2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
October 25-29, 2020, Las Vegas, NV, USA (Virtual)

Enabling Robot to Assist Human in Collaborative Assembly using Convolutional Neural Networks

Yi Chen, Weitian Wang, Venkat Krovi and Yunyi Jia*

Abstract— **Human-robot collaborative assembly consists of humans and automated robots, who cooperate with each other to accomplish complex assembly tasks, which are difficult for either humans or robots to accomplish alone. There has been some success in statistics-based and optimization-based approaches to realize human-robot collaboration. However, they usually need a set of complex modeling and setup efforts and the robots usually need to be programmed by a well-trained expert. In this paper, we take a new approach by introducing convolutional neural networks (CNN) into the teaching-learning-collaboration (TLC) model for collaborative assembly tasks. The proposed approach can alleviate the need for complex modeling and setup compared to the existing approaches. It can collect and automatically label the data from human demonstrations and then train a CNN-based robot assistance model to make the robot assist humans in the assembly process in real-time. We have experimentally verified our proposed approach on a human-robot collaborative assembly platform and the results suggest that the robot can successfully learn from human demonstrations to automatically generate right actions to assist human in accomplishing assembly tasks.**

I. Introduction

The human-robot collaborative assembly has become popular research in recent years due to the difficulty of automating the tasks. By involving human and robot into the same workspace, the human-robot collaborative assembly includes diverse research such as human intent prediction [1], object recognition and grasping, human-friendly trajectories generating [2], and robot teaching and learning techniques [3], [4], etc.

Early research on processing planning of assembly tasks was typically based on the task-allocation situation [5]. However, it is very challenging to apply these techniques to human-robot collaborative assembly situations because of the extensive uncertainties introduced by both task flexibilities and human operations. For instance, different human operators may prefer to assemble the same product with different gestures, different tools and/or different sequences of operations. These challenges have stimulated a variety of research in the field:

Tsarouchi et al. [6] has given a review on human-robot interaction related to task planning and programming with an emphasis on the manufacturing environment. Some of the principles such as human comfort, natural motion have been studied in human-aware robot navigation [7]. There were some studies on statistics-based approach: Calinon et al. [8] presented a task-parameterized movement learning approach, Rozo et al. [9] extend the approach to task-parameterized impedance learning in human-robot collaborative assembly task, which is based on a statistical representation of dynamical system that can be modulated with respect to task variables represented as candidate frames of reference. There were also some studies on optimization-based approach: Mainprice et al. [10] presented a framework to predict human reaching motion in collaborative tasks using inverse optimal control, which is based on Path Integral Inverse Reinforcement Learning (PIIRL) and Stochastic Trajectory Optimizer for Motion Planning (STOMP).

However, when applying these existing approaches to human-robot collaborative assembly, the collaborative tasks usually need a set of complex modeling and setup efforts [11], and robots usually need to be programmed by a well-trained expert. This increases the cost of applying collaborative robots in human-robot assembly and also makes the collaborative robots very complicated and very inconvenient for end-users to use.

Therefore, in order to address this challenge, in this paper, we introduce deep learning to make robots easily learn undefined task and workspace situation from human demonstrations and enable the trained robots to actively assist human operators in the human-robot collaborative assembly in real-time.

Deep learning methods have dramatically improved the state-of-art in visual object recognition and object detection [12]. Taking advantage of the outstanding performance of object recognition, many types of research have been done in robotic grasping based on convolutional neural networks (CNN) [13]. In robotic motion control perspective, Zhang et al. [14] presented a vision-based deep reinforcement learning

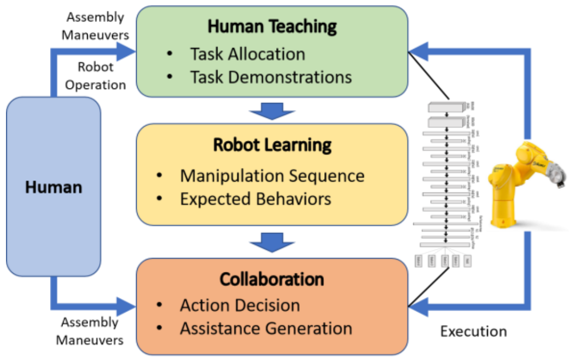

Fig. 1. The system diagram of the TLC model.

