Robotics and Computer Integrated Manufacturing 60 (2019) 34–48

Contents lists available at ScienceDirect

Robotics and Computer Integrated Manufacturing

journal homepage: www.elsevier.com/locate/rcim

A real-time human-robot interaction framework with robust background invariant hand gesture detection

Osama Mazhar*, Benjamin Navarro, Sofiane Ramdani, Robin Passama, Andrea Cherubini

LIRMM, Université de Montpellier, CNRS, Montpellier, France

ARTICLE INFO

Keywords:
Physical human-robot interaction
Safe collaborative robotics
Convolutional neural networks
Real-time vision
Transfer learning

ABSTRACT

In the light of factories of the future, to ensure productive and safe interaction between robot and human coworkers, it is imperative that the robot extracts the essential information of the coworker. We address this by designing a reliable framework for *real-time safe human-robot collaboration*, using static hand gestures and 3D skeleton extraction. OpenPose library is integrated with Microsoft Kinect V2, to obtain a 3D estimation of the human skeleton. With the help of 10 volunteers, we recorded an image dataset of alpha-numeric static hand gestures, taken from the American Sign Language. We named our dataset OpenSign and released it to the community for benchmarking. Inception V3 convolutional neural network is adapted and trained to detect the hand gestures. To augment the data for training the hand gesture detector, we use OpenPose to localize the hands in the dataset images and segment the backgrounds of hand images, by exploiting the Kinect V2 depth map. Then, the backgrounds are substituted with random patterns and indoor architecture templates. Fine-tuning of Inception V3 is performed in three phases, to achieve validation accuracy of 99.1% and test accuracy of 98.9%. An asynchronous integration of image acquisition and hand gesture detection is performed to ensure real-time detection of hand gestures. Finally, the proposed framework is integrated in our physical human-robot interaction library OpenPHRI. This integration complements OpenPHRI by providing successful implementation of the ISO/TS 15066 safety standards for "safety rated monitored stop" and "speed and separation monitoring" collaborative modes. We validate the performance of the proposed framework through a complete teaching by demonstration experiment with a robotic manipulator.

1. Introduction

The advent of Industry 4.0, as a modern trend of automation and data exchange in the manufacturing industry, has proposed the concept of smart factories of the future [1]. This evolving industry demands a more effective and involved collaboration between humans and robots, where each partner can constructively utilize the strengths of the others to increase productivity and work quality [2].

Safety of the human coworkers and an efficacious interaction between humans and robots are key factors of success in such an industrial setting [3,4]. To ensure safety, the ability of the robot to detect an external force, differentiate between intended and accidental forces and to adapt to the rapidity of the human coworker is essential [5]. Nevertheless, the sense of vision is also imperative for modern collaborative robots to monitor the behavior and actions of their human coworkers for communicating or preventing accidents [6].

Generally, robots are designed and programmed to perform specialized tasks. Hence, it is difficult for an unskilled worker to reprogram the robot for a new task [7]. The traditional robot teaching methods are tedious, non-intuitive and time consuming. Multi-modal interfaces that include vision-based gesture detection frameworks, constitute instances of natural and tangible user interfaces (NUIs and TUIs). NUIs exploit the user's pre-existing knowledge and actions – related to daily practices – to offer natural and realistic interactions. This allows humans to directly interact with robots through voice, gestures, touch and motion tracking rather than instructing them the same by typing commands [8]. In many industrial settings, communication through speech is not appreciated because of the interference produced by machines operations. The conventional use of teach pendants is itself too complicated for new users to learn. Portable devices are always required to be charged almost on daily-basis and may also have complex menu trees or networking problems in the interaction software. Manoeuvring the robot to specific target locations by hand, in physical human-robot interactions like in teach-by-demonstration applications, is the most intuitive way of interaction. To unburden the human coworker from carrying any extra device while s/he manoeuvres the robot with her/his

* Corresponding author.
E-mail address: osamazhar@yahoo.com (O. Mazhar).

https://doi.org/10.1016/j.rcim.2019.05.008
Received 15 June 2018; Received in revised form 9 May 2019; Accepted 10 May 2019
Available online 30 May 2019

hands, gestures are considered to be natural and intuitive ways to communicate/interact with the robot [9].

Hence, in this paper we propose a real-time robust and background independent hand gesture detection module based on transfer learning [10] with convolutional neural networks. The intuitiveness of our system comes from the fact that the human does not need to wear any specific suits (Motion capture suits or inertial sensors) nor to carry a specialized remote control or learn complicated teach pendant commands. Such additional burdens would make the interaction unnatural [11].

We integrate the proposed hand gesture detection module with our physical human-robot interaction library OpenPHRI [12] for robot control. On one hand, this ensures safety of the human coworker – by complementing the standard collaborative modes in OpenPHRI – while on the other provides a natural means for robot programming and reprogramming, through hand gestures.

Background and related work are described in Section 2. We summarize our contributions in Section 3, while Skeleton extraction and hand localization are detailed in Section 4. We describe our convolutional neural network for hand gesture detection in Section 5, while OpenPHRI integration and collaborative modes imposed by safety standards are briefly explained in Section 6. The robotic framework and example industrial application of the proposed framework are presented in Section 6. We conclude in Section 7.

2. Background and related work

The authors of [1] present the emerging concept of cyber-physical structure, which will employ extensive automation and self-organization of machines and component parts in complex manufacturing scenarios, using different sensor modalities. The primary role of human workers will be to dictate a production strategy and to supervise its implementation by the robotic systems.

To achieve this goal, safe Human-Robot Interaction is crucial. According to [13], collaboration can be achieved only when safety is first guaranteed. We will hereby survey the state of art with a similar perspective. We will review the literature on safety in collaborative robotics, gesture detection for interaction and sign language detection relevant to our research.

2.1. Safety in collaborative robotics

A recent survey on human-robot collaboration in industrial settings is presented in [8]. The authors talk about safety citing several ISO standards, discuss intuitive interfaces for robot programming/collaboration and explore different industrial applications of human-robot collaboration. With regards to safety, they recall the four collaborative modes from ISO 10218-1/2 and ISO TS 15066 [14–16]: “Safety-rated monitored stop”, “Hand guiding”, “Speed and separation monitoring” and “Power and force limitation”. Since in this work we addressed the first and third, let us now focus on works related to these modes.

In [17], the authors present a tire workshop assistant robot. SICK S300 laser sensors are utilized for navigation, obstacle avoidance and human detection. The authors define three areas surrounding the robot namely “Safe area”, “Collaboration area” and “Forbidden area”. The main disadvantage of this technology [18] is that several thousands of reflective landmarks are required for reliable navigation of the robot in a cluttered environment.

The authors of [19] thoroughly discuss several aspects of *speed and separation monitoring* in collaborative robot workcells. They analyze laser-based human tracking systems. The human coworkers are detected through centroid estimation of the detected objects and as the authors state, this varies based on the motion of legs, shifting of clothes, and sensor noise. The authors emphasize the technological advancement of safety-rated cameras and on-robot sensing hardware for enabling speed and separation monitoring in unstructured environments. Moreover, the importance of human-specific identification and localization methods is discussed for reliable physical human robot collaboration.

In [20], the authors present the preliminary results of their research on sensor-less radio human localization to enable *speed and separation monitoring*. A wireless device-free radio localization method is adopted with several nodes connected in mesh configuration, non-regularly spread over a large indoor area, so that the human-operator being localized does not need to carry an active wireless device. The concept of user tracking in wireless sensor networks is extended in [21]. This study considers the availability of the source attached to the human coworker’s body in the industrial scenario.

The idea of trajectory dependent safety distances is proposed in [22] to attain dynamic *speed and separation monitoring*. This method avoids fixed extra safety clearances and is optimized with respect to the functional task at hand.

Alternative sensing modalities for *speed and separation monitoring* include motion capture systems [23] and vision based depth cameras [13,24]. In this regard, [18] compares structured light depth cameras and stereo-vision cameras for mobile robot localization in the industry.

As all these works highlight, an advantage of vision over other sensors is that it does not require structuring the environment and/or operator. Furthermore, it is generally more rich, portable (a fundamental feature for mobile robots) and low-cost, even when depth is also measured by the sensor (as with Microsoft Kinect). While at present Kinect is far from being certifiable for safety, we are confident that in the near future similar RGB + D sensors will. For all these reasons, in this work we have decided to use RGB + D vision for addressing *safety-rated monitored stops* and *speed and separation monitoring*. As in [19], we adopt the idea of human-specific localization to effectively identify the presence of humans in cluttered environments. Our contributions on safety will be detailed in the subsequent section after reviewing literature on gesture detection in human-robot interaction.

2.2. Gesture detection in human-robot interaction

Once safety is guaranteed, collaboration is possible. To this end, researchers have proposed to use body gestures for communicating with the robot. The literature on gesture detection in human-robot interaction scenarios is enormous. Here, we focus on works that are related to the idea we propose, by relying mainly on RGB + D sensing.

A task-oriented intuitive programming procedure is presented in [25] to demonstrate human-like behavior of a dual-arm robot. The authors decompose complex activities in simpler tasks that are performed through task-oriented programming where the focus is given to “what to be done, rather than how to do it”. Moreover, through the development of intuitive human interfaces, high level commands are transferred to a sequence of robot motion and actions. For human-robot interaction, the authors use Kinect V1 [26] to extract human skeletal coordinates for gesture detection, and the built-in microphone array of Kinect V1 to detect the oral commands. Whole body gestures (extended arms) are used to achieve robot motion in a dashboard assembly case. Although the idea of task decomposition and controlling the robot through human gestures is beneficial, the considered gestures, as in [27], are counterintuitive and tiring.

