes, 6,720,000 bytes, the access time is 26.8 msec, 27 msec, 23 msec, 13.2 msec, 11.5 msec respectively. However, our proposed method only needs 5,880,000 bytes of memory size and the access time is 9.8 msec. In terms of memory usage of our proposed method,

we only need 42% of PointNet [13], 10% of KPConv [22], 8% of FPConv [23], 37% of SENet-PointNet [31], CBAM - 36% of PointNet [31], 43% of LAM-PointNet [31], 74% of SENet-PointNet++ [31], 72% of CBAM-PointNet++ [31], 88% of LAM-PointNet++ [31]. In terms of memory access time analysis, our method is 2.38 times faster than PointNet [13], 9.87 times faster than KPConv [22], 11.86 times faster than FPConv [23], and 2.73 times faster than SENet-PointNet [31]. 2.76 times faster than CBAM-PointNet [31], 2.35 times faster than LAM-PointNet [31], 1.35 times faster than SENet-PointNet++ [31], 1.38 times faster than CBAM-PointNet++ [31], and faster than LAM-PointNet++ [31]db@32 1.14 times faster. From this experiment, we can know that our proposed architecture has a high performance of high-speed parameter access. At the same time, our method can save a lot of memory.

It can be seen from the above analysis that the proposed lightweight FPConv algorithm sacrifices a little accuracy, but greatly simplifies the operation parameters and module size. The proposed approach provides a good solution for real-time computing, high-speed computing, edge computing, and portable consumer electronics for the 3D point cloud.

This section discusses the experiment results and compares the experimental data obtained from the lightweight and improved FPConv framework. The standards for each evaluation system (comprising oA, mAcc, and mIoU) are detailed, and the NN accuracy, object recognition accuracy of scenes from Area 5, and the network parameters of the proposed model are compared with those of models in the relevant literature. The results revealed the excellent performance of the proposed model.

## V. CONCLUSION

The present study proposed a 3D point cloud semantic segmentation system and fast convolution design based on lightweight FPConv. It is mainly designed and improved for the characteristics of large-scale NNs with highly accurate, but high computation complexity and low portability. The proposed method combines depth-wise separate convolution, quantization, and Winograd convolution technology to lighten and accelerate NN computation. In the present study, RandLA-Net was combined with FPConv, and MobileNet was employed for depth-wise separate convolution and quantization to accelerate convolution computation without a considerable reduction of neural network accuracy. Subsequently, a large-scale scene database provided by Stanford 3D AI Lab, S3DIS [29], was used to verify the designed NN. After comparing the designed NN with those proposed by relevant studies, the results revealed that despite the implementation of lightening compression, the proposed NN achieved satisfactory recognition rates. Additionally, the proposed NN maintained an acceptable level of overall accuracy. Our proposed method (FP32) outperforms PointNet [13] by 21.30% in the mIoU comparison. In the mAcc comparison, the proposed method (FP32) is 19.50% better than PointNet [13]. Besides, in the comparison of object recognition accuracy, our proposed method is 19.88% better than PointNet [13]. In addition, the proposed method is only 10.1% of KPConv [22] and only 8.4% of FPConv [23] in the comparison of the number of parameters. In the module size experiment, the proposed method is only 41.6% of KPConv [22] and only 36.9% of FPConv [23]. In terms of memory usage of our proposed method, we only need 42% of PointNet [13], 10% of KPConv [22], and 8% of FPConv [23]. In terms of memory access time analysis, our method is 2.38 times faster than PointNet [13], 9.87 times faster than KPConv [22], and 11.86 times faster than FPConv [23]. A series of experiments prove that our proposed lightweight FPConv algorithm greatly simplifies the operation parameters and module size, speeds up parameter access, saves a lot of memory, performs high-speed computing, and achieves a highly efficient 3D Point Cloud Semantic Segmentation System for portable consumer electronics applications.

## REFERENCES

[1] J. Liu, Q. Sun, Z. Fan, and Y. Jia, "TOF LiDAR development in autonomous vehicle," in *Proc. IEEE 3rd Optoelectron. Global Conf. (OGC)*, Shenzhen, China, Sep. 2018, pp. 185–190.

[2] Y. C. Fan, C. M. Yelamandala, T. W. Chen, and C. J. Huang, "Real-time object detection for LiDAR based on LS-R-YOLOv4 neural network," *J. Sensors*, vol. 2021, pp. 1–11, May 2021.

[3] Y. Jian, Y. Yang, Z. Chen, X. Qing, Y. Zhao, L. He, X. Chen, and W. Luo, "PointMTL: Multi-transform learning for effective 3D point cloud representations," *IEEE Access*, vol. 9, pp. 126241–126255, 2021.

[4] Y. Guo, H. Wang, Q. Hu, H. Liu, L. Liu, and M. Bennamoun, "Deep learning for 3D point clouds: A survey," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 43, no. 12, pp. 4338–4364, Dec. 2021.

[5] F. J. Lawin, M. Danelljan, P. Tosteberg, G. Bhat, F. S. Khan, and M. Felsberg, "Deep projective 3D semantic segmentation," in *Proc. Int. Conf. Comput. Anal. Images Patterns*, Ystad, Sweden, 2017, pp. 95–107.