This work was supported by the National Science Foundation under Grant IIS-1845779.

Yi Chen, Venkat Krovi and Yunyi Jia are with the Department of Automotive Engineering, Clemson University, Greenville, SC 29607, USA (e-mail: yc4@clemson.edu, vkrovi@clemson.edu, yunyij@clemson.edu).

Weitian Wang is with the Department of Computer Science, Montclair State University, NJ 07043, USA (e-mail: wangw@montclair.edu).

*Corresponding author: Yunyi Jia (e-mail: yunyij@clemson.edu).

for robot motion control; the research of Pwhich et al. [15] revealed that the end-to-end training results in a better performance in control policy search and robots can learn control policies for a wide range of tasks on the same neural networks. These existing works motivate us to introduce CNN into human-robot collaborative assembly tasks. We propose to recognize the state of the assembly objects and the environment of the workspace through CNN and predict the corresponding responses of robots to actively assist human operators in human-robot collaborative assembly tasks.

The contributions of the paper can be summarized as (a) proposing a CNN based approach to make robot learn to assist humans in collaborative assembly from human demonstrations; (b) using CNN to alleviate the need of complex modeling and setup of traditional approaches; (c) online data collection and automatic image labeling for the training of CNN.

II. Problem Formulation

A. An overview of the TLC model

Human-robot collaboration in collaborative assembly requires robots to assist humans actively, which means the manipulations of robots must be deployed automatically without manually trigger actions from human operators. To solve this, we proposed a TLC model, which consists of human-teaching, robot-learning, and human-robot collaboration. The system diagram of the TLC model is shown in Fig. 1. In the human-phase, the collaborative assembly task is allocated to both the human and the robot. In general, the robot should cooperate with the human operator to handle the lower-precision and higher-strength jobs, while the human operator focuses on the higher-precision and lower-strength assembly operations. To demonstrate the collaborative assembly task, the human operates the robot through intuitively human-robot interactions, such as leading-through, joystick operation when conducting the assembly maneuvers. Therefore, human operators are able to teach the robot to accomplish collaborative assembly tasks through natural demonstrations. In the robot-learning phase, the robot learns the expected behaviors in the process of collaborative assembly tasks online based on the scene of the

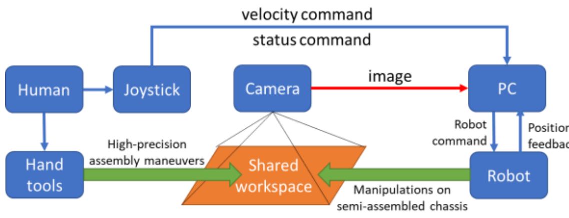

Fig. 2. Robot system configuration for learning from demonstration.

shared workspace captured by the version system and the

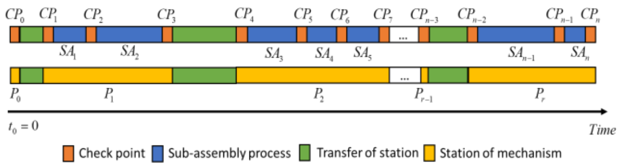

Fig. 3. The time series analysis of collaborative assembly tasks.

robot operations taught by the human operator. Based on the task knowledge learned from human demonstrations, in the human-robot collaboration phase, the robot makes action decisions and generates proper assistant behaviors by given real-time images of the shared workspace. In this work, by modeling a collaborative assembly task as a time series, CNN is introduced to map the real-time vision of the shared workspace to proper robot assistant behaviors.

B. Time Series Analysis of Collaborative Assembly Tasks

Human-robot collaborative assembly tasks are complex time series, which include massive strict constraints (e.g. force/torque, tolerance, etc.), plentiful flexible manipulations (e.g. personalized preference in gestures, tools, and sequence of maneuvers, etc.) and dynamic environments. These characteristics lead to challenges to pre-define and program every possible state that may happen in the collaborative assembly. However, any specific assembly process must follow a certain sequence of requirements to achieve a successful final assembly.