The authors of [28] present methods for obtaining human worker posture in a human-robot collaboration task of abrasive blasting. They compare the performance of three depth cameras, namely Microsoft Kinect V1, Microsoft Kinect V2 [29] and Intel RealSense R200 [30]. Kinect V1 uses a structured light approach to estimate the depth map, Kinect V2 is a time-of-flight sensor, while RealSense R200 has a stereoscopic infra-red setting to produce depth. In the blasting process, the abrasives are suspended in the air or fill the surrounding environment, and significantly decrease the scene visibility. The use of image-based methods to extract human worker pose is challenging in such environments. The experimental observations suggest that Kinect V1

performs best in the real blasting environment, although no concrete reason could explain this. They also present a novel camera rig with an array of four Kinect V1 to cover a 180° horizontal field of view. The use of Kinect V1, to extract human pose for a marker-less robot control method is also presented in [31].

In [32], the authors present an online robot teaching method that fuses speech and gesture information using text. Kinect V2 localizes the position of hands in the scene, while their orientation is measured by an inertial measurement unit. The gesture and speech data are first converted into a description text, then a text understanding method converts the text to robot commands. The proposed method is validated by performing a peg-in-hole experiment, placing wire-cut shapes, and an irregular trajectory following task.

To ensure safe interaction, [33,34] proposes a virtual reality training system for human-robot collaboration. A virtual game simulation is developed for real-time collaboration between industrial robotic manipulators and humans. A realistic virtual human body, including a simple first person shooter view, simulates the user's vision. A head mount display and a Kinect V1 track the human head and skeleton pose respectively. Several interaction tasks are accomplished. These include selection of objects, manipulation, navigation and robot control. This technique is useful to establish the acceptability of a collaborative robot among humans in a shared workspace, as well as to tackle mental safety issues.

In [7], the authors present a strategy to use speech and a Wii controller to program a Motoman HP6 industrial robot. This helps workers, with no knowledge of typical programming languages, in teaching different activities to the robot. A neural network is trained to recognize hand gestures using features extracted from the accelerometer output of the Wii-controller. In [35], the authors train artificial neural networks to classify 25 static and 10 dynamic gestures to control an industrial 5 degrees-of-freedom robotic arm. A data glove, CyberGlove II, and a magnetic tracker, Polhemus Liberty, are used to extract a total of 26 degrees-of-freedom.

The authors of [5] present a study for measuring trust of human coworkers in fence-less human-robot collaboration for industrial robotic applications. To ensure safety of the human coworkers, it is essential to equip the robot with vision sensors, so that it can understand the environment and adapt to the worker's behavior. The paper also discusses the use of RGB + D cameras to detect pointing gestures and proximity monitoring for safety using the depth information. In [36], authors use human gestures to navigate a wheeled robot through pointing gestures directed on the floor. The interaction scheme also includes detection of facial gestures which often fails, as stated by the authors, because the untrained users make those gestures subtly.

In [37], the authors propose object recognition through 3D gestures using Kinect V1. They exploit the depth information from Kinect V1 to subtract the object background. This strategy often fails if predefined environmental assumptions are not met. Moreover, a histogram matching algorithm is used to recognize the objects placed on a white color table. Such techniques have recently been outperformed by modern deep learning ones like convolutional neural networks [38]. The authors of [39] propose a human-robot interaction system for the navigation of a mobile robot using Kinect V1. The point cloud acquired from Kinect V1 is fit on a skeleton topology with multiple nodes, to extract the human operator pose. This technique is not reliable to obtain the skeletal pose unless the human body non-linear anatomical constraints are modeled in the design of the skeleton topology.

According to [40], sign language is among the most structured set of gestures. Hence, in our work, we proposed the use of American Sign Language (ASL) for communicating with the robot. In the following subsection, we discuss previous works on sign language detection.

2.3. Sign language detection

Hand gesture detection techniques can be mainly divided into two categories: electronic/glove-based systems and vision-based methods. Some researchers prefer the use of wearable sensors to deal with occlusions or varying light conditions [41]. These sensors are expensive, counterintuitive and limit the operator's dexterity in his/her routine tasks. The vision-based methods can be further divided into two categories; (a) methods that use markers and (b) pure vision-based methods [42]. Since pure vision-based methods do not require the users to wear any data-gloves or markers, they offer ease-of-use for the operators to interact with the robots/machines. Furthermore, in Section 2.1 we have highlighted the advantage of using vision for safety monitoring. For these two reasons, here we opt for a pure vision-based method and review only works on vision-based sign language detection.

Early research on purely vision-based methods for ASL recognition dates back to 1992 [43]. In this work, the authors use motion detection to capture start/stop instances of the sign/gesture, hand location tracking to record the trajectory of the gesture, trajectory shape (using curve eccentricity) and detection of hand shapes at each stop position. The hand shapes are classified using the Hough Transform method described in [44]. The authors in [42] utilize a similar method as in [43]. It consists of a Canny filter that detects the hand edges in the scene, followed by a Hough Transform that extracts the feature vector of size 200. A neural network is then devised to classify the hand gestures. The dataset used to train the neural network is extremely small and it is assumed that the image background is uniform. The authors do not mention the hands' localization in the scene during the recognition phase. Thus, it is assumed that the system only works if the hand appears in a specific region of the image.

One of the initial works in detecting ASL gestures through Hidden Markov models is discussed in [40]. The authors propose two settings in this research i.e., the second person view (desk based recognizer) and the first person view (wearable based recognizer). The proposed system recognizes sentences of the form "personal pronoun, verb, noun and adjective, pronoun" generated through 40 randomly chosen words. In both systems, videos were recorded and analyzed offline for ASL translation. An a priori model of the skin color is used to segment hands in the scene, while the absolute positions of the detected blobs in the image are used to distinguish left and right hand. The use of absolute positions of the hands in addition to a cumbersome wearable camera and computer system on the head significantly constrain the movement of the "signer" in the scene.

Recently in [45], researchers proposed an ASL translation system using binary hand images by keeping the edge information in the image intact. They use an image cross-correlation method to identify the signs by comparison with gesture images in a database. A similar hand gesture detector based on binary images is proposed in [46]. The author proposes a color-independent (using preprocessed binary hand images) hand gesture detector that relies on a convolutional neural network (CNN), inspired by LeNet [47]. The classification accuracy of such system depends largely on the preprocessing steps of image segmentation performed with color or intensity thresholding, while CNNs are inherently able of learning color features robustly, as shown in [48]. Such systems also normally require a plain or white background for hand segmentation, which is hard to obtain in realistic human-robot interaction scenarios.

The use of depth cameras is becoming increasingly popular in applications like hand gestures detection or sign language translation. A thorough survey on 3D hand gesture recognition is presented in [49]. The depth information from such sensors can be used to segment the hands in cluttered backgrounds, by setting a depth threshold, while the normalized depth image of the hand adds the information for correct classification of the hand gestures [50]. The accuracy of such techniques depends on the range and resolution of the depth sensors. Nevertheless, the use of depth sensors is beneficial, since it aids the detection of fine-grained hand gestures [51]. Latest works in deep learning have allowed the extraction of 2D hand skeletons from conventional RGB images [52,53]. This can be used as a basis to fit a 3D

kinematic hand model through an appropriate optimization technique as described in [54], thus eliminating the need of depth sensors for this purpose.

In recent years, the idea of deep learning has made a concrete impact on computer vision research and has been reported to even surpass human-level performance in image classification [48]. Hence, in our work we chose to exploit convolutional neural networks to recognize static hand gestures. We localize and crop the image regions containing the hands, by exploiting the data from our integration of Kinect V2 with a state-of-the-art 2D skeleton extractor library. Then, we perform background substitution and image processing operations (e.g., histogram equalization, introduction of salt and pepper noise etc.) on the cropped hand images to increase data variance, before training the CNN. This allows the network to learn robust hand features, by avoiding time-consuming rigid image processing methods during the recognition phase.

3. Our contributions

This paper is an extension of our previous work proposed in [55] which presented a tool handover task between robot and human coworker through static hand gestures. A convolutional neural network, inspired mainly by LeNet [47] was developed, to classify four hand gestures. The aim of the previous work was to build a robust hand gesture detection system. However, the dataset was small, and the hand images were recorded only by one individual. This could not guarantee correct detection of hand gestures made by other individuals and with backgrounds having rich textures.

We extend our work by training a hand gesture detector on ten gestures instead of four presented in [55]. Moreover, the backgrounds are now replaced with random pattern/indoor-architecture images to make the detection robust and background invariant. The vision pipeline is then integrated with OpenPHRI [12] to complement the library with two collaborative modes of the ISO/TS 15066 safety standards. This integration is detailed in Section 6. We propose an interaction setting where a human coworker can safely instruct commands to the robot via gestures. In summary the contributions of this paper are the following:

- Development of a real-time hand gesture detection framework that localizes hands through asynchronous integration of OpenPose 2D skeleton detector and classifies hand-gestures at frame-rate of approximately 20fps.
- Integration of Kinect V2 depth map with the obtained 2D skeleton to get a pseudo 3D skeleton, which is used for “speed and separation monitoring” to ensure the safety of the human coworker.
- Training a background-invariant hand-gesture detection system through transfer learning from Inception V3 convolutional neural network.
- On-line release of hand gestures database of Kinect V2 recordings for benchmarking and comparison.
- Integration of the developed hand gesture detection module with our safe physical human robot interaction framework, namely OpenPHRI.
- Validation of the proposed framework for robot teaching and control of Kuka LWR 4+ arm with the detected hand-gestures.

The overall pipeline of the proposed framework is illustrated in Fig. 1. Each named box is a cyclic process and dotted arrows represent asynchronous communications between these processes. Each process is described in the following sections in detail.

