Received March 26, 2020, accepted April 17, 2020, date of publication May 4, 2020, date of current version May 26, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.2992068

Fast Prediction of a Worker's Reaching Motion Without a Skeleton Model (F-PREMO)

SHOGO ARAI[1], (Member, IEEE), ANDREAS LENNART PETTERSSON[2], AND KOICHI HASHIMOTO[2], (Member, IEEE)
[1]Department of Robotics, Graduate School of Engineering, Tohoku University, Sendai 980-8577, Japan
[2]System Information Sciences, Graduate School of Information Sciences, Tohoku University, Sendai 980-8577, Japan

Corresponding author: Shogo Arai (arai@tohoku.ac.jp)

This work was supported in part by the Grant-in-Aid for Scientific Research on Innovative Areas under Grant 16H06536, and in part by the Grant-in-Aid for Young Scientists (A) of Japan Society for the Promotion of Science under Grant 26709012.

ABSTRACT This paper proposes a fast and highly accurate prediction method for the reaching motions performed by a worker in cooperative work with a robot, called fast prediction of reaching motion (F-PREMO). Cooperative work is permitted by a 2011 ISO revision under the condition that risk assessment has been performed. To increase production effectivity, it is essential to predict the worker's movement and control the robot based on the prediction. This paper focuses on the movement of a worker's hand, called the reaching motion, which is relevant to many types of assembling tasks. The most common existing methods require attaching markers to the workers or installing three-dimensional (3D) sensors in front of the workers since these methods require the real-time estimation of skeleton models of the workers. These requirements lead to difficulties in introducing prediction methods at production sites. To solve this problem, we propose a prediction method for the reaching motion that neither requires the attachment of markers nor limits the placement of the 3D sensor. The proposed method first computes a feature vector of the reaching motion and then performs the prediction using random forest with the computed feature vector as input. Experimental results show that the proposed method can predict the reaching motion from the initial 50% of the movement of the worker with more than 80% accuracy in less than 5.2 [ms].

INDEX TERMS Collaborative work, reaching motion, assembly, bin-picking.

I. INTRODUCTION

There is an increasing need for cooperative work between humans and robots in fields such as object transportation [1], [2], nursing care [3], [4], and product assembly [5], [6]. Particularly in the industrial field, many collaborative industrial robots have been developed and operated without any separation between the workers and industrial robots, primarily due to the following reasons:

1) The safety fences can be removed after a risk assessment has been performed, as defined in ISO10218-1:2011 and ISO10218-2:2011 (revised in 2011);
2) Cooperative work has great potential for improving production efficiency by decreasing production time, shortening the production line, and decreasing the number of workers.

The associate editor coordinating the review of this manuscript and approving it for publication was Tao Wang.

For these two reasons, a number of robots have been developed for cooperative work, including CR-35*i*A [7], duAro [8], NEXTAGE [9], YuMi [10], LBRiiWa [11], and Baxter [12]. For example, the FANUC corporation developed an industrial robot for cooperative work, called CR-35iA. This robot has the following three safety functions [13]:

1) Contact stop function, which is activated on the physical impact between the robot and worker;
2) Push to escape function, under which the robot may be shoved in any direction by a human operator; and
3) Shock absorption function, which is achieved by covering the robot's surface with cushioned materials.

ABB has also produced a collaborative industrial robot called YuMi [10]. YuMi can execute emergency stops within less than 10 [ms] after detecting an unexpected impact, and the surface of the robot is covered with soft pads for absorbing the physical impacts. In addition, the positions of YuMi's joints and the length of each joint are carefully designed such that there is no gap where the worker's fingers can get stuck.

These robots ensure worker safety by utilizing emergency stop functions when collisions are detected and by computing trajectories for collision avoidance based on the distance between the robot and the worker [14]–[16]. In other words, these safety functions are based on the current state of the robot and the worker. In addition to these functions, predicting the worker's movement can greatly improve production efficiency by decreasing the number of emergency stops, increasing the running speed of the robot, and computing more efficient trajectories for collision avoidance [17].

The prediction of worker movements at production sites is categorized into the following two types. The first is the prediction of the movement of the worker's center of mass [18], [19], while the second is the prediction of the motion of the worker's arm, which is referred to as the reaching motion [29]–[31]. The former is mainly treated as a problem of predicting the movement of a mass point. In contrast, predicting the reaching motion is more difficult since the dynamics of motion of a human arm are highly nonlinear [32]. However, in many assembly tasks, the worker's arm moves with nearly no change in the worker's center of mass.

Therefore, this paper focuses on predicting the reaching motion of a worker. A number of prediction methods for reaching motion have been proposed. A prediction method that uses a Bayes classifier and assumes that the positions of each joint are recognized and measured is reported in [31]. In their method, the reaching motion is considered to be divided into two layers: the upper layer defines the type of task for the reaching motion and the lower layer provides the motion. The method proposed herein predicts the type of task and motion, and subsequently, integrates each prediction. Their paper reports that a prediction accuracy of greater than 70% is achieved based on 1/3 of the trajectory of the reaching motion.

