Application of Depth Camera-based Action Recognition with Graph Convolutional Networks in Elderly Care and Smart Education

ZHANG Qingqi*

WU Ren**

GE Qi-Wei***

(Abstract)

This study explores the innovative applications of depth cameras combined with Graph Convolutional Networks (GCN) for action recognition in two critical domains: elderly care and smart education. We harness the capabilities of depth cameras to capture spatial and temporal features, alongside our robust GCN algorithm, to develop models capable of accurately recognizing and classifying human actions. In elderly care, our model is particularly focused on detecting and analyzing falls, which are crucial for enhancing care safety and supporting the independence of elderly individuals. Experimental results demonstrate that our depth camera-based action recognition model achieved an impressive average accuracy of 96.3% in fall detection within real-world scenarios, while also maintaining low rates of false positives and false negatives. In the realm of smart education, our depth camera-based model is specifically designed to recognize students' hand-raising actions in real-time, which is crucial for comprehensively assessing student engagement in the class, and accordingly adjusting teaching strategies. Experimental results show that our model achieves an average accuracy of 89.7% in real-world scenarios, while maintaining low rates of false positives and false negatives. Overall, this study showcases the powerful potential of integrating depth cameras with GCNs for action recognition, significantly enhancing both the safety and efficiency of elderly care, as well as the interactivity and educational quality of smart education.

Keywords: Action Recognition; Depth Camera; Graph Convolutional Network; Elderly Care; Smart Education

1. Introduction

Human action recognition is a challenging and engaging research task within computer vision. It has a wide range of applications including video understanding, smart surveillance, robotics, industrial automation, healthcare, and education [1,2,3]. In recent years, many researchers have dedicated efforts to recognizing and analyzing human actions from RGB videos [4,5]. However, methods based on RGB often fail to achieve satisfactory results in practical

* The Graduate School of East Asian Studies, Yamaguchi University

** Department of Information Science, Shunan University

*** The Graduate School of East Asian Studies, Yamaguchi University

Journal of East Asian Studies, No.23, 2025.3. (pp.79-97)

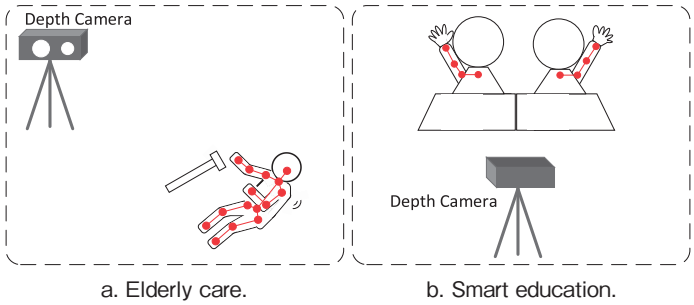


Figure 1. Applications of depth cameras in real-world scenarios. (a) Application of depth cameras in elderly care, focusing on the recognition and analysis of falling actions. (b) Application of depth cameras in smart education, concentrating on detecting students' hand-raising actions. (Source: Created by the authors)

applications due to their inability to robustly handle environmental noise such as changes in viewpoint, lighting conditions, background colors, and clothing [6,7].

The rapid advancement of depth camera technology has opened up new possibilities for action recognition. Depth cameras, such as Azure Kinect DK [8], utilize Time-of-Flight (ToF) technology to capture depth information, enhancing understanding of complex environments and interactions. Moreover, depth cameras offer advantages such as being unaffected by lighting conditions and preserving privacy [9]. Consequently, methods that use three-dimensional coordinates of human joints as input have attracted widespread attention [10,11]. However, in specific practical applications, such as elderly care and smart education, technologies based on depth cameras and Graph Convolutional Networks (GCNs) have not been fully developed or applied.

In elderly care, medical surveys have shown that falls are a leading cause of both fatal and non-fatal injuries among the elderly [12]. The incidence of falls among the elderly ranges from 32% to 42%, and timely medical intervention after a fall can reduce the risk of death by 80% [13,14]. Therefore, accurately and effectively monitoring elderly falls is of great importance. However, due to the sudden nature of falls and the rapidity of the falling process, traditional monitoring methods often fail to capture the entire event in real-time. This necessitates the adoption of more advanced technological means, such as depth cameras, which can precisely capture the entire process of a fall and provide detailed three-dimensional spatial information, thereby laying a solid foundation for the accurate identification of falls. Furthermore, by utilizing GCNs to analyze these complex three-dimensional data, we can extract key features from structured spatial relationships and the continuous temporal dimension, achieving more accurate recognition and analysis.

In smart education, recognizing students' hand-raising behavior plays a crucial role as it helps analyze classroom engagement to assess teaching processes and optimize educational strategies [15]. Traditionally, many studies have used two-dimensional cameras, i.e., RGB videos, to record and analyze student behavior to evaluate teaching quality and student attitudes

[16]. However, this method inherently lacks depth information of the targeted objects and cannot robustly handle environmental noise. Depth camera technology, which provides richer three-dimensional spatial information, has the potential to significantly enhance the accuracy and reliability of student action recognition, yet its application in smart education is still in an exploratory stage. Depth cameras, combined with the analytical power of GCNs, offer a promising avenue for capturing and understanding complex student behaviors through enhanced three-dimensional data analysis.

This study initially explores the application of depth cameras and Graph Convolutional Networks (GCNs) for action recognition, focusing on elderly care and smart education, as illustrated in Figure 1. We introduce a method that integrates these technologies to develop a Spatio-Temporal Fusion Graph Convolutional Network (STF-GCN), tailored for real-time action recognition. Our approach involves deploying depth cameras to capture three-dimensional data in practical scenarios, processed and analyzed by the STF-GCN. This network effectively exploits the spatio-temporal features of the depth data, ensuring precise action recognition. To facilitate real-time applications, we convert the trained STF-GCN model into the ONNX format, significantly enhancing its portability and deployment efficiency on depth cameras. We conducted extensive experiments on fall detection in elderly care and hand-raising detection in educational settings. The results validate our method's effectiveness and demonstrate the promising potential of integrating depth cameras with GCNs for future applications.

