Gesture Recognition for Initiating Human-to-Robot Handovers

Jun Kwan, Chinkye Tan and Akansel Cosgun
Monash University, Australia

***Abstract*—Human-to-Robot handovers are useful for many Human-Robot Interaction scenarios. It is important to recognize when a human initiates handovers, so that the robot does not try to take objects from humans when a handover is not intended. We pose the handover gesture recognition as a binary classification problem in a single RGB image. Three separate neural network modules for detecting the object, human body key points and head orientation, are implemented to extract relevant features from the RGB images, and then the feature vectors are passed into a deep neural net to perform binary classification. Our results show that the handover gestures are correctly identified with an accuracy of over 90%. The abstraction of the features makes our approach modular and generalizable to different objects and human body types.**

I. Introduction

Robots are under rapid development in sectors such as manufacturing, automation and hospitality. Collaborative robots are increasingly being used in industry, and is expected to be useful in home environments in the future. One of the expected capabilities for collaborative robots is the ability to perform *object handovers*. Object handovers can happen in two directions: Robot-to-Human where the robot delivers a requested object to a human, and Human-to-Robot where the robot acquires an object from a human. The object handover problem has been studied extensively in the robotics literature, with most of the published work focusing on the Robot-to-Human handover scenario. Human-to-Robot handover scenario is arguably the more difficult problem because, as opposed to the Robot-to-Human scenario, the robot responsible for grasping the object from the human partner's hand, which raises safety issues. This makes the perception problem very critical for the Human-to-Robot handover scenario. The perception system is responsible for detecting when and where a human partner wants to engage in a handover, detect the object the human wants to pass, and ensure that a finger or body part of the human is not grasped. For example, a human might be holding a cellphone near the robot, and the robot should not unexpectedly reach out and grasp it from the human's hand. In this paper, we focus on detecting if a human is initiating a Human-to-Robot object handover.

Human-to-Robot handovers have been demonstrated on a physical robot system by a handful of researchers [1]–[6]. Even though successful demonstrations has been shown, in these works the robot is programmed for a single task: object handovers. In real scenarios where the robot is expected to engage in many tasks, the robot must first recognize the action of the human handing an object over to the robot. There are many communication cues that humans use to recognize handover intent, including direct cues such as verbal communication, and indirect cues such as eye gaze and body gestures. In this work, we focus on the binary classification of whether a human partner is handing over an object from body gestures only.

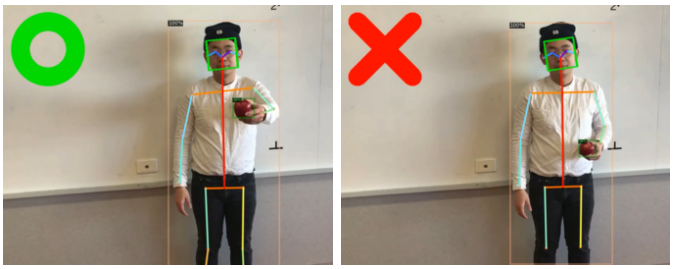

Fig. 1: We detect whether a user is initiating a handover from a single RGB image. Handover gesture is correctly detected (left column). No handover activity is detected (right column).

Our approach is based on extracting features relevant to the task from three independent modules, and learning a classifier to detect the existence of a handover gesture, based on the extracted features. Each of these modules is a neural network, described below:

- Object Detector: Detect the presence and pixel coordinates of the bounding box of an object.
- Human Body Pose Detector: Detect the human body pose with specific key points.
- Head Orientation Detector: The head orientation of the person, represented in Euler angles with respect to the camera frame.

We train a deep neural network that converts the resulting feature vector generated by the modules into a binary classification result. Our system uses the Faster Region Based Convolutional Neural Network (Faster R-CNN) [7] for object detection, Keypoint R-CNN [8] to estimate the human body pose, multi-loss Resnet50 [9] architecture to estimate the head pose of a person, and a fully connected deep net to produce the final result for recognition. We represent the human body key points in the object coordinate frame so that the detection is independent of where the human is in the image frame. Our approach is 1) *modular* because each module can be replaced as long as the output type is the same 2) *generalizable* because it is independent of the object and human appearance, and the relative position of the human in the image.

II. Approach

Our approach utilizes a total of four modules as shown in the system diagram (Fig. 2). The input RGB image is passed into the three neural networks to obtain relevant features in parallel: An object detector detects the pixel coordinates of the object of interest, multiple keypoints of a person's body is detected, and the head pose of the person is estimated. The resulting features are processed and compressed into a feature vector, which

is then passed into the final multi-layer perceptron (MLP) to generate a binary output, which is the estimation of whether the human is ready for a handover.