With technological advances on three-dimensional measurement [20], [21], a lot of robotic tasks have been achieved with 3D sensors, such as, positioning [22], [23], pose estimation [24]–[26], picking [27], [28], and measurement of worker's movement. Researches [29] and [30] also employed motion-capture systems to detect the arms and utilized the Gaussian Mixture Model (GMM) algorithms to predict the reaching motion of the arm. Their studies focused on learning models that describe how humans cooperate, which can be applied to cooperative human-robot work. Other similar techniques used to investigate human motion have been studied by using Hidden Markov Models (HMMs) [33].

In these models, the positions of the worker's joints, particularly the arm joints, are measured using a motion-capture system, which requires (1) attaching markers to the worker and (2) placing at least three cameras. These requirements increase the cost of installing a cooperative production system and place limitations on the working environment.

Alternative methods for predicting the reaching motion without a motion-capture system have been proposed using a skeleton model of the worker.

For example, a prediction method for the reaching motion based on gesture recognition was proposed in [34]. In this method, the various types of worker movements are registered in a database in an offline process. The movements are then categorized into classes, and one GMM is assigned to each class. The volume swept by the worker is also computed for each class. Next, in an online process, the current motion is first categorized based on the recognized gesture, and the time series of the volume swept by the worker is predicted. This method achieved prediction accuracies of 50% and 92% based on measured data of 43% and 80% initial trajectories, respectively.

Ravichandar *et al.* proposed a prediction method for the trajectory of reaching motion and reaching point based on the dynamics of a human arm model using a neural network (NN) [32]. In their experiment, a skeleton model of the worker was tracked in real-time with a Kinect sensor. The prediction accuracy and prediction time of this algorithm were 87% (seven correct predictions out of eight trials) and 50 [ms], respectively.

Jiang *et al.* [35] proposed a method that identifies the skeleton model and predicts the reaching motion by using the identified skeleton model. They represented the skeleton model as a mapping between high- and low-dimensional representations in the skeleton model by using latent conditional random fields to model the spatial and temporal contexts of human activities. The method of Jiang *et al.* took 11.2 [s] to compute 10 [s] of the reaching motions.

The above methods estimate the joint positions of the worker by using the estimated skeleton model in real-time. However, to develop the skeleton model, it is necessary to place a 3D sensor in front of the worker in order to capture the upper body. In other words, these methods place limitations on the placement of the 3D sensor.

To overcome these limitations, we propose a fast and highly accurate prediction method, called fast prediction of reaching motion (F-PREMO) for the prediction of reaching motion without the need to attach markers to the worker or place a 3D sensor in front of the worker. The proposed method divides the common working area of the worker and robot into different regions and predicts which region the worker's hand will reach into. The proposed method mainly comprises the three following processes. First, the method extracts a set of 3D points comprising of the worker, which is called the ''worker's cluster'' in this paper, from the obtained 3D point cloud at each time step. Next, the method calculates a feature vector of the reaching motion from the time series composed of the worker's clusters. Finally, the proposed method predicts the region into which the worker's hand will reach by using a random forest. We confirmed the performance of the proposed method using an assumed assembling task. We conducted an experiment using a Microsoft Kinect v2 [36] as the 3D sensor; the sensor was mounted above the worker rather than in front of the worker. The experimental results showed that the proposed method could accurately predict the reaching motion based on 50% initial data for the reaching motion. In addition, the average computational time for the prediction was 5.2 [ms]. These results show that

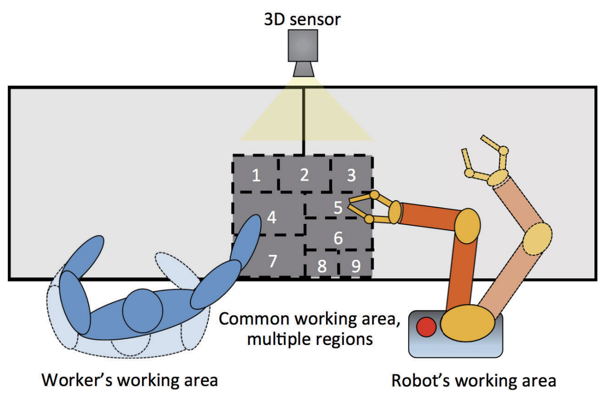


FIGURE 1. Situation for cooperative work considered in this paper. The working space consists of three types of areas: the robot's working area, common working area, and the worker's working area. In the robot's and worker's working areas, only a robot and worker can be located, respectively. The common working area is divided into several regions, each of which is assigned a label. A 3D sensor is located at a position where it can observe the common working area and movements of the worker.

the proposed method can quickly and accurately predict the reaching motion without requiring marker attachments or 3D sensors in front of the workers.