2. Related Works

Next, we will discuss the related works from two perspectives, including action recognition in elderly care and action recognition in smart education.

2.1 Action Recognition in Elderly Care

Action recognition plays a pivotal role in enhancing the safety and quality of life for the elderly, a demographic that often requires continuous monitoring due to a higher risk of falls that could necessitate immediate medical attention. Jang et al. [1] proposed a network called Four Stream Adaptive CNN (FSA-CNN), which has three main properties: robustness to spatio-temporal variations, input-adaptive activation function, and extension of the conventional two-stream approach. Shu et al. [17] proposed a Expansion-Squeeze-Excitation Fusion Network (ESE-FN), which learns modal and channel-wise ESE attentions for attentively fusing the multi-modal features in the modal and channel-wise ways. Zin et al. [18] introduced a system that integrates feature extraction methods from previous works and utilizes depth frame sequences provided by depth cameras. The system locates individuals by extracting different Regions of Interest (ROI) from UV-disparity maps. Tabbakha et al. [19] introduced the development and testing of a wearable device featuring motion detection and indoor localization based on a

random forest classifier. The action recognition phase utilizes a gyroscope and an accelerometer to detect various types of movements.

Among the methods discussed, only several methods [18,19] involved actual model deployment and testing. The remaining methods did not implement the proposed algorithms in practical settings. Although Zin et al. [18] employ depth cameras, they only use depth information for locating individuals and do not fully utilize them in the action recognition phase. Instead, the action recognition phase of their method involves a strategy of randomly sampling frame sequences, a process that can lead to inconsistent results and potentially miss critical motions, reducing the overall effectiveness and accuracy of the action recognition. In contrast, the method proposed in this paper utilizes depth cameras to capture the three-dimensional information of all human joints and employs our proposed STF-GCN in the action recognition phase. Finally, we convert our model into a deployable ONNX format to conduct tests in real-time within real-world scenarios.

2.2 Action Recognition in Smart Education

As educational technologies evolve towards personalized learning and data-driven feedback, action recognition in smart education becomes increasingly important. Recognizing specific actions, such as hand-raising, plays a key role in assessing student engagement, facilitating classroom interaction, and optimizing teaching strategies. Therefore, some studies have focused on detecting hand-raising. Zhou et al. [20] proposed an algorithm for recognizing hand-raising actions. It accomplishes the recognition of hand-raising through three tasks: hand detection, pose estimation, and heuristic matching. Si et al. [21] integrated a feature pyramid into their model architecture, thus designing a region-based fully convolutional network to detect hand-raising gestures. This design somewhat improves the detection capability for low-resolution hand gestures. The method proposed by Liao et al. [22] includes two stages: pose estimation and gesture recognition. The goal of the pose estimation stage is to estimate human joint positions, which are then used to recognize gestures through predefined relationships among the joints. Similarly, Lin et al. [2] employed a process akin to that of Liao et al. [22], utilizing CNNs to classify actions.

Although these methods have achieved commendable results, they are all based on RGB videos, which lack depth information and are highly susceptible to variations in lighting and clothing. Additionally, these methods typically segment an action into frames to perform recognition, which can lead to fragmentation of contextual information and potential loss of continuity in action sequences. In contrast, our approach leverages depth camera technology combined with our STF-GCN to overcome these limitations.

3. Methodology

In this section, we will delineate our methodology across three aspects, including the system architecture, skeleton data preprocessing and the STF-GCN.

3.1 System Architecture

The overview of the proposed system is shown in Figure 2. Initially, the system uses a depth camera to capture real-time three-dimensional skeletal data of individuals. This is followed by a skeletal data preprocessing stage, where the collected skeletal sequences are converted into a format suitable for input into the STF-GCN, a graph neural network specifically designed by us. Structurally, the STF-GCN includes two encoding modules, i.e., TDE (Temporal Dimension Encoding) and SDE (Spatial Dimension Encoding), and five STF (Spatial Temporal Fusion) modules, followed by GAP (Global Average Pooling) and FC (Fully Connected) layers for action classification, as detailed in Section 3.3. After the training phase, we obtain a trained STF-GCN model. The next step is to convert this trained model into an ONNX model that can be deployed on depth cameras, which offers the advantages of being lightweight and fast for inference. Finally, in the inference phase, we can use the depth cameras equipped with the ONNX model to detect actions such as falls and hand-raising in real-time.

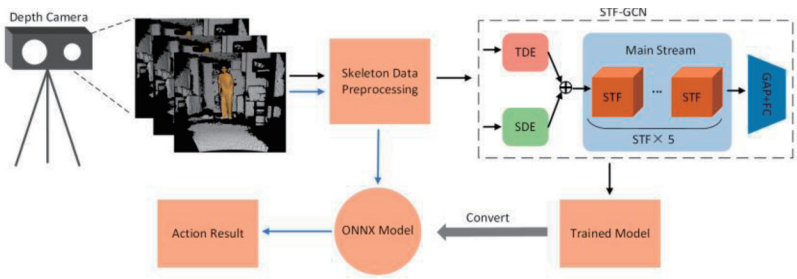


Figure 2. System overview. The black arrow indicates the training phase, while the blue arrow indicates the inference phase. TDE stands for Temporal Dimension Encoding, SDE stands for Spatial Dimension Encoding, STF denotes Spatial Temporal Fusion, GAP represents Global Average Pooling, and FC refers to the Fully Connected layer. (Source: Created by the authors)

3.2 Skeleton Data Preprocessing

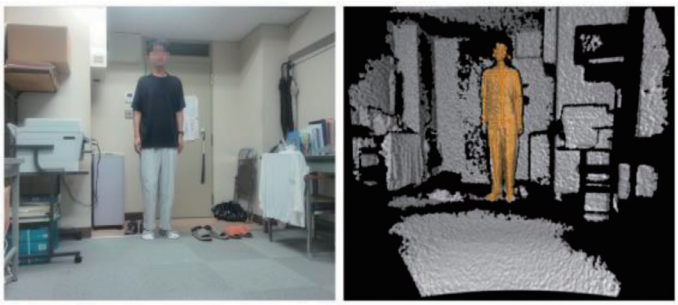
This study utilizes the Femto Bolt depth camera. Figure 3 demonstrates visual examples of the depth camera's RGB field of view (Figure 3a) and infrared intensity visualization (Figure 3b) from a distance of 1.5 meters from the front. It is evident that the RGB field of view does not contain spatial information about the scene, such as the distance of objects. In contrast, Figure 3b represents the intensity of infrared reflection captured by the depth camera, where the grayscale variations indicate the strength of the reflected infrared signal. Depth information is stored as pixel-wise attributes in the depth data. And then, using the API provided by Azure

Kinect Body Tracking SDK [23], we can capture 32 human body joints (marked in yellow in Figure 3b). Figure 4a displays the specific meanings of these 32 joints.