Fig. 3 illustrates the time series of a collaborative assembly task. From a mechanical perspective, *checkpoints* CP_i represent a series of discrete states, which are necessary and order-sensitive for the mechanism targeting for a successful final assembly. A *sub-assembly* SA_i is defined as a set of maneuvers, which make the state of mechanism transfer from the current checkpoint to the next checkpoint. The sub-assembly processes often include plenty of flexible operations accomplished by human operators which are not order-sensitive and highly based on the humans' personal preferences. A *station* P_i represents a position and orientation of the semi-assembled machine that leads to the comfortable and convenient installation of new parts for human operators corresponding to the current sub-assembly section. In summary, an entire assembly task can be regarded as a time series consists of checkpoints, sub-assembly processes, hold and transfer of stations of the semi-assembled machine. The robot must generate proper behaviors to move from one station to another in real-time based on the state of the assembly task and the behavior of the human operator.

For an arbitrary human-robot collaborative assembly task, the configurations of stations are always discrete and finite though their values are unknown in advance. The configurations of stations are mainly determined by the design of the mechanism and humans' personal performances. Therefore, the problem is formulated as two steps:

Step 1: the robot learns the applicable station set $\{P\}$ for the collaborative assembly. In this process, the information of the tasks, such as the operations of the human operator and the states of semi-assembled mechanism, are represented through sequences of camera frames. Meanwhile, the applicable station set $\{P\}$ is abstracted from the position and speed feedback of the robot in human demonstrations.

Step 2: the robot generates proper behavior to assist the human in the collaborative assembly. In this human-robot collaboration process, the robot should generate proper behavior based on the real-time images of the shared workspace. This is achieved by a trained CNN, which maps the situation of the shared workspace to a set of proper behaviors learned in the human demonstrations.

III. Learning from Demonstrations using CNN

A. Robot System Configuration

The robot system configuration for the human-teaching process is shown in Fig. 2. The robot holds the semi-assembled mechanism in its gripper, the state of the workspace is captured by a camera. In the process of human demonstrations, the human operator uses a joystick to control the motion of the robot. After moving the robot to a station that is convenient for him/her to conduct the following sub-assembly maneuvers, the human operator accomplishes the desired sub-assembly maneuvers by selecting correct parts and assembling them to the semi-assembled mechanism with hand tools.

For each round of demonstration, we can obtain four datasets of time series data: a series of timestamped images, which include the information of mechanism status and human operations; the moments when the robot speed turns to zero, which indicate the human intends to hold the position of the end-effect; the moments when the robot starts to move, which indicate the human wants to move the robot to the next station; and the positions of the end-effector when the robot speed turns to zero, which indicate the set of stations selected by the human operator for the assembly task. Based on these four datasets, the images can be automatically and effectively labeled according to their timestamps.

```
Algorithm 1: Data Acquisition and Automatic Image Labeling
Initialization
    Initialize the list of "robot moving moment"
    Initialize the list of "robot stopping moment"
    Initialize the list of "robot stopping pose"
    Initialize the robot position and velocity
    Initialize the folder for temporary image storage

Human demonstration
    While the current round of demonstration is not finished, do
        Save the timestamped image of workspace
        Read current robot speed
        Read current robot position

        If the robot is stopping, then
            Append the current robot position to "robot stopping pose"
            Append the current timestamp to "robot stopping moment"

        If the robot is starting to move, then
            Append the current timestamp to "robot moving moment"

        If the current round of demonstration is finished, then
            Break the while loop

Automatic image labeling
    Set the proper time interval $\Delta t$
    For image in temporary image storage do
        Get the timestamp of the image $t_i$
        Find the nearest timestamp $t_s$ in the list of "robot stopping
        moment"
        Find the nearest timestamp $t_m$ in the list of "robot moving
        moment"
        If $t_i$ is earlier than $t_s$ and $t_i$ is later than $t_m - \Delta t$, then
            Get the corresponding robot stopping pose $P$ at $t_s$
            Label the image as "Moving to $P$"

        If $t_i$ is later than $t_s$ and $t_i$ is earlier than $t_m - \Delta t$, then
            Label the image as "Stop at current position"
```

B. Automatically Image Labeling

Unlike many deep learning cases, whose datasets are manually labeled, the image frames of the workspace in the collaborative assembly process are automatically labeled based on the timestamps. Fig. 4 illustrates the timing sequence to label the image time series data sampled in the human demonstration. The algorithm of automatic image labeling is shown in Algorithm 1. When the robot is stopped at a station, the position of the robot end-effector is recorded through the robot feedback. When the human operator is conducting the assembly maneuvers, the real-time images are sampled and saved to the computer, and all the images are timestamped. All these timestamped images are mapped to a robot behavior, which is the robot should stop at this specific station to wait until the sub-assembly is accomplished.