The remainder of the paper is organized as follows. Section II describes the problem considered in this paper: predicting the reaching motion of a human doing cooperative work with a robot. Section III describes the processes for computing the feature vector showing the worker's reaching motion from 3D point cloud data obtained by a 3D sensor. Section IV describes the semi-automatic process for creating training data and the process to construct a predictor using a random forest. Section V experimentally examines the performance of F-PREMO, including prediction accuracy and computation time. Section VI concludes this paper.

The following notations are used in this paper. A sphere with its center located at $p \in \mathbb{R}^3$ and radius $r \in \mathbb{R}$ is denoted by $S(p, r)$. Elements of set $\mathbb{A}$ are represented by $a_i (i = 1, 2, \cdots)$. The symbol $|\mathbb{A}|$ is defined as the number of elements of a set $\mathbb{A}$. Using the above notations, we can write

$$\mathbb{A} := \{a_1, a_2, \cdots, a_{|\mathbb{A}|}\}. \tag{1}$$

II. PROBLEM STATEMENT

As described in the previous section, this paper aims to predict a human's reaching motion during cooperative work between the human and the robot. In cooperative work, there is a chance of collision in the common working area where the robot and worker co-exist.

To address the above problem, we consider the working situation shown in Fig.1. This paper assumes three types of areas: the robot working area, common working area, and worker's working area. The common working area is divided into several regions, each of which is given a label. We will

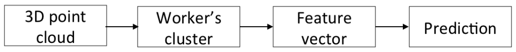


FIGURE 2. Prediction process from 3D point cloud data to prediction in the proposed method. Based on the 3D point cloud data, the method categorizes the point cloud and identifies 3D points representing a worker, as shown in Sec. III-B. We refer to the 3D points as a worker's cluster. The proposed method then computes a feature vector using the time-series data of the worker's clusters, as shown in Sec. III-C. Finally, the method predicts the reaching motion from the current time t to $t + T$ by using a random forest, as shown in Sec. IV.

propose a method that predicts the area that the worker's arm is reaching toward.

In addition, the proposed method does not require attaching markers to the worker or placing the 3D sensor in front of the worker. The only requirement concerning the placement of the 3D sensor is to mount it such that it can observe the worker movements in the common working area and in the worker's working area.

We can summarize the problem described above as follows:

Problem 1: There exists a common working area for a worker and a robot, which is divided into n regions represented by $\Omega_1, \Omega_2, \cdots, \Omega_n$. The time-series data of the 3D point cloud in the common working and worker's working areas are assumed to be given from the initial time 0 to the current time t, and are denoted by $\mathbb{P}(0), \mathbb{P}(1), \cdots, \mathbb{P}(t)$. The region located closest to the robot's working area S^{robot} and where the worker's hand will reach from the current time t to $t + T$ is determined as

$$i^*(t : T) = \arg\min_{i \in \hat{\mathbb{I}}(t+1:t+T)} d(\Omega_i, S^{\text{robot}}), \tag{2}$$

where $d(X, Y)$ denotes the distance function for the regions X and Y. $\hat{\mathbb{I}}(t + 1 : t + T)$ is the predicted index set of regions toward which the worker's hand will reach from time $t + 1$ to $t + T$, and is defined by

$$\mathbb{I}(t_s : t_g) = E(\mathbb{I}(t_s), \mathbb{I}(t_s + 1), \cdots, \mathbb{I}(t_g)), \tag{3}$$

$$\mathbb{I}(\tau) = \{i \mid \|(\hat{\mathbb{V}}^{\text{worker}}(\tau) \cap \Omega_i\|_{\text{volume}} > V_{\text{th}}\}, \tag{4}$$

where $E(\mathbb{J}_1, \mathbb{J}_2, \cdots, \mathbb{J}_n)$ is a mapping from the multi-set $\{\mathbb{J}_1, \mathbb{J}_2, \cdots, \mathbb{J}_n\}$ to a unique set in which each element has a multiplicity of 1. The symbol V_{th} represents a threshold for volume. The symbol $\hat{\mathbb{V}}^{\text{worker}}$ represents the predicted area occupied by the worker at time τ and $\|\mathbb{X} \cap \mathbb{Y}\|_{\text{volume}}$ denotes the volume of the intersection between regions $\mathbb{X}$ and $\mathbb{Y}$.

To solve Problems 1 and 2, we first propose a method to extract the worker's cluster from the obtained 3D point cloud data; this method is a solution to Problem 1, shown in Sec. III-B. We then propose a process to compute a feature vector that represents the reaching motion, as described in Sec. III-C. Finally, we present a method to estimate the region Ω_{i^*} that satisfies (2) by using the time-series data of the feature vectors; This method is a solution to Problem 2.