According to preprocessing protocols established by prior research [9,17], which have been proven effective and widely adopted, the input joint count for GCNs is set at 25, as shown in Figure 4b. Since the number and sequence of the 32 joints captured by the depth camera do not align with the input format required by the GCNs, it is necessary to map these captured joints to the 25 joints required by the GCN. This mapping is detailed in Table 1, which provides a comprehensive cross-reference of each joint from the depth camera output to its corresponding joint in the GCN input.

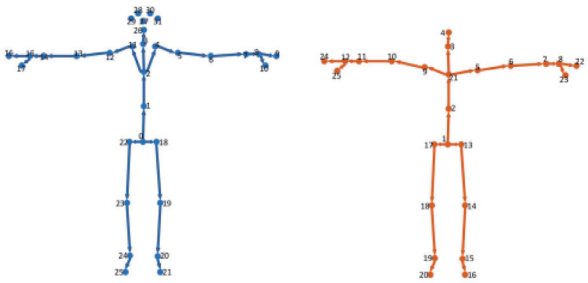
Based on the mapping relationships outlined in Table 1, we first use Algorithm 1 to convert the one-dimensional vector of 32 joints captured by the depth camera into a one-dimensional vector containing 25 GCN joints. The input format for the GCN is a five-dimensional tensor, i.e. [Batchsize, Channels, Fram__Count, Joint_Count, Person_Count]. Subsequently, we employ Algorithm 2 to transform the one-dimensional vector of 25 joints into a five-dimensional tensor suitable for GCN input.

After successfully constructing the five-dimensional tensor, we can then use our proposed STF-GCN to perform model training and inference on the action data captured by the depth camera.



a. RGB field of view. b. Infrared intensity visualization.

Figure 3. Visualization of depth camera views. (Source: Created by the authors)



a. Azure Kinect DK format. b. Input format for GCNs.

Figure 4. Comparison of depth camera-derived human skeletal outputs and GCNs input formats. (Source: Created by the authors)

Table 1. Joints mapping from Kinect DK output to GCN input. (Source: Created by the authors)

Joint Name	Kinect DK Joint Number	GCN Input Joint Number
Pelvis	0	1
Spine Nanal	1	2
Neck	3	3
Head	26	4
Shoulder Left	5	5
Elbow Left	6	6
Wrist Left	7	7
Hand Left	8	8
Shoulder Right	12	9
Elbow Right	13	10
Wrist Right	14	11
Hand Right	15	12
Hip Left	18	13
Knee Left	19	14
Ankle Left	20	15
Foot Left	21	16
Hip Right	22	17
Knee Right	23	18
Ankle Right	24	19
Foot Right	25	20
Spine Chest	2	21
Handtip Left	9	22
Thumb Left	10	23
Handtip Right	16	24
Thumb Right	17	25

Algorithm 1. Convert kinect skeletons to GCN skeletons

Input: Kinect_Skeletons // A one-dimensional vector composed of 32 joints captured by the depth camera.

Output: GCN_Skeletons // A one-dimensional vector composed of 25 GCN joints.

```
kinectToGCNMap = [0, 1, 3, 26, 5, 6, 7, 8, 12, 13, 14, 15, 18, 19, 20, 21, 22, 23, 24, 25, 2, 9, 10, 16, 17]
Initialize GCN_Skeletons as an empty vector of floats.
Initialize gcnJointsTemp as an array of floats of size Joint count * 3. // Each joint is represented by (x, y, z)
For each person in Person_Count do
    For each skeletonPair in Kinect_Skeletons do
        For gcnIndex from 0 to Joint_Count - 1 do
            kinectIndex ← kinectToGCNMap[gcnIndex]
            Set gcnJointsTemp[gcnIndex*3+0] to skeletonP air.person.joints[kinectIndex].position.xyz.x
            Set gcnJointsTemp[gcnIndex*3+1] to skeletonP air.person.joints[kinectIndex].position.xyz.y
            Set gcnJointsTemp[gcnIndex*3+2] to skeletonP air.person.joints[kinectIndex].position.xyz.z
        End For
        Append gcnJointsTemp to GCN_Skeletons
    End For
End For
```

Algorithm 2. Transform a one-dimensional vector to a five-dimensional tensor, i.e., [Batchsize, Channels, Frame_Count, Joint_Count, Person_Count]

Input: GCN_Skeletons
Output: GCN_Tensor // A five-dimensional tensor

```
Initialize rearranged_data as a vector of floats with size Batchsize*Channels*Frame_Count*Joint_Count*Person_Count
Initialize index to 0
For channel from 0 to Channels – 1 do
  For frame from 0 to Frame_Count – 1 do
    For joint from 0 to Joint_Count – 1 do
      For person from 0 to Person_Count – 1 do
        Calculate original_index using: original_index ← frame * Joint_Count * Person * Channels + joint * Channels + person * Channels * Joint_Count + channel
        Set rearranged_data[index] to GCN_Skeletons[original_index]
        Increment index
      End For
    End For
  End For
End For
Convert rearranged_data to GCN_Tensor by using torch::from_blob function
```