After the current sub-assembly process being finished, the human operator uses a joystick to move the robot to the next station. The real-time images of the process that the robot transfers from the current station to the next station are also captured by the camera and all the images are timestamped as well. When the robot arrived at the next proper station for the human operator to conduct the following assembly maneuvers, the robot is stopped by the human via the joystick. The images in this period should be mapped to a robot motion, which is the robot end-effector transfers from the previous station to the current station. The current station is also recorded via robot position feedback.

Since the human operator uses a joystick to operate the robot and uses hand tools to conduct the assembly maneuvers, there is a time interval Δt when the human operator switches

TABLE I. Structure of CNN

Type	Stride	Input Size	Filter Shape
Conv	1	800 x 300 x 3	3 x 3 x 3 x 16
Max Pool	2	800 x 300 x 3	Pool 2 x 2
Conv	1	400 x 150 x3	3 x 3 x 3 x 32
Max Pool	2	400 x 150 x3	Pool 2 x 2
Conv	1	200 x 75 x3	3 x 3 x 3 x 32
Max Pool	2	200 x 75 x3	Pool 2 x 2
Conv	1	100 x 38 x 3	3 x 3 x 3 x 64
Max Pool	2	100 x 38 x 3	Pool 2 x 2
Conv	1	50 x 19 x 3	3 x 3 x 3 x 64
Max Pool	2	50 x 19 x 3	Pool 2 x 2
Conv	1	25 x 10 x 3	3 x 3 x 3 x 64
Max Pool	2	25 x 10 x 3	Pool 2 x 2
Flat	N/A	13 x 5 x 64	N/A
FC	1	1x1x4160	4160 x 512
FC	1	1 x 1 x 512	512 x 256
Softmax	1	1 x 1 x 256	Classifier

* Conv is the convolutional layer
* FC is the fully connected layer
* Filter shape is noted by width, height, channel and number of filters

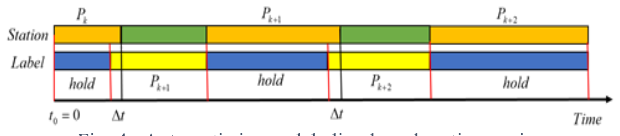

Fig. 4. Automatic image labeling based on time series.

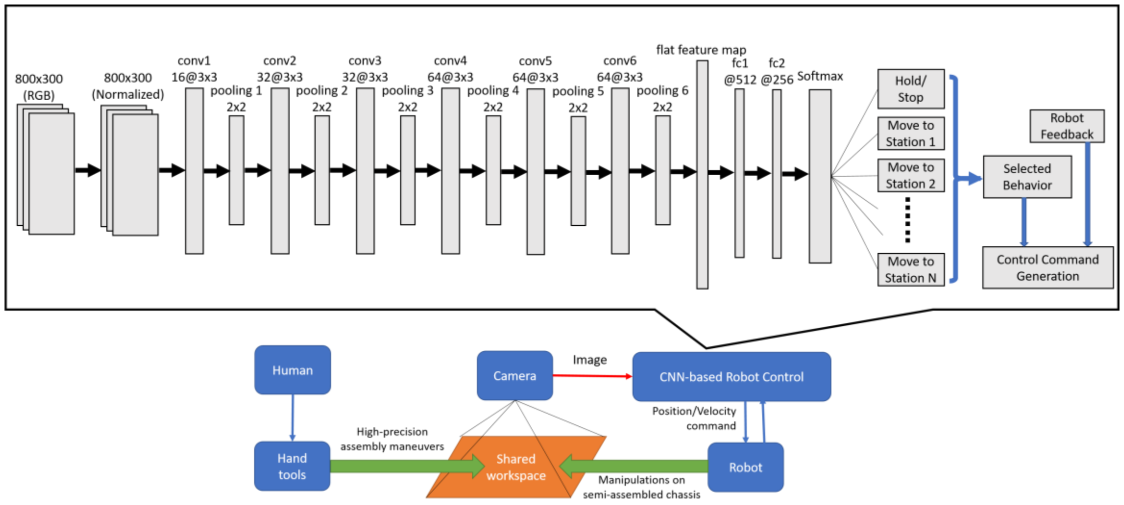

Fig. 5. The structure of CNN and robot configuration for CNN implementation.