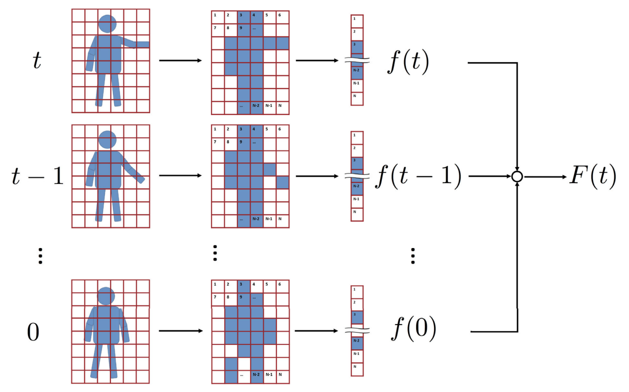


FIGURE 3. Process for computing a feature vector for the worker's reaching motion. This computation consists of four steps. In the first step, the worker's cluster is determined from point clouds obtained by a 3D sensor. In the second step, the 3D space is divided into voxels, and each voxel is binarized such that whether the number of points for the worker's cluster located in each voxel exceeds a threshold or not. After this process, the binary vector $f(t)$ is obtained in the third step. In the fourth step, the feature vector of the worker's reaching motion $F(t)$ is computed from the time series of the binary vectors $f(t), f(t-1), \cdots$.

III. COMPUTATION OF FEATURE VECTOR

This section describes the process of computing a feature vector of the worker's reaching motion. The process mainly consists of four steps, as shown in Fig.3.

In the first step, the worker's cluster is extracted from 3D point cloud data. In the second step, the working area is divided into voxels and each voxel is binarized to indicate the presence of the worker. Subsequently, the binary vector $f(t)$ is obtained in the third step. Finally, in the fourth step, a feature vector $F(t)$ is computed based on the time series of binary vectors $f(t), f(t-1), \cdots$.

Section III-A will explain preparation needed to be performed for the process of computing the feature vector. The first step is described in sec. III-B and the rest of the steps are shown in sec. III-C.

A. PREPARATION

First, it is necessary to install a 3D sensor in a place where it can observe the common working area and the worker's working area. It is not necessary for the 3D sensor's observation area to cover the entirety of the worker's working area. In addition, the 3D sensor is assumed to be able to obtain depth images of the scene. Thus, each 3D point obtained by the 3D sensor will have a corresponding 2D representation (u, v) in the camera's view, which can be represented by

$$s\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = g(x, y, z), \tag{5}$$

where (x, y, z) and s are the obtained 3D point's coordinates and a scale parameter, respectively. A number of models have been proposed for the function g [37]. The simplest one is the pin-hole camera model, which can be represented by

$$g(x, y, z) = K\begin{bmatrix} x \\ y \\ z \\ 1, \end{bmatrix} \tag{6}$$

where K is the intrinsic matrix and includes the focal length of the camera.

Second, a rigid body conversion matrix $T_{\mathrm{M}}^{\mathrm{S}}$ to transform the manipulator's coordinate system to the 3D sensor's coordinate system can be obtained by offline calibration. Next, the 3D shape of the manipulator is approximated by a primitive shape χ (e.g., a cuboid or a sphere) [14], and a function

$$\mathbb{V}^{\mathrm{M}} = g(\chi, p_1^{\mathrm{joint}}, p_2^{\mathrm{joint}}, \cdots, p_J^{\mathrm{joint}}) \tag{7}$$

is identified, where $\mathbb{V}^{\mathrm{M}}$ and p_i^{joint}, $i = 1, 2, \cdots, J$ represent the approximated occupied space of the manipulator and each joint position of the manipulator, respectively. As a final phase in the preparation, the 3D sensor obtains a point cloud data for background where the manipulator is set to be in an initial pose, and the worker is not present.

B. RECOGNITION OF WORKER BASED ON 3D POINT CLOUD DATA

This section describes the process for extracting the worker's cluster of 3D point cloud data from the obtained 3D point cloud data at each time point.

First, the 3D points consisting of the manipulator are recognized in the 3D point cloud data as follows. Here, the 3D point cloud data are denoted by $\mathbb{P}(t) := \{p_1(t), p_2(t), \cdots, p_{n_t}(t)\}$. For the extraction, each joint position of the manipulator $q_i^{\mathrm{M}}, i = 1, 2, \cdots, J$ in the robot coordinate system is computed by encoders from the angles of the joints. The joint positions are converted into the 3D sensor's coordinate system by

$$q_i^S(t) = T_{\mathrm{M}}^{\mathrm{S}} q_i^{\mathrm{M}}(t), \quad i = 1, 2, \cdots, n_{\mathrm{joint}}. \tag{8}$$

From the computed joint positions, we can specify the space occupied by the manipulator as

$$\mathbb{V}^{\mathrm{M}}(t) = g(\chi, q_1^S(t), q_2^S(t), \cdots, q_J^S(t)). \tag{9}$$

The 3D points corresponding to the manipulator are recognized by

$$\mathbb{P}^{\mathrm{M}}(t) := \{p_i(t) \mid p_i(t) \in \mathbb{V}^{\mathrm{M}}(t)\}. \tag{10}$$

Next, we perform the background subtraction for the point cloud and extract the changed points to satisfy

$$\psi_z(p_{(u,v)}(t)) - \psi_z(p_{(u,v)}(0)) \geq \epsilon_{\mathrm{bs}}, \tag{11}$$

where $p_{(u,v)}(t) \in \mathbb{R}^3$ represents the 3D point corresponding to the 2D representation (u, v) and $\psi_z(p)$ is a function that returns the z coordinate of a 3D point p. Here, the set of points satisfying (11) is defined by $\mathbb{P}^{\mathrm{bs}}(t) := \{p_{(u,v)}(t) \mid \text{satisfies eq.(11)}\}$.