[6] A. Boulch, B. Le Saux, and N. Audebert, "Unstructured point cloud semantic labeling using deep segmentation networks," in *Proc. Eurographics Workshop 3D Object Retr.*, Lyon, France, 2017, pp. 17–24.

[7] B.-T. Wu, P.-C. Li, J.-H. Chen, Y.-J. Li, and Y.-C. Fan, "3D environment detection using multi-view color images and LiDAR point clouds," in *Proc. IEEE Int. Conf. Consum. Electron.-Taiwan (ICCE-TW)*, Taichung, Taiwan, May 2018, pp. 367–368.

[8] Y.-C. Fan, J.-C. Chiou, and Y.-H. Jiang, "Hole-filling based memory controller of disparity modification system for multiview three-dimensional video," *IEEE Trans. Magn.*, vol. 47, no. 3, pp. 679–682, Mar. 2011.

[9] B. Wu, A. Wan, X. Yue, and K. Keutzer, "SqueezeSeg: Convolutional neural nets with recurrent CRF for real-time road-object segmentation from 3D LiDAR point cloud," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, Brisbane, QLD, Australia, May 2018, pp. 1887–1893.

[10] F. N. Iandola, S. Han, M. W. Moskewicz, K. Ashraf, W. J. Dally, and K. Keutzer, "SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and <0.5 MB model size," in *Proc. Int. Conf. Learn. Represent.*, Toulon, France, Apr. 2017, pp. 1–13.

[11] B. Wu, X. Zhou, S. Zhao, X. Yue, and K. Keutzer, "SqueezeSegV2: Improved model structure and unsupervised domain adaptation for road-object segmentation from a LiDAR point cloud," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, Montreal, QC, Canada, May 2019, pp. 4376–4382.

[12] A. Milioto, I. Vizzo, J. Behley, and C. Stachniss, "RangeNet++: Fast and accurate LiDAR semantic segmentation," in *Proc. IEEE/RSJ Int. Conf. Intell. Robot. Syst. (IROS)*, Macau, China, Nov. 2019, pp. 4213–4220.

[13] C. R. Qi, H. Su, K. Mo, and L. J. Guibas, "PointNet: Deep learning on point sets for 3D classification and segmentation," in *Proc. 30th IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Honolulu, HI, USA, Jan. 2017, pp. 77–85.

[14] B.-S. Hua, M.-K. Tran, and S.-K. Yeung, "Pointwise convolutional neural networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Salt Lake City, UT, USA, Jun. 2018, pp. 984–993.

[15] S. Wang, S. Suo, W.-C. Ma, A. Pokrovsky, and R. Urtasun, "Deep parametric continuous convolutional neural networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Salt Lake City, UT, USA, Jun. 2018, pp. 2589–2597.
[16] F. Engelmann, T. Kontogianni, A. Hermans, and B. Leibe, "Exploring spatial context for 3D semantic segmentation of point clouds," in *Proc. IEEE Int. Conf. Comput. Vis.*, Venice, Italy, Aug. 2017, pp. 716–724.
[17] Y.-C. Fan, L.-J. Zheng, and Y.-C. Liu, "3D environment measurement and reconstruction based on LiDAR," in *Proc. IEEE Int. Instrum. Meas. Technol. Conf. (IMTC)*, Houston, TX, USA, May 2018, pp. 1–4.
[18] Z. Zhao, M. Liu, and K. Ramani, "DAR-Net: Dynamic aggregation network for semantic scene segmentation," 2019, *arXiv:1907.12022*.
[19] L. Landrieu and M. Simonovsky, "Large-scale point cloud semantic segmentation with superpoint graphs," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)* Salt Lake City, UT, USA, Jun. 2018, pp. 4558–4567.
[20] C. R. Qi, L. Yi, H. Su, and L. J. Guibas, "PointNet++: Deep hierarchical feature learning on point sets in a metric space," in *Proc. Neural Inf. Process. Syst. (NIPS)*, Long Beach, CA, USA, 2017, pp. 1–10.
[21] Q. Hu, B. Yang, L. Xie, S. Rosa, Y. Guo, Z. Wang, N. Trigoni, and A. Markham, "RandLA-Net: Efficient semantic segmentation of large-scale point clouds," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Seattle, WA, USA, Jun. 2020, pp. 11108–11117.
[22] H. Thomas, C. R. Qi, J.-E. Deschaud, B. Marcotegui, F. Goulette, and L. Guibas, "KPConv: Flexible and deformable convolution for point clouds," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Seoul, Korea (South), Oct. 2019, pp. 6411–6420.
[23] Y. Lin, Z. Yan, H. Huang, D. Du, L. Liu, S. Cui, and X. Han, "FPConv: Learning local flattening for point convolution," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Seattle, WA, USA, Jun. 2020, pp. 4293–4302.
[24] N. Ma, X. Zhang, H.-T. Zheng, and J. Sun, "ShuffleNet v2: Practical guidelines for efficient CNN architecture design," 2018, *arXiv:1807.11164*.
[25] *[DL] Winograd Fast Convolutional Algorithm*. Accessed: Nov. 13, 2019. [Online]. Available: https://Martin20150405.github.io/2019/11/13/dl-winograd-kuai-su-juan-ji-suan-fa/
[26] S. Winograd, "Signal processing and complexity of computation," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process. (ICASSP)*, Denver, Colorado, Apr. 1980, pp. 94–101.
[27] J. Shen, Y. Huang, M. Wen, and C. Zhang, "Toward an efficient deep pipelined template-based architecture for accelerating the entire 2-D and 3-D CNNs on FPGA," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 39, no. 7, pp. 1442–1455, Jul. 2020.
[28] K.-Y. Liao, Y.-S. Xiao, and Y.-C. Fan, "3D point cloud semantic segmentation system," in *Proc. 10th Int. Conf. Comput. Commun. Manag.*, Okayama, Japan, Jul. 2022, pp. 68–72.
[29] I. Armeni, O. Sener, A. R. Zamir, H. Jiang, I. K. Brilakis, M. Fischer, and S. Savarese, "3D semantic parsing of large-scale indoor spaces," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Las Vegas, NV, USA, Jun. 2016, pp. 1534–1543.
[30] L. Tchapmi, C. Choy, I. Armeni, J. Gwak, and S. Savarese, "SEGCloud: Semantic segmentation of 3D point clouds," in *Proc. Int. Conf. 3D Vis. (3DV)*, Qingdao, China, Oct. 2017, pp. 537–547.
[31] Y. Cui, Y. An, W. Sun, H. Hu, and X. Song, "Lightweight attention module for deep learning on classification and segmentation of 3-D point clouds," *IEEE Trans. Instrum. Meas.*, vol. 70, pp. 1–12, 2021.

