Taylor & Francis
Taylor & Francis Group

Applied Artificial Intelligence
An International Journal

ISSN: (Print) (Online) Journal homepage: https://www.tandfonline.com/loi/uaai20

Hand Gesture Recognition of Methods-Time Measurement-1 Motions in Manual Assembly Tasks Using Graph Convolutional Networks

Alexander Riedel, Nico Brehm & Tobias Pfeifroth

To cite this article: Alexander Riedel, Nico Brehm & Tobias Pfeifroth (2022) Hand Gesture Recognition of Methods-Time Measurement-1 Motions in Manual Assembly Tasks Using Graph Convolutional Networks, Applied Artificial Intelligence, 36:1, 2014191, DOI: 10.1080/08839514.2021.2014191

To link to this article: https://doi.org/10.1080/08839514.2021.2014191

Published online: 21 Dec 2021.

APPLIED ARTIFICIAL INTELLIGENCE
2022, VOL. 36, NO. 1, e2014191 (1303 pages)
https://doi.org/10.1080/08839514.2021.2014191

Taylor & Francis
Taylor & Francis Group

Hand Gesture Recognition of Methods-Time Measurement-1 Motions in Manual Assembly Tasks Using Graph Convolutional Networks

Alexander Riedel, Nico Brehm, and Tobias Pfeifroth

Department of Industrial Engineering, Ernst-Abbe University of Applied Sciences, Jena, Germany

ABSTARCT

Gesture recognition is gaining popularity in many fields, including gesture control, robotics, or medical applications. However, the technology is barely used in industrial manufacturing processes due to high costs, a time-consuming configuration, and changes in the workflow. This paper proposes a minimally invasive approach to recognize workers' hand motions in manual assembly tasks. The novelty of this approach is the use of only one camera instead of any other sensors and the application of state-of-the-art graph neural networks. The method relies on monocular RGB video data to predict the basic motions of the industry standard motion-time system Methods-Time Measurement-1. Our two-stage neural network composed of hand key point extraction and adaptive graph convolution delivers accurate classification results in real-time. To train and validate the model, we created a dataset containing 22,000 frames of real-world assembly tasks. The data produced by this method in a production line can be used for motion time verification, assembly-line design, or assembly cost estimation. In a use-case study, we show that the proposed approach can generate Methods-Time Measurement analysis tables. These have so far only been accurately created by human experts. Source code: https://github.com/alexriedel1/Hand-Gesture-Recognition-in-manual-assembly-tasks-using-GCN

ARTICLE HISTORY
Received 28 July 2021
Revised 25 November 2021
Accepted 30 November 2021

Introduction

The key challenge for manufacturing companies is a globally continuing trend for shorter life cycles and highly customized products (Stump and Badurdeen 2012). To meet this challenge, assembly lines need to be complex but also fast to be planned and established (Manns, Otto, and Mauer 2016). In highly customized production environments, manual product assembly is often still the most popular choice when determining the level of automation (Salmi et al. 2016).

CONTACT Alexander Riedel alexander.riedel@eah-jean.de Alexander Riedel Ernst-Abbe University of Applied Sciences Carl-Zeiss-Promenade 207745 Jena Germany

Nico Brehm, https://dblp.org/pid/52/1674.htmlTobias Pfeifroth, https://www.researchgate.net/scientific-contributions/Tobias-Pfeifroth-2117172088

Studies find that manual assembly processes take up to 50% of the total production time and 20% of the total manufacturing cost in industrial manufacturing (Fan and Dong 2003). To ensure a high productivity, correct cost estimation and workers' health, a well-designed and well-planned assembly line is essential. Once in production use, the line might be monitored, evaluated, and further improved.

Getting access to data from manual production lines would not only improve the planning processes but also deliver useful insights for quality control, logistics planning, and management decisions. Key information derived from manual assembly lines are the types of performed motions, their duration, and their sequence. To not only use these data in the planning process, but also during production, the data need to be acquired in real-time, at a certain level of accuracy, and without intervening in the workers' assembly process.

In particular, these data contain the durations for specific motion tasks in a manual assembly line. The most widely used system for the classification of motion sequences and their duration uses the five basic motions grasp, move, position, release, and reach (Maynard, Stegemerten, and Schwab 1948). The corresponding motion duration is measured and defined by human experts for different kinds of distances, difficulty levels, and object shapes. In the manual assembly industry, it is desirable to automate the process of (1) identifying a basic motion and (2) measuring the motion duration.

There is a broad variety of motion capturing technology available, ranging from inertial motion units (Roetenberg, Luinge, and Slycke 2009), smartphone sensors (Qi, Hang, and Aliverti 2020), electromagnetic fields, electromyographic signals (Su et al. 2020) or depth cameras (Shafaei and Little 2016). Despite giving accurate results, wearable sensors are invasive and interfere with the workplace or disrupt the workflow. Also, installation and usage might not be trivial and cost intensive. Camera solutions are available at low cost, simple installation, and without interfering with the worker or the workplace. However, their performance heavily relies on the used data processing algorithms (Menolotto et al. 2020). This work provides a deep learning-based method to accurately predict the basic manual assembly motions from a single RGB camera signal. To our knowledge, there is no procedure described in the literature that can accomplish this task.

We propose a two-stage neural network for predicting Methods-Time Measurement 1 (MTM-1) motions from monocular RGB-Video data in real-time (see Figure 1). The first stage is a lightweight hand-pose model to extract 3D joint coordinates. These joints form a static hand graph, feeding a state-of-the-art spatio-temporal recurrent Graph Convolutional Network (GCN) for predicting motions from consecutive temporal signal frames. To train this neural network, we created and provide a dataset of roughly 21,000 frames containing manual assembly tasks that are labeled according to MTM-1 motions.

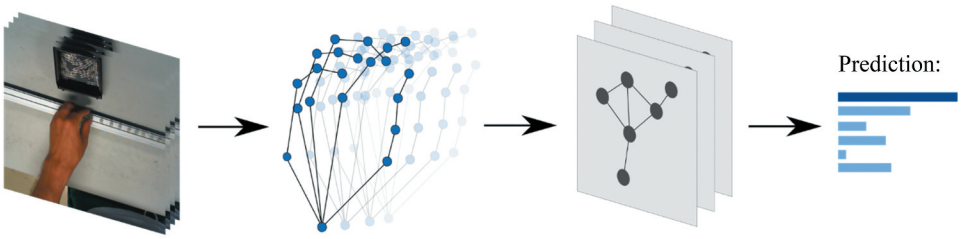

Figure 1. Two-stage approach of extracting hand key points and applying spatio-temporal adaptive graph convolution for predicting MTM-1 motions from live video data.

Time Studying in Assembly Processes

Time studies for measuring the production process can be divided into (1) stopwatch timing, (2) instantaneous observations, and (3) predetermined motion-time systems. Each of these depends on estimation, statistical tools and/or are affected by human factors (Polotski, Beauregard, and Franzoni 2019). Studies prove that both the spatial (Agethen et al. 2016) and temporal (Almeida and Ferreira 2009; Baines et al. 2003) dimension of these methods show inconsistency between theoretical planning and real-world assembly.