The manipulator's points are then removed from $\mathbb{P}^{\mathrm{bs}}(t)$; that is,

$$\bar{\mathbb{P}}^{\mathrm{bs}}(t) := \mathbb{P}^{\mathrm{bs}}(t)\backslash\mathbb{P}^{\mathrm{M}}(t). \tag{12}$$

The clustering is performed for points $\bar{\mathbb{P}}^{\mathrm{bs}}(t)$ using either the 3D representation of the points [38] or the projected 2D representation of the points [39]:

$$\bar{\mathbb{P}}^{\mathrm{bs}}(t) = \{\mathbb{P}_1^{\mathrm{bs}}(t), \mathbb{P}_2^{\mathrm{bs}}(t), \cdots\}. \tag{13}$$

The clustered points are then obtained.[1]

The clusters in $\bar{\mathbb{P}}^{\mathrm{bs}}(t)$ with more than $\epsilon_{\mathrm{worker}}$ points are considered to be the worker; that is,

$$\mathbb{P}^{\mathrm{worker}}(t) := \{\bar{\mathbb{P}}_i^{\mathrm{bs}} \mid i \in \{1, 2, \cdots\},\ |\bar{\mathbb{P}}_i^{\mathrm{bs}}| > \epsilon_{\mathrm{worker}}\}, \tag{14}$$

where $\epsilon_{\mathrm{worker}}$ is the user-defined threshold, which should be set depending on the worker's volume.

We write the above process for extracting the worker's cluster of 3D point cloud data from $\mathbb{P}(t)$ as

$$\mathbb{P}^{\mathrm{worker}}(t) = \xi(\mathbb{P}(t), \mathbb{P}(0)), \tag{15}$$

where $\mathbb{P}(0)$ represents the 3D point cloud obtained as the background. The function ξ in (15) is a solution to Problem 1.

C. FEATURE VECTOR OF THE WORKER's REACHING MOTION

This section describes the process of computing a feature vector of the worker's movement from the obtained worker's cluster derived in the previous section. The reaching motion is predicted using a random forest [40] with the feature vector as input.

First, we assume that the worker's cluster of 3D point cloud data is given from the initial to current time $\mathbb{P}^{\mathrm{worker}}(\tau)$, $\tau \in \{0, 1, \cdots, t\}$ by using the method shown in sec. III-B.

The 3D space is then divided into voxels with length L_{voxel} on each side. Each voxel is turned into a binary state to indicate the presence of the worker inside a voxel. If at least one point from $\mathbb{P}^{\mathrm{worker}}(t)$ is mapped to the voxel, the voxel takes the value 1; otherwise, the voxel takes the value 0. After this process, the 3D voxel space is mapped into a 1D space, creating a binary vector denoted by $f(\tau) \in \{0, 1\}^{n_f}$, $\tau = t, t-1, \cdots$, as shown in Fig.3.

Now, we are ready to define the feature vector as

$$F(t) := \begin{bmatrix} f(t) \\ \delta(0, \kappa_0, \kappa_1) \\ \delta(1, \kappa_0, \kappa_1) \\ \vdots \\ \delta(k_0, \kappa_0, \kappa_1) \end{bmatrix} \in \{-1, 0, 1\}^{n_F}, \tag{16}$$

where κ_0 and κ_1 are user-defined constants for time delay. The function δ represents the time change for a worker's pose and is defined by

$$\delta(k, \kappa_0, \kappa_1) = f(t - k\kappa_0) - f(t - k\kappa_0 - \kappa_1). \tag{17}$$

[1]We have used an approach similar to that of [39] in sec. V.

IV. PREDICTION OF REACHING MOTION WITH FEATURE VECTOR

This section explains the prediction process by using random forest as a classifier. Before use, the random forest is trained on the data pairs of the feature vectors and region labels.