4. Skeleton extraction and hand localization

For safe Human-Robot Interaction, it is essential for the robot to understand its environment, particularly the human coworker. In this research, we opted for Microsoft Kinect V2 as the main sensor to capture the visual information of the human coworker. Kinect V2 is a time-of-flight sensor and provides a larger field-of-view and higher resolution RGB and depth images than its predecessor Kinect V1. This allows the robot to extract functional information from the scene, like human (s) presence or object/obstacle detection, including depth perception.

4.1. Skeleton extraction module

In our work we utilize OpenPose [52,53], to extract skeletal joint coordinates, as in [56,57]. This library returns 2D skeletal coordinates (x_i, y_i, c_i), for $i = 1, \ldots, 18$, from a RGB image, using confidence maps and parts affinity fields in a multi-person scene; x_i and y_i are the abscissas and ordinates respectively of 18 COCO body parts [58], while c_i represent their confidence measure. OpenPose works on the principle of “convolutional pose machines” described in [52].

OpenPose is a robust skeleton extractor and is not trained on predefined body poses. It extracts each joint independently from the overall body pose. Hence, it is preferred over libraries like OpenNI and Microsoft SDK as they are often not accurate in skeleton extraction, require initialization pose and constraint the user to face the sensor. For real-time skeleton extraction, this method requires a multi-GPU hardware with the output frame-rate mainly dependent on the number of persons appearing in the scene. The average frame rate that is achievable using two Nvidia GeForce GTX 1080 on full-HD Kinect V2 RGB image is approximately 14 fps. Since we currently employ only one GPU in our framework, we obtain 6 fps with 1 person in the scene.

4.2. Image acquisition and hand localization module

The strategy to localize human body and its sub-parts (i.e., hands or face) depends mainly on the output of the sensor of choice. In [59] the authors use skin color for hand segmentation using a conventional RGB camera, as in [46]. In [60], human body localization is performed using laser sensors, and its sub-parts are obtained through Kinect with the OpenNI library as in [61]. In [39], the authors localize the human body, inspired by Munaro et al. [62], by merging clusters of the point cloud obtained from the Kinect V1 after voxel filtering and ground plane removal. Lately, infrared based sensors e.g., Leap Motion, are developed to track fingers of a hand in the near proximity (within 25 to 600 mm) of the sensor. However this range is too close for our application. In [54], authors adapt a state-of-the-art object detection deep learning technique namely YOLOv2, adapted to localize hands and head/face of a person in a scene. The authors have utilized OpenPose to first extract hands and face images from recorded videos with human activity, and then used these images to train YOLOv2 to detect hands and the face of the person in the scene in real-time. The face is detected to differentiate left hand from the right one. This is an efficient method to detect hands in the scene in real-time but requires a separate training/adaptation of YOLOv2 for hands and faces.

In our research, since we obtain the skeletal joint coordinates from OpenPose, training a separate hand detector to localize hands is not required. To estimate the hands position, we fit a line between the elbow joint and the wrist joint returned by OpenPose and extend this line to one-third of its original length (empirical value) in the direction of hand to approximately reach the center of hand. A bounding box is then centered at the approximated hand center at an angle which the forearm makes with the horizontal. This makes the hand image acquisition rotation invariant. The size of the bounding box is determined by the mean depth value of a 6×6 matrix centered at the wrist joint, obtained through Kinect V2 depth map as shown in Fig. 2. The hand images are cropped with reference to the tilted bounding box, re-scaled to size 224×224 pixels and rotated again such that the cropped image becomes vertical.

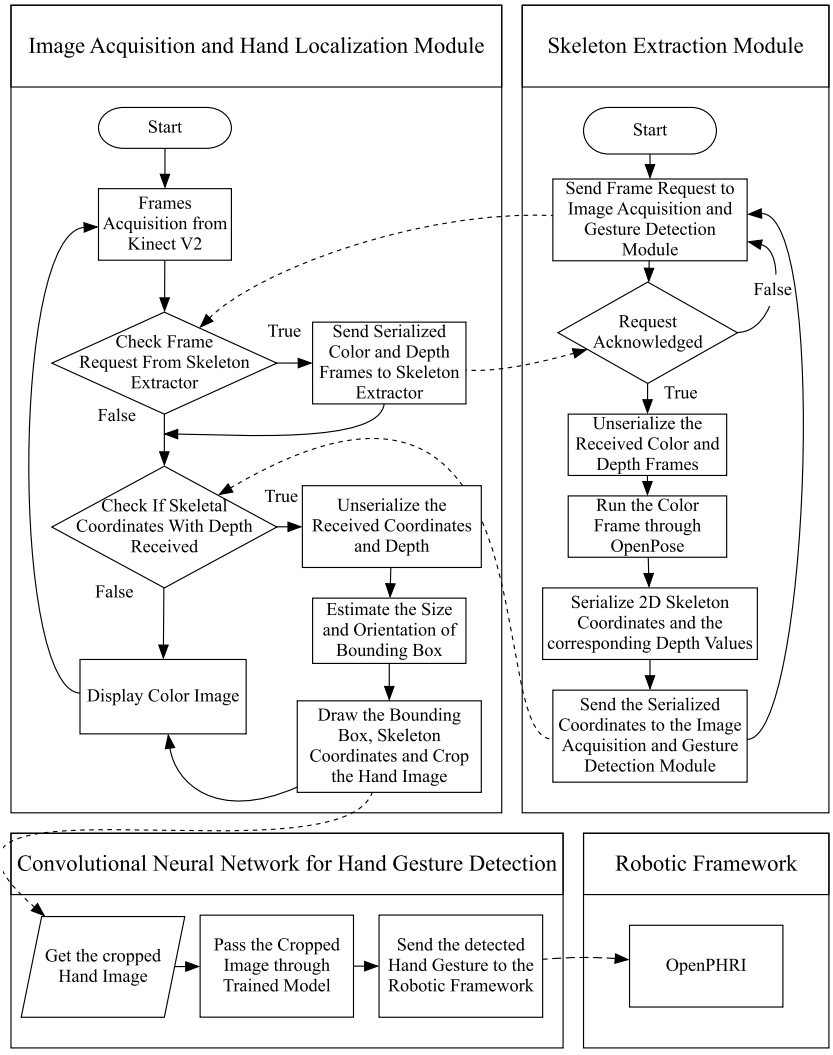


Fig. 1. The overall pipeline of our framework for pHRI using hand gestures.

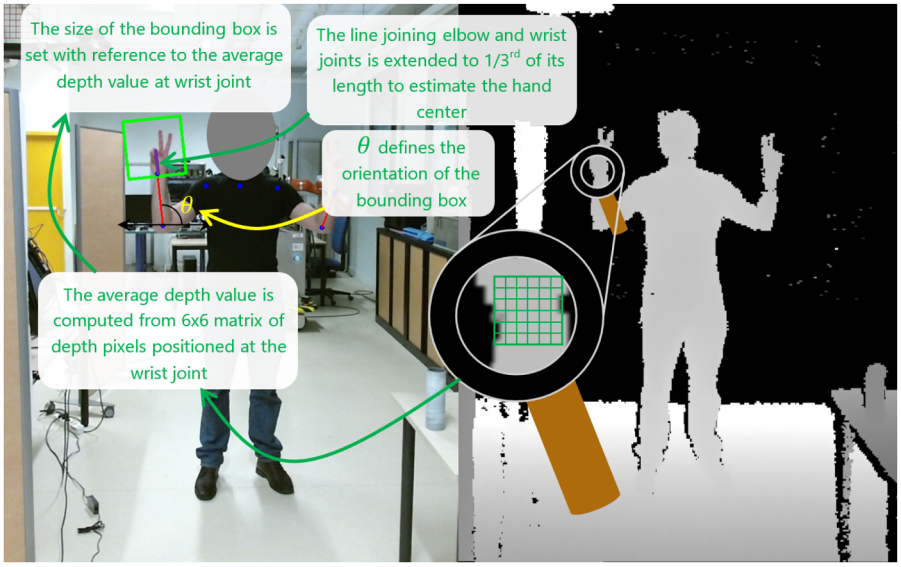


Fig. 2. Localization of hand through OpenPose is illustrated. The bounding box is titled with an angle that the forearm makes with horizontal, while the size of bounding box is determined by the mean depth value of the wrist joint. The mean depth value is computed by averaging the depth pixel values of a 6 × 6 matrix centered at the wrist joint.

4.3. Asynchronous integration of the modules

In our previous work [55], we integrated OpenPose with gesture recognition sequentially to obtain an overall temporal resolution of approximately 4 fps. In this work, an inter-process distributed system is designed through nanomsg socket library[1] which has drastically increased the frame rate of the vision pipeline. The afore-mentioned inter-process distributed system works using a "request-reply" communication pattern, known as scalability protocol. Furthermore, it ensures that no frames are lost during communication. Fig. 1 illustrates this asynchronous communication between the proposed framework modules via dotted lines. The image acquisition and hand localization module retrieves the image stream from Kinect V2 and checks if a frame request has arrived from the skeleton extraction module. When a frame request is received, the current RGB and depth image are first serialized through flatbuffers[2] and then passed to the skeleton extraction module. The skeleton extraction module unserializes the received frames with flatbuffers and then pass the RGB image through the forward-pass of OpenPose which returns a vector of 2D skeleton coordinates (x_i, y_i, c_i). The calculated mean depth values, as described in the previous section, are concatenated with the 2D skeleton coordinates and this 3D vector (x_i, y_i, d_i) is then sent to the image acquisition and hand localization module. The integration of Kinect V2 depth map with the 2D skeleton coordinates from OpenPose however does not represent the actual 3D coordinates of the joints and represents the surface depth value of the joints. There is a possibility that a joint is occluded in the scene by an object or the body itself. To prevent false detection of depth hence preventing potential accident, we use the confidence measure for each joint returned by OpenPose. The depth value of each joint is only updated if $c_i > 0.5$ (this is an empirical value), otherwise the previous depth value is kept. The image acquisition and hand localization module expects to receive coordinates from skeleton extraction module in each execution cycle. Once the coordinates are received, the hand is segmented and cropped image (described in Section 4.2) runs through the forward-pass of trained convolutional neural network for hand gesture detection. The detected hand gesture label is sent to the robot controller running OpenPHRI to pilot the experiment. The overall frame rate of our gesture detection pipeline is approximately 20 fps while the skeleton is extracted and the hand location is updated at around 5 fps. This significantly improves the execution performance of the vision system as compared to that in [55], which finally leads to a system that better reacts to human commands.