Predetermined motion-time systems (PMTS) are used to provide standard target times for predefined assembly tasks. They are based on the fundamental assumptions, that a predetermined time value can be assigned to every basic motion and that the duration of all basic motions sums up to the complete process time (Genaidy, Mital, and Obeidat 1989). The most widespread PMTS include Methods-Time Measurement (MTM) (Maynard, Stegemerten, and Schwab 1948) and Maynard Operation Sequence Technique (MOST) (Zandin 2020), with MTM as the de-facto standard in western industrial countries (Bures and Pivodova 2015). In the MTM basic system MTM-1, assembly tasks are analyzed, structured, and dissected into sequences of five basic movements: grasp, move, position, release, and reach (see Figure 2). Each basic motion is assigned a specific duration, based on defined influence factors, e.g., distance, complexity, or physical effort (Bokranz and Landau 2012). The five MTM-1 motions make up 80–85% of human movements during an assembly task and can be composed to sequences of moves (e.g., "Grasp and Release" or "Put in Place") for further analysis (Almeida and Ferreira 2009). Conducting a workplace planning using MTM-1 is a time-consuming task because every motion and its corresponding influence factor need to be determined by a qualified MTM expert (Bures and Pivodova 2015). The outcome of an MTM-1 workplace analysis is a table of the basic motions performed during an explicit task and their duration, as shown in Table 1. The motions are encoded as MTM codes that contain information about motion time influence factors like difficulty or distance. The time unit for measuring movements is referred to as Time Measuring Unit (TMU) with 1

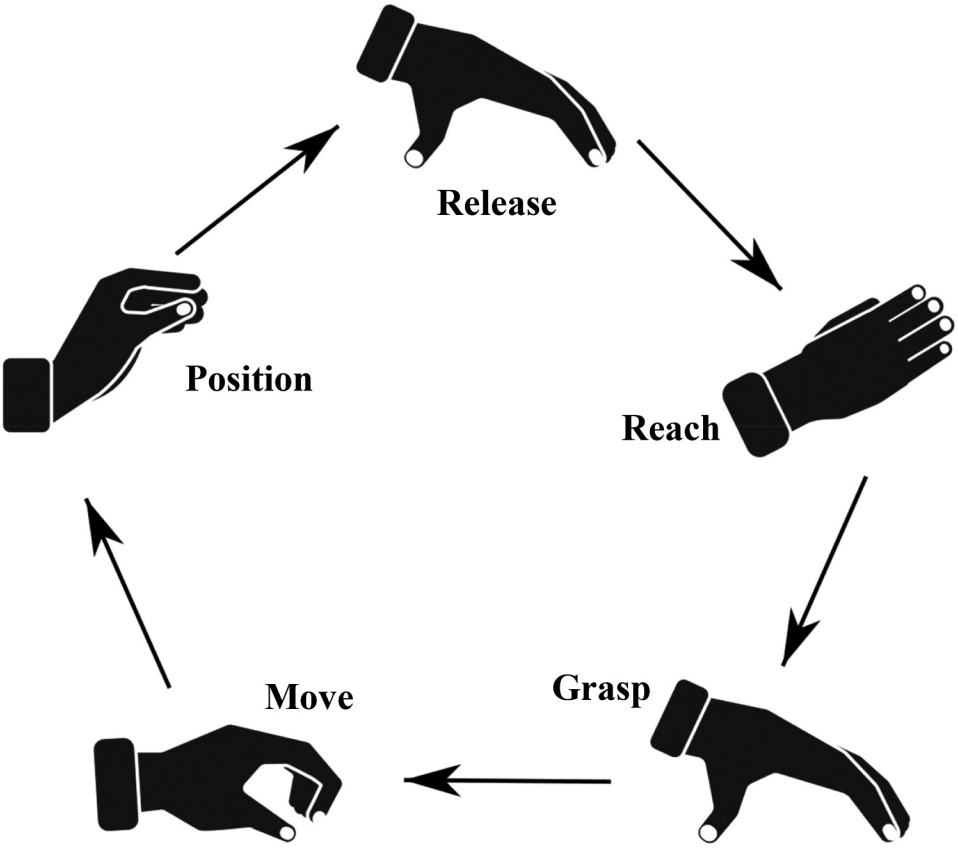

Figure 2. The five Methods-Time measurement basic motions in the commonly used sequence.

Table 1. Generic MTM-1 analysis for one task, conducted by an MTM expert.

No.	Basic motion	MTM-Code	TMU	Time in s
1	Reach	R 30 C	14.1	0.508
2	Grasp	G 4 C	12.9	0.464
3	Move	M 30 C	15.1	0.544
4	Position	P 1 D S	16.0	0.576
5	Release	RL 1	2.0	0.072
Total			**60.1**	**2.164**

The MTM-Code and the corresponding duration (TMU) is defined for every motion and distance by an international MTM consortium.

TMU = 0.036 s. Predetermined motion-time systems mainly use TMU instead of seconds to avoid decimal places and keep the high accuracy. The use of TMU in this work is due to this convention. Gathering the MTM-1 motions in real-time during production is only viable through the use of technological solutions such as wired sensors, wireless sensors, or camera-based solutions (Fantoni et al. 2021).

Recent Methods for Acquisition of Motion Data in Production

To identify, track, and monitor a worker's position during assembly, there is a wide range of more or less invasive or costly solutions. Most of these can be classified into systems that involve (1) fixed sensors, (2) wearable sensors, (3) virtual reality, or (4) camera monitoring.

Fixed sensor (1) solutions are based on laser-barriers (Yin et al. 2019) or ultra-sonic sensors (Pham et al. 2007) to detect the motion in specified areas, e.g., grabbing into a laser-barrier equipped box. Physical motions can be observed in a direct way wherever a sensor is placed. The downsides of using fixed physical sensors are their high cost, their labor-intensive set-up, and their high grade of workplace invasiveness (Fantoni et al. 2021).

Wearable sensors (2) combine all kinds of motion identification systems that are attached directly to the worker or to the processed material. Wireless RFID tags can be used to allocate the worker's or material's position (Zhong et al. 2013) from which motion data can be derived. By using wearable inertial measurement units (IMU) to directly access workers movements, the motion data can be enhanced (Fang and Zheng 2020). Despite being cheap and widely available, RFID sensors cannot be attached to every product and might interfere with the employee's workflow when being worn directly. Similar to IMUs, Qi et al. propose a method to use smartphone sensors to predict 12 fine-grained classes of human activities (Qi, Hang, and Aliverti 2020). The work proves the superiority of deep neural networks over classic machine learning methods for activity recognition. Also, the use of smartphones gives a widely available and inexpensive alternative to industrial IMUs. Only whole-body activities are examined, leaving doubts that the method works for fine-grained hand gestures.

Hang Su et al. show an approach of surface electromyographic (EMG) signals, obtained by a wristband and labeled via depth-camera, to predict hand gestures (Su et al. 2020). The multi-channel EMG wristband provides data for accurately predicting 10 gestures.

The discussed wearable solutions provide accurate body pose or hand gesture prediction; however, they need to be attached to the human body and might interfere with the workers' workflow.

Virtual or mixed reality solutions (3) have been extensively studied for the use of manual assembly verification (Gomes de Sá and Zachmann 1999; Chryssolouris et al. 2000; S. Li et al. 2009). They are proven to be suitable for virtual assembly simulations and deliver good approximations for assembly time studies. However, they are costly to implement, highly invasive and still show a low spread and a low expertise level in industrial environments (Masood and Egger 2019).

Camera-based monitoring systems (4) are a common method to analyze and track shop-floor data about the worker in manufacturing environments (Fang and Zheng 2020). Typically, those solutions rely on

motion tracking markers to extract positions from video data. These markers are either attached to the worker or to the workpiece (Puthenveetil et al. 2015; Wang, Ong, and Nee 2016). With the rise of low-cost depth cameras (RGB-D) like Microsoft® Kinect™ or Intel® RealSense™, marker-less camera motion monitoring became popular. Experimental setups using multiple RGB-D cameras have been successfully used for reconstructing walking paths (Agethen et al. 2016) to compare planned and real paths in manual assembly lines. Rude et al. show the use of RGB-D cameras to distinguish assembly tasks like fetching, painting and loading in clearly defined environments (Rude, Adams, and Beling 2018). Other studies demonstrate the use of these cameras for predicting fine-grained, but very general worker motions like "jerky motions in x-axis" (Prabhu et al. 2016).

Ovur et al. propose an alternative camera method for hand pose estimation, based on multiple infrared "Leap Motion Controllers" (LMC) (Ovur et al. 2021). The sensor fusion method, which incorporates multiple signal sources, reduces the risk of prediction failure due to occlusion. However, it remains unclear if LMCs are accurate enough to not only predict the hand pose as quaternions but also fine-grained gestures.

Depth-based camera methods such as LMC or RGB-D are crucial for estimating the exact position in a 3-dimensional space. For real-time hand gestures prediction though, true depth information may not be relevant and increases the computational cost, resulting in longer inference time.

The visual-only detection of the widely used and universally applicable MTM-1 motions has, to our knowledge, not yet been studied in the literature.

Many of the mentioned methods for predicting workers' motions in assembly tasks rely on the use of Hidden-Markov-Models (HMM). HMM tend to be useful in static situations where clear, pre-defined assumptions about state transitions and only minor temporal relations between states exist (Rabiner 1989). Neural networks, however, are capable of learning those assumptions, importance of far previous timesteps and other complex relations between features. When being trained on sufficient training data, neural networks are superior to HMM in terms of prediction accuracy (Tadayon and Pottie 2020; L. Liu, Lin, and Reid 2019).