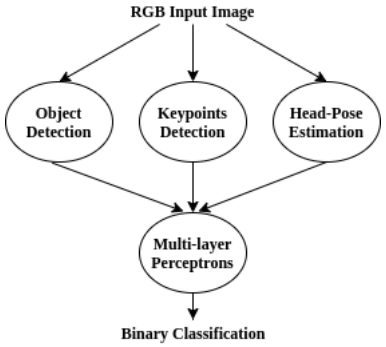

Fig. 2: System Diagram

A. Object Detection

Faster R-CNN [7] is implemented to detect pixel coordinates of an object held by a person. When an object is detected, a bounding box will be generated by this module. In the case where multiple objects are detected, the object with largest bounding box area is selected to be the target. Faster R-CNN is a modification of the original Region Based CNN [10] and the improved Fast R-CNN [11]. R-CNNs introduces a selective search algorithm to propose regions of interest in the image. These regions are then individually passed into a CNN which extracts the features of the regions. These features are subsequently passed into an SVM which determines the presence of the object in the proposed regions. Fast R-CNN improves the performance of R-CNN by passing the image through the CNN first to extract the feature map which is then used to generate the region proposals. Fast R-CNN is faster than basic R-CNN because the image is only passed into the CNN once whereas in R-CNN the large number of region proposals are all passed into the CNN slowing down the entire network. Faster R-CNN takes this improvement further by switching out the selective search algorithm used by R-CNN and Fast R-CNN to generate the region proposals with a region proposal network.

B. Body Keypoints Detection

Human body keypoints is commonly used in robotics applications, such as for ensuring safety for Human-Robot Interaction [6], [12] and recognizing gestures such as a pointing gestures [13].

Mask R-CNN [8] is implemented to detect various keypoints of a human body, such as shoulders, elbows, wrists, etc. Mask R-CNN is a modification of Faster R-CNN, where in parallel to the class and bounding box prediction it also adds a branch that outputs a binary mask for each Region of Interest (RoI). Mask R-CNN is able to perform semantic segmentation on each separate RoI, allowing the model to identify the boundaries of objects at the pixel level.

For keypoints detection, each keypoint is treated as a separate class during the training process. The segmentation branch of Mask R-CNN outputs k binary masks representing k different keypoints. In each binary mask, only one pixel is labelled as the foreground by the model representing the location of the keypoint. The coordinates of each of these points are then taken as the estimation of the pose. This modification of the Mask R-CNN model is called Keypoint R-CNN.

C. Head Pose Estimation

Multi-loss Resnet50 architecture, proposed by Ruiz et al. [9], is utilised for head pose estimation. Multiple losses are involved in the work and a loss is designated for each Euler angle. Thus, there are a total of three losses which are designated for yaw, pitch and roll separately. A binned pose classification and a regression loss are included in each loss. Furthermore, ResNet50 is used as the backbone of the network and three fully connected layers are attached to it for Euler angle predictions. A softmax layer and a cross entropy loss are implemented for bin classification, and hence there are three cross entropy losses in total. These losses are then back-propagated through the network. A mean squared error loss is added to the network as a regression loss as well. In addition, multi-task cascaded convolution networks (MTCNN) [14], proposed by Zhang et al., is used for face detection before computing the head pose estimation of the person.

D. Gesture Recognition from Extracted Features

We used a Multi-Layer Perceptron that consists of an input layer, four hidden layers, and an output layer. The network architecture is illustrated in Fig. 3. Features extracted by the previous modules are processed and concatenated into a feature vectore before passing into the MLP. Only keypoints of upper body of a person are included in the feature vector. We implemented two ways to input the feature vector, according to how the body keypoints are represented: relative to the object, or absolute pixel coordinates. Below we explain each method.

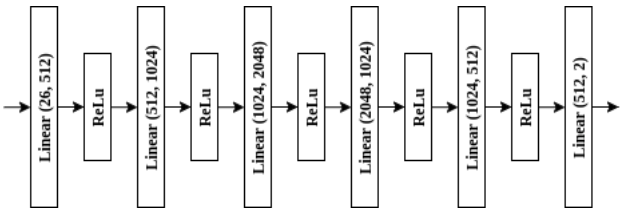

Fig. 3: Neural network architecture for the MLP to detect handover gestures

- **Absolute Pixels:** The input feature vector has a size of 29. The following features are concatenated into a vector.
  - Object (4 values): The pixel coordinates of the centroid, width and height of the object bounding box.
  - Absolute Human body keypoints (22 values): Pixel coordinates of each of the 11 keypoints belonging to upper body.
  - Head orientation (3 values): Yaw, pitch and roll angles in degrees, represented in camera frame.
- **Relative to Object:** The input feature vector has a size of 26. The following features are concatenated into a vector.
  - Object presence (1 value): Binary value of whether an object of interest is detected.

