Real-Time Multi-Modal Human–Robot Collaboration Using Gestures and Speech

Haodong Chen[1]
Department of Mechanical and Aerospace Engineering,
Missouri University of Science and Technology,
Rolla, MO 65409
e-mail: h.chen@mst.edu

Ming C. Leu
Department of Mechanical and Aerospace Engineering,
Missouri University of Science and Technology,
Rolla, MO 65409
e-mail: mleu@mst.edu

Zhaozheng Yin
Department of Biomedical Informatics &
Department of Computer Science,
Stony Brook University,
Stony Brook, NY 11794
e-mail: zyin@cs.stonybrook.edu

As artificial intelligence and industrial automation are developing, human–robot collaboration (HRC) with advanced interaction capabilities has become an increasingly significant area of research. In this paper, we design and develop a real-time, multi-model HRC system using speech and gestures. A set of 16 dynamic gestures is designed for communication from a human to an industrial robot. A data set of dynamic gestures is designed and constructed, and it will be shared with the community. A convolutional neural network is developed to recognize the dynamic gestures in real time using the motion history image and deep learning methods. An improved open-source speech recognizer is used for real-time speech recognition of the human worker. An integration strategy is proposed to integrate the gesture and speech recognition results, and a software interface is designed for system visualization. A multi-threading architecture is constructed for simultaneously operating multiple tasks, including gesture and speech data collection and recognition, data integration, robot control, and software interface operation. The various methods and algorithms are integrated to develop the HRC system, with a platform constructed to demonstrate the system performance. The experimental results validate the feasibility and effectiveness of the proposed algorithms and the HRC system. [DOI: 10.1115/1.4054297]

Keywords: human–robot collaboration, gesture recognition, speech recognition, real-time, multi-modal, multiple threads

1 Introduction

During the era of modern manufacturing, human–robot collaboration (HRC) has emerged as one of the next-generation technologies in automated systems because of its ability to integrate the flexibility of humans and the productivity of robots. An HRC system should ideally function like a human–human collaboration using multiple communication channels, and the system can monitor, interact with, control, and respond to the physical environment in real time. Despite the existence of various existing communication methods and object recognition algorithms, it remains challenging for a robot to realize natural, precise, and real-time identification and respond to human actions in a manufacturing environment, so as to maintain the HRC's efficient performance. This is because factory-based human–robot communication methods and real-time multi-modal collaboration systems are still lacking. Without the ability to operate in real time, it is difficult for an HRC system to conduct realistic applications and perform tasks with concurrent recognition of human actions and decision-making of robot responses [1–6]. In order to mitigate the communication issues on the factory floor, we have designed a real-time, multi-modal HRC system, which includes a series of HRC communication protocols that combine the designed arm gestures and natural speech for humans to communicate with robots in real time on the factory floor.

1.1 Related Work

1.1.1 Human–Robot Communication. Researchers have been using a wide variety of methods, both verbal and non-verbal, to facilitate communication between humans and robots. Through verbal communication technology, speech signals can be understood and expressed fairly clearly. Zinchenko et al. [7] designed a speech recognition interface for surgical robot control. Bingol and Aydogmus [8] proposed an interactive speech control with an industrial robot using speech recognition software in the Turkish language. Two studies were conducted by Kuhn et al. [9] for the HRC to explore the intentional influence of speech metaphors on human workers. Unlike the verbal communication, the non-verbal communication usually uses body language and biological signals to convey information, such as gestures, electroencephalogram (EEG), and electromyography (EMG). In these communication methods, gestures are widely used for the human–robot communication. Coupeté et al. [10] designed a user-adaptive method which increased the accuracy rate of the dynamic gesture recognition algorithm by around 3.5% by incorporating less than 1% of the previous database. Unhelkar et al. [11] designed the prediction of motion with a look-ahead time of up to 16 s. Pinto et al. [12] compared different networks in static hand gesture recognition. Li et al. [13] constructed an attention-based model for human action recognition, which could effectively capture the long-range and long-distance dependencies in actions. Apart from gestures, wearable sensors are used for specific communication systems. Tao et al. [14] proposed a method for worker activity recognition on a factory floor using inertial measurement unit and EMG signals. Treussart et al. [15] designed an upper-limb exoskeleton robot using a wireless EMG signal to help carry unknown loads without the use of force sensors. Although wearable sensors are widely adopted and proved to be useful in HRC systems, it needs to be pointed out that they often limit the movements of human workers due to their attachments. Regarding the gesture and speech communication, the lack of associated gesture vocabulary for intuitive HRC, the heavy reliance on features of the interaction objects' environments (such as the skin color model), the effects of lighting, the high computing power requirement for speech processing with hosted services, and the fluctuating manufacturing background noise limit the flexibility and robustness of their applications. This paper focuses on gesture and speech

[1]Corresponding author.
Manuscript received December 8, 2021; final manuscript received April 6, 2022; published online June 10, 2022. Assoc. Editor: Chinedum Okwudire.

command only as they are the two most common modalities. Although they are rather mature technologies, there is still important research to be done to improve either of the two modalities as well as their integration.

1.1.2 Multi-Modal Sensors. Recent applications of hybrid control systems combine different sensors into a multi-modal communication, which provides higher accuracy and robustness than the use of a single type of sensors. Ajoudani et al. [16] combined EMG and EEG sensors to extract motion features of human workers in a collaborative setup. A human–robot interaction system was developed by Yongda et al. [17], which combined finger movements and speech for training robots to move along a pre-designed path. A multi-modal-based network architecture to improve the fused decision-making of human–robot interaction was devised by Lin et al. [18]. It used various sensors, including EEG, blood pressure, body temperature, and other sensors. These sensor fusions used in HRC show the robustness of multi-modal communication. However, when humans and robots interact in a shared working environment, the lack of intuitive and natural multi-modal communication between humans and robots limits the transferability of an HRC system between different tasks, which also undermines the symbiotic relationship between humans and robots to facilitate environmental and task changes during collaboration [19].

1.1.3 Real-Time Recognition in Human-Robot Collaboration. To collaborate with humans on a factory floor in a seamless manner, robots in HRC systems should recognize human activities in a real-time modality. Yu et al. [20] proposed a real-time recognition of human object interaction using depth sensors, and it achieved an accuracy of 73.80%. Shinde et al. [21] proposed a you only look once-based human action recognition and localization model with an accuracy of 88.37%, which used real-time video streams as inputs. Sun et al. [22] designed a locally aggregated kinematic-guided skeleton-based method to recognize human gestures in real time, and it achieved an accuracy of around 95%. Yu et al. [23] proposed a discriminative deep learning model to recognize human actions based on real-time feature fusion and temporal attention, and the model achieved 96.40% accuracy on a real data set. Although these studies show that HRC systems with real-time recognition modality can powerfully perform many dynamic tasks, the single frame or channel-based recognition with limited features affects recognition performance of human actions, and the real-time recognition with high accuracy of multiple communication modalities in an HRC system remains a challenging task. This is because the calculation of deep networks for multiple human behavior recognitions, as well as the requirement for integrating and coordinating multiple concurrent processes may impact the system's efficiency.

1.2 Contributions of This Article. This paper presents the development of a real-time, multi-modal HRC system that can detect and recognize a human worker's dynamic gesture and natural speech commands, thereby allowing the worker to collaborate with a designated robot based on the gesture and speech inputs. An overview of the proposed HRC system is depicted in Fig. 1. This system operates several parallel threads simultaneously in real time, including input data capturing, gesture/speech recognition, multi-modal integration, robot control, and interface operation.

The main contributions of this paper are as follows:

- Overall, we propose a real-time, multi-modal HRC system which combines our designed dynamic gestures and natural speech commands. This system can recognize gesture and speech signals in real time and conduct collaboration between a human and a robot in real time.
- Regarding the communication of the HRC system, a data set of dynamic gestures is designed and constructed, which will be shared with the community via our GitHub website. A gesture recognizer is built using a multi-view data set and a convolutional neural network (CNN). Robust speech recognition is achieved through improvements to an online open-source speech recognizer.
- Regarding the operation of the HRC system, a real-time motion history image (MHI) generation method is designed to extract dynamic features of gestures to represent motion information in real time. A multi-threading architecture is built to conduct the seven threads shown in Fig. 1 simultaneously in real time.

1.3 Organization of This Article. The remainder of this paper is organized as follows. Section 2 presents the designed dynamic gestures, real-time MHI for feature extraction, and a deep learning model for gesture recognition. Section 3 describes our method of speech recognition. Section 4 describes the integration of the gesture and speech recognition results, and the construction of a multi-threading model for real-time system operation. Section 5 presents evaluation and demonstration with the HRC system and details the system performance through experiments between a human worker and a robotic arm. Section 6 presents the conclusion.

2 Design and Recognition of Dynamic Gestures

This section describes the design of dynamic gestures, the construction of a gesture data set, and the design of a real-time gesture recognition model. A set of dynamic gestures is designed and an associated data set is constructed in Sec. 2.1. The MHI method is used to extract dynamic gesture features. The video-based MHI is applied in Sec. 2.2 to extract action features from gesture videos, allowing for the construction of a training data set for the recognition model. Then, a real-time MHI method is proposed in Sec. 2.3 to realize real-time gesture recognition.

2.1 Gesture Design and Data Collection

2.1.1 Design of Dynamic Gestures. Dynamic gestures are generated by moving the two upper limbs with both the upper and lower arms but without movement of fingers. Different from static gestures which mainly rely on the shapes and flexure angles of limbs, dynamic gestures rely on limb trajectories, orientations, and motion speeds saved in the temporal sequence, as well as shapes and limb angles. These features allow dynamic gestures to have a higher number of configuration options and contain more information than static gestures [24].