These training data are semi-automatically created from time 0 to T_{train} using the following process. The training data, which are constructed of pairs of the feature vector $F(t)$ and the region label $\ell(t)$, are created in real-time without post-processing. To create the training data, the worker only needs to indicate the starting time t_i for the i-th reaching motion. The end time for each reaching motion is automatically identified as follows. Once the worker's hand is detected inside the region ℓ_i in the common working area at time t_i^{reach}, a timer will run for the duration s^{timer}. If the worker's hand has been inside the region ℓ_i during the whole duration of s^{timer}, the training dataset $\mathbb{D}_i$ for the i-th reaching motion is constructed by

$$\mathbb{D}_i := \{(F(t_i), \ell_i), (F(t_i{+}1), \ell_i), \cdots, (F(t_i{+}T_i), \ell_i)\}, \tag{18}$$

where T_i represents prediction horizon, which is defined as $t_i^{\mathrm{reach}} - t_i + s^{\mathrm{timer}}$. The collection of n_{train} reaching motion samples is combined as

$$\mathbb{D}^{\mathrm{predict}} := \{\mathbb{D}_1, \mathbb{D}_2, \cdots, \mathbb{D}_{n_{\mathrm{train}}}\}. \tag{19}$$

Not all feature vectors from time 0 to T_{train} are included in $\mathbb{D}^{\mathrm{predict}}$. The feature vectors that are not included in $\mathbb{D}^{\mathrm{predict}}$ (i.e., the worker is not performing a reaching motion toward the common working area) are denoted by $F(t_i^{\mathrm{no}})$ and collected in

$$\mathbb{D}^{\mathrm{no}} := (F(t_1^{\mathrm{no}}), -1), (F(t_2^{\mathrm{no}}), -1), \cdots, (F(t_{m_{\mathrm{no}}}^{\mathrm{no}}), -1), \tag{20}$$

where the label -1 indicates no reaching motion toward the common working area. Here, note that

$$\{t_1^{\mathrm{no}}, t_2^{\mathrm{no}}, \cdots, t_{m_{\mathrm{no}}}^{\mathrm{no}}\} = \{0, 1, \cdots, T_{\mathrm{train}}\}\backslash\{t_1, t_2, \cdots, t_{n_{\mathrm{train}}}\}. \tag{21}$$

Finally, we combine all training datasets generated by the above process into one training dataset:

$$\mathbb{D} := \{\mathbb{D}^{\mathrm{predict}}, \mathbb{D}^{\mathrm{no}}\}. \tag{22}$$

The random forest classifier consisting of n_{tree} decision trees is then built using the training dataset. Each decision tree in the random forest is trained on randomly chosen data pairs $(F(t), \ell(t))$ from $\mathbb{D}$, where $F(t)$ is the input vector to a decision tree and $\ell(t)$ is supervised data. In the online prediction process, the output of the random forest is the integrated output from each decision tree, as shown in Fig.4. The output of each tree is a probability prediction score for each region, including the special label -1. The output $i_{\mathrm{region}}^{\mathrm{predict}}(t)$ from the random forest represents the region into which the worker's hand will reach with high probability from time t to $t + T$.

Representing the reaching motion with the feature vector $F(t)$ allows us to store the current and previous positions of the worker's poses in a single vector without imposing any

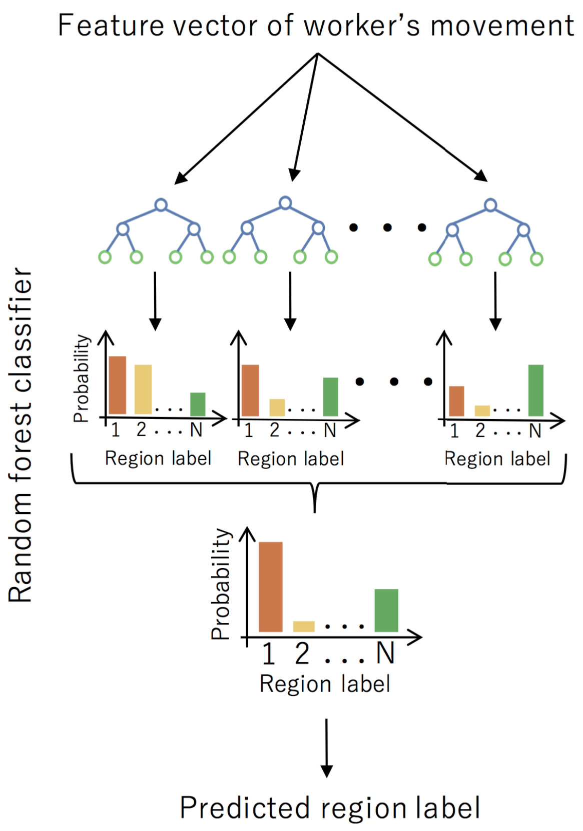


FIGURE 4. Process of predicting the region toward which the worker's hand will reach from current time t to $t + T$ by using a random forest. First, a feature vector $F(t)$ of the worker's reaching motion is computed from the point cloud data $\mathbb{P}(t)$. The feature $F(t)$ is then inputted to the random forest, which is obtained by the semi-automatic training process described in Sec. IV. A random forest is a set of binary decision trees, and each tree outputs the probability of the worker's hand reaching a certain region. The outputs of each tree are integrated, and the region with the highest probability is the prediction output from the random forest.