Materials and Methods

Recently, deep neural networks have been widely investigated in general gesture and human action recognition, but merely in manual assembly motion prediction. Tran et al. present a method for extracting features from RGB videos via 2D convolutions followed by a temporal 1D convolution, resulting in a "(2 + 1)D" convolutional block (Tran et al. 2018). Min et al. propose to first extract temporal point clouds that are fed as features into a recurrent long

short-term memory sequence model (LSTM) to preserve more spatial features than from RGB data only (Min et al. 2020). Another feature representation method that has proven to be well applicable for action recognition tasks is based on human skeletal joints (Yan, Xiong, and Lin 2018). The main advantage over RGB data lies in the sparsity and compactness of such data, where only pose information is included. This results in robustness against noisy backgrounds, changing lightning conditions or bad camera positions. The gesture of a human hand, for example, can be represented as a sequence of 3D coordinates of the hand's skeletal joints. For gesture recognition, this information can be fed into an LSTM or other recurrent networks (J. Liu et al. 2017; P. Zhang et al. 2017). As human actions heavily rely on the spatial information between the skeletal joints and not only on the temporal information, standard recurrent networks only work to a certain extent. Spatio-temporal graph convolutional networks (ST-GCN) (Yan, Xiong, and Lin 2018) have been the most promising approach to capture spatial and temporal relations between the joints and the sequences of joints. For this purpose, a skeleton graph is pre-defined, according to the natural human body structure. This approach, however, shows several downsides. Defining the skeletal graph edges according to the body structure might be misleading as the GCN could gather more information from alternative connections. Fixing the graph over every GCN layer is counterintuitive to the fact that the neural network might learn different features in each layer. Predicting different classes of action might require different connections between the skeletal joint. To overcome these issues, Shi et al. propose a Two-Stream Adaptive GCN (2s-AGCN) with adaptive graphs and an attention mechanism (Shi et al. 2018). The graph topology is an optimized parameter in the network and unique for each layer, but also a residual branch to achieve stability in the network. The spatial and temporal dimension of a sequence of hand graphs is visualized in Figure 3, also displaying the adaptive importance of each node. The 2s-AGCN model was designed to work well on human body skeletal datasets like NTU-RGBD or Kinetics-Skeleton where it achieves state-of-the-art accuracy. We are aiming to use 2s-AGCN on MTM-1 skeletal motion data that show (1) much more fine-grained motions, (2) a low inter-class variance (3) a limited amount of available data and (4) a naturally imbalanced class distribution. To adapt the model to the new scope of application, several changes considering the architecture of the model, the underlying graph and the input features will be evaluated. To better understand how to implement the described methods in a real-world use case, each stage of the pipeline will be briefly explained.

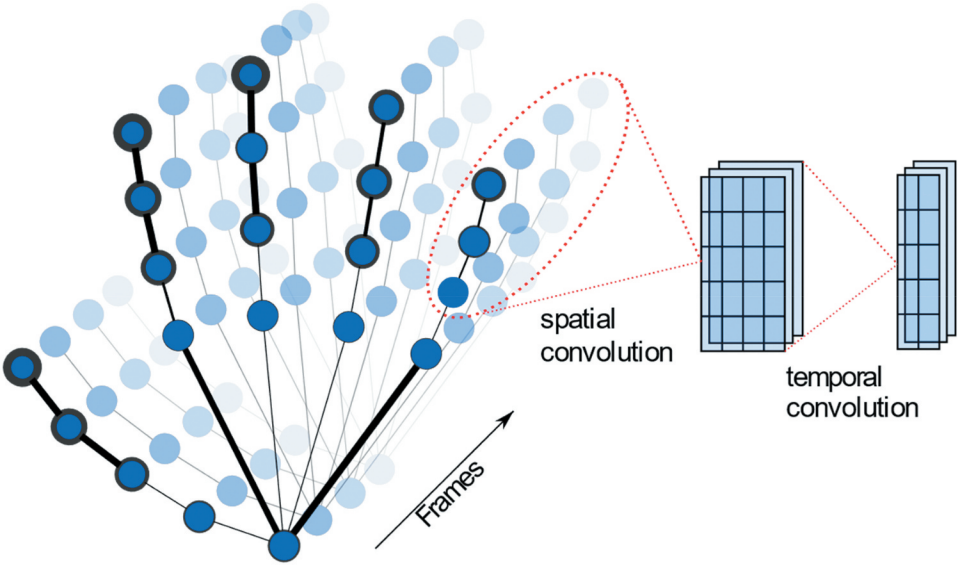

Figure 3. A sequence of hand graphs during a motion. The line width of the graph vertices denotes the learned joint importance of the adaptive GCN. Spatial and temporal convolution is performed on neighboring graph nodes, increasing the number of feature maps and decreasing the number of temporal columns.

Key Point Extraction

To extract the hand key points coordinates from RGB video data, a pre-trained, ready-to-use solution is preferred. Most of the available methods rely on the use of convolutional neural networks and are trained on large amounts of data (Cao et al. 2021; Kreiss, Bertoni, and Alahi 2019). Due to its fast inference speed, high accuracy, and simple use, the framework chosen in this work is MediaPipe Hands (F. Zhang et al. 2020). It allows to extract 21 x-y-z hand key points, following the structure shown in Figure 3. The implemented SSD model achieves 95.7% accuracy for detecting palms and an MSE of 10.05 pixels for predicting the key points. With an inferencing time of 5.3 ms on an iPhone 11, it allows the use in real-time applications even on low-performance edge devices. The hand key point data was acquired via the Python API of MediaPipe Hands, using a minimum detection confidence of 0.4 and a minimum tracking confidence of 0.5 for tracking key points in consecutive frames.

Defining the Hand 2s-AGCN

Spatio-temporal GCNs for hand gesture recognition use an embedded sequence of joints and the corresponding adjacency matrix as input. The proposed AGCN is fed by a four-dimensional matrix in the shape of [N, C, T, V] where N denotes the batch size, C denotes the number of input features

(x-y-z-coordinate), T denotes the input time steps and V denotes the graph vertices (joints). The adjacency matrix for AGCN is composed of three sub-matrices: (1) the inward-links starting from the wrist joint, (2) the outward-links pointing in the opposite direction, and (3) the self-links of each joint. Thus, the matrix is of the shape [V, V, 3], where V is 21 in this work.

The Hand-AGCN model used in this work is a stack of 7 AGCN blocks with increasing output feature dimensions. A preceding batch normalization layer is added for input data normalization. A global average pooling layer (GAP) followed by a fully connected layer (FC) maps the output features to the corresponding output classes. The model structure is shown in Table 2. One AGCN-block is composed of a spatial graph convolution followed by a temporal graph convolution with respective kernel sizes.

The adaptive spatial graph convolution layer uses a combination of both the provided adjacency matrix as well as parameterized and optimized adjacency matrices. Equation (1) represents the general formula to perform this adaptive spatial graph convolution.

$$\mathrm{f}_{out} = \sum_{k}^{K_v} \mathrm{W}_k * f_{in}(\mathrm{A}_k + \mathrm{B}_k + \mathrm{C}_k) \tag{1}$$

where K_v denotes the kernel size of the spatial dimension (set to 3 in this work), W_k represents the weight vector of the $1*1$ convolution, f_{in} is the input feature vector and A_k, B_k, C_k are the adjacency matrices used. The first matrix A_k is the originally provided adjacency matrix. B_k is a completely learnable, parameterized adjacency matrix, able to learn new connections. C_k is a data-dependent adjacency matrix, obtained by embedding the input features through a $1*1$ convolutional operation and a *softmax* function. The spatial graph convolutional is followed by batch norm, a ReLU activation, and a dropout layer with 0.5 dropout rate. Consecutively, the temporal graph convolutional is performed as proposed by (Yan, Xiong, and Lin 2018),

Table 2. Layers and dimensions of the implemented AGCN model.