3.3 STF-GCN

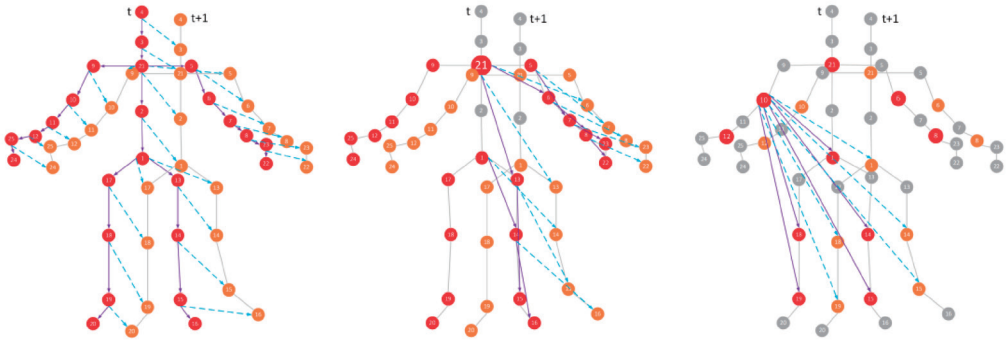
To learn the features of the skeletal data captured by the depth camera, we have developed the STF-GCN. The overall network structure of the STF-GCN is illustrated in Figure 2. It includes SDE, TDE and STF modules.

3.3.1 SDE and TDE

SDE represents features related to joints, bones, and spatial angles [24, 25]. The TDE we propose focuses on two types of temporal dimension features, i.e., temporal motion features and temporal angle features. Each type of temporal feature includes three encoding strategies, i.e., general encoding, reinforced encoding and united encoding.

As depicted in Fig.5a, the general encoding of the human skeleton is rooted in the natural connections between skeletal joints. By analyzing the temporal dimension relationships of each joint across two distinct frames, this encoding strategy comprehensively captures the subtle temporal movement and angle variations that spatial dimension features may overlook. Specifically, the temporal angle is formed by the vector (shown as the purple vector) that connects a joint at frame t to its neighboring joints, and the temporal motion feature (portrayed as the blue vector) pertinent to that joint.

Different actions typically manifest distinct limb motion patterns and positional variations. Additionally, the positions and movements of human limbs often convey semantic cues related to the performed actions. Therefore, we introduce the reinforced encoding scheme, designed to capture the intricate characteristics and semantic nuances inherent in limb motions. As



a. General encoding of human skeleton.

b. Reinforced encoding of human limbs.

c. United encoding of human limbs.

Figure 5. Three encoding strategies. Red and orange dots represent skeletal joints at frames t and t+1, respectively. Blue vectors symbolize temporal motion features; angles between blue and purple vectors are temporal angle features. (b) For clarity, features of only the right arm and leg are depicted. (c) Visualized features between joint No. 10 of the upper body and the joints of the lower body. (Source: Created by the authors)

depicted in Fig.5b, we designate joint No. 21 as the rootnode and select the limb joints of the human body (shown as red and orange dots) to constitute four joint sequences: left arm, right arm, left leg and right leg. Reinforced encoding is suitable for actions dominated by either the upper limbs or lower limbs.

Human actions typically involve intricate interactions and coordination among various limbs. Furthermore, the synchronized motions of these limbs can indicate the consistency and coherence of specific actions. To capture these inter-limb relationships and enhance the model's capacity to distinguish between diverse actions, we introduce the united encoding approach for human limbs. As depicted in Fig.5c, considering the shoulder, elbow and wrist joints account for the primary range of motion in the upper limbs, we select joint No. 21 from the upper body, accompanied by both elbow and wrist joints, as representative of the upper body's movements. Similarly, we choose the joint No. 1, along with both knee and ankle joints, to represent the lower body's motions. Subsequently, we apply united encoding to these selected joint sequences. United encoding is designed for actions involving coordinated movements of all limbs.

3.3.2 STF Module

As depicted in Fig.6, the STF module is designed to process input features through a multi-layered architecture. $C \times T \times V$ represents the dimensions of the input features, where C stands for the number of channels, T for the number of temporal frames and V for the number of joints. The principal components of STF encompass the Spatial Graph Convolution (SGC) [11], the Temporal Receptive Field (TRF) block. $\oplus$ represents concatenation. $\ominus$ represents pairwise subtraction.

Within each TRF block, as shown on the right side of Figure 6, large kernel and small

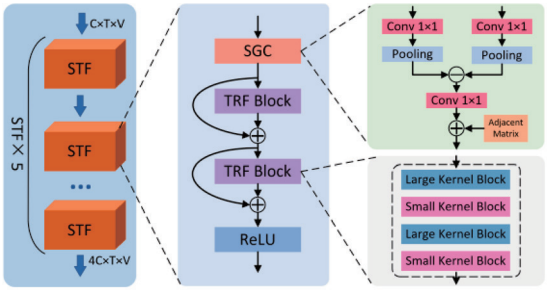


Figure 6. Overview of STF module. (Source: Created by the authors)

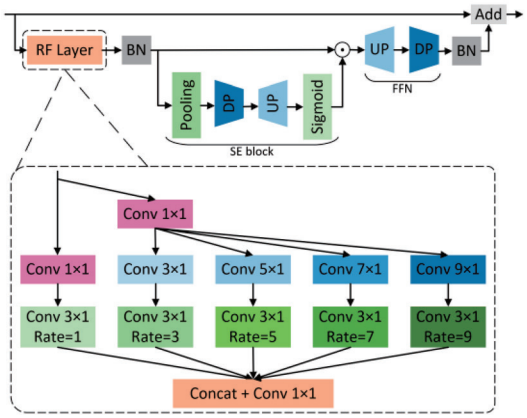


Figure 7. Architectural design of Large Kernel Block. (Source: Created by the authors)

kernel blocks are arranged in an alternating cascading sequence. This configuration advocates for the combined use of large and small kernel convolutions. The small kernels excel at identifying fine-grained and small-scale temporal patterns. Conversely, large kernels are pivotal for capturing sparse and extensive temporal features, vital for analyzing time series with significant relational dependencies.