5. Convolutional neural network for hand gestures detection

In our previous work [55], we designed the CNN architecture for hand images with relatively plain backgrounds, while the number of gestures were set to 4 and the gestures were recorded by a single person. In this research, we used 10 static hand gestures recorded by 10 volunteers (8 males and 2 females) of age 22 to 35 and the backgrounds of the hand images are substituted with random pattern and indoor architecture images (explained in Section 5.2). This makes the recognition problem more complex as compared to the one presented in [55], where only 1 volunteer and 4 gestures had been considered. Therefore we opted for transfer learning for gesture recognition, exploiting state-of-the-art CNNs pre-trained on large image data from the ImageNet Large Scale Visual Recognition Challenge (ILSVRC) [63]. In particular, Inception V3 [64] which is state-of-the-art in image classification for 1000 classes, is adapted for our background-independent hand-gesture recognition task. Inception V3 is available in Keras python library [65] with pre-trained weights. Fig. 3 shows samples of the static gestures we trained our framework on. The gestures include 9 letters/numbers taken from ASL [66] and a **None** gesture that is not among these nine. The letters/numbers are chosen such that they resemble with each other (like F, 7 and W; A, L and Y) so as to challenge the training and ensuring robustness of the CNN. The **None** gesture is important to determine if the person does not intend to interact with the robot. Our system also generates this label when the line joining the elbow and wrist joint (forearm) of the user is too low (in the lower two quadrants of the axes centered at the elbow joint). In this case, the robot controller ignores this command and does not initiate any action (which is likely undesired, since the user's hand is low). We could have excluded the **None** gesture from the trained network, by applying a predefined threshold to the nine (one per class) network scores. However, since the relative scores of the ten classes vary a lot according to the operating conditions, it is not possible to fix a priori such threshold.

5.1. Preparation of dataset/dataset recordings

To create a dataset for gesture recognition and off-line development, RGB and depth image streams from Kinect V2 are saved in the local workstation. The frames are saved with an approximate frame rate of 20 fps. Each gesture is recorded by each volunteer for around 12 s with both hands (see Fig. 4), at three distances of 5, 3 and 1.5 m away from the sensor. The depth information near Kinect V2 is rich and accurate, thus the images recorded at the distance of 1.5 m are used for the fine-tuning of Inception V3 (discussed in Section 5.3). However, since the network is trained only on RGB images, the hand gestures can also be recognized at other distances. We are releasing our dataset OpenSign[3] online. OpenSign contains RGB and depth (registered) frames of volunteers recording 10 gestures. The RGB images are saved in *png* format, while the float data of the depth images are saved in *bin* files. The total number of images in our dataset is 20950. These include 8,646 original images, and 12,304 synthetic images obtained by substituting the background with the technique that we will explain in Section 5.2.

We divide the dataset of 20,950 images with a ratio of 3:1:1, i.e., 12,570 train images and 4,190 images each for cross validation and test. Train images go through extensive preprocessing (explained in Section 5.2), while only selective preprocessing operations are applied to cross-validation images to keep them near those obtained during recognition in the robotic interaction experiments.

5.2. Background substitution and preprocessing of the hand images

Background substitution is performed so the network is trained to detect hand gestures independently from the background. We use nearly 1100 images of random pattern and indoor architectures which are freely available on the internet[4]. The background substitution process is illustrated in Fig. 5. A binary mask for background substitution is created using the depth information from Kinect V2. All the pixels that lie at distance within $\pm$ 18 % (empirical value) of the mean depth value computed at the wrist joint (obtained through OpenPose) are set to 1, while the rest are zeroed.

This binary mask is broadcasted into three channels and then multiplied by the cropped RGB hand image to get a background subtracted hand. An inverted mask is also created by simply applying a "NOT" operation on the mask originally computed. The background pattern images are cropped to multiple 224 $\times$ 224 sized images (as it is the set size of hand images) which are subsequently multiplied by the inverted hand mask. The hand image with subtracted background and the pattern images multiplied with the inverted binary mask are then added in the final step of background substitution. Fig. 6 shows the samples of gestures with original and substituted backgrounds. As discussed in Section 5.1, all the training images (images with substituted

[1] https://nanomsg.org
[2] https://google.github.io/flatbuffers.
[3] http://dx.doi.org/10.17632/k793ybxx7t.1
[4] https://pixabay.com/

Fig. 3. Samples of the gestures considered for training. The labels represent the letters and the numbers taken from American Sign Language. The last gesture is one of the several **None** gestures included in the training set.

Fig. 4. A volunteer recording '7' gesture in the laboratory.

and original backgrounds) go through several preprocessing steps. Image processing operations of histogram equalization and introduction of Gaussian and salt and pepper noise are applied on 30% of training images each while the remaining 10% are left unprocessed. Fig. 7 shows random samples of original and processed images after the addition of Gaussian noise and histogram equalization. For robust gesture detection, we also use the real-time data augmentation feature of Keras library.

Keras real-time data augmentation is designed to be iterated by the model fitting process, creating augmented image data in defined batch size during training. This reduces the memory overhead of the computer but adds additional time cost during model training. The image processing operations that are applied on all training images (after the addition of noise and histogram equalization as discussed above) using the Keras library include channel shift, zoom, shearing, rotation, axes flip and position shift.

The batch size for model fitting is set to 100 training images. These transformations are applied in real-time during model training. So the number of train images remains the same while each batch for training

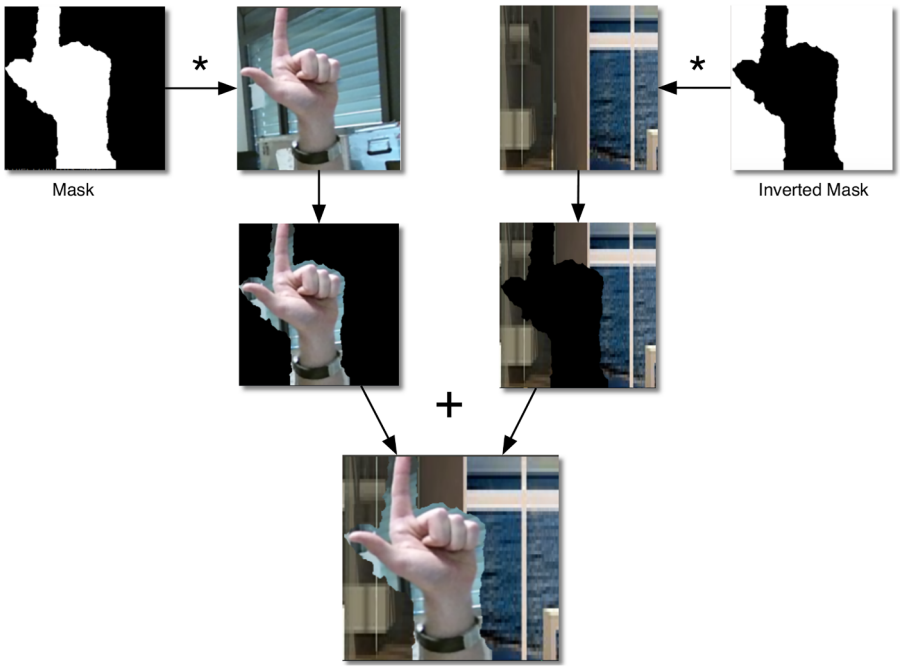


Fig. 5. The process of background substitution.

Fig. 6. Samples of hand gesture images with original (labeled images) and substituted backgrounds (below originals). Note the remnants of the original backgrounds. This phenomenon is due to dilation of the binary masks. While it could be avoided by using techniques like chroma key, we do not intend to use a uniform background, to avoid bringing any extra apparatus in operation. In the experimental results (Section 5.4), we show that despite these remnants, gesture detection is highly accurate.

is applied with selected – yet randomly chosen – transforms. In Fig. 8, we show samples of processed training images with Keras being passed to the CNN.

5.3. Adapting Inception V3 to gesture recognition

In image classification problems, the input data i.e., an image, is formed by low-level edges, curves and color combinations irrespective of the type of object that the image represents. It is therefore assumed that the early layers in the pre-trained state-of-the-art networks have learned to efficiently extract those features from the images thus they need to be preserved. Inception V3 is trained to recognize 1000 classes of objects as explained in Section 5. To adapt Inception V3 to classify only 10 gestures, the last softmax activation layer of this network with 1000 neurons should be replaced with a new layer of 10 neurons. As implemented in Keras, the Inception V3 has 10 trainable inception blocks. We perform training in three phases. In the first phase all the layers (hence inception blocks) in the network are frozen with the exception of the new layer added and the CNN is trained for 10 epochs only. This fine-tune the weights of the new layer exploiting the knowledge of all pre-trained inception blocks. Then we unfreeze last two inception blocks and trained the CNN for 10 epochs, and then we trained top four inception blocks so the network is fine-tuned properly on our dataset. This gradual unfreezing of inception blocks prevents damaging the pre-trained weights and thus avert over-fitting.

The validation set is used to chose the best performing weights and then the network is tested on the unseen test set to quantify/estimate the accuracy of the selected weights. Fig. 9 illustrates the training curve of validation accuracy and loss of our dataset. Each epoch took approximately 130 s to pass and the network was able to achieve validation accuracy of 99.12% at 745th epoch taking around 27 hours of training.