The gestures used in HRC systems should be simple to sign, socially acceptable, and minimize the cognitive workload. McNeill [25] proposed a classification scheme of gestures with four categories: iconic (gestures present images of concrete entities and/or actions), metaphoric (gestures are not limited to depictions of concrete events), deictic (the prototypical deictic gesture is an extended "index" finger, but almost any extensible body part or held object can be used), and beats (using hands to generate time

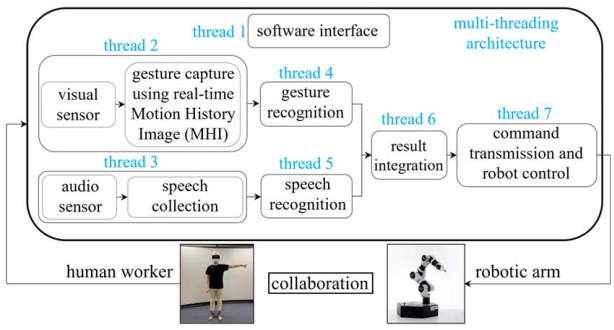

Fig. 1 System overview

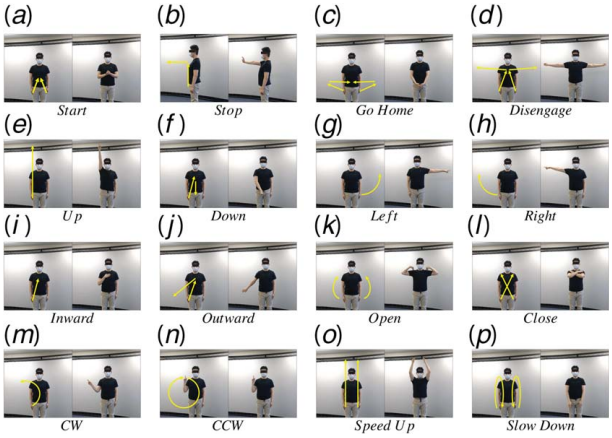

Fig. 2 Illustration of the designed 16 dynamic gestures (CW, clockwise; CCW, counterclockwise): (*a*) gesture 1, (*b*) gesture 2, (*c*) gesture 3, (*d*) gesture 4, (*e*) gesture 5, (*f*) gesture 6, (*g*) gesture 7, (*h*) gesture 8, (*i*) gesture 9, (*j*) gesture 10, (*k*) gesture 11, (*l*) gesture 12, (*m*) gesture 13, (*n*) gesture 14, (*o*) gesture 15, and (*p*) gesture 16

beats). The metaphoric gestures put abstract ideas into a more literal and concrete form, which is not straight-forward. The beats gestures are only used to keep the rhythm of speech, and they usually do not convey any semantic content [26]. Therefore, we create 16 dynamic gestures in iconic and deictic categories for the HRC system, as shown in Fig. 2.

The gestures in Fig. 2 are mainly designed for robot calibration and operation in a HRC system. Calibration gestures allow the robot to perform routine procedures, including gestures 1–4 (*Start*, *Stop*, *Go Home*, and *Disengage*). The *Start* gesture commands a robot to calibrate the kinematic structure, which is the initialization process of identifying the initial values of parameters in the robot's kinematic structure, and move the robot into a working position for upcoming tasks. The *Stop* gesture commands a robot to stop its movement and keep it in its current pose. The *Go Home* gesture commands the robot to go to its home position, such as keeping the robotic arm straight up. When the robot gets stuck because it is commanded to move outside a joint limit, the *Disengage* gesture command can disengage the robot and make it mandatory to keep the joints away from the end of stroke and move back to the standard workspace. In addition to calibration gestures, an operation gesture (gestures 5–16) instructs the robot to move its end-effector in a different direction, change its speed of motion, open or close the gripper, conduct clockwise (CW) or counterclockwise (CCW) rotation of the end-effector, etc. The instructions for performing the proposed gestures are given in Appendix A.

2.1.2 Multi-view Data Collection. As shown in Fig. 3, two red–green–blue (RGB) cameras in Fig. 3(*a*) are used to collect gesture features from multiple views. Each human subject keeps staying in the 3D cubic space of both camera views while turning

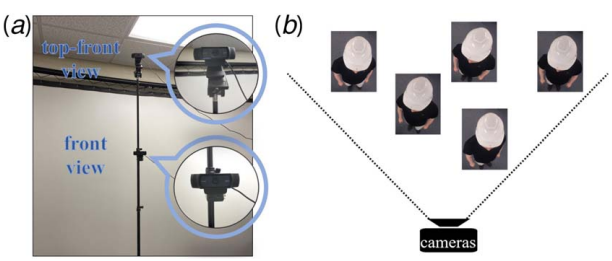

Fig. 3 Illustration of the multi-view dynamic gesture collection: (*a*) Two camera views and (*b*) multi-view gestures

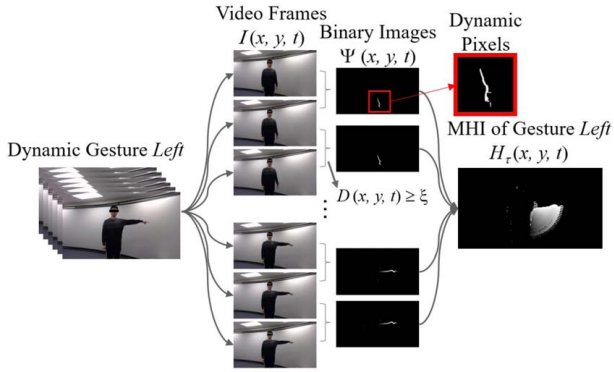

Fig. 4 Generation of a video-based MHI

around arbitrarily and changing locations in the gesture data collection (Fig. 3(*b*)). The gesture data set includes the 16 dynamic gestures of eight human subjects. In particular, the data set of seven unilateral gestures (*Stop*, *Up*, *Down*, *Inward*, *Outward*, *Clockwise (CW)*, *Counterclockwise* (*CCW*)) include gestures performed by the left or right upper limb. Image augmentation is applied through brightness variation, horizontal/vertical shifts, zooms, and perspective transformations.

2.2 Feature Extraction and Data Set Construction Using Video-Based MHI. In order to extract the features of the designed dynamic gestures into static images, the motion history image (MHI) approach is used because it can show the cumulative object motion with a gradient trail. This approach is a view-based template approach that records the temporal history of a movement and converts it into a static scalar-valued image, i.e., the MHI, where more recently moving pixels are brighter and vice versa [27]. As shown in Fig. 4, a video-based MHI $H_\tau(x, y, t)$ is generated from binary images of the input video's sequential frames using the following formula:

$$H_\tau(x, y, t) = \begin{cases} \tau & \text{if } \Psi(x, y, t) = 1 \\ \max(0,\ H_\tau(x, y, t-1) - \delta) & \text{otherwise} \end{cases} \quad (1)$$

where x and y are pixel coordinates of the image and t is the time. $\Psi(x, y, t)$ denotes the binary image which saves the movement of an object in the current video frame, which is called for each new video frame analyzed in the sequence, where white pixels have a value of 255 and black pixels have a value of 0. The duration τ denotes the temporal extent of a movement (e.g., in terms of frames), which is set to the same value as the number of frames in the video clips. This is because a τ smaller than the number of frames decays earlier motion information in an MHI, whereas a considerable value of τ obscures brightness changes (pixel value changes) in the MHI. The decay parameter δ denotes decrease in previous pixel value when a new frame is given. If no new motion covers some specific pixels that were covered by an earlier motion while loading the frames, the values of these pixels will be reduced by δ [28]. The decay parameter δ is usually set to 1 [29].

In Fig. 4, frame subtraction is conducted to obtain the binary image. If the difference $D(x, y, t)$ between two adjacent frames is greater than the threshold ξ, a binary image $\Psi(x, y, t)$ is generated as follows:

$$\Psi(x, y, t) = \begin{cases} 1 & \text{if } D(x, y, t) \geq \xi \\ 0 & \text{otherwise} \end{cases} \quad (2)$$

where $\Psi(x, y, t)$ denotes a binary image at the tth frame, and ξ is a threshold removing the background noise from MHIs. The value of ξ is set to 10 based on the validation experiments described later in Sec. 5.1.1 [30]. The frame difference $D(x, y, t)$ is defined as

$$D(x, y, t) = |\, I(x, y, t) - I(x, y, t - \triangle)\,| \quad (3)$$

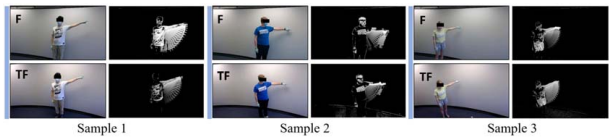

Fig. 5 Illustration of the MHIs of the gesture *Left* in two different views (F, front camera view; TF, top-front camera view)

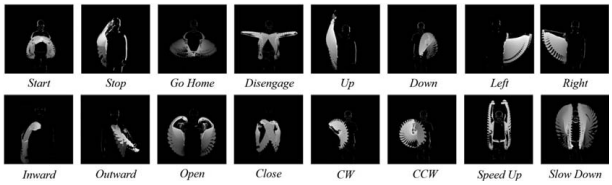

Fig. 6 Illustration of the 16 dynamic gesture MHIs

where $I(x, y, t)$ represents the intensity value of pixel location with the coordinates (x, y) at the tth frame of the image sequence and the intensity value range is [0 255]. $\triangle$ denotes the temporal difference between two pixels in the same location, which is set to 1 to take all frames into account [27].

Figure 5 shows consistent features in MHIs of the gesture *Left* in different views, i.e., front (F) and top-front (TF) views, which shows the feasibility of multi-view data collection for dynamic gestures in Sec. 2.1. After setting up all parameters for MHI generation, the MHIs for the 16 gestures are obtained, as illustrated in Fig. 6, where these MHIs successfully exhibit appearance differences for different dynamic gestures.

2.3 Real-Time Gesture Recognition. To enable real-time system operations in the feature extraction and recognition of dynamic gestures, MHIs should be generated in real time from input camera frames. Therefore, a real-time MHI method is proposed in this section.

2.3.1 Generation of Real-Time Motion History Image. According to Eqs. (1)–(3), there are three key parameters in the generation of a video-based MHI, i.e., the duration τ, decay parameter δ, and threshold ξ. The primary goal of this section is to generate the MHI in real time using continuous input frames through representing the decay parameter δ using the duration τ during the MHI generation. Since the real-time MHI method takes each new input frame into account, the difference $D(x, y, t)$ between adjacent frames and the threshold ξ are removed from the real-time MHI calculation, which means that each new input frame triggers a new binary image $\Psi(x, y, t)$. The process of the real-time MHI generation is illustrated in Fig. 7, in which the camera continuously captures input intensity value images $I(x, y, t)$ of input frames, and each new recorded input frame triggers a new binary image $\Psi(x, y, t)$ by subtracting the intensity values of each pair of adjacent frame images. Next, the program continuously combines the new binary image $\Psi(x, y, t)$ into the constantly updating real-time MHI $H_\tau(x, y, t)$. In the meantime, the value of all previous pixels in the updating MHI is reduced by δ until it equals to zero. The value of a pixel in an eight-bit grayscale image ranges from 0 (black) to 255 (white) [28].