Layer	Input size	Channels	Temporal stride	Output size
Input	(32, 21, 3)			(32, 21, 3)
Batch normalization				
AGCN 1	(32, 21, 3)	64	1	(32, 21, 64)
AGCN 2	(32, 21, 64)	64	1	(32, 21, 64)
AGCN 3	(32, 21, 64)	64	1	(32, 21, 64)
AGCN 4	(32, 21, 64)	128	2	(16, 21, 128)
AGCN 5	(16, 21, 128)	128	1	(16, 21, 128)
AGCN 6	(16, 21, 128)	128	1	(16, 21, 128)
AGCN 7	(16, 21, 128)	128	2	(8, 21, 128)
Global average pool	(8, 21, 128)			(1, 128)
Fully connected	(1, 128)	6		(1, 6)

The input vector is organized the following: (time steps, key points, features). The number of channels refers to the number of convolutional feature maps both in the temporal and spatial convolution. The temporal kernel stride is set to down sample and embed the temporal dimension.

Table 3. Details of the adaptive graph convolutional (AGCN) layer introduced in Table 3.

Layer	Kernel size
Residual	
Adaptive spatial convolution	(1,1)
Batch norm + ReLU	
Dropout	
Temporal convolution	(1,1) ⊕
Batch norm + ReLU	

⊕ denotes the element-wise summation in the residual layer.

again followed by a batch norm layer and ReLU activation. Each of the AGCN blocks has a residual connection for stabilizing the training. See Table 3 for the AGCN layer summary. The implementation details of all layers and the model can be further investigated in the provided GitHub repository and can be found in (Shi et al. 2018).

The hand joint coordinates can be utilized to derive second-order information, i.e., the bone length. To benefit from this additional information, bone representations will be calculated and used as input features.

These bone representations are defined as vectors pointing from the source joint to the target joint. To keep information about the direction, a simple subtraction of the joint positions is performed rather than calculating the actual distance. The bone that is connecting joints v_1 and v_2 is defined by $e_{1,2} = (x_1 - x_2, y_1 - y_2, z_1 - z_2)$. To not mix the information about joint coordinates and bone lengths as vertex features, two models will be trained separately. The joint-stream (J-Stream) and the bone-stream (B-Stream) predictions will be averaged for the final prediction score. This ensemble of models represents the 2-Stream AGCN architecture.

Hand Graph Modeling

The MediaPipe Hands key point extraction method predicts the x-y-z-coordinates of 21 hand joints; four joints per finger plus an additional wrist joint (Figure 4). For the definition of the underlying graph, each of the joints is connected to its natural neighbor, resulting in a graph of 21 vertices and 20 edges. As Y. Li et al. 2019 propose, these might be too few connections for the fine-grained hand movements and the low inter-class variability of the MTM-1 motions. To obtain more semantic information in the hand graph, two types of additional joints were added, as displayed in Figure 4. The first type of added joints links the fingertips to the base of the right neighbor finger. The second type of additional joints links the fingertips to the middle segment of the same finger. These supplementary links help to retrieve more information about different states of the hand. The first type contains data

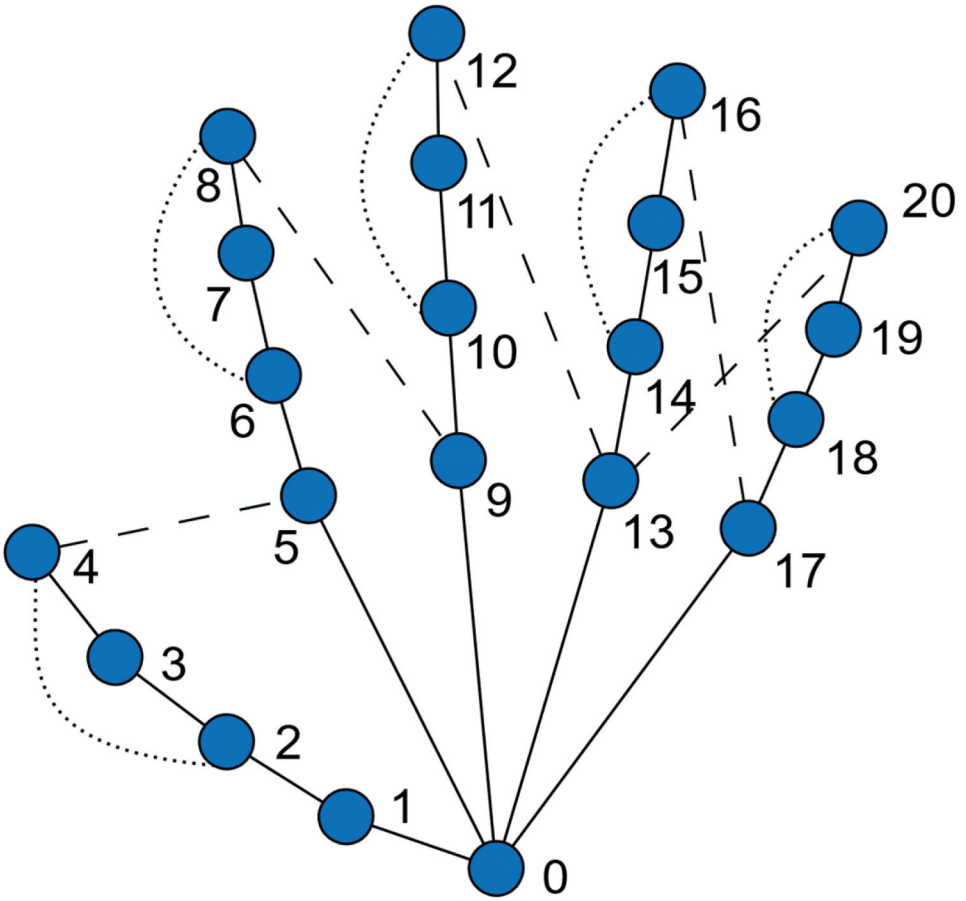

Figure 4. Arrangement of the hand key points extracted by MediaPipe Hands. Solid lines denote the natural joint links. Dashed lines denote first type of additional links. Dotted lines denote the second type of additional links.

about the horizontal and vertical distance of two fingers and can therefore help to encode overlapping or spreading of two fingers. The second type encodes bending of the fingers and thus creates information about, e.g.,, grabbing motions. These additional links are applied to the J-Stream but also to the B-stream model. This will result in "pseudo-bones" that might carry important length information for some basic motions.

As the model adaptively learns the importance of links between features, unnecessarily introduced edges will vanish during the training process and meaningful edges will solidify. Thus, adding explainable links might not decrease the model's prediction quality, whereas simply connecting every joint with all the others might overfit the model.

Two types of models are trained with one including the motion class "Release" and one excluding "Release." Including "Release" in a model is theoretically necessary to predict all MTM-1 classes. For the practical use of

the model, including "Release" is not found to be obligatory, as in practice, the duration associated with this motion is mostly set to be 0. By excluding this class, the accuracy of all other classes can be improved without affecting the practical usefulness of the model predictions. The prediction results of both models, however, will be presented.

MTM-1 Data Acquisition

The MTM-1 dataset was acquired performing a variety of one-handed manual assembly tasks under different workplace conditions. The workpiece as well as the components to be assembled were put in different positions to reflect various real-world situations. Single parts of diverse shapes, with diameters from 1 cm to 10 cm were assembled using different manual assembly techniques, like plugging, screwing, and placing. The assembly was performed by different people to capture a range of motion and assembly characteristics. The sequences were filmed from varying distances and angles, using a Microsoft LifeCam HD-3000 720p RGB webcam at 25 frames per second. The camera was mounted in different positions from 0.5 m to 1 m above the assembly station at angles ranging from 0° to 30°. Labeling the data was done manually per frame, using the five MTM-1 basic motions "Grasp," "Release," "Reach," "Position" and "Move." Frames without the presence of hands were labeled "Negative," resulting in a total of six classes for the dataset. The motions can be well distinguished during a manual assembly task, e.g., "Reach" is the basic element when the purpose of motion is to move the hand to definite location. "Move" is the motion performed when an object is to be transported to a definite destination. "Grasp" is performed to secure control over an object. "Position" is the element employed to align and engage one object with another one. "Release" is associated with opening the fingers to break contact between hand and object. In real-life, the motions always appear in a certain order and hence single motions cannot be sampled without sampling the other motions. This explains the degree of class imbalance that can be found in the dataset.