5.4. Quantification of the trained CNN

To validate and quantify the results even further, the accuracy of the trained CNN is tested with a test set of 4190 new images. The overall test accuracy of the trained CNN is found to be 98.9% on test set. The normalized confusion matrix in Fig. 10 shows the accuracy of each gestures and misinterpretation of one gesture against the others. It can be observed that despite 94.3% accuracy of the **None** gesture, it was misinterpreted the most among all. The reason for this lower accuracy is that the **None** gesture defines all gestures that do not appear like the other 9 as well as all transitional gestures.

It is difficult to include all the transitional gesture possible to be classified as **None** gesture. Moreover, it can be observed from a close inspection of the test results that the CNN is very accurate in identifying a gesture as **None** when a person is holding an object in his hand. It is inferred that if the CNN is additionally trained on a gesture like “an object in hand”, this gesture will be easily distinguished. Meanwhile, this misinterpretation can be dealt by adding a software constraint, as explained in Section 5, of not invoking gesture detector until the arm is in the upper two quadrants of the axes centered at the elbow joint of the person, as we did in [55]. We release the source code of our hand gesture detection

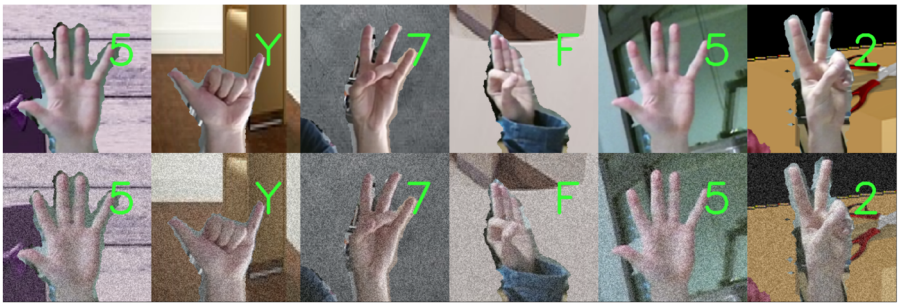
6. OpenPHRI integration

To control the robot and to remain safe during human-robot collaboration, we have used OpenPHRI open-source control library. This library allows to describe the task to perform using force and velocity inputs in both the joint and task spaces while enforcing safety constraints such as velocity limitations, speed and separation monitoring or safety-rated monitored stops.

As discussed in Section 2.1, ISO 10218-1/2 and ISO/TS 15066 have imposed safety requirements for industrial robot systems. Moreover, these ISO standards have identified four collaborative modes which are briefly explained as follows:

- *Safety-rated monitored stop* - This states that the human and robot can operate in a shared space but not at the same time. As soon as the human operator occupies the shared space, the robot must stop until the human exits the shared space.
- *Hand guiding* - In this mode, the human coworker can teach the robot

(a) Samples of training images after histogram equalization



(b) Samples of training images after the introduction of Gaussian noise

(c) Samples of training images after the introduction of salt and pepper noise.

Fig. 7. Image processing operations of histogram equalization, introduction of Gaussian and salt and pepper noise are performed on the training images. First row in each sub-image shows unprocessed image while the processed images are shown in the second rows.

Fig. 8. Image processing operations applied to the training images include color-shift, zoom, shear, rotation, axes flip and position shift processes.

positions/waypoints by physically moving the robot without any means of an intermediate interface.

- *Speed and separation monitoring* - This defines three zones of the shared space say red, yellow and green. The operation of robot depends on the presence of human in each zone. If human coworker is in the green zone the robot operates at its full speed, at reduced speed in yellow zone and it should stop in the red zone.
- *Power and force limiting* - This mode prescribes the limitation of power and force to allow humans to work side-by-side with the robot. The robot should be able to handle collisions with the human to prevent any harmful consequences.

OpenPHRI inherently is able to adopt all four collaborative modes efficiently. The first and the third modes however, require safety-rated monitoring sensors. As described in the previous sections, our proposed framework obtains a pseudo 3D human skeleton, which is used to

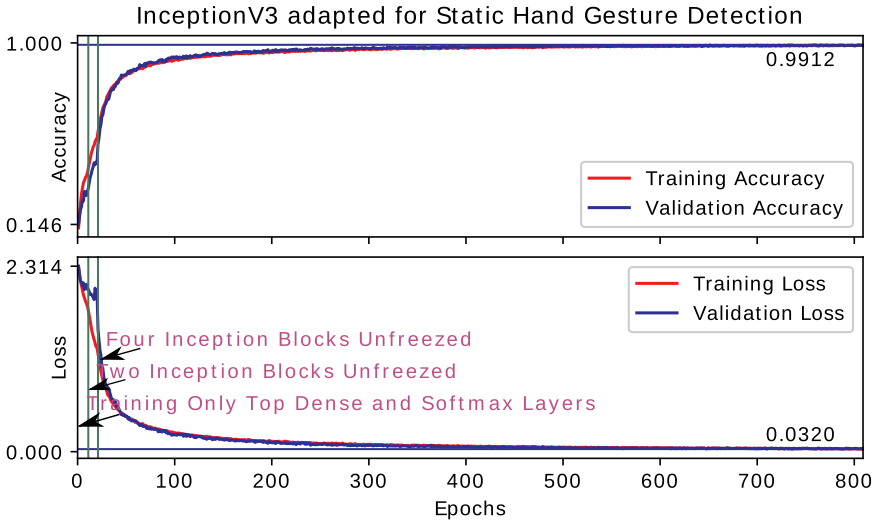


Fig. 9. Plot of validation accuracy (top) and validation loss (bottom).

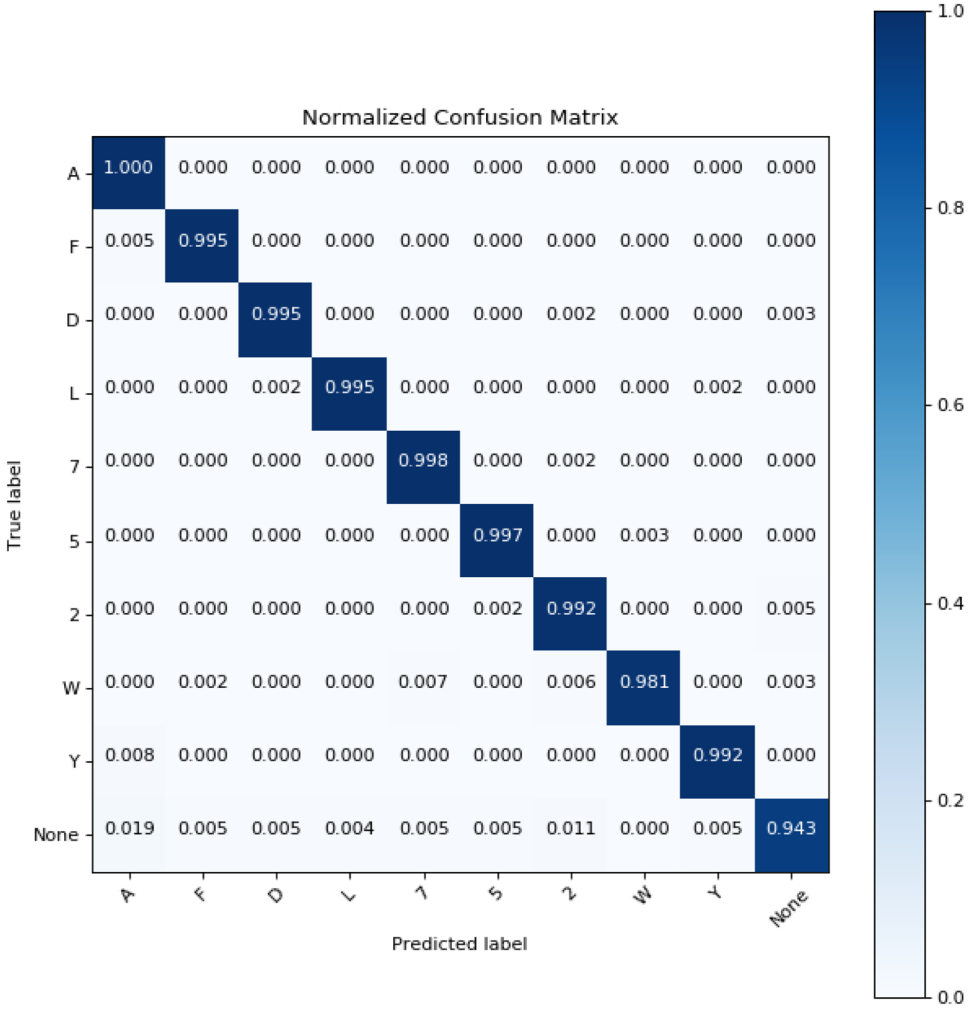


Fig. 10. Normalized confusion matrix quantified on test-set.

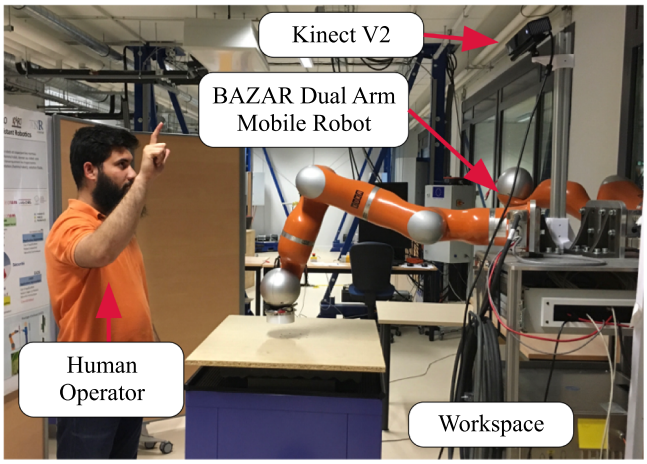


Fig. 11. Safe physical human robot interaction setup.

determine the distance of the closest body part of the human coworker to the robot. This is integrated with OpenPHRI to complement the two collaborative modes.