Since each new input frame triggers a new binary image, the number of frames, i.e., the duration τ, needs to be determined. The value of the duration τ is determined by the time it takes to perform a gesture. The length of all the gesture videos in the data set are $\leq$75 frames, where the frame rate in this system is 30 frames per second (fps). Therefore, the duration τ is set to 75 frames to record all gestures completely.

The initial value of white dynamic pixels in binary images is 255, and these white pixels turn black after $\tau = 75$ times decay to maximize the brightness difference between different binary images in an MHI, i.e., the motion history features. Since each new input frame updates the real-time MHI by adding a new binary image $\Psi(x, y, t)$ and reducing the values of previous pixels by δ, the decay parameter δ can be denoted by the duration τ through $255/\tau \approx 3.40$.

The real-time MHI generator enables the real-time and continuous MHI generation of current frames. Furthermore, the fact that a new input frame triggers a new real-time MHI means that every

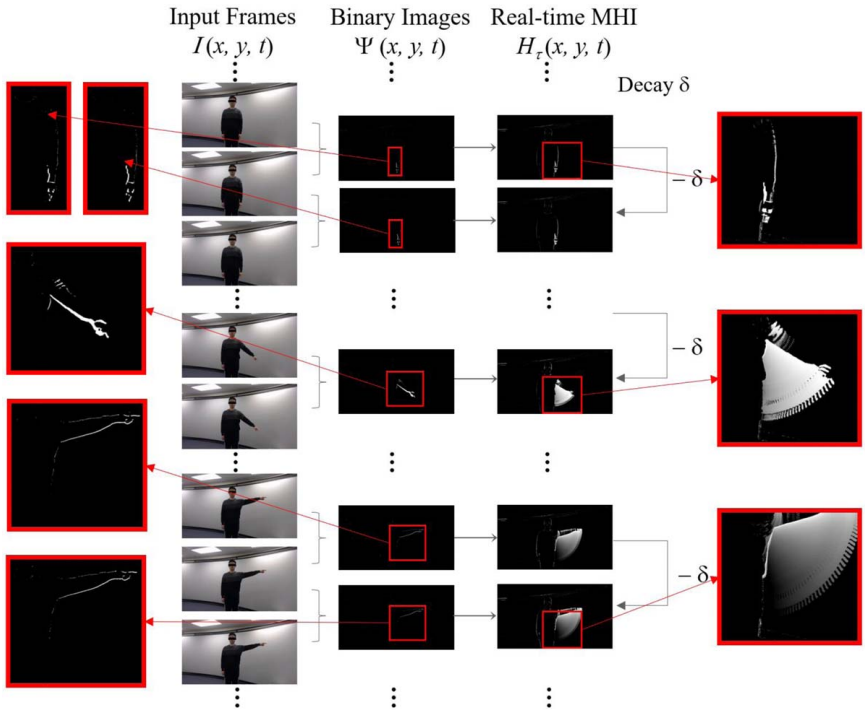

Fig. 7 Generation of real-time MHIs

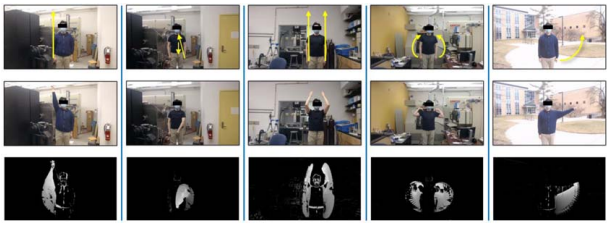

Fig. 8 MHI generation in five different environments

input frame is recorded in the real-time MHI, i.e., every movement (more than one frame) of the human worker can update more than one similar MHI, which is critical in providing high error tolerance and recognition accuracy for later real-time gesture recognition. The pseudo-code for the real-time MHI generation is given in Algorithm 1.

Algorithm 1 Real-time MHI generation

Input: I_p /*previous frame image*/, I_c /*current frame image*/
Output: MHI_f /*frame-based real-time MHI*/
function MHI_f I_p, I_c
  $\text{MHI}_f = O$; /*Initialization of the MHI_f*/
  for each I_c
    $I_d = |I_c - I_p|$;
    Values of different pixels between I_c and I_p in $I_d = 255$;
    $I_b = I_d$; /*I_b : binary image*/
    $\text{MHI}_f = \max(0, \text{MHI}_f - 3.40)$; /*$\delta = 3.40$*/
    $\text{MHI}_f = \text{MHI}_f + I_b$;
    $I_p = I_c$;
  end for
end function

To demonstrate the robustness of the MHI generation, we collected gesture data under five different environments (Fig. 8). According to the MHI results in Fig. 8, only dynamic gesture pixels are extracted, and the stationary environments have no effect on MHI generation. In future work, we will combine the skeleton model and the MHI method to process gestures in dynamically changing environments.

2.3.2 Gesture Recognition. A deep learning model is built using convolutional neural networks (CNNs) to recognize the designed dynamic gestures. Figure 9 depicts the overall architecture of the CNN model. The input images are MHIs that have been resized to 32 × 32 (width × height). The model consists of two convolution layers, and each layer is followed by a max-pooling layer. The sizes of the convolution kernels, feature maps at each layer, and the pooling operators are shown in Fig. 9. The dropout occurs after the second pooling layer, which randomly drops units from the neural network during training and can effectively avoid over-fitting issues [31]. Following the second pooling layer, a 5 × 5 × 40 feature map is obtained. It is flattened into a 400 feature vector, and then followed by a 100-neuron fully connected layer.

The output of this network is a 16-dimensional score vector layer transformed by the softmax function. The softmax function can carry out the normalization process by limiting the function output to the interval [0, 1] and making the components add up to 1. A very confident event is denoted by 1, while an impossible event is denoted by 0 [32]. This process allows for the computation of class-membership probabilities for the 16 gestures, and the larger output components correspond to larger confidence [33]. The softmax function is calculated as follows:

$$P(x_i) = \frac{e^{x_i}}{\sum_{k=1}^{16} e^{x_k}} \quad \text{for } i = 1, \ldots, 16 \tag{4}$$

where $P(x_i)$ denotes the confidence values of the sample x belonging to the class i, x_k denotes the weighted input of the softmax layer, and 16 is the number of gestures.

The distribution with a single spike yields a more confident result, and the spike label is therefore the final recognized gesture label. When the result is a uniform distribution without spikes, it indicates that the input gesture does not belong to any of the 16 gestures, such as walking in front of the camera. In our case, if the confidence value is greater than 90%, we assume that the associated gesture has been successfully recognized. Due to the fact that the real-time MHI generator can create a new MHI for each input frame, a series of continuous MHIs for a single gesture can be

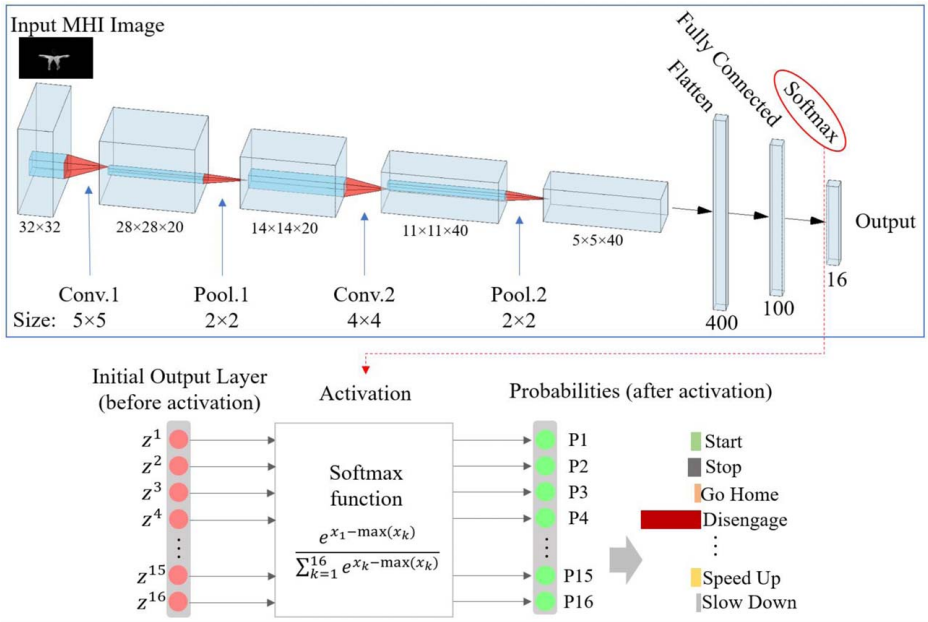

Fig. 9 Overview of the CNN architecture for the recognition of the designed dynamic gestures (the terms "Conv." and "Pool." refer to the convolution and pooling operations, respectively)

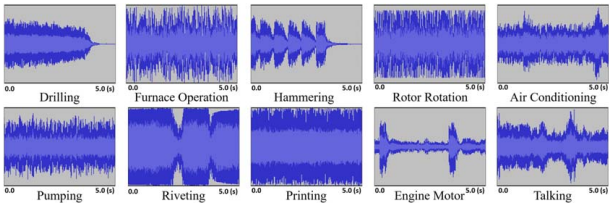

Fig. 10 Waveform samples of the ten different background noises

obtained. While the real-time recognition discerns each of these MHIs individually, the final result is the most frequently occurring label of high-confidence recognition.

3 Recognition of Speech Commands

To improve the robustness and flexibility of the proposed HRC system, speech data from human workers can be combined with the aforementioned real-time gestures to generate a more reliable recognition result. The Google open-source speech recognizer[2] is used to recognize speech data because it is capable of converting more than 120 different languages from speech to text. The log-Mel filterbank is used in the Google open-source speech recognizer to reduce the dimension of all speech log-spectral magnitude vectors, resulting in improved speech recognition performance for a deep neural network with complex parameters [34,35]. To evaluate the effectiveness of this speech recognizer in a factory floor environment, experiments are conducted in which the speech is recognized in ten different background noises. The waveforms of the 10 background noises used in our speech experiments are depicted in Fig. 10. These audio files are extracted from video and audio files on the internet and recorded at a level of 60–70 decibels (dB). Volume levels greater than 70 dB are not included in tests because normal human speaking and communication are around 60 dB, and background noise levels greater than 70 dB can submerge the normal speech tone [36].