The large kernel block is illustrated in Figure 7. It comprises a Receptive Field (RF) layer, a Squeeze-and-Excitation (SE) block [26], a Feed Forward Network (FFN) and Batch Normalization (BN). The symbol ⊙ represents element-wise multiplication. The only difference between a small kernel block and a large kernel block is that the former replaces the RF layer with a *Conv* 3×1. Initially, the RF layer applies a convolution operation to the input data. The sequence of convolutions, starting from *Conv* 1×1 to *Conv* 9×1, with increasing dilation rates from 1 to 9, allows for an expansive temporal analysis, covering both immediate and extended temporal contexts. The parallel pathways process the input through *Conv* 3×1 kernels at varying dilation rates, enabling the block to assimilate temporal features with different granularities. SE block is an efficient structure that performs both inter-channel communications and temporal aggregations to increase the depth of network.

4. Experiments

Our experiments focus on fall detection in elderly care and hand-raising detection in smart education. Initially, we train our proposed STF-GCN using publicly available datasets [1,27] and use it as a pre-trained model. Subsequently, we utilize transfer learning to fine-tune our model on fall and hand-raising data collected by the depth camera. Finally, the trained model is converted into an ONNX format, which can be deployed on depth cameras for testing and evaluation.

4.1 Datasets

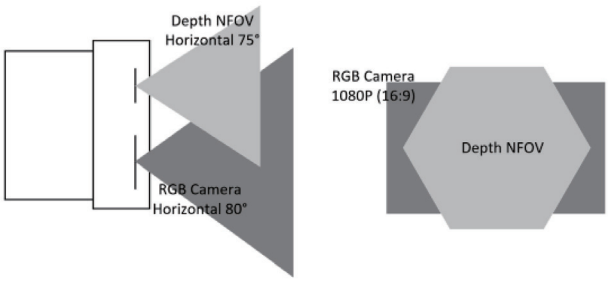
ETRIActivity3D (EA3D) [1] is currently the largest elderly action recognition dataset collected in real-world monitoring environments, comprising 112,620 samples of 50 elderly individuals and 50 young adults performing actions across 8 synchronized sensors. The elderly participants range in age from 64 to 88 years, while the young adults are around 20 years old. All videos are categorized into 55 types of actions.

NTU-RGB+D 60 (NTU60) [27] is a large-scale laboratory indoor dataset provided by [15], comprising 60 action categories. The authors of the dataset recommend two benchmarks: (1) Cross-Subject (X-sub), which includes 40,320 training samples and 16,560 evaluation samples, dividing 40 subjects into two groups. (2) Cross-View (X-view) uses videos captured by cameras 2 and 3 as training samples (37,920 videos) and videos captured by camera 1 as evaluation samples (18,960 videos).

We collected fall and hand-raising data by recording videos, each lasting one minute. Following the data preprocessing protocols established in [11,17], we set the *Frame_Count* in Algorithms 1 and 2 to 64, a setting that is suitable for both falling and hand-raising actions. We created 88 samples from one-minute videos using a stride of 20 frames. To increase the quantity and diversity of samples, we also produced 58 samples from one-minute videos using a stride of 30 frames. Thus, a single one-minute video could generate 146 samples. We collected a total of 6,000 samples using the depth camera, with 3,000 samples for each type of action to fine-tune the STF-GCN model. Of these, 2,500 samples were used as the training set and 500 samples as the test set.

4.2 Depth Camera Settings

Figure 8 illustrates the specific setup details of the depth camera used during data collection. Specifically, the RGB camera was set to 1080p mode with a frame rate of 30. In depth mode, we used the NFOV Unbinned (640x576) mode, also at a frame rate of 30. The distance between the camera and the subject ranged from 1.5 to 2.5 meters, which aligns with the optimal operating conditions of the depth camera. And the camera was positioned 70 cm above the ground.



a. Field of view of depth and RGB. b. Field of view from 1.5m front.

Figure 8. Field of view for depth and RGB cameras (the perspective seen by the sensors) and field of view at a distance of 1.5 meters from the front. (Source: Created by the authors)

4.3 Implementation Details

For the training phase, programming was conducted in Python on Ubuntu 22.04. We use SGD optimization with 0.1 as the base learning rate and a weight decay of 0.0005. All the models are trained on two GeForce RTX 3090 with a batch size of 200, using ReduceLROnPlateau to update the learning rate. For fall detection, we initially pre-trained using the EA3D dataset, setting epochs to 100, and then fine-tuned the model with datasets collected by the depth camera. Similarly, for hand-raising recognition, we pre-trained using the NTU 60 dataset with epochs set at 100, followed by fine-tuning with datasets collected by the depth camera. The fine-tuned model is in PyTorch format. Finally, we convert the PyTorch model into an ONNX format model that can be deployed on a depth camera.

For the inference phase, programming was conducted in C++ using Microsoft Visual Studio Community 2019. The computer used for this purpose was equipped with a 12th Gen Intel(R) Core (TM) i9-12900 and 32GB of RAM. The depth camera employed was the Femto Bolt, which is a collaborative production by Microsoft and Orbbec.

4.4 Evaluation Metrics

We evaluate our model's performance using several key metrics: FPR (False Positive Rate), FNR (False Negative Rate), Acc (Accuracy), PT (Preprocessing Time) and IT (Inference Time). These metrics collectively assist in assessing the effectiveness, efficiency and operational speed of our system.