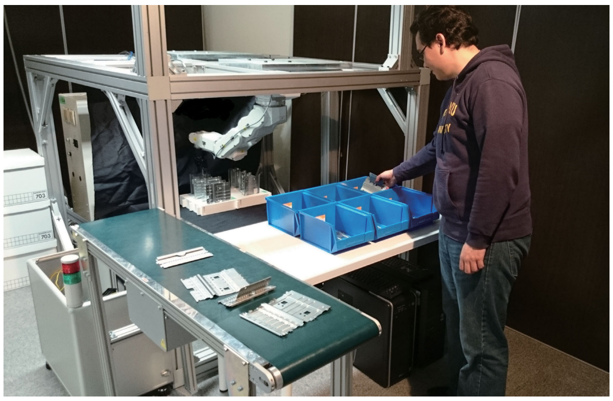


FIGURE 5. Experimental system. A detailed explanation is given in the caption of Fig.6.

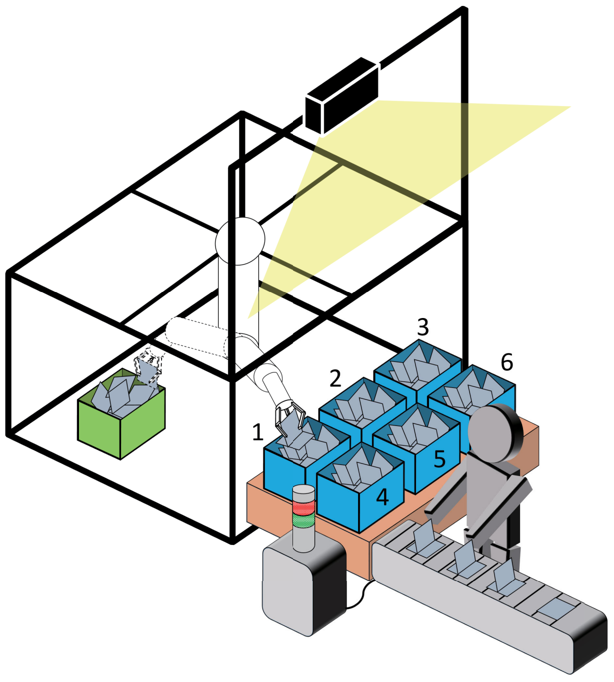


FIGURE 6. Illustration of the experimental system. The six blue boxes are located in the common working area; the worker reaches and picks up pieces from the boxes needed for the assembly task, and the robot supplies the boxes with the pieces. The common working area is monitored by a 3D sensor, which generates point clouds of the scene.

kinematic model for the worker's body. Even though $F(t)$ is a high-dimensional binary vector, the fast computation of the reaching motion is achievable. The random forest classifier can ignore unimportant elements inside $F(t)$ (e.g., voxels that have never been occupied by the worker). Each decision tree is trained to identify and disregard unimportant feature values inside the first binary decision tree layers, thereby quickly reducing the feature space for the prediction process.