Figure 11 displays waveforms of our speech commands in a quiet environment. Figure 12 compares the waveforms of the *Start* command without background noise, a drilling noise without speech commands, and the *Start* command in the drilling noise environment.

To reduce the influence of background noise on speech recognition, we use the spectral subtraction method, which estimates the speech signal spectrum and the noise spectrum, and subtracts the noise spectrum from the speech signal, resulting in an improved speech signal [37,38]. The first 0.5 s of our speech data set contains only the background noise. Therefore, we use the first 0.5 s of an audio file to represent the background noise spectrum, and the rest of the audio file as the speech signal spectrum. The spectral subtraction technique then deducts the background noise spectrum from the speech signal spectrum. To achieve real-time denoising, the microphone is activated 0.5 s before the camera is turned on, i.e., prior to the worker giving gestures and speech, and the additional 0.5 s of audio is recorded as the background noise.

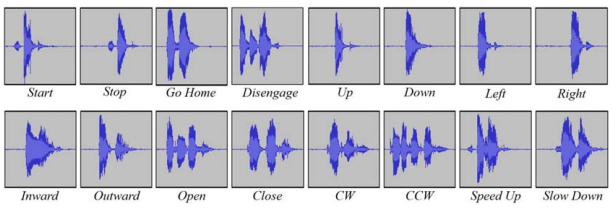

Fig. 11 Waveform samples of the 16 commands

[2]https://pypi.org/project/SpeechRecognition/

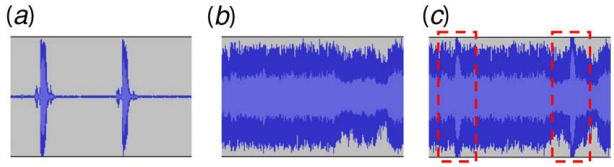

Fig. 12 Waveform samples of the *Start* command in the drilling noise environment. (*a*) *Start* command, (*b*) drilling noise, and (*c*) *Start* command in drilling noise environment.

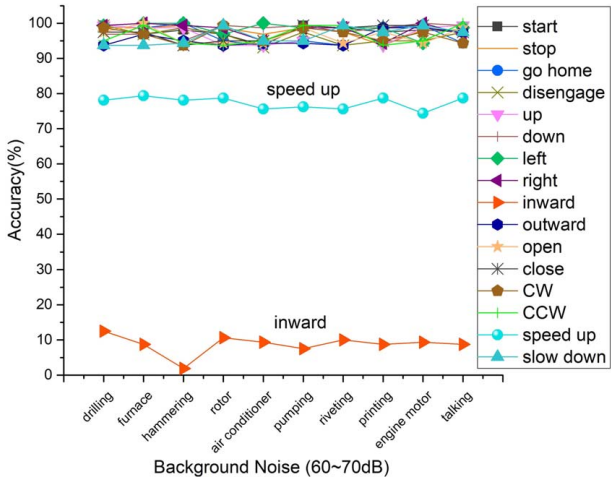

Fig. 13 Performance (%) of the speech recognizer on our speech data set

The audio data set contains 16 different speech commands of two native English-speaking subjects (one male and one female) under ten different background noises. The data set contains about 1600 audio files. To evaluate the performance of the recognition model, we define the speech recognition accuracy of a particular class C as the ratio of the number of correct recognitions to the total number of samples within C. The recognition results are shown in Fig. 13, in which most speech commands have >90% recognition accuracy. However, the command *speed up* has a recognition accuracy of around 75%, while the command *inward* has a recognition accuracy of less than 15%.

Table 1 displays low-accuracy transcriptions of the above two commands. For each sample, the Google open-source speech recognizer gives several possible transcripts during recognition of an input command and calculates the confidence of all possible transcripts. Confidence is a value between 0% and 100% used to evaluate the reliability of recognition results. The transcription with the highest level of confidence outputs as the recognized result. In the first sample of the command *inward* in Table 1, it is recognized as several possible transcripts, including *uworld*, *in word*, *inward*, *keyword*, and *in world*. The *uworld* is the output with the highest confidence (69.70%). The command *speed up* in its first sample is recognized possibly as *beat up*, *speed up*, *beat-up*, *bead up*, and *the beat up*. The output *beat up* is the one with the highest confidence (94.07%). In the second sample of these two commands, even though they are correctly recognized, their confidence levels (61.21% for *inward* and 86.89% for *speed up*) are lower than those of the other commands as shown in Fig. 13 (>90%).

According to the literature on speaking habits, the reason for the low accuracy of the command *inward* is that the *in* is stressed, and there is a short pause between *in* and *ward*. For the *speed up*, the *s* is generally pronounced at a low volume [39]. These speaking habits reduce the recognition accuracy of these two speech commands. To address this issue, we use the fuzzy matching strategy [40,41] for the command "*inward*" and "*speed up*," in which all possible

Table 1 Performance (%) of the speech recognizer in recognizing the *inward* and *speed up* commands

Command	Sample	Possible transcripts	Output	Confidence of the output (0–100)
"*inward*"	1	"*uworld*," "*in word*," "*inward*," "*keyword*," "*in world*"	"*uworld*"	69.70
	2	"*inward*," "*in-word*," "*in word*," "*uworld*," "*any word*"	"*inward*"	61.21
"*speed up*"	1	"*beat up*," "*speed up*," "*beat-up*," "*bead up*," "*the beat up*"	"*beat up*"	94.07
	2	"*speed up*," "*subitup*," "*to beat up*," "*who beat up*," "*the beat up*"	"*speed up*"	86.89

Note: The recognizer outputs the transcription with the highest confidence, and the lowercase/capitalization does not affect the recognition result).

transcripts in Table 1 are set as correct recognition results in the speech database. This strategy is feasible because those possible transcripts are very distinct and generally cannot be confused with other commands in HRC scenarios on the factory floor.

We further design a real-time keyword extraction technique for the 16 speech commands, which allows the speech recognizer to extract keywords from a short sentence, such as the "*speed up*" from the sentence "*speed up the process*" and the "*inward*" from the short sentence "*go inward*." The experimental results in Sec. 5 show that the recognition accuracy of the two commands *inward* and *speed up* are increased to >90% using our designed technique.

4 System Integration

This section describes the integration strategy of gesture and speech results under several different cases. Following that, the construction of a multi-threading architecture is demonstrated for system operation on several parallel tasks.

4.1 Integration of Gesture and Speech Recognition Results. In the recognition of human commands, our system evaluates both the recognition result of gesture and/or speech commands and the corresponding recognition confidence. The recognition confidence is a value between 0% and 100% used to evaluate the reliability of recognition results, which represents the probability that the output is correct. Our system assumes that a gesture and/or speech command is successfully recognized only if the confidence level is ≥90%. If the confidence is <90%, the system shows that the result is invalid on the interface, gives a beep sound to remind the worker that a new gesture/speech needs to be given, and no robot movement is performed. Using the recognition confidence enables the system to prevent most false-positive and false-negative results from occurring and maintain the system robustness.

The integration strategy of gesture and speech recognition results is shown in Algorithm 2. The human worker can use gesture and speech in the collaboration with the robot. First, the intensity of environmental noise is assessed, and only if it is less than 70 dB, the speech result is considered in integration. Following that, the results of gesture and speech recognition, as well as the corresponding probability/confidence, are input and five cases are evaluated:

Case 1: If the gesture and speech results are identical and valid (i.e., the recognition result is within the designed 16 labels), the corresponding result label is given.
Case 2: If the gesture and speech results are different but both valid, the integration operator compares the confidence of the speech result and the probability of the gesture result. The integration result is the one with the larger value.
Case 3: If the gesture result is valid but the speech result is not, the integration yields the gesture result.
Case 4: If the speech result is valid but the gesture result is not, the speech result is selected.
Case 5: If both the gesture and speech results are invalid, the system displays "invalid" indicating that it is waiting for the next input, and the system continuously collects new gesture and speech data.

Only the gesture is considered if the environmental noise is greater than 70 dB, and cases 6 and 7 are evaluated:

Case 6: If the gesture result is valid, it is output as the result.
Case 7: If the gesture result is invalid, no result is obtained and transmitted into the robot, and the system stands by to continuously collect new gesture and speech data.

Algorithm 2 Integration of gesture and speech recognition results

Input: G_r /*gesture result*/ and associated confidence, S_r /*speech result*/ and associated confidence, C /*16 command labels*/
Output: I_r /*integrated result*/

```
function Integration G_r, S_r, C
    if Sound Intensity ≤ 70, dB; then /* high speech confidence */
        if G_r == S_r and G_r and S_r both ∈ C; then /* case1*/
            I_r = G_r (or S_r);
        else if G_r != S_r, G_r and S_r both ∈ C; then /*case2*/
            I_r = the one with larger probability/confidence value;
        else if G_r ∈ C, and S_r ∉ C; then /*case3*/
            I_r = G_r;
        else if S_r ∈ C, and G_r ∉ C; then /*case4*/
            I_r = S_r;
        else /* no valid result, i.e., G_r and S_r are not ∈ C */ /*case5*/
            I_r = [ ]; /*no output*/
        end if
    else /* low speech confidence */
        if G_r ∈ C; then /*case6*/
            I_r = G_r;
        else /*case7*/
            I_r = [ ]; /*no output and wait for the next input*/
        end if
    end if
end function
```

4.2 Multi-Threading Architecture of the HRC System. To achieve high-speed operation and response, a multi-threading model is designed to manage multiple tasks simultaneously. The structure and procedure of the multi-threading model are shown in Fig. 14. Seven threads are created in this model to handle seven concurrent tasks, including interface operation, gesture and speech capturing, real-time gesture and speech recognition, result integration, and robot control. To meet the requirements of real-time system operation, all threads must operate continuously and