FPR is defined as the proportion of negative cases that were incorrectly classified as positive. It represents the probability of falsely identifying a negative instance as positive and is calculated using the formula (1). FNR is the proportion of positive cases that were incorrectly classified as negative. It measures the probability of failing to identify a positive instance and is calculated as (2). Acc measures the proportion of true results (both true positives and true negatives) among the total number of cases examined. It provides an overall effectiveness of the model and is expressed as (3). PT and IT specifically measure the time our system takes during the data preprocessing and model inference phases, respectively, highlighting the

operational efficiency.

$$FPR = \frac{FP}{FP+TN} \tag{1}$$

$$FNR = \frac{FN}{FN+TP} \tag{2}$$

$$Acc = \frac{TP+TN}{TP+FN+FP+TN} \tag{3}$$

4.5 Experimental Results of Fall Detection

The STF-GCN model for fall detection was initially tested on the EA3D dataset, achieving an accuracy of 92.9% as detailed in Table 2. The test results of the STF-GCN on the EA3D dataset are presented using a confusion matrix, as illustrated in Figure 9. To enhance its performance specifically for fall detection tasks, the model underwent a fine-tuning process. This adjustment involved retraining the model using our collected falls data, which improved its accuracy to 97.6%. These results not only affirm the model's improved effectiveness but also underscore the benefits of fine-tuning for specific scenarios.

Table 2. Accuracy of the STF-GCN model on EA3D dataset before and after fine-tuning specifically for fall detection. (Source: Created by the authors)

	Acc
Initial Testing on EA3D	92.9%
After Fine-tuning for Fall Detection	97.6%

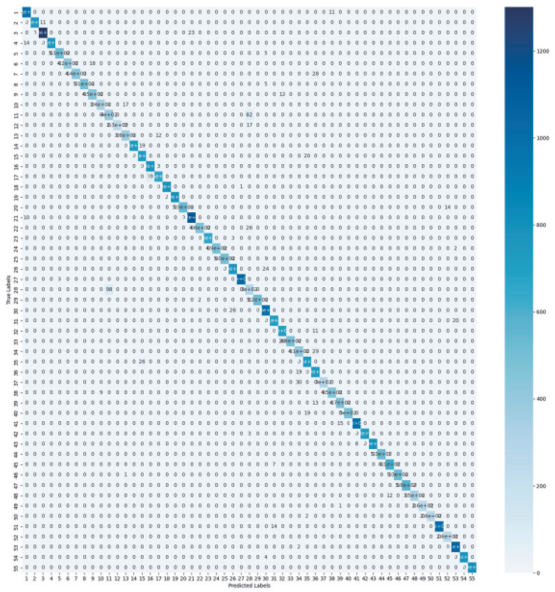


Figure 9. The confusion matrices of our STF-GCN model on EA3D. The horizontal axis represents the predicted labels, while the vertical axis represents the true labels. (Source: Created by the authors)

Table 3. Fall detection accuracy and processing times from various orientations. (Source: Created by the authors)

	Front	Back	Left	Right	Avg
FPR	25%	0	0	0	6.25%
FNR	0	33.3%	0	0	8.33%
Acc	92.9%	92.3%	100%	100%	96.3%
PT	0.148s	0.136s	0.139s	0.138s	0.14s
IT	0.047s	0.043s	0.047s	0.044s	0.045s

After fine-tuning, the model size is 8.07 MB. We then convert the fine-tuned model into an ONNX format that can be deployed on a depth camera. The size of the ONNX model is only 1.55 MB.

After deploying the ONNX model to the depth camera, we conducted real-time performance tests in real-world scenarios. Due to the limited detection range of the depth camera, these tests were performed with only one individual. Specifically, we simulated falls of an elderly person from four different orientations including front, back, left, and right. The results presented in Table 3 demonstrate our method's effectiveness and efficiency from various orientations. The system achieves 100% accuracy in both the Left and Right orientations, illustrating the robustness and reliability of our fall detection system in these views. However, the accuracy for Front and Back orientations does not reach 100%, primarily due to increased false positive rates in the Front and elevated false negative rate in the Back orientation. These discrepancies highlight challenges in accurately detecting falls when the individual is facing towards or away from the camera, which may be influenced by the depth camera's angle and the overlap of body parts in these orientations. Table 3 also details preprocessing and model inference times, further highlighting the operational efficiency of our system in real-world scenarios. These metrics collectively demonstrate the capability to our method in real-world scenarios.

Figure 10 provides visual examples of the detection process from these various

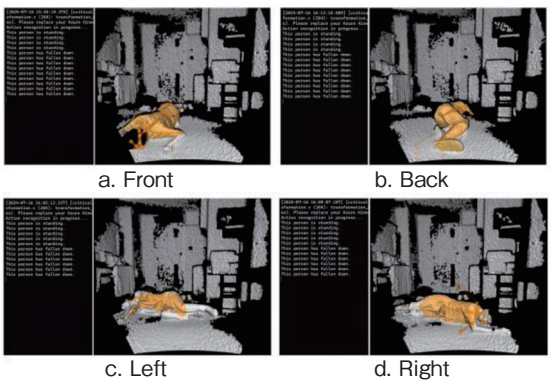


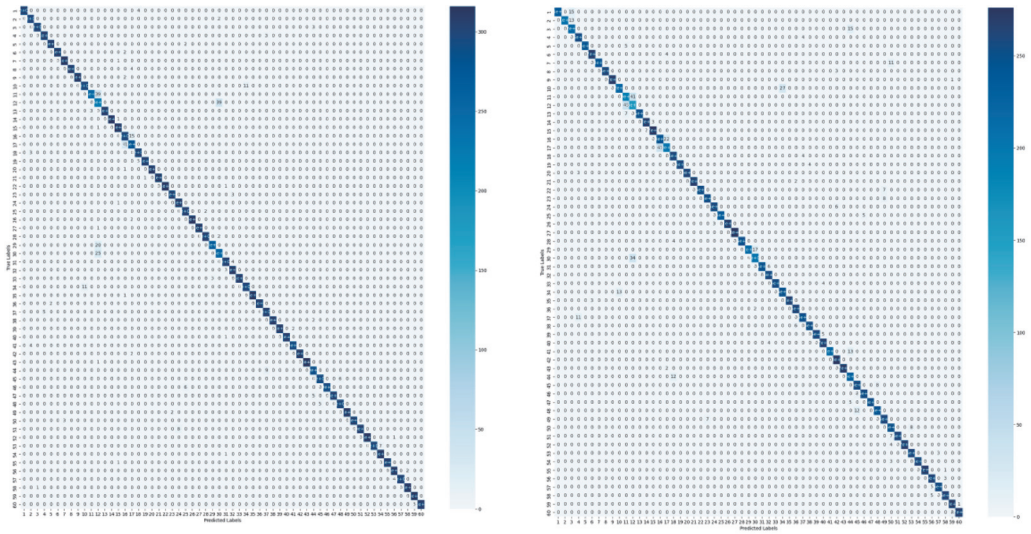
Figure 10. Visual examples of fall detection tests conducted from four orientations using a depth camera. Each subfigure (a) Front, (b) Back, (c) Left and (d) Right displays the detection outcome: "This person is standing" when the individual is upright, and "This person has fallen down" following a fall. (Source: Created by the authors)

orientations. These images demonstrate the system's real-time response and its ability to accurately recognize falls, which is crucial for ensuring the safety of the elderly.