Results

Dataset

The training dataset comprises 11 videos with a total of 17,535 frames. The categories are distributed as of the following: "Grasp" 4,863, "Position" 4,293, "Move" 3,579, "Reach" 2,863, "Release" 682, "Negative" 1,255 frames. The validation dataset comprises three videos with a total of 4,241 frames. The categories are allocated as of the following: "Grasp" 919, "Position" 1,359,

"Move" 881, "Reach" 716, "Release" 95, "Negative" 271 frames, resembling a natural distribution for manual assembly tasks. To obtain a meaningful train and test split, complete assembly videos are used for validation showing different tasks than the training data.

The video footage was processed using the MediaPipe Hands Python API to obtain the 21 x-, y- and z-hand joint coordinates per frame. The processed dataset will hence be referred to as hand graph dataset. During the training process, the hand graph data was augmented by generating vertically, horizontally, and both ways flipped graphs of the original data. Doing so, the training dataset could be enlarged four times and the resulting model will be invariant to flipping and more robust.

Training Details

The experiments were performed using the PyTorch framework (Paszke et al. 2019). The training and validation data is organized as a [N, C, T, V] tensor, with the batch size N, the input features C, the consecutive frames (time steps) T and the hand vertices V. For the training process, we set N to be 64 and C to be 3 (x-y-z coordinates). The batch size was determined in a preliminary experiment using mini-batches of 32, 64, and 128, with 64 resulting in the highest accuracy in the validation set. The relatively small batch sizes were considered, as they might lead to smoother minimizers and higher generalization ability (Keskar et al. 2016). The number of time steps T was empirically specified to be 32 and V was set to be 21, resembling the available hand joints. Thirty-two consecutive frames, correspondingly 1.3 s in a 25 FPS video stream, were found to be sufficient for a human labeling expert to identify the correct motion. Increasing the number of frames increases the computational cost both during training and inferencing time and does not lead to higher accuracy.

The training was conducted using ADAM optimization with a starting learning rate of 1e-4 and a plateau learning rate scheduler set on the validation accuracy with a 0.5 reduction factor and 5 epochs patience. The learning rate and corresponding scheduler parameters were determined in a series of experiments using varying learning rates, ranging from 1e-3 to 1e-5, and reduction factors, ranging from 0.1 to 0.5.

The models were trained for a maximum of 30 epochs or until no further validation accuracy increase was measured. During the experiments, it was observed that after 30 epochs none of the models showed any substantial accuracy increase.

Ablation studies were performed using Sharpness-Aware Minimization (SAM) as a wrapper to the ADAM optimizer (Foret et al. 2020) for improving the models generalization ability. Considering the high class-imbalance, a class-weighted binary cross-entropy (BCE) loss function as well as focal loss (Lin et al. 2017) were applied. The weighted BCE loss function acts similarly to

oversampling underrepresented classes. Focal loss reduces the importance of easily classified and focuses on hardly classified examples which tackles the problem of over-confidence in unbalanced dataset. Collecting additional data to handle the class imbalance does not lead to the desired result as the class distribution is naturally determined for a manual assembly task.

Ablation Study

We examined the effectiveness of training optimization, the implementation of additional joints as well as joint and bone model ensembling. The experiments were conducted for the joint-stream adaptive graph convolutional network, which was trained and validated on the proposed MTM-1 dataset. For the ablation evaluation, the overall accuracy was used as a metric to achieve a general comparability between the experiments (see Table 4).

For the baseline model, no additional joints were added, and weighted BCE loss was used in the training process resulting in a mean accuracy of 78.01%.

The class imbalance was addressed by using focal loss as it focuses the training more on hard to classify examples. The proposed use of focal loss led to an accuracy improvement of 3.52% from the baseline.

Besides the already implemented dropout and batch normalization layers, we introduced Sharpness-Aware Minimization (SAM) to the ADAM optimizer, to further prevent overfitting. Wrapping ADAM with SAM gave an additional accuracy boost of 2.04% and stabilized the validation loss when training for many epochs.

The additional joints type 1 and 2 improved the model's prediction capabilities by 0.92%, by enabling the model to encode more information on certain hand poses.

Through ensembling the separately trained joint-stream model and bone-stream model (2s-AGCN), the accuracy could be improved by another 1.29%, see Table 5. This leads to the conclusion that both streams learn individual representations each of which are meaningful for the motion prediction.

Table 4. Comparison of the validation accuracy when applying different training techniques and adding joints to the natural hand graph to the joint-stream-AGCN.

+ Joints 1	+ Joints 2	SAM	Loss	Accuracy (%)
			BCE	78.01
			Focal	81.53
		✓	Focal	83.57
✓		✓	Focal	82.94
	✓	✓	Focal	83.47
✓	✓	✓	Focal	84.39

Table 5. Accuracies and average F1-scores of the two stream methods and two-stream ensemble.

Method	Accuracy (%)	Ø F1-score (%)
Joint-Stream AGCN	84.39	76.82
Bone-Stream AGCN	81.43	74.83
2-s AGCN	85.68	78.21

To give more insights about the individual class performance, the F1-score (harmonic mean of precision and recall) per class is calculated for the final 2s-AGCN in Table 6. The F1-Score is considered due to the high class-imbalance as it does not correspond to the underlying class distribution, unlike accuracy.

The model predicts all classes at an accuracy of at least 82.75% except for the "Release" class, showing 23.44% accuracy. The high variance can be explained by the underrepresentation of the "Release" class in the training dataset with only 3.8% of the data showing the "Release" motion. Moreover, "Release" shows a high intra-class variability in terms of motion and duration and cannot even be observed by the human eye for certain assembly tasks.

The confusion matrix (Figure 5) shows that "Release" is mainly mistaken with "Position." This is explained by the natural sequence of the manual assembly process where positioning an object is followed by a release operation. Other mistaken predictions are also mainly neighbors in the motion sequence cycle (Figure 3).

Table 6. Class-wise F1-scores and unweighted F1-scoresof the final 2s-AGCN with all classes compared to the 2s-AGCN trained without the "Release" class. Bold values imply higher F1-score.

Class	F1-score (%)	F1-score w/o release (%)
Grasp	86.79	**86.87**
Move	**86.07**	85.39
Position	86.47	**91.07**
Reach	82.75	**83.83**
Release	**23.44**	-
Negative	98.96	**99.00**
Ø F1-score	78.21	**89.23**

Table 7. Comparison of LSTM, Spatio-Temporal-GCN and two-stream AGCN accuracy and average F1-score on the MTM-1 hand graph dataset.

Method	Accuracy (%)	Ø F1-score (%)
LSTM	70.75	64.32
ST-GCN	75.14	66.02
2s-AGCN	85.68	78.21

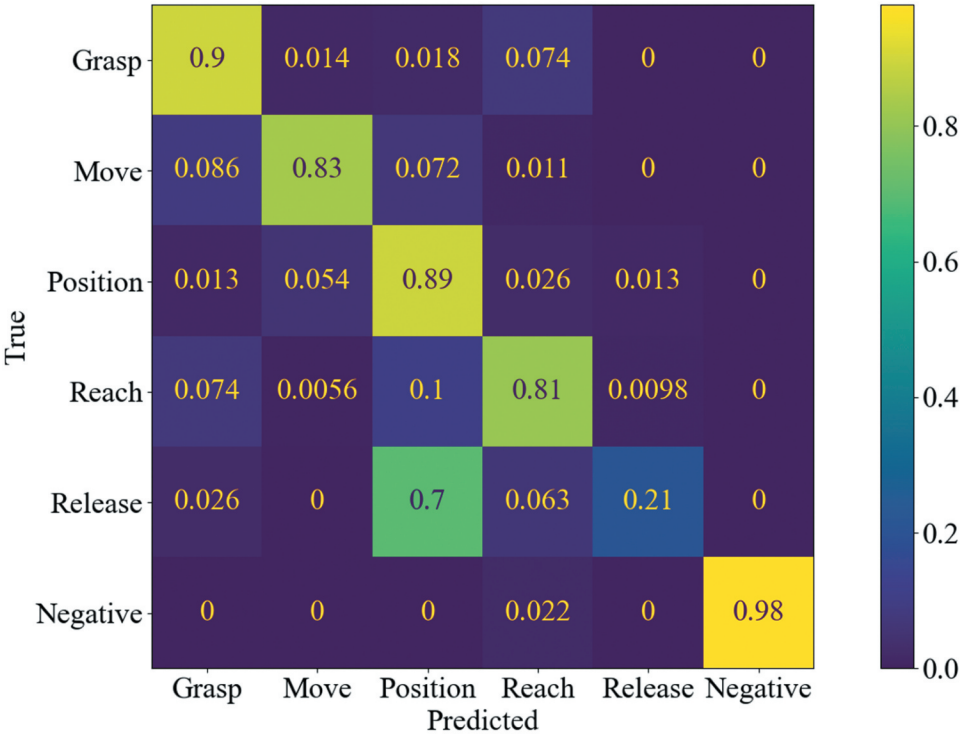

Figure 5. Normalized confusion matrix of the final 2s-AGCN model.