his/her hand between the joystick and hand tools. In this time interval, the last sub-assembly has been finished and the state of the assembly has already arrived at the checkpoint, therefore, the images in this period should be mapped to the next robot motion. In our experiment, we found that the time interval Δt is generally kept consistent in serval rounds of human demonstrations, and is affected by the level of proficiency of the human operator. By selecting a proper value of the time interval, the error rate of image labeling can be controlled from 1% to 2%, which is normally acceptable for CNN training.

C. Structure of CNN

The structure of CNN and system diagram we used in this work is shown in Fig. 5. The detail of the structure of CNN is given in Table I. The input size is notated by image width, image height, and number of channels. The filter shape of convolutional layers is noted by filter height, filter width, filter height, number of channels, and number of filters. The image of the workspace is captured by a webcam and the sampling frequency is 2Hz. This sampling frequency is selected based on the normal operating speed of the human operator, which can obtain enough data to present the assembly process and avoid too many repetitive images. The original RGB images obtained by the camera are first cropped and resized to 800 x 300 x 3. Then the image is pixel-wise normalized in each channel by

$$\mu_x = \frac{1}{H \times W} \sum_{i=1}^{H \times W} x_i \tag{1}$$

$$\sigma_x = \sqrt{\frac{1}{H \times W} \sum_{i=1}^{H \times W} (x_i - \mu_x)^2} \tag{2}$$

$$\hat{x}_i = \frac{x_i - \mu_x}{\sigma_x} \tag{3}$$

where H is the height of the image, W is the width of the image, and $x_i \in [0, 255]$ is the pixel value at a specific position in one channel of the RGB image. The parameters of the CNN applied in this work are illustrated in Table I. It includes six convolution-pool sections after the image normalization. The number of filters is variant corresponding to the convolutional layers. The filter size and the stride of the max pool are set as 2 x 2 and 1. After the convolution-pool sections, the output is flattened as an array with 4160 elements. Then, there are two fully connected layers, which have 512 and 256 neurons respectively. A Softmax classifier is used to calculate the loss function and map the probability to each robot's behavior. The probability of each potential robot behavior can be written as

$$\sigma(z)_i = \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}} (\forall i \in 1, 2, ..., K) \tag{4}$$

where K is the total number of robot behaviors to predict in the collaborative assembly task, which is learned from the human demonstrations. The sparse cross-entropy loss is applied for the measurement of classification measurement. The Adam optimizer is implemented for the training of the CNN. The initial learning rate is set as 0.002 with an exponential decay rate of 0.98. Considering the limitation of the memory on our workstation, the batch size of training,

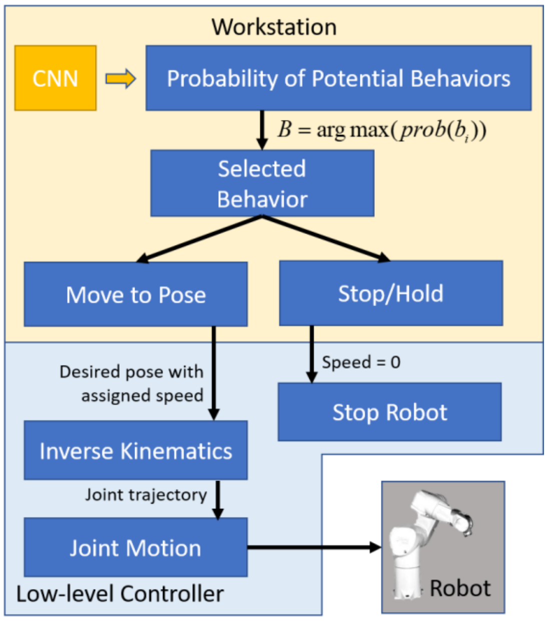

Fig. 6. Robot control diagram for CNN implementation.

validation, and testing are set as 100 images. After every 500 iterations, the updated CNN is validated throughout the overall validation dataset. If the validation result is better than the previous validation, then the current parameters of the CNN are saved to a file. The maximum training epochs are set as 100 rounds. Once the training is finished, the CNN parameters with the minimum validation error are selected for robot assistance generation.