4.6 Experimental Results of Hand-Raising Detection

The STF-GCN model tailored for hand-raising detection was first evaluated on NTU 60 X-sub and NTU 60 X-view, where it reached accuracies of 91.4% and 95.9% as detailed in Table 4. The testing outcomes on the NTU 60 datasets are depicted in the confusion matrices shown in Figures 11a and 11b. To optimize it further for hand-raising actions, the model was fine-tuned. This enhancement process included retraining the model with our specifically collected hand-raising data, which boosted its accuracies to 95.1% and 98.5%. These findings effectively underscore the improvements in model performance due to specialized fine-tuning for dedicated tasks.

For the two fine-tuned models listed in Table 4, we selected the model with the higher accuracy to be converted into an ONNX model for deployment on a depth camera. The sizes of the fine-tuned model and the ONNX model are 8.07MB and 1.55MB, respectively. Finally, we deployed the ONNX model on a depth camera and conducted subsequent testing in real-time within real-world scenarios.



a. Confusion matrix on NTU60 X-view.

b. Confusion matrix on NTU60 X-sub.

Figure 11. The confusion matrices of our STF-GCN model on NTU60 X-view and NTU60 X-sub. The horizontal axis represents the predicted labels, while the vertical axis represents the true labels. (Source: Created by the authors)

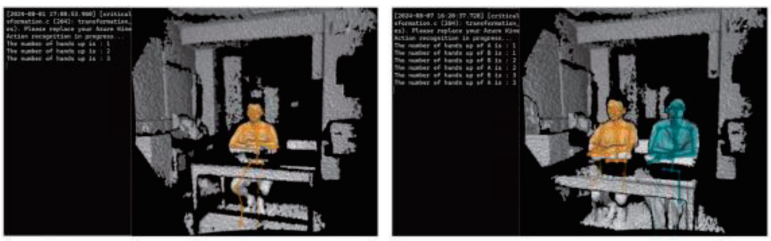
Table 4. Accuracy of the STF-GCN model on NTU 60 dataset before and after fine-tuning specifically for hand-raising. (Source: Created by the authors)

	Acc
Initial Testing on NTU 60 X-sub	91.4%
After Fine-tuning for Hand-raising	95.1%
Initial Testing on NTU 60 X-view	95.9%
After Fine-tuning for Hand-raising	98.5%

Due to the limited detection range of the depth camera, tests were performed only in scenarios involving one person and two people. Specifically, we simulated the detection of student hand-raising actions in a classroom setting and recorded the number of hand-raising in real-time. Table 5 presents the FPR, FNR, Acc, PT and IT for scenarios involving one and two people. For one person, the system achieved a higher accuracy of 93.1% with lower false positive and false negative rates compared to the two-people scenario, where accuracy dropped to 86.2%. This indicates the system's efficiency in simpler scenarios, while highlighting challenges in more complex settings with multiple individuals.

Table 5. Hand-raising detection accuracy and processing times for one and two scenarios. (Source: Created by the authors)

	One Person	Two People	Avg
FPR	5.9%	13.3%	9.6%
FNR	8.3%	14.3%	11.3%
Acc	93.1%	86.2%	89.7%
PT	0.142s	0.137s	0.14s
IT	0.02s	0.039s	0.03s



a. One person. b. Two people.

Figure 12. Visual examples of hand-raising detection tests conducted using a depth camera, for scenarios involving one person (a) and two people (b). The left side of each sub-plot displays real-time updates by our system that records the number of hand-raising actions detected. (Source: Created by the authors)

Figure 12 provides visual examples of hand-raising detection tests conducted using a depth camera for scenarios involving (a) one person and (b) two people. On the left side of each sub-plot, the system's real-time updates are displayed, which include the number of hand-raising actions detected. These images illustrate the system's capability to accurately monitor and record hand-raising actions, showcasing its real-time effectiveness in different individuals settings.

5. Conclusion

In this study, we demonstrated the effectiveness and efficiency of our proposed STF-GCN model in two key applications, namely fall detection in elderly care and hand-raising detection in smart education. (a) For fall detection, our model initially achieved an accuracy of 92.9% during the pre-training phase and was subsequently fine-tuned for specific fall scenarios, which boosted its accuracy to 97.6%. In real-world scenario tests, the model demonstrated substantial accuracy and rapid response capabilities, achieving an average accuracy of 96.3%, with a false positive rate of 6.25% and a false negative rate of 8.33%. The average inference time was a mere 0.045 seconds. (b) For hand-raising detection, targeted fine-tuning improved the model's accuracy from 95.9% to 98.5%. Tests in real-world scenarios confirmed the model's excellent performance, maintaining an average accuracy of 89.7%, a false positive rate of 9.6%, and a false negative rate of 11.3%. The average inference time was only 0.03 seconds. In summary, this study highlights the potential of the STF-GCN model to enhance action recognition in both elderly care and smart education, demonstrating high accuracy and efficiency in real-world scenarios.

This study represents the initial phase of our exploration into the integration of depth cameras and GCNs for action recognition. The findings detailed in this study not only validate the viability of our approach but also underscore its considerable potential for future advancements. This study is merely the first step in our research trajectory. Moving forward, we plan to expand our dataset significantly and establish benchmarks that will aid in the systematic evaluation of our methodology. Additionally, future work will involve a thorough assessment of external environmental factors that could impact the efficacy of our system. By addressing these elements, we aim to refine our models further and enhance their applicability in real-world scenarios, thereby continuing to advance the field of action recognition.