**YU-CHENG FAN** (Senior Member, IEEE) received the B.S. and M.S. degrees in electrical engineering from National Cheng Kung University, in 1997 and 1999, respectively, and the Ph.D. degree in electrical engineering from National Taiwan University, in 2005. From 1999 to 2000, he was an IC Design Engineer with the Industrial Technology Research Institute (ITRI). In 2006, he joined the Department of Electronic Engineering, National Taipei University of Technology (NTUT), Taipei, Taiwan, where he is currently a Full Professor. He was a Visiting Scholar with the University of Washington, USA, from July 2019 to February 2020. He was also an Associate Dean of the Electrical Engineering and Computer Science College, NTUT, from 2021 to 2023. His research results have been published in over 200 journals and conference papers. His research interests include multimedia systems, consumer electronics, 3-D display systems, image and video systems, and VLSI/SoC design. He has received the Best Paper Awards, including the IEEE ICCE 2003, the CIASPCD 2010, the APDSC 2010, the IEEE ISNE 2013, the IDW/3DSA 2016, the IMID/3DSA 2017, the IEEE ISNE 2018, the IEEE ICCE 2020 (Best Poster Video Award), the IEEE ICIET 2021, and the IEEE ICCCI 2021. He was presented with the Dr. Shechtman Young Researcher Award, in 2015. He received the IEEE Consumer Electronics Society Service Award, in 2016, 2017, 2018, and 2019. He also received the Excellent Research Teacher Award, in 2015; the Research Progress Awards, in 2010 and 2015; the Excellent Teaching Awards from NTUT, in 2010, 2014, 2016, and 2018; and the Industry–University Cooperation Outstanding Achievement Awards from the Ministry of Science and Technology (MOST), Taiwan, in 2011, 2013, 2014, 2016, 2017, 2018, and 2022. Since 2022, he has been serving as the Vice Chair for the CTSoC VAR Technical Committee. He is currently the Asia–Pacific Regional Director of the IEEE Consumer Technology (Electronics) Society and a Voting Member of IEEE TRANSACTIONS ON GAMES and IEEE TRANSACTIONS ON COGNITIVE AND DEVELOPMENTAL SYSTEMS. He is a Scholastic Honor Member of Phi Tau Phi. He is an Associate Editor of IEEE ACCESS and IEEE TRANSACTIONS ON CONSUMER ELECTRONICS.

**KUAN-YU LIAO** received the B.S. degree from the Ming Chi University of Technology, in 2017, and the M.S. degree from the National Taipei University of Technology, Taipei, Taiwan, in 2021. His research interests include digital image processing, digital IC design, and VLSI design.

**YOU-SHENG XIAO** received the M.S. degree in electronic engineering from the National Taipei University of Technology, in 2022. He is currently an IC Design Engineer with Richtek Technology Corporation. His research interests include digital IC design, image processing, and deep learning application.

**MIN-HUA LU** received the M.S. degree in electronic engineering from the National Taipei University of Technology, in 2022. She is currently an Engineer with Taiwan Semiconductor Manufacturing Company. Her research interests include deep learning applications and SoC design.

**WEI-ZHE YAN** received the M.S. degree in electronic engineering from the National Taipei University of Technology, Taipei, in 2022. He is currently an Engineer with Macronix International Company Ltd. His research interests include deep learning applications, image processing, and digital IC design.