D. CNN-based Robot Assistance Generation

The system configuration of the robot system for CNN implementation is illustrated in Fig. 5. In this case, the human does not use the joystick to control the robot behavior. The diagram of the robot control logic is illustrated in Fig. 6. The scenario of the shared workspace is sampled at a frequency of 10Hz, which is a normal frequency used in real-time control. The same pre-process including cropping, resizing and normalization is conducted for the image as the human demonstration before feeding to the CNN. The trained CNN can generate the probability of each potential robot behaviors, which are learned from human demonstration. The robot behavior with the highest probability is selected to generate the robot assistant manipulation.

If the selected behavior is to move the robot end-effector to a pose $P=\{x,y,z,\alpha,\beta,\gamma\}$, the desired pose with assigned robot speed will be sent to the lower level controller. The lower level controller compares the received desired position with the real-time robot pose feedback. If the current robot pose is different from the desired robot pose, then a joint motion trajectory is generated based on the inverse kinematics of the robot in the low-level controller. The robot executes the joint trajectory to move to the desired robot pose with the assigned robot speed. If the selected robot behavior is to stop and hold the robot at the current pose, a zero robot velocity command will send to the low-level robot controller, which makes the robot stop immediately.

IV. Experimental Results and Analysis

A. Experimental Setup

The real robot system setup for the experiment is shown in Fig. 7(a). The experiment is conducted based on a Staubli TX40 industrial robot. A web camera is used to record the state in the human-robot shared workspace of the collaborative assembly. The system integration for robot control, joystick operation, computer version are based on Robot Operating System (ROS). The CNN is built, trained and deployed with TensorFlow. The low-level motion planning and robot speed control is implemented by the joint motion function of Staubli TX40 controller.

The vehicle model (Fig. 7(c)) is disassembled as four wheels, the front bumper, the rear bumper, the semi-assembled chassis and the corresponding screws and washers for each component (Fig. 7(b)). In the human

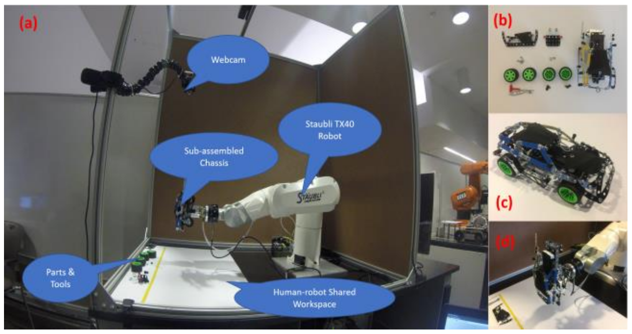

Fig. 7. Robot setup for human-robot collaborative assembly.

demonstrations, human uses the joystick to operate the robot to move the semi-assembled chassis to a proper location, which is comfortable and convenient for him/her to conduct the following sub-assembly maneuvers. The image of the workspace and the position of the robot are recorded for the CNN training. In the process of the CNN implementation, the real-time images are acquired by the webcam at the same location and the robot behavior is triggered automatically based on the image input of the CNN, which enables the robot to move to a position that is comfortable and convenient for the human worker to accomplish the assembly maneuvers.

B. CNN Validation and Testing Results

In the learning efficiency perspective, the robot should accomplish the learning process in a few demonstrations of specific tasks for the collaborative assembly applications. In our experiment, we have created four datasets (D1- D4) from various human demonstrations of accomplishing the assembly tasks. Based on these datasets we have conducted the training, validation, and testing with two different configurations.