For the real-world use of tracking assembly data on the shopfloor and verifying planned assembly tasks, the motion "Release" might not be crucial to identify correctly. In MTM-1 it is either measured with a fixed value for actively releasing an object (2 TMU = 0.072 s) or not even considered for passive releasing (0 TMU) (Bokranz and Landau 2012). Thus, we trained additional models replacing "Release" with "Position," showing superior classification results for most of the other classes, as shown in Table 6.

The lightweight architecture of the two-stage pipeline allows real-time inference on edge devices. The model combination of MediaPipe Hands and 2s-AGCN shows a total of 3.6 M parameters and 30 ms inference time on the Nvidia Jetson Nano.

Visualization and Explainability

We can show that the learned adjacency matrix of the AGCN model's last layer is more diverse than the original one and reflects intuitive assumptions about the importance of hand joints for gesture recognition. Across the networks

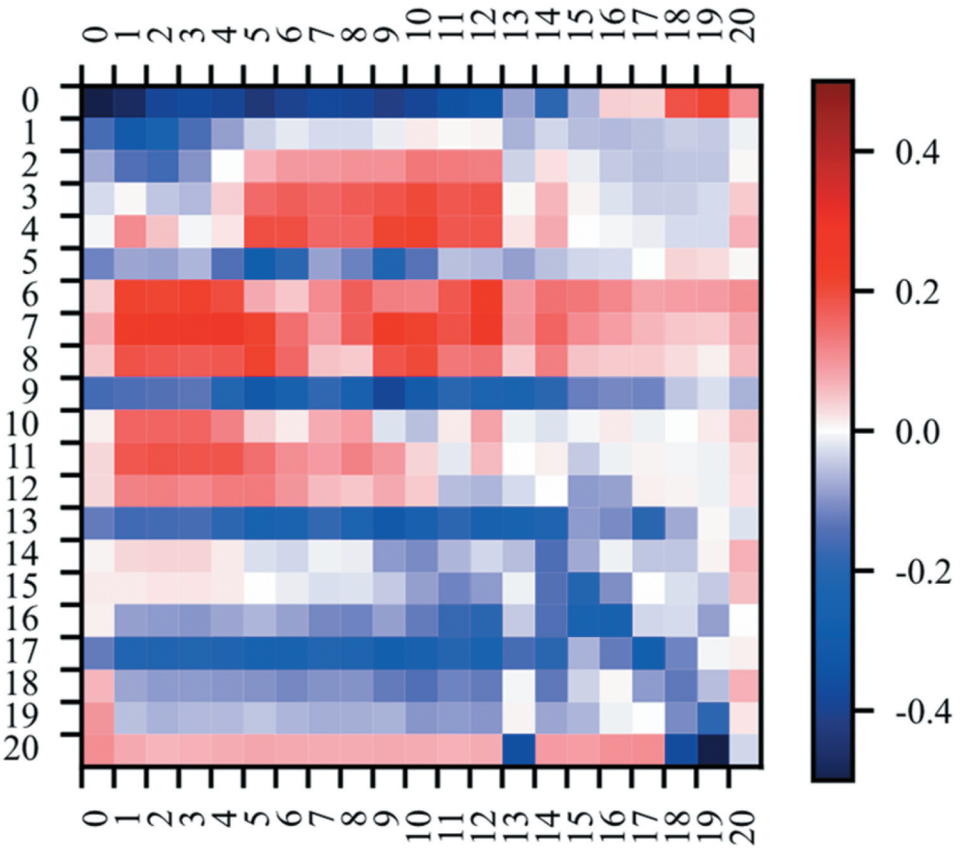

Figure 6. Absolute difference between original hand joint adjacency matrix and learned adjacency matrix. The figure shows how connections between joints 5–13 (upper-left section) are strengthened in the learning process. Connections between joints of the last two fingers (joints 13–20) are weakened (lower section) or unaffected (upper-right section).

layer, the adjacency matrices show largely varying appearances, leading to the assumption that each layer evolves a specific graph structure according to its learned representations.

The analysis of the difference between the original and the learned adjacency matrix shows how connections between the first three fingers evolve, whereas other connections vanish (Figure 6). Thumb, index, and middle fingers are of special importance for recognizing positioning and grasping movements, which can explain the models' stronger focus on these connections. Also, the joints from almost every finger to the index finger (joints 6–8) strengthened, which emphasizes the natural importance of this finger for the considered motions.

State-of-the-Art Comparison

For comparing different approaches on MTM-1 motion recognition from hand key points, the experiments were also performed using a simple LSTM model (Hochreiter and Schmidhuber 1997) and a non-adaptive ST-GCN (Yan, Xiong, and Lin 2018). The LSTM was defined by 120 hidden nodes on two layers, followed by fully connected output layer. The ST-GCN was implemented as provided by the PyTorch Geometric Temporal extension library (Rozemberczki et al. 2021) with 16 spatial channels and a fully connected output layer. All models were trained on the joint data.

The superiority of AGCN (85.68% accuracy) over LSTM (70.75% accuracy) and ST-GCN (75.14% accuracy) on the MTM-1 dataset can be clearly proven, underlining previous research results on other motion datasets such as NTU-RGB or Kinetics (Table 7).

Case Study

In a use-case study, we demonstrate how the proposed method can be used to generate MTM-1 analysis tables of manual assembly tasks during the manufacturing process. For this purpose, we analyze the inference results of validation video 1 of the MTM-1 dataset.

The achieved validation accuracy of the final model is sufficient to make useable predictions that generally follow the ground truth signal, shown in Figure 7. This diagram resembles the class-wise F1-Scores of the final model. Analyzing Figure 7 also reveals that the low F1-score of the "Release" class might be partially explainable by the wrong timing of the prediction, rather than by a missing prediction.

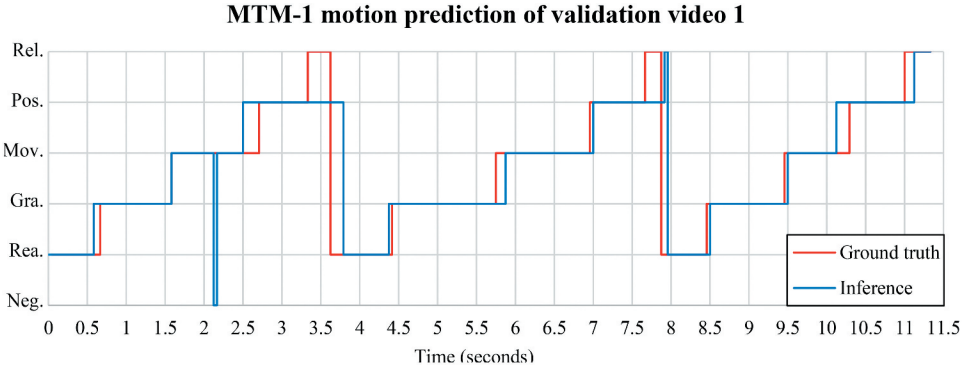

Figure 7. Prediction of the MTM-1 motions with our proposed method on validation video 1 of the MTM-1 dataset. The inferencing is generally accurate compared to the ground truth in terms of the predicted classes and motion duration. The key point extraction method fails to predict the hand coordinates, leading to a "Negative" prediction at 2.2 s. Also, the algorithm shows difficulties in predicting the "Release" class at 3.4 s or predicts the class too late (7.7 s).

Table 8. Algorithmic MTM-1 analysis of the assembly task shown in validation video 1, MTM-1 dataset.