Acknowledge

This work was supported by JST SPRING, Grant Number JP-MJSP2111.

References

[1] J. Jang, D. Kim, C. Park, et al., "ETRI-activity3D: A large-scale RGB-D dataset for robots to recognize daily activities of the elderly," *IEEE International Conference on Intelligent Robots and Systems (IROS)*, pp.10990-10997, 2020.

[2] F. C. Lin, H. H. Ngo, C. R. Dow, et al., "Student behavior recognition system for the classroom environment based on skeleton pose estimation and person detection," *Sensors*, vol.21, no.16, pp.5314, 2021.

[3] L. Pan, J. Lu and X. Tang, "Spatial-temporal graph neural ODE networks for skeleton-based action recognition," *Scientific Reports*, vol.14, no.1, pp.7629, 2024.

[4] D. Lee, J. Lee and J. Choi, "CAST: cross-attention in space and time for video action recognition," *Advances in Neural Information Processing Systems*, 2024.

[5] S. Grover, V. Vineet and Y. Rawat Y, "Revealing the unseen: Benchmarking video action recognition under occlusion," *Advances in Neural Information Processing Systems*, 2024.

[6] C. Feichtenhofer, "X3d: Expanding architectures for efficient video recognition," *IEEE Conference on Computer Vision and Pattern Recognition*, pp.203-213, 2020.

[7] C. Y. Wu, Y. Li, K. Mangalam, et al., "Memvit: Memory-augmented multiscale vision transformer for efficient long-term video recognition," *IEEE Conference on Computer Vision and Pattern Recognition*, pp.13587-13597, 2022.

[8] Z. Zhang, "Microsoft kinect sensor and its effect," *IEEE multimedia*, vol.19, no.2, pp.4-10, 2012.

[9] J. Lee, M. Lee, D. Lee, et al., "Hierarchically decomposed graph convolutional networks for skeleton-based action recognition," *IEEE International Conference on Computer Vision*, pp.10444-10453, 2023.

[10] L. Wang and P. Koniusz, "3mformer: Multi-order multi-mode transformer for skeletal action recognition," *IEEE Conference on Computer Vision and Pattern Recognition*, pp.5620-5631, 2023.

[11] N. Trivedi, R. K. Sarvadevabhatla, "Psumnet: Unified modality part streams are all you need for efficient pose-based action recognition," *European Conference on Computer Vision*, pp.211-227, 2022.

[12] X. Zheng, J. Cao, C. Wang, et al., "A High-Precision Human Fall Detection Model Based on FasterNet and Deformable Convolution," *Electronics*, vol.13, no.14, pp.2798, 2024.

[13] X. Wang, J. Ellul and G. Azzopardi, "Elderly fall detection systems: A literature survey," *Frontiers in Robotics and AI*, vol.7, no.71, 2020.

[14] L. Ren and Y. Peng, "Research of fall detection and fall prevention technologies: A systematic review," *IEEE Access*, vol.7, pp.77702-77722, 2019.

[15] O, Díaz-Parra, A, Fuentes-Penna, R. A. Barrera-Cámara, et al., "Smart Education and future trends," *International journal of combinatorial optimization problems and informatics*, vol.13, no.1, pp.65, 2022.

[16] S. Mouti, H. Al-Chalabi, "A smart system for student performance assessment (SPA)," *Scientific Reports*, vol.14, no.1, pp.20679, 2024.

[17] X. Shu, J. Yang, R. Yan, et al., "Expansion-squeeze-excitation fusion network for elderly activity recognition," *IEEE Transactions on Circuits and Systems for Video Technology*, vol.32, no.8, pp.5281-5292, 2022.

[18] T. T. Zin, Y. Htet, Y. Akagi, et al., "Real-time action recognition system for elderly people using stereo depth camera," *Sensors*, vol.21, no.17, pp.5895, 2021.

[19] N. E. Tabbakha, W. H. Tan, C. P. Ooi, "Elderly action recognition system with location and motion data," *International Conference on Information and Communication Technology (ICoICT)*, pp.1-5, 2019.

[20] H. Zhou, F. Jiang and R. Shen, "Who are raising their hands? Hand-raiser seeking based on object detection and pose estimation," *Asian Conference on Machine Learning*, pp.470-485, 2018.

[21] J. Si, J. Lin, F. Jiang, et al., "Hand-raising gesture detection in real classrooms using improved R-FCN," *Neurocomputing*, vol.359, pp.69-76, 2019.

[22] W. Liao, W. Xu, S. C. Kong, et al., "A two-stage method for hand-raising gesture recognition in classroom," *International Conference on Educational and Information Technology*, pp.38-44, 2019.

[23] Microsoft, "Azure Kinect Body Tracking SDK 1.1.x documentation," [Online]. Available: https://microsoft.git hub.io/Azure-Kinect-Body-Tracking/release/1.1.x/index.ht ml. [Accessed: Feb. 6, 2025].

[24] L. Shi, Y. Zhang, J. Cheng, and H. Lu, "Two-stream adaptive graph convolutional networks for skeleton-based action recognition," *IEEE Conference on Computer Vision and Pattern Recognition*, pp.12026-12035, 2019.

[26] Z. Qin, Y. Liu, P. Ji, D. Kim, L. Wang, R. Mckay, S. Anwar, and T. Gedeon, "Fusing higher-order features in graph neural networks for skeleton-based action recognition," *IEEE Transactions on Neural Networks and Learning Systems*, 2022.

[26] J. Hu, L. Shen, and G. Sun, "Squeeze-and-excitation networks," *IEEE Conference on Computer Vision and Pattern Recognition*, pp.7132-7141, 2018.

[27] A. Shahroudy, J. Liu, T. T. Ng and G. Wang, "Ntu rgb+ d: A large scale dataset for 3d human activity analysis," *IEEE Conference on Computer Vision and Pattern Recognition*, pp.1010-1019, 2016.