Firstly, we used only the dataset D1 to train and the dataset D3 to determine the parameters of the neural networks. The fourth demonstration was used as a test dataset for the trained neural networks. The training process was totally run for 100 epochs, meanwhile, the entire images of the 3rd demonstration as the validation set were fed to the neural networks for every 500 iterations. The parameters of the neural networks were saved whenever we get a better result in the validation. The best validation result is 96.21% correct prediction which was achieved in 54000 iterations. The trained neural networks

TABLE II. Training, Validation and Testing Results

Training Dataset	Validation Dataset	Test Dataset	Validation (%)	Test (%)
D1	D3	D4	96.21	98.34
D1 & D2	D3	D4	98.26	98.49

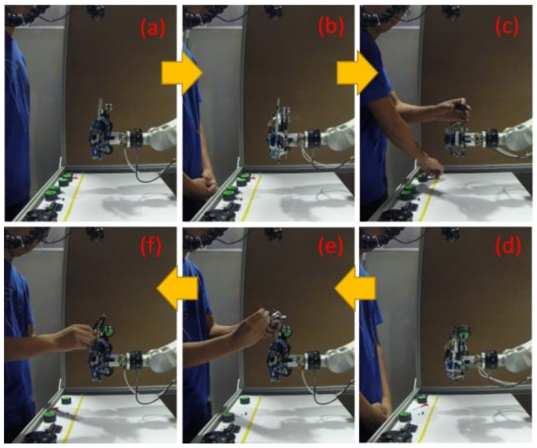

Fig. 8. The robot supportive behaviors for two wheels and front bumper assembly.

with the parameter corresponding to the minimum validation error get an average prediction accuracy as 98.34% with the test dataset.

Secondly, we used the dataset D1 and D2 to train the neural networks so that the training dataset increased to 1334 images in total. The dataset D3 and D4 demonstrations were still used for validation and testing. The best validation result is 98.26% correct prediction which was achieved in 120000 iterations. The trained neural networks with the parameter corresponding to the minimum validation error get an average prediction accuracy as 98.49% with the test dataset. The training, validation and teasing results of both configurations are summarized in Table II.

According to the results of the two different training-validation-testing configurations, the prediction accuracies on the validation dataset are both higher than 95%. Through the CNN trained by the former configuration has a lower validation accuracy, the average test accuracies of both configurations are similar to each other, which are 98.34% and 98.49% respectively. In our experiment, both CNNs successfully assisted the human operator in accomplishing the model vehicle assembly task.

The human-robot collaboration in the process of the two wheels and front bumper assembly is shown in Fig. 8. The robot can hold or turn the proper station to make the human operator to install the parts easily. The scenario of an intelligent emergency stop function of the robot is shown in Fig. 9. Once the features of human hands were detected, the control commands to stop the robot were generated by the neural networks. The robot stopped immediately before collision when the human operator's hand suddenly approached the moving chassis.

These experimental results demonstrated the trained CNN can generate the proper supportive behaviors automatically in the human-robot collaboration to help the human operator in the assembly of the model vehicle. In the process of human demonstrations, when the human hands are working on assembly maneuvers in the shared workspace, the robot is always stopping and holding on a specific station. The feature of human hands is successfully abstracted by the CNN, which enables the robot to stop immediately in many other states besides the learned stations when the human hands are approaching the moving robot arm.

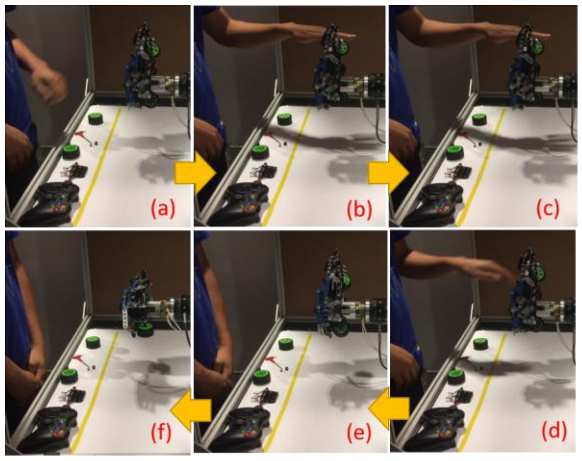

Fig. 9. The robot stops immediately when human hands approaching the moving object.