7. Example industrial application of the proposed framework

To demonstrate the effectiveness of the proposed approached, we set up an industrial-like experiment where multiple operators can safely interact sequentially with the robot using both hand gestures and physical contact. The experiment is decomposed into two phases: (1) a teaching by demonstration phase, where the user manually guides the robot to a set of waypoints and (2) a replay phase, where the robot autonomously goes to every recorded waypoint to perform a given task, here force control.

BAZAR robot used for the experiments is composed of two Kuka LWR 4+ arms with two Shadow Dexterous Hands attached at the end-effectors and a Kinect V2 mounted on top of it [67]. The arms are attached to a Neobotix MP700 omnidirectional mobile platform. In our scenario, shown in Fig. 11 the mobile base is kept fixed and only the left arm, without the hand, is used. The communication with the embedded arm controller is done using the FRI library.[5] The external force applied to the arm's end-effector is estimated by the embedded controller (based on joint torque sensing and on knowledge of the robot's dynamic model) and retrieved by FRI. The control rate is set to 5ms.

To orchestrate the experiment, we have designed a finite state machine (FSM), depicted in Fig. 12. The transitions between the states are either automatic (no text), depending on sensory information (arrow with text) or triggered by gestures (hand sign with text).

A video of the experiment is available online[6] and snapshots are given in Fig. 13. The experiment goes as follow. First, the robot goes to a predefined initial joint configuration before initializing the *Teach* phase. Once this initialization is performed, the robot is ready to be manually guided and taught the waypoints where the tasks have to be performed during the *Replay* phase.

Each time a **Record** gesture (L letter sign) is detected, the current end-effector pose is recorded. When a **Replay** gesture (A letter sign) comes in, the *Teach* phase is ended and the *Replay* phase is initialized. Then, the robot goes to the first recorded waypoint while limiting its velocity thus ensuring safety of the human worker (speed and separation monitoring in the FSM) according to the distance of the closest detected body part. This distance corresponds to the depth value given by Kinect V2 at the joint image coordinates obtained from OpenPose as explained in Section 4.3. If the closest body part is occluded by the robotic arm, the depth value (that will then correspond to the depth value of the robot itself) is discarded while the next closest body part visible in the scene is considered a reference for depth.

This estimation of body parts distance is not available with the default output of OpenPose but it is possible, thanks to our integration, of Kinect V2 depth map. This amplifies the usefulness of OpenPose skeleton extraction while assuring a safe interaction of a human coworker with the robot. While in autonomous motion, the robot can be stopped at any time (Soft Stop constraint in the FSM) using a **Stop** gesture (number 5 sign). Making this gesture will slow down the robot until a full stop is reached. This is useful if an operator must enter the robot workspace without fearing any injury. The **Resume** gesture (Y letter sign) can be made to resume normal operation. When the robot reaches the waypoint, it switches to the task execution. In this scenario the task is to apply a 30N force for 2 s along the vertical axis. Once the task has been executed, the robot goes back to its waypoint and moves to the next ones to repeat the same operations. If the task has been performed at all the waypoints, the *Replay* phase ends and the next action is determined by the operator. A **Reteach** gesture (number 7 sign) will move the FSM to the *Teach* phase while a **Repeat** gesture (F letter sign) will repeat all the tasks at the recorded waypoints. If no other operation is needed, an **End** gesture (number 2 sign) will end the experiment.

Experimental results are show in Fig. 14. The time axis has been limited to the 132–185 s range for better readability. The top graph displays the result of the hand gesture detection where each vertical dashed line corresponds to the detection of a gesture. To filter out false positives, a gesture is considered valid if it appears in five consecutive frames.

Considering the hand-gesture detection frame rate of 20 Hz, this gives a 250 ms delay between the making of the gesture and its detection. This delay should not impact human-robot interaction since the average human reaction time usually lies within the 200–250 ms range.[7] Once the same gesture has been detected five times in a row, the corresponding signal is activated. False positives can be observed, e.g. at t = 139 s when the first record signal ends, but thanks to the filtering systems no incorrect signal activation is made. The two following graphs in Fig. 14 show the end-effector translational velocity and force. It can be seen that through the *Teach* phase, i.e. until t = 135 s, the velocity simply follows the force applied to the robot. Then, the *Replay* phase starts and the end-effector velocity is now the result of the motion made to reach the waypoints and also by the force regulation applied at these locations. Between the two task executions (t = 153 s and t = 170 s), one can observe some force applied to the robot at t = 162 s. A safety feature is programmed to prevent accidents due to unexpected contact between the operator and the robot, leading to a monitored stop. In this situation, the robot stays still until the contact disappear and then resumes its motion to the second waypoints. The fourth graph displays the distance to the closest body part. The values are the raw ones provided by the Kinect V2 and are unitless. As mentioned previously, this distance is used to adapt the velocity limitation so that the robot can move quickly when nobody is around but slows down when an operator is approaching. The velocity limit is at a minimum of 0.02 m/s at a distance of 300 and at a maximum of 0.3 m/s at a distance of 600. The effect of this limitation can be observed multiple times, including after the beginning of the *Replay* phase where the distance suddenly drops below 300, enforcing a very slow motion of the robot. The last graph shows the evolution of the scaling factor computed by OpenPHRI. A value equals to one means that no velocity reduction has to be performed to comply with the constraints (velocity and acceleration limits, speed and separation monitoring and safety-

[5] https://cs.stanford.edu/people/tkr/fri/html/
[6] https://www.youtube.com/watch?v=lB5vXc8LMnk
[7] http://humanbenchmark.com/tests/reactiontime

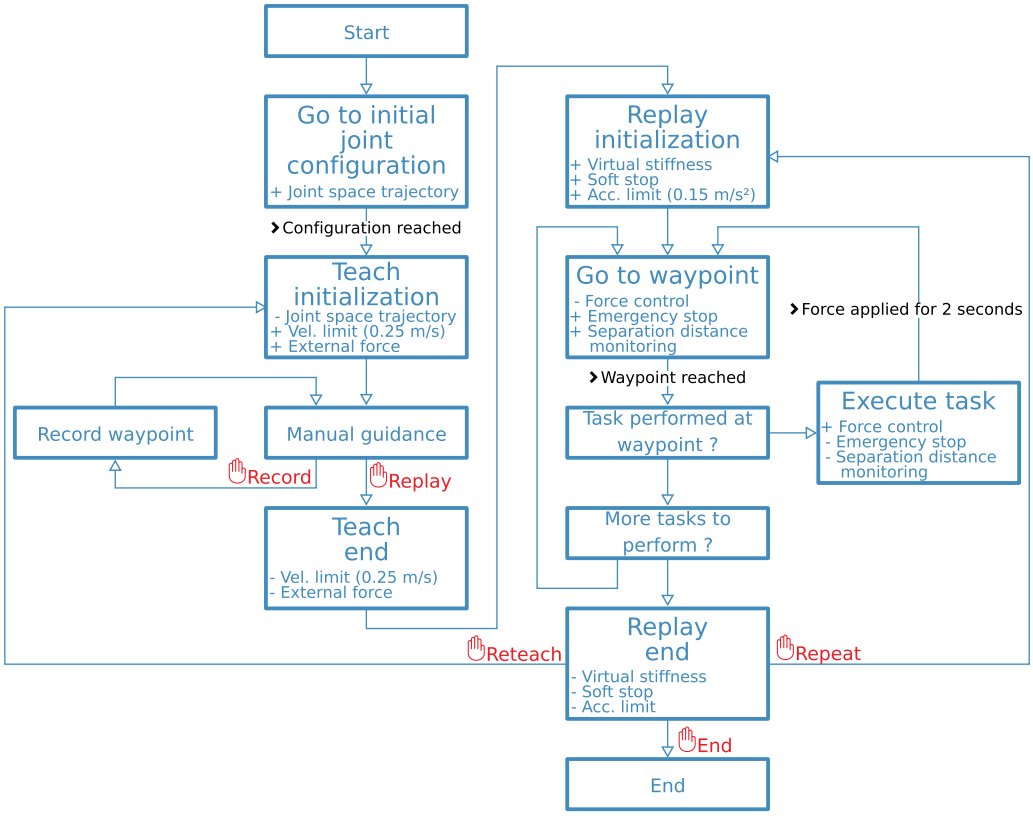


Fig. 12. The FSM used for the experiment. A plus sign indicates an addition to the controller (a new constraint or new input) while a minus indicates a removal.

rated monitored or soft stop). When at least one constraint would not be respected considering the current inputs, the scaling factor decreases below one to make sure that all constraints are satisfied. When the value reaches zero, the robot is at a complete stop. Using this technique allows to easily slow down the robot only when it is necessary.

8. Conclusion

In the perspective of smart factories – also known as factories of the future – we have introduced a real-time human-robot interaction framework for robot teaching using hand gestures. The proposed framework relies on our novel rotation and background invariant robust hand gesture detector. This is achieved by adapting a pre-trained state-of-the-art convolutional neural network, namely Inception V3, to the classification of 10 hand gestures. The CNN is trained on an image dataset of 10 hand gestures, recorded with the help of 10 volunteers. The dataset OpenSign, is open and available to the computer vision community for benchmarking. We also release the source code of our hand gesture detector.[8]

The accuracy of the trained CNN is validated with a set of test images and is found to be 98.9%. To reaffirm the quality of the hand gesture detector and to validate it on a mock-up example industrial scenario, we perform a robotic experiment. Safety and effectiveness of the experiment are guaranteed by our physical human-robot interaction library, OpenPHRI. Besides, real-time operation is established by asynchronous integration of the different modules present in our framework. The experiment proves the efficiency of the proposed framework, that ensures a natural means for robot programming. The robot is also aware of its distance from the human worker thanks to the integration of Kinect V2 and OpenPose. To guarantee the safety of the human coworker in close vicinity, the robot slows down using the velocity scaling feature of OpenPHRI.