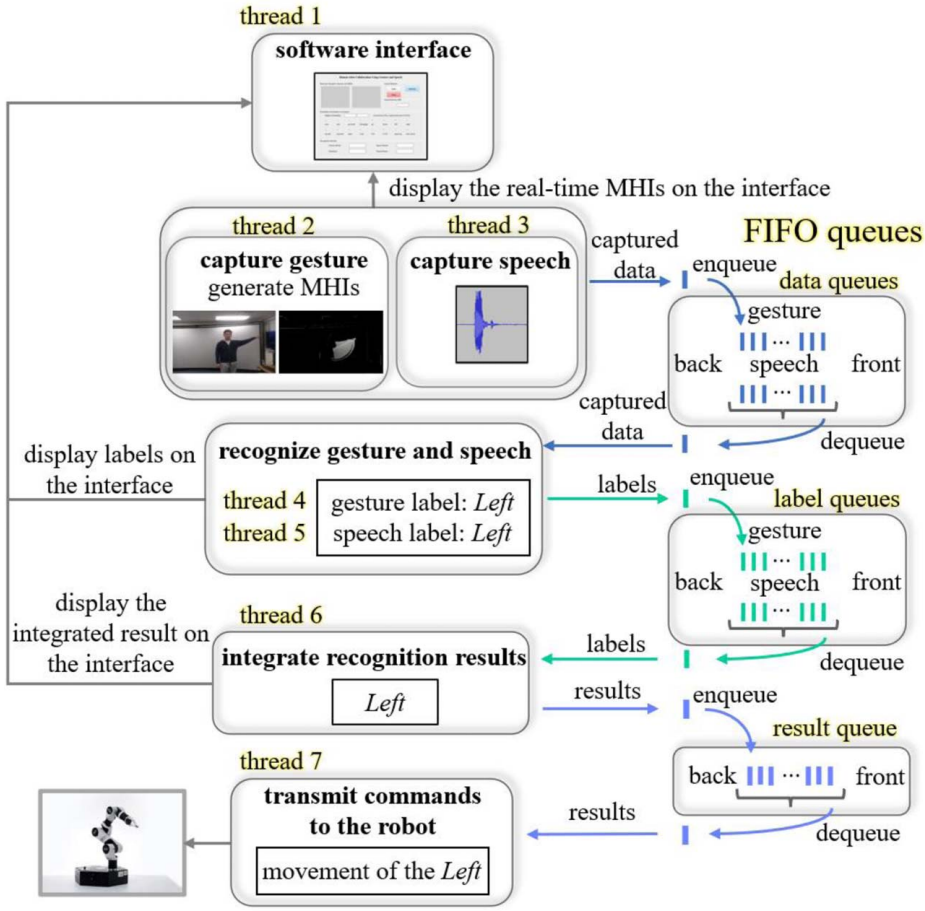

Fig. 14 Multi-threading model of the proposed real-time HRC system with an example

simultaneously, and the functions and algorithms used in real-time operation must be isolated from other functions and algorithms to minimize interference. As a result, all concurrent threads are independent. The queues shown in Fig. 14 are linear data structures that store items in a first in first out manner [42].

We use the Pycharm application programming interface (API) and python language in our system. In prediction, our hardware (24-core CPU, two Nvidia GeForce GTX 1080Ti GPUs, and 64G memory) can conduct the recognition of one gesture MHI in ~0.026 s (using camera input at 30 fps, or 0.033 s per frame), which means the video processing is real-time. One speech command is about 3 s in our system, and the Google speech recognition API can recognize it in ~0.218 s (i.e., a 3-s audio signal can be processed in 0.218 s, faster than real-time). Therefore, the CNN and the Google speech recognition API do not affect the real-time capability of our framework.

Thread 1 is constantly active when the system is running to ensure that the interface functions properly. Threads 2 and 3 read data from the RGB camera and microphone, respectively. The camera and microphone turn off after each execution time and automatically turn back on when threads 6 and 7 finish. Thread 2 and 3 collect gesture and speech data in an execution time, which is a set variable that depends on the time required for the worker to perform a command (gesture and/or speech). The real-time MHI is displayed on the interface window and saved in a gesture data queue. The speech data in an execution time is also stored in a speech data queue. Given that our data set shows that all gestures and speech take less than 2.5 s to complete, the execution time for collecting gesture and speech data (described in Sec. 4.2) is set to 2.5 s to collect complete gesture and speech information. To record background noise for speech denoising, the microphone is turned on 0.5 s before the camera is activated. Meanwhile, threads 4 and 5 begin recognizing the MHIs and speech data that popped out of the data queues. Along with labeling the recognition results, threads 4 and 5 display them on the interface and store them in label queues. After obtaining the gesture and/or speech results from the label queues, thread 6 integrates them and saves the final result in a result queue. Thread 7 continuously pops out the results from the result queue and sends them to the robotic arm to perform the associated tasks.

5 System Evaluation and Demonstration

The experimental platform is Ubuntu 16.04 on a computer equipped with 64343M memory and two NVIDIA GeForce GTX 1080 Ti graphics cards. Gestures and speech are captured using a Logitech HD Webcam C920, which consists of an RGB camera and a microphone. The following experiments are carried out: (i) evaluation of the proposed gesture recognition model, including the hold-out and leave-one-out experiments, (ii) evaluation of the improved techniques proposed for speech recognition, and (iii) demonstration of the HRC system.

5.1 Evaluation of the Proposed Gesture Recognition Model

5.1.1 Validation Experiments of the Proposed Gesture Recognition Model. Evaluation Results of Hold-Out Experiment. There are approximately 57,000 MHI samples after data augmentation, with approximately 3500 samples for each gesture class. The hold-out experiment is carried out to evaluate the performance of the deep learning model constructed in Sec. 2, in which the data set is randomly divided into a training data set (80% of the total) and a testing data set (20% of the total). The threshold ξ in Eq. (2) of Sec. 2 is determined using an eight-fold validation experiment. The training data set is divided into eight folds. Each fold is

Table 2 Performance (%) of different ξ in gesture recognition

Threshold ξ	5	10	15	20	25	30	35	40
Accuracy	92.24	98.83	96.56	95.08	95.05	93.44	91.09	91.04

Table 3 Performance (%) of the gesture recognition model in the hold-out experiment

Class	Accuracy	Precision	Recall	*F*-score
Start	99.50	97.00	94.98	95.98
Stop	99.41	95.21	95.28	95.24
Go Home	99.54	95.46	97.23	96.34
Disengage	99.64	96.33	98.01	97.16
Up	99.45	95.53	95.64	95.59
Down	98.71	88.65	90.93	89.78
Left	99.85	98.36	99.27	98.81
Right	99.65	96.37	98.14	97.25
Inward	98.82	91.00	90.03	90.51
Outward	98.83	92.53	88.80	90.63
Open	99.59	97.59	95.76	96.66
Close	99.34	94.37	95.12	94.75
CW	98.75	89.29	91.13	90.20
CCW	99.11	93.76	91.79	92.77
Speed Up	99.43	96.17	94.68	95.42
Slow Down	99.66	96.87	97.66	97.26

treated as a pseudo-test set in turn, and the other seven folds are pseudo-train sets. We calculate the average recognition accuracy of 16 gestures for each threshold ξ, which shows the ability to classify a sample gesture x from a certain class C correctly. The results are summarized in Table 2, which indicates that the accuracy threshold ξ should be set to 10 for the maximum accuracy.

Table 4 Performance (%) of the gesture recognition model in the leave-one-out experiment

"Left Out" subject	Accuracy	Precision	Recall	*F*-score
1	99.36	90.89	99.80	95.14
2	99.57	95.15	98.10	96.60
3	99.06	91.26	94.00	92.61
4	98.82	90.11	91.13	90.62
5	99.63	97.96	96.01	96.97
6	99.00	94.68	89.00	91.75
7	99.82	97.74	99.40	98.56
8	98.94	91.09	91.98	91.53

Several widely used metrics are used to assess classification performance:

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{FN} + \text{FP} + \text{TN}} \tag{5}$$

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{6}$$

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{7}$$

$$F\text{−score} = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} \ + \ \text{recall}} \tag{8}$$

where true positive (TP) refers to a sample x belonging to a particular class C that is correctly classified as C. True negative (TN) indicates that a sample x from a "*not C*" class is correctly classified as a member of the "*not C*" class. The false positive (FP) is defined as when a sample x from a "*not C*" class is incorrectly classified as