No.	Basic motion	TMU	Time in s
1	Reach	16.1	0.58
2	Grasp	27.7	0.997
3	Move	25.4	0.914
4	Position	35.8	1.289
5	Reach	16.2	0.583
6	Grasp	41.6	1.498
7	Move	31.2	1.123
8	Position	25.4	0.914
9	Release	1.2	0.043
10	Reach	15.0	0.54
11	Grasp	27.7	0.997
12	Move	17.3	0.623
13	Position	27.7	0.997
14	Release	5.8	0.209
Total		314.1	11.308

The inference results of validation video 1 can be directly used to create an MTM-1 analysis table (Table 8) for planning, verification, or monitoring the manual assembly task. The only post-processing method applied to the data was filtering out short-time predictions of the "Negative" class, to eliminate noisy inter-class transitions. Restrictions about the permissible order of the predictions might further improve the accuracy in the real-world use. For example, "Reach" cannot be followed by "Move" without a "Grasp" prediction in between.

Discussion

The system proposed in this work has been proven to meet the requirements for real-time MTM-1 motion prediction in an industrial manual assembly environment. As the results show, our two-stage approach of a key point extractor and a graph convolutional network is capable of this challenging task, using data from only one RGB camera. The issue of a naturally imbalanced dataset and the demanding training process of the AGCN were overcome by using focal loss and regularization techniques. Also, we were able to briefly explain the models learned representations by visualizing the adaptive adjacency matrix.

The low workplace invasiveness and the high prediction quality differentiate the proposed method from other system in the literature that rely on physical sensors or inferior prediction models.

A real-world case study demonstrates that the graph convolutional networks predictions can be used to generate MTM workplace analysis tables, including the motion duration for each assembly phase. This qualifies non-experts to identify assembly times for products and to validate existing MTM analyses on many workstations simultaneously.

For applying the approach in a wide range of real-world use cases, trained models can be fine-tuned to specific scenarios as the MTM-1 motions are predefined and universally used.

The difficulties in predicting the "Release" class can partly be overcome by additional training data. As this motion is not even observable by human experts in many assembly situations, "Release" might also be considered a fix prediction for a defined number of frames or simply not considered at all.

Since the proposed method relies on RGB video data, the operator's hands must be fully visible at all times to make meaningful predictions. As this cannot be guaranteed for all manual assembly activities and workplaces, the scope of application is limited.

In future research, more real-world use cases and their benefits to companies should be studied and discussed. These also include the evaluation of efficiency gains and cost reduction by using the proposed MTM-1 motion model in production. Neural network architectures that are able to predict more complex gestures derived from the MTM-1 basic motions should also be investigated, as they can provide additional useful insights. The additional MTM motions might include full-body motions like lifting and walking or eye movements. Body and head key point models are widely available and could be used analogously to the hand model.

To overcome the limits of visibility, multi-sensor data fusion combining RGB, electromyographic signals, and depth signals might be a promising research direction for gesture recognition in industrial environments.

Disclosure Statement

No potential conflict of interest was reported by the author(s).

ORCID

Alexander Riedel http://orcid.org/0000-0002-1466-3206

References

Agethen, P., M. Otto, F. Gaisbauer, and E. Rukzio. 2016. Presenting a novel motion capture-based approach for walk path segmentation and drift analysis in manual assembly. *Procedia CIRP* 52:286–91. https://doi.org/10.1016/j.procir.2016.07.048. doi:10.1016/j.procir.2016.07.048.

Almeida, D., and J. Ferreira. 2009. Analysis of the Methods Time Measurement (MTM) methodology through its application in manufacturing companies. 19th International Conference on Flexible Automation and Intelligent Manufacturing (FAIM 2009), Middlesbrough, United Kingdom. Middlesbrough, 10.13140/RG.2.1.2826.1927.

Baines, T., L. Hadfield, S. Mason, and J. Ladbrook. 2003. Using empirical evidence of variations in worker performance to extend the capabilities of discrete event simulations in manufacturing. Proceedings of the 2003 International Conference on Machine Learning and Cybernetics (IEEE Cat. No. 03EX693), Xi'an, China, 1210–16. New Orleans, IEEE. 10.1109/WSC.2003.1261552.

Bokranz, R., and K. Landau. 2012. *Handbuch Industrial Engineering: Produktivitätsmanagement Mit MTM*. 2nd ed. Stuttgart: Schäffer-Poeschel.

Bures, M., and P. Pivodova. 2015. Comparison of time standardization methods on the basis of real experiment. *Procedia Engineering* 100: 466–74. doi:10.1016/j.proeng.2015.01.392.

Cao, Z., G. Hidalgo, T. Simon, S.-E. Wei, and Y. Sheikh. 2021. OpenPose: Realtime multi-person 2D pose estimation using part affinity fields. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 43 (1):172–86. doi:10.1109/TPAMI.2019.2929257.

Chryssolouris, G., D. Mavrikios, D. Fragos, and V. Karabatsou. 2000. A virtual reality-based experimentation environment for the verification of human-related factors in assembly processes. *Robotics and Computer-Integrated Manufacturing* 16 (4):267–76. doi:10.1016/S0736-5845(00)00013-2.

Fan, J., and J. Dong. 2003. Intelligent virtual assembly planning with integrated assembly model. SMC'03 Conference Proceedings. 2003 IEEE International Conference on Systems, Man and Cybernetics. Conference Theme - System Security and Assurance (Cat. No. 03CH37483), Washington, DC, USA, 5: 4803–8vol.5. doi:10.1109/ICSMC.2003.1245743.

Fang, W., and L. Zheng. 2020. Shop floor data-driven spatial–temporal verification for manual assembly planning. *Journal of Intelligent Manufacturing* 31 (4):1003–18. doi:10.1007/s10845-019-01491-y.

Fantoni, G., S. Q. Al-Zubaidi, E. Coli, and D. Mazzei. 2021. Automating the process of method-time-measurement. *International Journal of Productivity and Performance Management* 70 (4):958–82. doi:10.1108/IJPPM-08-2019-0404.

Foret, P., A. Kleiner, H. Mobahi, and B. Neyshabur. 2020. Sharpness-Aware minimization for efficiently improving generalization, October. http://arxiv.org/abs/2010.01412.

Genaidy, A. M., A. Mital, and M. Obeidat. 1989. The validity of predetermined motion time systems in setting production standards for industrial tasks. *International Journal of Industrial Ergonomics* 3 (3):249–63. doi:10.1016/0169-8141(89)90025-5.

Gomes de Sá, A., and G. Zachmann. 1999. Virtual reality as a tool for verification of assembly and maintenance processes. *Computers & Graphics* 23 (3):389–403. doi:10.1016/S0097-8493(99)00047-3.

Hochreiter, S., and J. Schmidhuber. 1997. Long short-term memory. *Neural Computation* 9 (8):1735–80. doi:10.1162/neco.1997.9.8.1735.

Keskar, N. S., D. Mudigere, J. Nocedal, M. Smelyanskiy, and P. T. Peter Tang. 2016. On large-batch training for deep learning: generalization gap and sharp minima, September. http://arxiv.org/abs/1609.04836.

Kreiss, S., L. Bertoni, and A. Alahi. 2019. PifPaf: Composite fields for human pose estimation. March. http://arxiv.org/abs/1903.06593.

Li, S., T. Peng, X. Chi, F. Yan, and Y. Liu. 2009. A mixed reality-based assembly verification and training platform BT - virtual and mixed reality . In ed. R. Shumaker, Virtual and Mixed Reality, 576–85. Berlin, Heidelberg: Springer Berlin Heidelberg.

Li, Y., H. Zihang, Y. Xiang, H. Zuguo, and K. Han. 2019. Spatial temporal graph convolutional networks for skeleton-based dynamic hand gesture recognition. *EURASIP Journal on Image and Video Processing* 2019 (1):78. doi:10.1186/s13640-019-0476-x.

Lin, T.-Y., P. Goyal, R. Girshick, H. Kaiming, and D. Piotr. 2017. Focal loss for dense object detection. August. http://arxiv.org/abs/1708.02002.

Liu, J., G. Wang, H. Ping, L.-Y. Duan, and A. C. Kot. 2017. Global context-aware attention LSTM networks for 3D action recognition. 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 3671–80. IEEE. 10.1109/CVPR.2017.391.