Our approach requires the user to know the gestures the robot can perceive. However, once s/he has memorized these gestures, it will be more natural for her/him to communicate with the robot. Integrating face identification algorithms in this framework, could also be a security feature. It will allow only selected people to interact with the robot without entering any passwords or fingerprints scanning which might require the users to come in close proximity to the robot.

Despite the quantified accuracy and experimental results, the capabilities of our system are limited by the depth range of the vision sensor. Moreover, the system is trained and tested in indoor settings and may fail in bright light due to the resulting contrast in RGB images. Backgrounds with intense texture may also compromise detection. To handle this, distinct background images should be substituted in the hand images to train the proposed network. Nevertheless, we believe that the preliminary results presented in this paper are a very promising step towards the development of vision-based robot programming. We encourage researchers interested in these topics to profit from our open image dataset for benchmarking their algorithms, and to enrich the dataset with more images.Acknowledgements

This research was supported by a grant from the French Ministère de l'Education Nationale, de l'Enseignement Supérieur et de la Recherche.

[8] https://github.com/OsamaMazhar/openhandgesture

Fig. 13. Screenshots from the robotic experiment by operators Op1 and Op2 (a) Op1 manually guiding the robot to a waypoint in the workspace. (b) Op1 records the way-points using Record gesture. (c) Op1 replay the taught waypoints by Replay gesture. (d) Op2 stands far from the robot so it moves with full speed. (e) Op2 stops the robot by applying external force (or accidental touch). (f) Op2 stands near the robot, so it moves slowly ensuring operator's safety. (g) Op2 gives Reteach command to the robot. (h) Op2 sets the new waypoints manually. (i) Op2 gives Record command. (j) Op2 stops the robot by Stop gesture. (k) Op2 resumes the robot operation by Resume gesture. (l) Op1 ends the robot operation by giving End command.

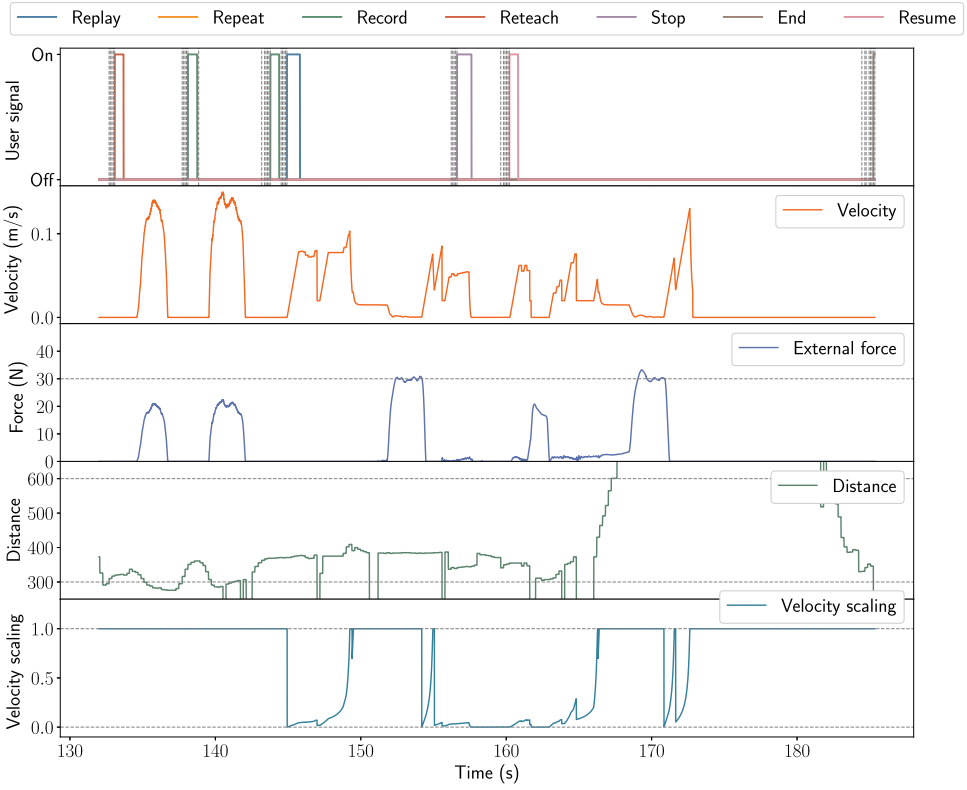


Fig. 14. Experimental results. From top to bottom: hand gesture detection (dashed lines correspond to detection instants and plain line to the activation signals), control point translational velocity, external force at the end-effector, distance between the camera and the closest human body part and velocity scaling factor computed by OpenPHRI to slow down the motion.

Conflict of Interest

There is no known conflict of interest.

References

[1] D. Gorecky, M. Schmitt, M. Loskyll, D. Zühlke, Human-machine-interaction in the industry 4.0 era, Proceedings of the IEEE International Conference on Industrial Informatics, (2014), pp. 289–294.

[2] B. Gleeson, K. MacLean, A. Haddadi, E. Croft, J. Alcazar, Gestures for industry: intuitive human-robot communication from human observation, Proceedings of the 8th ACM/IEEE International Conference on Human-robot Interaction, IEEE Press, Piscataway, NJ, USA, 2013, pp. 349–356.

[3] A. Cherubini, R. Passama, A. Crosnier, A. Lasnier, P. Fraisse, Collaborative manufacturing with physical human–robot interaction, Robot. Comput. Integr. Manuf. 40 (2016) 1–13.

[4] C. Schlenoff, Z. Kootbally, A. Pietromartire, M. Franaszek, S. Foufou, Intention recognition in manufacturing applications, Robot. Comput. Integr. Manuf. 33 (2015) 29–41.

[5] I. Maurtua, A. Ibarguren, J. Kildal, L. Susperregi, B. Sierra, Human-robot collaboration in industrial applications: safety, interaction and trust, Int. J. Adv. Robot. Syst. 14 (2017) 1–10.

[6] S.S. Rautaray, A. Agrawal, Vision based hand gesture recognition for human computer interaction: a survey, Artif. Intell. Rev. 43 (2015) 1–54.

[7] P. Neto, J.N. Pires, A.P. Moreira, High-level programming and control for industrial robotics: using a hand-held accelerometer-based input device for gesture and posture recognition, Ind. Robot. 37 (2010) pp137–147.

[8] V. Villani, F. Pini, F. Leali, C. Secchi, Survey on human–robot collaboration in industrial settings: safety, intuitive interfaces and applications, Mechatronics 55 (2018) 248–266.

[9] P. Neto, D. Pereira, J.N. Pires, A.P. Moreira, Gesture recognition for human-robot collaboration: a review, Proceedings of the 7th Swedish Production Symposium, (2016), pp. 1–12.

[10] J. Ruiz-del-Solar, P. Loncomilla, N. Soto, A survey on deep learning methods for robot vision, (2018) arxiv:1803.10862.

[11] R. Yang, S. Sarkar, B. Loeding, Handling movement epenthesis and hand segmentation ambiguities in continuous sign language recognition using nested dynamic programming, IEEE Trans. Pattern Anal. Mach. Intell. 32 (2010) 462–477.

[12] B. Navarro, A. Fonte, P. Fraisse, G. Poisson, A. Cherubini, In pursuit of safety: an open-Source library for physical human-robot interaction, IEEE Robot. Autom. Mag. 25 (2018) 39–50.

[13] A. De Luca, F. Flacco, Integrated control for pHRI: collision avoidance, detection, reaction and collaboration, Proceedings of the 4th IEEE RAS & EMBS International Conference on Biomedical Robotics and Biomechatronics, IEEE, 2012, pp. 288–295.

[14] ISO 10218-1:2011 - Robots and robotic devices - Safety requirements for industrial robots - Part 1: Robots. ISO; 2011(2011).

[15] ISO 10218-2:2011 - Robots and robotic devices - Safety requirements for industrial robots - Part 2: Robot systems and integration. ISO; 2011 (2011).

[16] ISO/TS 15066:2016 - Robots and robotic devices - Safety requirements for collaborative industrial robot systems and the work environment: Collaborative robots. ISO/TS; 2016 (2011).

[17] A. Levratti, G. Riggio, A. De Vuono, C. Fantuzzi, C. Secchi, Safe navigation and experimental evaluation of a novel tire workshop assistant robot, Proceedings of the IEEE International Conference on Robotics and Automation, (2017), pp. 994–999.

[18] L. Sabattini, A. Levratti, F. Venturi, E. Amplo, C. Fantuzzi, C. Secchi, Experimental comparison of 3D vision sensors for mobile robot localization for industrial application: Stereo-camera and RGB-D sensor, Proceedings of the 12th IEEE International Conference on Control Automation Robotics Vision, (2012), pp. 823–828.

[19] J.A. Marvel, R. Norcross, Implementing speed and separation monitoring in collaborative robot workcells, Robot. Comput. Integr. Manuf. 44 (2017) pp.144–155.

[20] V. Rampa, F. Vicentini, S. Savazzi, N. Pedrocchi, M. Ioppolo, M. Giussani, Safe human-robot cooperation through sensor-less radio localization, Proceedings of the 12th IEEE International Conference on Industrial Informatics, (2014), pp. 683–689.