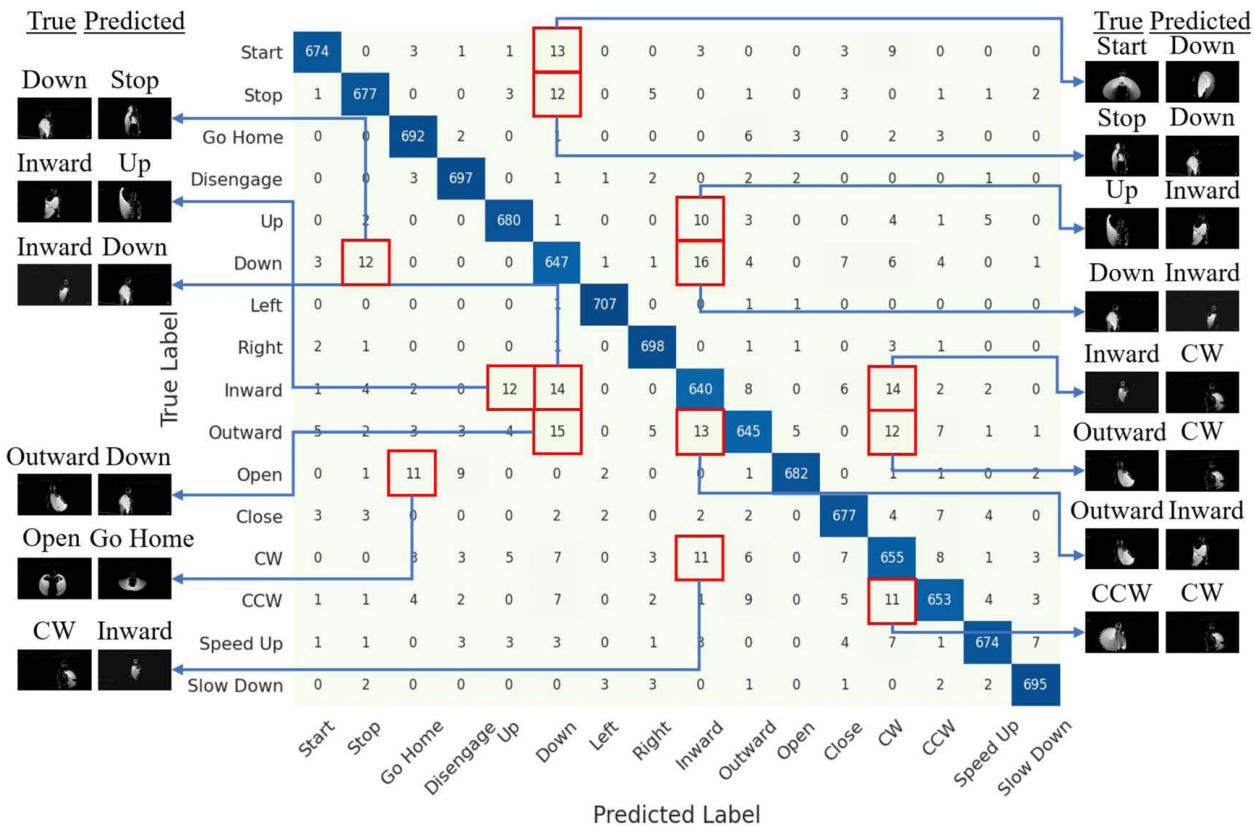

Fig. 15 Confusion matrix and the most confusing pairs of the hold-out experiment

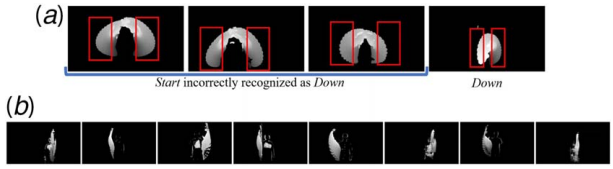

Fig. 16 Analysis of failure cases in the hold-out experiment: (*a*) some failure cases of the *Start* incorrectly recognized as *Down* and (*b*) gesture *Up* performed by different subjects

class C. The false negative (FN) describes a situation in which a sample x from class C is misclassified as belonging to "*not C*" classes. The $F-$score is the harmonic mean of the precision and recall, which ranges in the interval [0,1] [43,44]. The values of the metrics of the classification results on the testing data set are shown in Table 3. It can be observed that the recognition accuracy is >98% for all gestures, and the other metrics also yield good results (all >88%). The results demonstrate how well the trained model recognized various gestures.

Evaluation Results of Leave-One-Out Experiment. The leave-one-out experiment treats the data set of each individual human subject as a test set in turn, and the rest of them as training sets. The leave-one-out experiment results are summarized in Table 4. The metrics for the test results of each "left out" subject are the average value of the 16 class metrics of the subject. As illustrated in Table 4, the gestures of eight subjects are correctly recognized with an accuracy >98%, and the other metrics for the subjects are ≥89%. The precision, recall, and F-scores for subjects 3, 4, 6, and 8 are below 95% due to their high subject-wise variability. The results show how well the trained model recognized the gestures of a new untrained subject.

5.1.2 Failure Analysis of Validation Experiments. This subsection discusses validation experiment failures. As illustrated in Fig. 15, we compute the confusion matrix of the hold-out experiment for our multi-view gesture classification model.

The confusion matrix is an error matrix that is used to visualize classification performance. Each column of the matrix denotes instances belonging to a predicted class, while each row denotes instances belonging to a ground truth class. The most confusing pairs of gestures are marked by squares in Fig. 15, which includes *Down–Start*, *Down–Stop*, *Down–Inward*, *Down–Outward*, *Inward–Up*, *Inward–Outward*, *Open–Go Home*, *CW–Inward*, *CW–Outward*, and *CW–CCW*.

By reviewing the failure cases, we found that the high degree of similarity between the confusing pairs makes them challenging to distinguish. For example, Fig. 16(*a*) illustrates three samples of the gesture *Start* that were incorrectly recognized as *Down*. A likely reason is that they share a similar texture and shape in the square marks shown in Fig. 16(*a*). Besides, the subject-wise difference for the same gesture makes it challenging to learn these unseen feature variations in advance, and this subject-wise variation can be seen in the samples of the gesture *Up* performed by different subjects shown in Fig. 16(*b*). To address these failure cases and further improve the performance of gesture recognition, the following future work will be considered: (i) more subjects can be added to include additional gesture features in model training and (ii) depth sensors can be used in the MHI generation process to obtain depth information of gestures.

5.2 Evaluation of Speech Recognition. To assess the proposed fuzzy matching strategy in Sec. 3, we constructed a data set of two non-native English-speaking subjects under the 10 noise backgrounds in Sec. 3 to show the robustness of our speech recognition on different English accents. The speech test data set contains 160 samples under every background noise for each command, including short sentences, such as "*move inward*," "*go left*," etc. The performance of speech recognition is shown in Table 5, where the recognition accuracy of "*Inward*" and "*Speed*

Table 5 Performance (%) of the speech recognition on non-native English-speaking subjects (labels 1–10 are the 10 different background noises in Fig. 10 of Sec. 3)

	Recognition accuracy under ten background noises										
Class	1	2	3	4	5	6	7	8	9	10	Average accuracy
Start	98.75	96.88	98.13	96.88	93.75	99.38	98.75	97.50	98.13	97.50	97.57
Stop	98.13	98.75	99.38	98.75	96.88	98.75	97.50	93.75	98.13	98.13	97.82
Go Home	99.38	98.75	100.00	95.00	93.75	95.00	93.75	98.75	99.38	94.38	96.81
Disengage	99.38	96.88	98.75	96.88	93.13	97.50	93.75	95.00	95.00	97.50	96.38
Up	99.38	98.75	98.75	93.75	93.75	95.00	99.38	93.75	98.75	99.38	97.06
Down	96.88	96.88	95.00	99.38	98.75	99.38	97.50	98.75	98.75	99.38	98.07
Left	99.38	98.75	100.00	96.88	94.38	98.75	98.75	98.75	98.75	98.75	98.31
Right	99.38	100.00	99.38	98.75	95.00	98.75	98.75	94.38	100.00	96.88	98.13
Inward	91.25	92.50	90.63	90.63	93.75	90.63	91.88	91.25	91.25	90.63	91.44
Outward	95.00	96.88	95.00	93.75	94.38	94.38	93.75	98.75	98.75	97.50	95.81
Open	97.50	100.00	94.38	94.38	99.38	98.75	95.00	97.50	94.38	98.75	97.00
Close	95.00	97.50	93.75	95.00	95.00	99.38	98.75	99.38	99.38	97.50	97.06
CW	98.75	96.88	93.75	99.38	95.00	98.75	97.50	95.00	97.50	100.00	97.25
CCW	95.00	99.38	94.38	93.75	95.00	99.38	99.38	93.75	95.00	100.00	96.50
Speed Up	97.50	97.50	98.75	96.88	98.75	94.38	99.38	95.00	95.00	97.50	97.06
Slow Down	93.75	93.75	94.38	99.38	95.00	95.00	99.38	97.50	99.38	97.50	96.50

Table 6 Performance of the proposed integration technique of gesture and speech results (sound intensity: 60–70dB)

	Number of experiments		Number of correct recognition		Correct rate	
Input commands	Subject 1	Subject 2	Subject 1	Subject 2	Subject 1	Subject 2
Gesture only	213	176	198	166	92.96%	94.32%
Speech only	297	223	277	209	93.27%	93.72%
Gesture and speech both	210	199	205	190	97.62%	95.48%

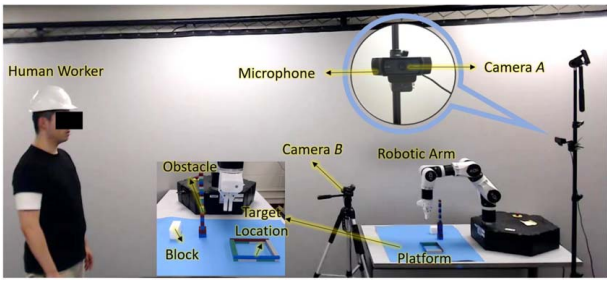

Fig. 17 Overview of the system demonstration

Up" is increased greatly to >90%, and all the other commands have high accuracy of >95%. The performance data in Table 5 show that the fuzzy matching strategy increases the performance of the speech recognition in our HRC system.

5.3 Evaluation of Gesture and Speech Integration. To evaluate the proposed multi-modal integration strategy in Sec. 4, comparison experiments were carried out to compare system performance when both gesture and speech are included or only one of them is used. The background sound intensity is 60–70 dB. The experimental results of two human subjects are shown in Table 6, and the correct rate of our proposed integration method performs better than when only gesture or speech is used.

5.4 Demonstration of the Human-Robot Collaboration System. In this section, we apply the proposed real-time, multi-modal HRC system to perform a pick-and-place task using a six-degree-of-freedom (6-DOF) e.DO robotic arm to demonstrate the system performance. A view of this system is shown in Fig. 17, where camera A and a microphone are equipped to collect gestures and speech from the human worker, and camera B is equipped to monitor the platform. The two-camera setting for MHI data collection in Sec. 2.1 is to have a more comprehensive data set on human gestures from different views in order to develop a robust CNN to recognize gestures. The platform includes a white block, an obstacle, and a target location. After the system is started, it continuously runs in the background to listen to and watch the human operator. The goal is to command the robot using gestures and speech to move the block to the target location along a collision-free path. As described in Sec. 4.2, the execution time for collecting gesture and speech data is set to 2.5 s. The microphone is activated 0.5 s prior to the camera A being activated, and the additional 0.5 s of audio is recorded as background noise.

Figures 18(*a*)–18(*c*) illustrate the experimental result of the *Start* command. To realize system visualization, a software interface is designed in Fig. 18(*a*), where two windows are used to update camera frames and real-time MHIs, respectively. The sound intensity level and recognition probability distribution of the current gesture are displayed in real time. The final identification results, i.e., the gesture, speech and integration results, are displayed at the bottom of the interface. In Fig. 18(*a*), the sound intensity is less than 70 dB. The speech of the human worker, i.e., *Start*, is selected as the final result since no gesture is given. In Fig. 18(*b*), the sound intensity is greater than 70 dB, and the gesture of the human worker is recognized as the final result of *Start*. Note that a gesture result is considered valid only when it is the most frequently occurring label of high-confidence recognition. After the integration result is transmitted to the robot, the end-effector performs the movements as shown in Fig. 18(*c*).