Liu, L., Y.-C. Lin, and J. Reid. 2019. Improving the performance of the LSTM and HMM model via hybridization. July. http://arxiv.org/abs/1907.04670.

Manns, M., M. Otto, and M. Mauer. 2016. Measuring motion capture data quality for data driven human motion synthesis. *Procedia CIRP* 41:945–50. doi:10.1016/j.procir.2015.12.068.

Masood, T., and J. Egger. 2019. Augmented reality in support of industry 4.0—implementation challenges and success factors. *Robotics and Computer-Integrated Manufacturing* 58:181–95. doi:10.1016/j.rcim.2019.02.003.

Maynard, H. B., G. J. Stegemerten, and J. L. Schwab. 1948. *Methods-Time Measurement*. *Methods-Time Measurement*. New York, NY, US: McGraw-Hill.

Menolotto, M., D.-S. Komaris, S. Tedesco, B. O'Flynn, and M. Walsh. 2020. Motion capture technology in industrial applications: A systematic review. *Sensors* 20 (19):5687. doi:10.3390/s20195687.

Min, Y., Y. Zhang, X. Chai, and X. Chen. 2020. An efficient pointLSTM for point clouds based gesture recognition. 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Seattle, WA, USA, 5760–69. IEEE. doi:10.1109/CVPR42600.2020.00580.

Ovur, S. E., S. Hang, Q. Wen, E. De Momi, and G. Ferrigno. 2021. Novel adaptive sensor fusion methodology for hand pose estimation with multileap motion. *IEEE Transactions on Instrumentation and Measurement* 70:1–8. doi:10.1109/TIM.2021.3063752.

Paszke, A., S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Köpf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J., Chintala, S. 2019. PyTorch: An imperative style, high-performance deep learning library. Advances in Neural Information Processing Systems , edited by H. Wallach, H. Larochelle, and A. Beygelzimer, F d\textquotesingle Alché\textquotesingle AlchéBuc, E Fox, and R Garnett. Vol. 32. Curran Associates, Inc., Vancouver, Canada. https://proceedings.neurips.cc/paper/2019/file/bdbca288fee7f92f2bfa9f7012727740-Paper.pdf.

Pham, V. T., Q. Qiu, A. A. P. Wai, and J. Biswas. 2007. Application of ultrasonic sensors in a smart environment. *Pervasive and Mobile Computing* 3 (2):180–207. doi:10.1016/j.pmcj.2006.07.002.

Polotski, V., Y. Beauregard, and A. Franzoni. 2019. Combining predetermined and measured assembly time techniques: Parameter estimation, regression and case study of fenestration industry. *International Journal of Production Research* 57 (17):5499–519. doi:10.1080/00207543.2018.1530469.

Prabhu, V. A., B. Song, J. Thrower, A. Tiwari, and P. Webb. 2016. Digitisation of a moving assembly operation using multiple depth imaging sensors. *The International Journal of Advanced Manufacturing Technology* 85 (1–4):163–84. doi:10.1007/s00170-015-7883-7.

Puthenveetil, S. C., C. P. Daphalapurkar, W. Zhu, M. C. Leu, X. F. Liu, J. K. Gilpin-Mcminn, and S. D. Snodgrass. 2015. Computer-automated ergonomic analysis based on motion capture and assembly simulation. *Virtual Reality* 19 (2):119–28. doi:10.1007/s10055-015-0261-9.

Qi, W., S. Hang, and A. Aliverti. 2020. A smartphone-based adaptive recognition and real-time monitoring system for human activities. *IEEE Transactions on Human-Machine Systems* 50 (5):414–23. doi:10.1109/THMS.2020.2984181.

Rabiner, L. R. 1989. A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of the IEEE* 77 (2):257–86. doi:10.1109/5.18626.

Roetenberg, D., H. Luinge, and P. J. Slycke. 2009. Xsens MVN: Full 6DOF human motion tracking using miniature inertial sensors. In.

Rozemberczki, B., P. Scherer, H. Yixuan, G. Panagopoulos, A. Riedel, M. Astefanoaei, O. Kiss, Ferenc, B., Lopez, G., Collignon, N., Sarkar, R. 2021. PyTorch geometric temporal: Spatiotemporal signal processing with neural machine learning models CIKM '21: Proceedings of the 30th ACM International Conference on Information & Knowledge ManagementGold Coast, Queensland, Australia. April. doi:10.1145/3459637.3482014.

Rude, D. J., S. Adams, and P. A. Beling. 2018. Task recognition from joint tracking data in an operational manufacturing cell. *Journal of Intelligent Manufacturing* 29 (6):1203–17. doi:10.1007/s10845-015-1168-8.

Salmi, A., P. David, E. Blanco, and J. D. Summers. 2016. A review of cost estimation models for determining assembly automation level. *Computers & Industrial Engineering* 98:246–59. doi:10.1016/j.cie.2016.06.007.

Shafaei, A., and J. J. Little. 2016. "Real-time human motion capture with multiple depth cameras.2016 13th conference on computer and Robot Vision (CRV), Victoria, BC, Canada, 24–31. IEEE. doi:10.1109/CRV.2016.25.

Shi, L., Y. Zhang, J. Cheng, and L. Hanqing 2018. Two-stream adaptive graph convolutional networks for skeleton-based action recognition. May. http://arxiv.org/abs/1805.07694.

Stump, B., and F. Badurdeen. 2012. Integrating lean and other strategies for mass customization manufacturing: A case study. *Journal of Intelligent Manufacturing* 23 (1):109–24. doi:10.1007/s10845-009-0289-3.

Su, H., S. E. Ovur, X. Zhou, Q. Wen, G. Ferrigno, and E. De Momi. 2020. Depth vision guided hand gesture recognition using electromyographic signals. *Advanced Robotics* 34 (15):985–97. doi:10.1080/01691864.2020.1713886.

Tadayon, M., and G. Pottie. 2020. Comparative analysis of the hidden Markov model and LSTM: A simulative approach. August. http://arxiv.org/abs/2008.03825.

Tran, D., H. Wang, L. Torresani, J. Ray, Y. LeCun, and M. Paluri. 2018. A closer look at spatiotemporal convolutions for action recognition.*2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, Salt Lake City, UT, USA, 6450–59. IEEE. doi:10.1109/CVPR.2018.00675.

Wang, X., S. K. Ong, and A. Y. C. Nee. 2016. Real-virtual components interaction for assembly simulation and planning. *Robotics and Computer-Integrated Manufacturing* 41 (October):102–14. doi:10.1016/j.rcim.2016.03.005.

Yan, S., Y. Xiong, and D. Lin. 2018. Spatial temporal graph convolutional networks for skeleton-based action recognition, January. http://arxiv.org/abs/1801.07455.

Yin, X.-Q., W. Tao, Y. Feng, H.-W. Yang, H. Qiao-Zhi, and H. Zhao. 2019. A robust and accurate breakpoint detection method for line-structured laser scanner. *Optics & Laser Technology* 118:52–61. doi:10.1016/j.optlastec.2019.03.037.

Zandin, K. B. 2020. *MOST® work measurement systems*. Edited by T. M. Schmidt, Fourth edition Boca Raton, FL: CRC Press. 2021. | On title page the registered trademark follows "MOST" in the title.: CRC Press. doi:10.1201/9780429326424.

Zhang, F., V. Bazarevsky, A. Vakunov, A. Tkachenka, G. Sung, C.-L. Chang, and M. Grundmann. 2020. MediaPipe Hands: On-device real-time hand tracking, June. http://arxiv.org/abs/2006.10214.

Zhang, P., C. Lan, J. Xing, W. Zeng, J. Xue, and N. Zheng. 2017. View adaptive recurrent neural networks for high performance human action recognition from skeleton data.2017 IEEE International Conference on Computer Vision (ICCV), Venice, Italy, 2136–45. IEEE. doi:10.1109/ICCV.2017.233.

Zhong, R. Y., Q. Y. Dai, T. Qu, G. J. Hu, and G. Q. Huang. 2013. RFID-enabled real-time manufacturing execution system for mass-customization production. *Robotics and Computer-Integrated Manufacturing* 29 (2):283–92. doi:10.1016/j.rcim.2012.08.001.