V. Conclusions

This paper proposes a CNN based approach to learn and assist humans in assembly tasks from human demonstrations. Experimental results show that CNN is effective in robot learning during collaborative assembly and the robot can be trained to actively assist humans in the human-robot collaborative assembly process in real-time. The datasets for training, validation, and testing can be created and labeled online from human demonstrations. The approach can help alleviate the need for complex modeling and setup compared to the existing approaches. Our approach also suggests a potential way by which the robot can be personalized by its users to assist them in their preferred ways.

References

[1] M. S. Ryoo, T. J. Fuchs, L. Xia, J. K. Aggarwal, and L. Matthies, "Robot-Centric Activity Prediction from First-Person Videos: What Will They Do to Me?," *ACM/IEEE Int. Conf. Human-Robot Interact.*, vol. 2015-March, pp. 295–302, 2015.

[2] E. C. Grigore, A. Roncone, O. Mangin, and B. Scassellati, "Preference-Based Assistance Prediction for Human-Robot Collaboration Tasks," *IEEE Int. Conf. Intell. Robot. Syst.*, pp. 4441–4448, 2018.

[3] T. Hamabe, H. Goto, and J. Miura, "A programming by demonstration system for human-robot collaborative assembly tasks," *2015 IEEE Int. Conf. Robot. Biomimetics, IEEE-ROBIO 2015*, pp. 1195–1201, 2015.

[4] Y. Wang, R. Xiong, L. Shen, K. Sun, J. Zhang, and L. Qi, "Towards learning from demonstration system for parts assembly: A graph based representation for knowledge," *4th Annu. IEEE Int. Conf. Cyber Technol. Autom. Control Intell. Syst. IEEE-CYBER 2014*, pp. 174–179, 2014.

[5] N. Sugimura, R. Shrestha, and J. Inoue, "Integrated process planning and scheduling in holonic manufacturing systems-Optimization based on shop time and machining cost," in *Assembly and Task Planning, 2003. Proceedings of the IEEE International Symposium on*, 2003, pp. 36–41.

[6] P. Tsarouchi, S. Makris, and G. Chryssolouris, "Human–robot interaction review and challenges on task planning and programming," *Int. J. Comput. Integr. Manuf.*, vol. 29, no. 8, pp. 916–931, Aug. 2016.

[7] T. Kruse, A. K. Pandey, R. Alami, and A. Kirsch, "Human-aware robot navigation: A survey," *Rob. Auton. Syst.*, vol. 61, no. 12, pp. 1726–1743, 2013.

[8] S. Calinon, Z. Li, T. Alizadeh, N. G. Tsagarakis, and D. G. Caldwell, "Statistical dynamical systems for skills acquisition in humanoids," *IEEE-RAS Int. Conf. Humanoid Robot.*, pp. 323–329, 2012.

[9] L. Rozo, S. Calinon, D. Caldwell, P. Jimenez, C. Torras, and P. Jiménez, "Learning Collaborative Impedance-based Robot Behaviors," *AAAI Conference on Artificial Intelligence*. pp. 1422–1428, 2013.

[10] J. Mainprice and D. Berenson, "Human-robot collaborative manipulation planning using early prediction of human motion," in *Intelligent Robots and Systems (IROS), 2013 IEEE/RSJ International Conference on*, 2013, pp. 299–306.

[11] L. Johannsmeier and S. Haddadin, "A Hierarchical Human-Robot Interaction-Planning Framework for Task Allocation in Collaborative Industrial Assembly Processes," *IEEE Robot. Autom. Lett.*, vol. 2, no. 1, pp. 41–48, 2017.

[12] H. Wang, N. Wang, and D.-Y. Yeung, "Collaborative deep learning for recommender systems," in *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2015, pp. 1235–1244.

[13] S. Kumra and C. Kanan, "Robotic Grasp Detection using Deep Convolutional Neural Networks," 2016.

[14] F. Zhang, J. Leitner, M. Milford, B. Upcroft, and P. Corke, "Towards vision-based deep reinforcement learning for robotic motion control," *arXiv Prepr. arXiv1511.03791*, 2015.

[15] T. P. Lillicrap *et al.*, "Continuous control with deep reinforcement learning," *arXiv Prepr. arXiv1509.02971*, 2015.