The experimental result of multiple commands for a pick-and-place task is demonstrated in Fig. 19, in which the robot delivers the white block to the target location along a collision-free path based on

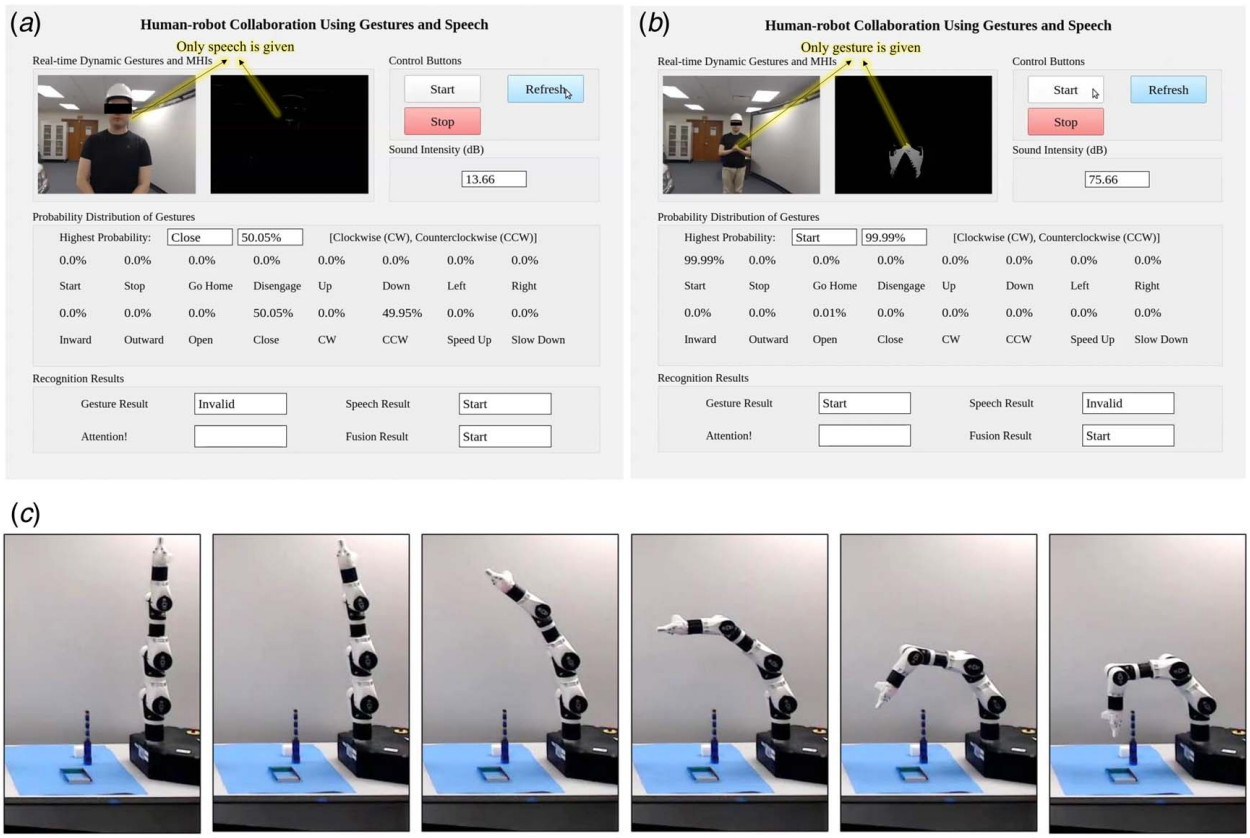

Fig. 18 Demonstration of the command *Start*. (*a*) Sound intensity ≤70 dB, (*b*) sound intensity >70 dB, and (c) robot response for the command *Start*.

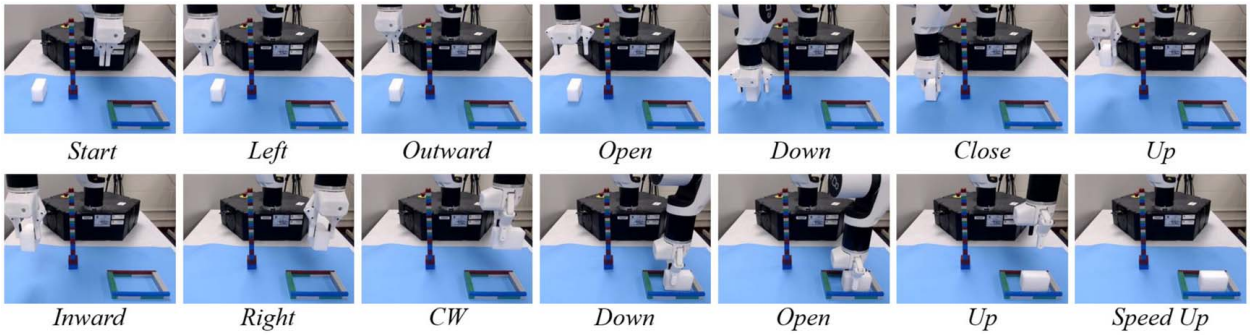

Fig. 19 Overview of the demonstration including various commands for the robot to perform a pick-and-place operation

commands from a human worker. During the delivery, the robot keeps the end-effector straight down, facing the platform. The human worker can use gestures and/or speech commands to communicate with the robot in real time. If a command is not recognized by the system, the human worker can continue to give it during the next execution time when the camera and microphone are turned on. When a command (such as *Left*) is received, the robot keeps moving (to the left) and stops until it reaches its joint limit or receives the *Stop* command. A video of the demonstration is available.[3]

6 Conclusion

We describe in this paper the design and development of a real-time, multi-modal human–robot collaboration (HRC) system using the integration of speech and gestures. For the communication, a set of 16 dynamic gestures is designed. A real-time motion history image (MHI) method is developed to extract dynamic gesture features. A convolutional neural network (CNN)-based recognition model is built for dynamic gesture recognition, and an open-source speech recognizer is adopted and improved for natural speech recognition. An integration strategy for combining gesture recognition and speech recognition is proposed, for which a multi-threading architecture is designed to enable the system to perform parallel tasks. The validation experiments demonstrate that the proposed gesture recognition model has an accuracy of >98% for the 16 designed dynamic gestures, and the speech experiments show that the improved speech recognizer achieves >95% accuracy for speech commands corresponding to the 16 dynamic gestures. A system demonstration using a 6-DOF robotic arm illustrates the performance of the proposed gesture–speech integration strategy and the feasibility of the proposed real-time HRC system.

Our system enables the worker to use both gesture and voice commands to control a robot, and integration of the two to increase the recognition accuracy. When a human is performing a task alongside a robot, the human will not be able to sign a gesture during the human–robot collaboration but will still have the option of using voice to command the robot, which has >91% accuracy (>96% in average) in our system. This still provides a natural and effective way for HRC. To further improve the proposed approach, future studies will include exploring other modalities to enhance HRC, such as brain waves and eye gaze, experimenting with other gesture representation methods to fully exploit the discriminative gesture features in dynamic backgrounds with moving objects and/or co-workers, performing object segmentation to generate MHI on the target human worker only, exploring speech recognition using a headset with a superb noise cancelation function in a noisy environment, developing system response to low-confidence cases by saying something like "Could you do it again?" for gesture and/or speech commands, and exploring energy-efficiency issues by designing sleep and activate modes for the system.

[3]https://youtu.be/GHGbFNu3lpQ

Acknowledgment

This research work was financially supported by the National Science Foundation grants CMMI-1646162 and CMMI-1954548 and also by the Intelligent Systems Center at Missouri University of Science and Technology. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.

Conflict of Interest

There are no conflicts of interest.

Data Availability Statement

The datasets generated and supporting the findings of this article are obtainable from the corresponding author upon reasonable request.

Appendix A

The ways to perform the proposed gestures as illustrated in Fig. 2 of the main text are as follows:

- Gesture 1 (*Start*): Clap in front of the chest.
- Gesture 2 (*Stop*): Raise the left/right arm until the hand reaches the height of the shoulder, and extend the arm with the palm facing the front, like a "stop" gesture in the traffic direction.
- Gesture 3 (*Go Home*): Straighten the two arms so that they are at an angle of 45 deg to the body, and then swing both arms inside and lock the fingers in front of the stomach.
- Gesture 4 (*Disengage*): Rotate the two arms, and move the two hands in front of the chest, and then extend the two hands to the sides until the hands reach the shoulder level.
- Gesture 5 (*Up*): Extend the left/right arm straight up.
- Gesture 6 (*Down*): Bend the left/right hand, and raise the wrist to the height of the chest, and then extend the hand straight down.
- Gesture 7 (*Left*): Swing the left arm straight out and up to the side until the arm reaches the height of the shoulder.
- Gesture 8 (*Right*): Swing the right arm straight out and up to the side until the arm reaches the height of the shoulder.
- Gesture 9 (*Inward*): Rotate the left/right forearm up around the elbow joint on the same side with the hand open, until the hand reaches the chest level. The palm faces back.
- Gesture 10 (*Outward*): Rotate the left/right forearm up around the elbow joint on the same side with the left hand open until the hand reaches the height of the chest, and then rotate the arm around the elbow joint on the same side until the arm is straight with about 30 deg from the body line. The palm faces back.

- Gesture 11 (*Open*): Bend each of the two arms up around each elbow on the side of the body until the hands touch the same side of the shoulder.
- Gesture 12 (*Close*): Bend the two arms and cross them in front of the chest with the two hands on different sides of the shoulder. The palms face backwards and the fingers are open.
- Gesture 13 (*Clockwise* (*CW*)): Rotate the forearm of the left/right arm clockwise around the elbow joint of the same side until the forearm reaches the shoulder level.
- Gesture 14 (*Counterclockwise* (*CCW*)): Rotate the forearm of the left/right arm counterclockwise around the elbow joint of the same side until the forearm reaches the height of the shoulder.
- Gesture 15 (*Speed Up*): Swing the two arms straight up in the front of the body.
- Gesture 16 (*Slow Down*): Swing the two forearms up around the elbow joints until both hands reach the height of the shoulder, and then swing the forearms back to the start position in front of the body.

References

[1] Burns, A., and Wellings, A., 2001, *Real-Time Systems and Programming Languages*, 3rd ed., Pearson Education, Harlow, UK.

[2] Nicora, M. L., Ambrosetti, R., Wiens, G. J., and Fassi, I., 2021, "Human–Robot Collaboration in Smart Manufacturing: Robot Reactive Behavior Intelligence," ASME J. Manuf. Sci. Eng., **143**(3), p. 031009.