- Relative human body keypoints (22 values): Pixel coordinates of each of the 11 keypoints belonging to upper body, represented in the 2D coordinate frame attached to the target object centroid.
- Head orientation (3 values): Yaw, pitch and roll angles in degrees, represented in camera frame.

The output of the MLP is passed through a sigmoid function resulting in a likelihood of gesture detection between [0,1], and compared against a threshold (set to 0.5 in current implementation) to obtain the result of binary classification.

III. Experiments

A. Datasets and Training

For object detection model, the Faster R-CNN is implemented with Facebook AI Research's Detectron2 engine [15]. Detectron2 engine contains a collection of state-of-the-art object detection algorithms as well as APIs to train and deploy multiple algorithms at ease. The Faster R-CNN model is loaded with the COCO 2017 pre-trained weights and retrained on a subset of the COCO 2017 dataset [16]. For this implementation, the object considered in the handover process is an apple, and the Faster R-CNN model is retrained on just the images of apples in the COCO 2017 dataset. Although the system is trained to recognize apples only, this choice is made arbitrarily given the objects at hand. Even though our system is implemented for a single object class, it will also work on generic object detectors as long as the object bounding box is generated.

For the body keypoints detection, the Keypoint R-CNN model is also implemented using the Detectron2 engine [15]. The model uses the pre-trained weights provided by the engine itself. The pre-trained weights are found to be sufficiently accurate at detecting the human poses usually encountered in a handover scenario.

For head pose estimation, we used the 300W across Large Poses (300W-LP) dataset [17] to train the network. The pre-trained model is loaded during the training phase. This dataset is initially used for 3D Dense Face Alignment (3DDFA) [17], whereby a convolutional neural network (CNN) is used to apply a dense 3D model to an image. Moreover, 300W-LP contains various 3D landmarks which will be useful for training to obtain Euler angles such as yaw, pitch and roll.

For the handover gesture detection, we created a custom dataset for training the MLP model. A total of 25 videos were recorded in a lab setting, containing a total of 2506 images. Each image was given a label denoting a handover scenario or otherwise. For handover scenarios, the image was given a label 1 and 0 for non-handover scenarios. The images were individually labelled by hand. We use a rough 80%/20% training and testing split for training, in which while enforcing balancing between positive and negative examples. We also store all features extracted from RGB images in JSON format.

The training process begins with slicing a raw RGB video input into images and feeding them into the object detection, body pose estimation, and head pose estimation modules sequentially. The respective feature outputs are then processed for the MLP model. At this stage, the feature outputs are vectorised and passed into the MLP model. Some computation is also performed in this stage to make the frame to be relative to the object. There are a total of 26 parameters used for training essentially. When no object is detected, the yaw, pitch and roll in the feature vector are set to a dummy variable (-999 in this case) and the rest is set to be 0. This is to indicate an invalid feature. Binary Cross Entropy (BCE) loss is implemented in the training of MLP.

B. Baselines

In addition to our approach with two variations (absolute and relative human body key points), as explained in Sec. II-D, we implemented two baselines:

- **End-to-end:** Alexnet [18] and Resnet50 [19] are used for this end-to-end image classification. A raw RGB image is provided as the input to the model to do feature extraction. Alexnet eventually classifies all images as 0 and hence it completely fails at the classification of the custom dataset. On the other hand, the accuracy of Resnet50 is significantly higher than Alexnet.
- **CNN on skeleton image:** Resnet50 [19] is used for skeleton image classification. A skeleton image is generated by colouring the corresponding coordinates of each estimation point, bounding box, and head orientation on a black background. The image is then passed into Resnet50 to perform feature extraction. Resnet50 can only successfully classify some skeleton images in the custom dataset to a small extent.

C. Quantitative Results

The detection accuracy for our method (averaged over 5 random train/test splits), as well the baselines are shown in Table I.

Method	Accuracy (%)
End-to-end (Alexnet)	50.0
End-to-end (Resnet50)	89.4
CNN on skeleton image	83.3
Ours (absolute pixels)	90.1
Ours (relative to object)	**90.6**

TABLE I: Handover gesture detection accuracy. Our method, with human pose represented relative to object, performs the best.

Our results show that the **End-to-End** method with Alexnet, which uses the raw RGB image, was no better than random guesses. This shows the difficulty of the problem, as the difference between positive and negative examples are very subtle. The **End-to-End** with Resnet50 and **CNN on Skeleton Image** approaches achieved 89.4% and 83.3% accuracy respectively.

Both variations of our approach outperformed the baselines. Our method that uses the feature vector relative to the object had 90.6% detection accuracy, performing slightly better than the feature representation that uses absolute pixel values.

D. Qualitative Results

Some qualitative results are shown in Fig. 4. From the experiments, object-centric approach performs better due to its robustness. When a subject stands at the corner (Fig. 4a), the model still can classify the event of handover accurately