[21] F. Vicentini, M. Ruggeri, L. Dariz, A. Pecora, L. Maiolo, D. Polese, L. Pazzini, L.M. Tosatti, Wireless sensor networks and safe protocols for user tracking in human-robot cooperative workspaces, Proceedings of the 23rd IEEE International Symposium on Industrial Electronics, (2014), pp. 1274–1279.

[22] F. Vicentini, M. Giussani, L.M. Tosatti, Trajectory-dependent safe distances in human-robot interaction, Proceedings of the Emerging Technology and Factory Automation, IEEE, 2014, pp. 1–4.

[23] P.A. Lasota, G.F. Rossano, J.A. Shah, Toward safe close-proximity human-robot interaction with standard industrial robots, 2014 IEEE International Conference on Automation Science and Engineering (CASE), Taipei, 2014, pp. 339–344.

[24] C. Morato, K.N. Kaipa, B. Zhao, S.K. Gupta, Toward safe human robot collaboration by using multiple kinects based real-time human tracking, J. Comput. Inf. Sci. Eng. 14 (2014).

[25] S. Makris, P. Tsarouchi, D. Surdilovic, J. Krüger, Intuitive dual arm robot programming for assembly operations, CIRP Ann. Manuf. Technol. 63 (2014) 13–16.

[26] Z. Zhang, Microsoft kinect sensor and its effect, IEEE Multimed. 19 (2012) 4–10.

[27] Y. Yang, H. Yan, M. Dehghan, M.H. Ang, Real-time human-robot interaction in complex environment using kinect v2 image recognition, Proceedings of the IEEE International Conference on Cybernetics and Intelligent Systems and IEEE Conference on Robotics, Automation and Mechatronics, (2015), pp. 112–117.

[28] M.G. Carmichael, D. Liu, A. Tran, R. Khonasty, S. Aldini, Robot co-worker for abrasive blasting: lessons learnt in worker posture estimation, Proceedings of the IEEE International Conference on Robotics and Automation, (2018).

[29] J. Sell, P. O'Connor, The xbox one system on a chip and kinect sensor, IEEE Micro 34 (2014) 44–53.

[30] L. Keselman, J.I. Woodfill, A. Grunnet-Jepsen, A. Bhowmik, Intel(R) RealSense(TM) Stereoscopic Depth Cameras, Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops, (2017), pp. 1267–1276.

[31] G. Du, P. Zhang, Markerless human–robot interface for dual robot manipulators using kinect sensor, Robot. Comput. Integr. Manuf. 30 (2014) pp.150–159.

[32] G. Du, M. Chen, C. Liu, B. Zhang and P. Zhang, Online Robot Teaching With Natural Human–Robot Interaction, in *IEEE Transactions on Industrial Electronics*, vol. 65, no. 12, pp. 9571-9581, Dec. 2018.

[33] E. Matsas, G.-C. Vosniakos, Design of a virtual reality training system for human–robot collaboration in manufacturing tasks, Int. J. Interact. Des. Manuf. 11 (2017) 139–153.

[34] E. Matsas, G.-C. Vosniakos, D. Batras, Prototyping proactive and adaptive techniques for human-robot collaboration in manufacturing using virtual reality, Robot. Comput. Integr. Manuf. 50 (2018) pp.168–180.

[35] M. SimÃ£o, P. Neto, O. Gibaru, Natural control of an industrial robot using hand gesture recognition with neural networks, Proceedings of the 42nd Annual Conference of the IEEE Industrial Electronics Society, (2016), pp. 5322–5327.

[36] G. Canal, S. Escalera, C. Angulo, A real-time human-robot interaction system based on gestures for assistive scenarios, Comput. Vis. Image Underst. 149 (2016) pp.65–77.

[37] J.L. Raheja, M. Chandra, A. Chaudhary, 3D Gesture based real-time object selection and recognition, Pattern Recognit. Lett. 115 (2017) 14–19.

[38] Y. LeCun, F.J. Huang, L. Bottou, Learning methods for generic object recognition with invariance to pose and lighting, Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2 (2004), pp. 97–104.

[39] K. Ehlers, K. Brama, A human-robot interaction interface for mobile and stationary robots based on real-time 3D human body and hand-finger pose estimation, Proceedings of the IEEE International Conference on Emerging Technologies and Factory Automation, (2016), pp. 1–6.

[40] T. Starner, J. Weaver, A. Pentland, Real-time american sign language recognition using desk and wearable computer based video, IEEE Trans Pattern Anal Mach Intell 20 (1998) 1371–1375.

[41] P. Neto, D. Pereira, J.N. Pires, A.P. Moreira, Real-time and continuous hand gesture spotting: An approach based on artificial neural networks, Proceedings of the IEEE International Conference on Robotics and Automation, (2013), pp. 178–183.

[42] Q. Munib, M. Habeeb, B. Takruri, H.A. Al-Malik, American sign language (ASL) recognition based on hough transform and neural networks, Expert Syst. Appl. 32 (2007) 24–37.

[43] C. Charayaphan, A. Marble, Image processing system for interpreting motion in american sign language, Med. Eng. Phys. 14 (1992) pp.419–425.

[44] D.H. Ballard, Generalizing the hough transform to detect arbitrary shapes, Pattern Recognit. 13 (1981) 111–122.

[45] A. Joshi, H. Sierra, E. Arzuaga, American sign language translation using edge detection and cross correlation, Proceedings of the IEEE Colombian Conference on Communications and Computing (COLCOM), (2017), pp. 1–6.

[46] P. Xu, A real-time hand gesture recognition and human-computer interaction system, (2017) arxiv:1704.07296.

[47] Y. LeCun, L. Bottou, Y. Bengio, P. Haffner, Gradient-based learning applied to document recognition, Proc. IEEE 86 (1998) 2278–2324.

[48] K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, (2016).

[49] H. Cheng, L. Yang, Z. Liu, Survey on 3D hand gesture recognition. IEEE Trans. Circuits Syst. Video Techn. 26 (2016) pp.1659–1673.

[50] M. Van den Bergh, L. Van Gool, Combining RGB and ToF cameras for real-time 3D hand gesture interaction, Proceedings of the IEEE Workshop on Applications of Computer Vision (WACV), (2011), pp. 66–72.

[51] I. Oikonomidis, N. Kyriazis, A.A. Argyros, Markerless and efficient 26-DOF hand pose recovery, Proceedings of the Asian Conference on Computer Vision, Springer, 2010, pp. 744–757.

[52] S.-E. Wei, V. Ramakrishna, T. Kanade, Y. Sheikh, Convolutional Pose Machines, Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, (2016).

[53] Z. Cao, T. Simon, S.-E. Wei, Y. Sheikh, Realtime Multi-Person 2D Pose Estimation using Part Affinity Fields, Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, (2017).

[54] P. Panteleris, I. Oikonomidis, A.A. Argyros, Using a Single RGB Frame for Real Time 3D Hand Pose Estimation in the Wild, Proceedings of the IEEE Winter Conference on Applications of Computer Vision, IEEE, 2018, pp. 436–445.

[55] O. Mazhar, S. Ramdani, B. Navarro, R. Passama, A. Cherubini, Towards Real-time Physical Human-Robot Interaction using Skeleton Information and Hand Gestures, Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems, (2018).

[56] S. Das, M. Koperski, F. Bremond, G. Francesca, Action recognition based on a mixture of RGB and depth based skeleton, Proceedings of the 14th IEEE International Conference on Advanced Video and Signal Based Surveillance, (2017), pp. 1–6.

[57] C. Zimmermann, T. Welschehold, C. Dornhege, T. Brox, W. Burgard, 3D human pose estimation in RGBD images for robotic task learning, Proceedings of the IEEE International Conference on Robotics and Automation, (2018).

[58] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár,

C.L. Zitnick, Microsoft COCO: Common Objects in Context, in: D. Fleet, T. Pajdla, B. Schiele, T. Tuytelaars (Eds.), Computer Vision – ECCV 2014, Springer International Publishing, Cham, 2014, pp. 740–755.
[59] R.C. Luo, Y.C. Wu, Hand Gesture Recognition for Human-Robot Interaction for Service Robot, Proceedings of the IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems, (2012), pp. 318–323.
[60] M.V. den Bergh, D. Carton, R.D. Nijs, N. Mitsou, C. Landsiedel, K. Kuehnlenz, D. Wollherr, L.V. Gool, M. Buss, Real-time 3D hand gesture interaction with a robot for understanding directions from humans, Proceedings of the RO-MAN, (2011), pp. 357–362.
[61] G. Cicirelli, C. Attolico, C. Guaragnella, T. D'Orazio, A kinect-based gesture recognition approach for a natural human robot interface, Int. J. Adv. Robot. Syst. 12 (2015) 22.
[62] M. Munaro, F. Basso, E. Menegatti, Tracking people within groups with RGB-D data, Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems, (2012), pp. 2101–2107.
[63] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A.C. Berg, L. Fei-Fei, Imagenet large scale visual recognition challenge, Int. J. Comput. Vis. 115 (2015) pp.211–252.
[64] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, Z. Wojna, Rethinking the inception architecture for computer vision, Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, (2016), pp. 2818–2826.
[65] F. Chollet, et al., Keras, (2015). https://keras.io
[66] R. Battison, Lexical Borrowing in American Sign Language 20901 Linstok Press, Inc., 9306 Mintwood Street, Silver Spring, Maryland, 1978.
[67] A. Cherubini, R. Passama, B. Navarro, M. Sorour, A. Khelloufi, O. Mazhar, S. Tarbouriech, J. Zhu, O. Tempier, A. Crosnier, P. Fraisse, S. Ramdani, A collaborative robot for the factory of the future: BAZAR, Int. J. Adv. Manuf. Technol. (2019) 1–17 Special Issue Design and Management of Digital Manufacturing & Assembly Systems in the Industry 4.0 era.