[3] Liu, S., Wang, L., and Wang, X. V., 2021, "Function Block-bBsed Multimodal Control for Symbiotic Human–Robot Collaborative Assembly," ASME J. Manuf. Sci. Eng., **143**(9), p. 091001.

[4] Arinez, J. F., Chang, Q., Gao, R. X., Xu, C., and Zhang, J., 2020, "Artificial Intelligence in Advanced Manufacturing: Current Status and Future Outlook," ASME J. Manuf. Sci. Eng., **142**(11), p. 110804.

[5] Chen, H., Leu, M. C., Tao, W., and Yin, Z., 2020, "Design of a Real-Time Human–Robot Collaboration System Using Dynamic Gestures," ASME International Mechanical Engineering Congress and Exposition, Virtual Conference, Nov. 16–19.

[6] Wang, X. V., and Wang, L., 2021, "A Literature Survey of the Robotic Technologies During the Covid-19 Pandemic," J. Manuf. Syst., **60**, pp. 823–36.

[7] Zinchenko, K., Wu, C.-Y., and Song, K.-T., 2016, "A Study on Speech Recognition Control for a Surgical Robot," IEEE Trans. Ind. Inf., **13**(2), pp. 607–615.

[8] Bingol, M. C., and Aydogmus, O., 2020, "Performing Predefined Tasks Using the Human–Robot Interaction on Speech Recognition for an Industrial Robot," Eng. Appl. Artif. Intell., **95**, p. 103903.

[9] Kuhn, M., Pollmann, K., and Papadopoulos, J., 2020, "I'm Your Partner-I'm Your Boss: Framing Human–Robot Collaboration With Conceptual Metaphors," Companion of the 2020 ACM/IEEE International Conference on Human–Robot Interaction, Virtual Conference, Mar. 24–26, pp. 322–324.

[10] Coupeté, E., Moutarde, F., and Manitsaris, S., 2016, "A User-Adaptive Gesture Recognition System Applied to Human–Robot Collaboration in Factories," Proceedings of the 3rd International Symposium on Movement and Computing, Thessaloniki, GA, Greece, July 5–6, pp. 1–7.

[11] Unhelkar, V. V., Lasota, P. A., Tyroller, Q., Buhai, R.-D., Marceau, L., Deml, B., and Shah, J. A., 2018, "Human-Aware Robotic Assistant for Collaborative Assembly: Integrating Human Motion Prediction With Planning in Time," IEEE Rob. Autom. Lett., **3**(3), pp. 2394–2401.

[12] Pinto, R. F., Borges, C. D., Almeida, A., and Paula, I. C., 2019, "Static Hand Gesture Recognition Based on Convolutional Neural Networks," J. Electr. Comput. Eng., **2019**.

[13] Li, J., Liu, X., Zhang, M., and Wang, D., 2020, "Spatio-Temporal Deformable 3d Convnets With Attention for Action Recognition," Pattern Recognit., **98**, p. 107037.

[14] Tao, W., Lai, Z.-H., Leu, M. C., and Yin, Z., 2018, "Worker Activity Recognition in Smart Manufacturing Using IMU and SEMG Signals With Convolutional Neural Networks," Procedia Manuf., **26**, pp. 1159–1166.

[15] Treussart, B., Geffard, F., Vignais, N., and Marin, F., 2020, "Controlling an Upper-Limb Exoskeleton by EMG Signal While Carrying Unknown Load," 2020 IEEE International Conference on Robotics and Automation (ICRA), Virtual Conference, May 31–Aug. 31, IEEE, pp. 9107–9113.

[16] Ajoudani, A., Zanchettin, A. M., Ivaldi, S., Albu-Schäffer, A., Kosuge, K., and Khatib, O., 2018, "Progress and Prospects of the Human–Robot Collaboration," Auton. Rob., **42**(5), pp. 957–975.

[17] Yongda, D., Fang, L., and Huang, X., 2018, "Research on Multimodal Human–Robot Interaction Based on Speech and Gesture," Comput. Electr. Eng., **72**, pp. 443–454.

[18] Lin, K., Li, Y., Sun, J., Zhou, D., and Zhang, Q., 2020, "Multi-sensor Fusion for Body Sensor Network in Medical Human–Robot Interaction Scenario," Inf. Fusion, **57**, pp. 15–26.

[19] Wang, L., Liu, S., Liu, H., and Wang, X. V., 2020, "Overview of Human–Robot Collaboration in Manufacturing," Proceedings of 5th International Conference on the Industry 4.0 Model for Advanced Manufacturing, Belgrade, Serbia, June 1–4, Springer, pp. 15–58.

[20] Yu, G., Liu, Z., and Yuan, J., 2014, "Discriminative Orderlet Mining for Real-Time Recognition of Human-Object Interaction," Asian Conference on Computer Vision, Singapore, Nov. 1–5, Springer, pp. 50–65.

[21] Shinde, S., Kothari, A., and Gupta, V., 2018, "Yolo Based Human Action Recognition and Localization," Procedia Comput. Sci., **133**, pp. 831–838.

[22] Sun, B., Wang, S., Kong, D., Wang, L., and Yin, B., 2021, "Real-Time Human Action Recognition Using Locally Aggregated Kinematic-Guided Skeletonlet and Supervised Hashing-by-Analysis Model," IEEE Trans. Cybern.

[23] Yu, J., Gao, H., Yang, W., Jiang, Y., Chin, W., Kubota, N., and Ju, Z., 2020, "A Discriminative Deep Model With Feature Fusion and Temporal Attention for Human Action Recognition," IEEE Access, **8**, pp. 43243–43255.

[24] Pisharady, P. K., and Saerbeck, M., 2015, "Recent Methods and Databases in Vision-Based Hand Gesture Recognition: A Review," Comput. Vis. Image Understand., **141**, pp. 152–165.

[25] McNeill, D., 2008, *Gesture and Thought*, University of Chicago Press, Chicago, IL.

[26] Holler, J., and Wilkin, K., 2009, "Communicating Common Ground: How Mutually Shared Knowledge Influences the Representation of Semantic Information in Speech and Gesture in a Narrative Task." Lang. Cogn. Process., **24**(2), pp. 267–289.

[27] Yin, Z., and Collins, R., 2006, "Moving Object Localization in Thermal Imagery by Forward–Backward MHI," 2006 Conference on Computer Vision and Pattern Recognition Workshop (CVPRW'06), New York, NY, June 17–22, IEEE, pp. 133–133.

[28] Ahad, M. A. R., Tan, J. K., Kim, H., and Ishikawa, S., 2012, "Motion History Image: Its Variants and Applications," Mach. Vision Appl., **23**(2), pp. 255–281.

[29] Bobick, A. F., and Davis, J. W., 2001, "The Recognition of Human Movement Using Temporal Templates," IEEE Trans. Pattern Anal. Mach. Intell., **23**(3), pp. 257–267.

[30] Chen, H., Tao, W., Leu, M. C., and Yin, Z., 2020, "Dynamic Gesture Design and Recognition for Human–Robot CCollaboration With Convolutional Neural Networks," International Symposium on Flexible Automation, Virtual Conference, July 8–9, American Society of Mechanical Engineers, p. V001T09A001.

[31] Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R., 2014, "Dropout: A Simple Way to Prevent Neural Networks From Overfitting," J. Mach. Learn. Res., **15**(1), pp. 1929–1958.

[32] Chen, B., Deng, W., and Du, J., 2017, "Noisy Softmax: Improving the Generalization Ability of Dcnn Via Postponing the Early Softmax Saturation," Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, July 21–26, pp. 5372–5381.

[33] Yeo, K., and Melnyk, I., 2019, "Deep Learning Algorithm for Data-Driven Simulation of Noisy Dynamical System," J. Comput. Phys., **376**, pp. 1212–1231.

[34] Kopparapu, S. K., and Laxminarayana, M., 2010, "Choice of Mel Filter Bank in Computing MFCC of a Resampled Speech," 10th International Conference on Information Science, Signal Processing and Their Applications (ISSPA 2010), Kuala Lumpur, Malaysia, May 10–13, IEEE, pp. 121–124.

[35] Li, B., Sainath, T. N., Narayanan, A., Caroselli, J., Bacchiani, M., Misra, A., Shafran, I., Sak, H., Pundak, G., Chin, K. K., and Sim, K. C., 2017, "Acoustic Modeling for Google Home," Interspeech, Stockholm, Sweden, Aug. 20–24, pp. 399–403.

[36] Rabinowitz, P., 2000, "Noise-Induced Hearing Loss," Am. Family Physician, **61**(9), pp. 2749–2756.

[37] Kamath, S., and Loizou, P., 2002, "A Multi-Band Spectral Subtraction Method for Enhancing Speech Corrupted by Colored Noise," ICASSP, Orlando, FL, May 13–17.

[38] Upadhyay, N., and Karmakar, A., 2015, "Speech Enhancement Using Spectral Subtraction-Type Algorithms: A Comparison and Simulation Study," Procedia Comput. Sci., **54**, pp. 574–584.

[39] Gilakjani, A. P., 2016, "English Pronunciation Instruction: A Literature Review," Int. J. Res. Engl. Educ., **1**(1), pp. 1–6.

[40] Amano, A., Aritsuka, T., Hataoka, N., and Ichikawa, A., 1989, "On the Use of Neural Networks and Fuzzy Logic in Speech Recognition," Proceedings of the 1989 International Joint Conference Neural Networks, Washington, DC, June 18–22, Vol. 301, pp. 147–169.

[41] Vani, H., and Anusuya, M., 2020, "Fuzzy Speech Recognition: A Review," Int. J. Comput. Appl., **177**(47), pp. 39–54.

[42] Karimov, E., 2020, *Data Structures and Algorithms in Swift*, Springer, New York City.

[43] Visentini, I., Snidaro, L., and Foresti, G. L., 2016, "Diversity-Aware Classifier Ensemble Selection Via F-Score," Inform. Fusion, **28**, pp. 24–43.

[44] Al-Amin, M., Tao, W., Doell, D., Lingard, R., Yin, Z., Leu, M. C., and Qin, R., 2019, "Action Recognition in Manufacturing Assembly Using Multimodal Sensor Fusion," Procedia Manuf., **39**, pp. 158–167.