V. VALIDATION OF THE PREDICTION METHOD

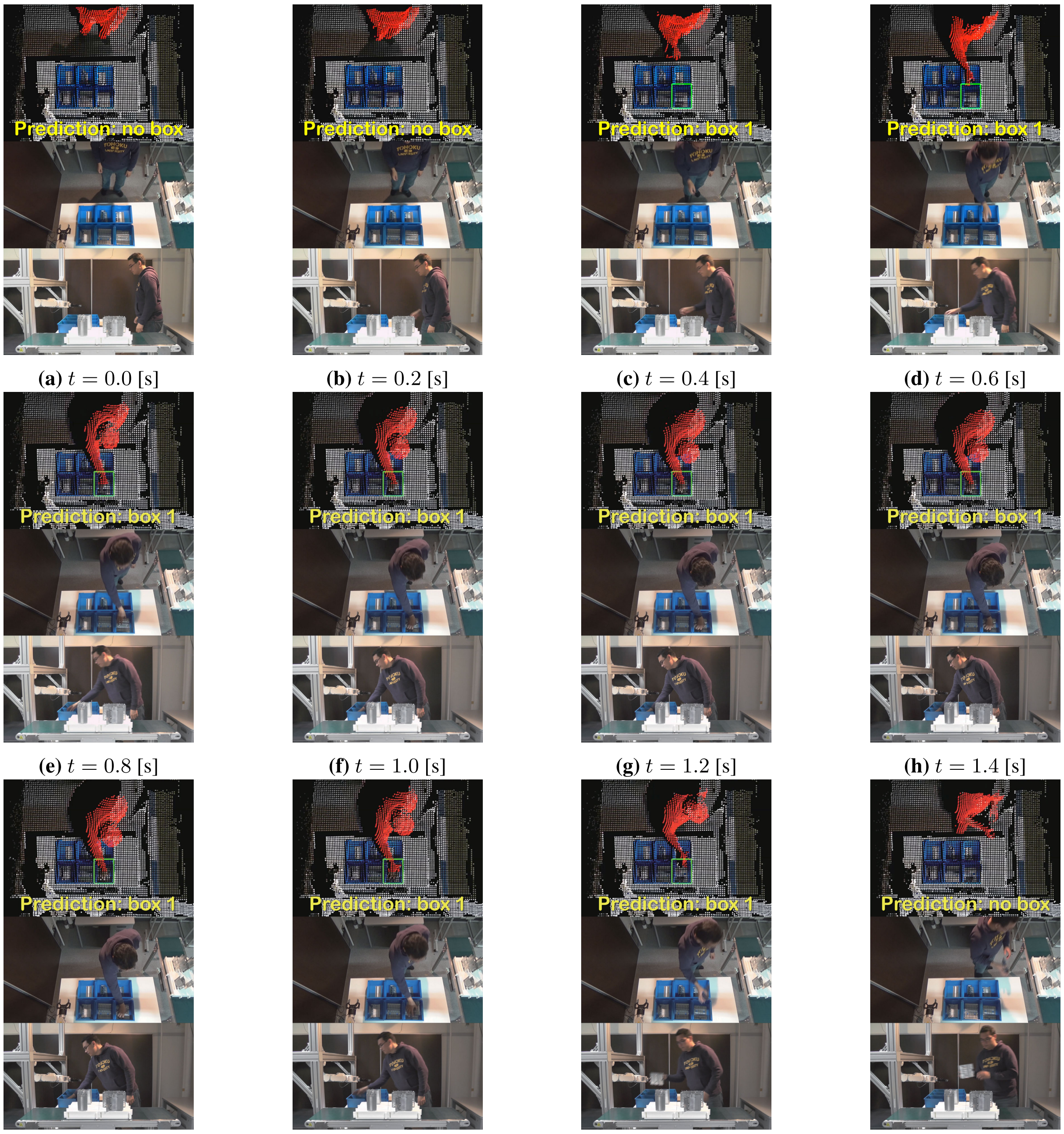
Figure 5 shows a photo of the experimental system used for validation.

The validation experiment involves a cooperative assembly task as follows.

1) A robot sequentially assembles six types of parts in the robot's working area.
2) The robot places an assembled part in the corresponding parts boxes (six boxes are located on a table).
3) Step-by-step assembly instructions for a product are shown to a worker.
4) The worker selects and picks up parts that are necessary for the product from the parts boxes, and assembles the product on a conveyor belt.

In practice, the robot does not assemble parts in the robot's working area, and six types of parts are prepared in the common working area because this situation is sufficient to evaluate the performance of the proposed method and verify the extraction of the worker's cluster from the 3D point cloud, including the robot's cluster.

In the verification experiment, the common working area is coincident with the area where the parts boxes are located. This common working area is divided into six regions, each of



(a) $t = 0.0$ [s] **(b)** $t = 0.2$ [s] **(c)** $t = 0.4$ [s] **(d)** $t = 0.6$ [s]

(e) $t = 0.8$ [s] **(f)** $t = 1.0$ [s] **(g)** $t = 1.2$ [s] **(h)** $t = 1.4$ [s]

(i) $t = 1.6$ [s] **(j)** $t = 1.8$ [s] **(k)** $t = 2.0$ [s] **(l)** $t = 2.2$ [s]

FIGURE 7. Prediction results for the worker's reaching motion in the 65th cycle from time 0.0 [s] to 2.2 [s]. Upper: 3D point cloud data obtained by the Kinect v2 camera installed above the part boxes and prediction results indicating which box was targeted by the worker's hand. Middle: time-series photos captured by a camera located next to the Kinect v2 camera. Lower: time-series photos captured by the camera located in front of the conveyor belt. Figures 7a–7d show the worker's hand approaching box 1 and Figs. 7i–7l show the worker's hand picking up the part. The prediction result of "no box" shown in Figs.7a and 7b indicates that the worker's hand will not reach the common working space from time *t* to *t* + *T*. The prediction result in Fig.7c illustrates that the proposed method outputs the correct prediction at 0.4 [s] after the initial pose of the worker. The results show that the proposed method generates correct predictions based on slight changes in the worker's posture.

which is represented by a box. In this situation, the proposed method predicts which box the worker's hand will reach.

The system configuration is shown in Fig.6. A Kinect v2 for Windows [36] is used as the 3D sensor and set at 1.2 [m] above the parts boxes. The Kinect v2 captures 512×424 depth images at a frequency of 30 Hz. The 3D point cloud data obtained by Kinect v2 are sent to a PC with an Intel Core i7-4720HQ CPU and 16 GB of DDR3 SDRAM, and processed for prediction. A point cloud library [41] is used for processing the 3D point cloud data.

To create the training dataset, the worker performs 300 cycles (50 cycles for each box). Here, starting from the

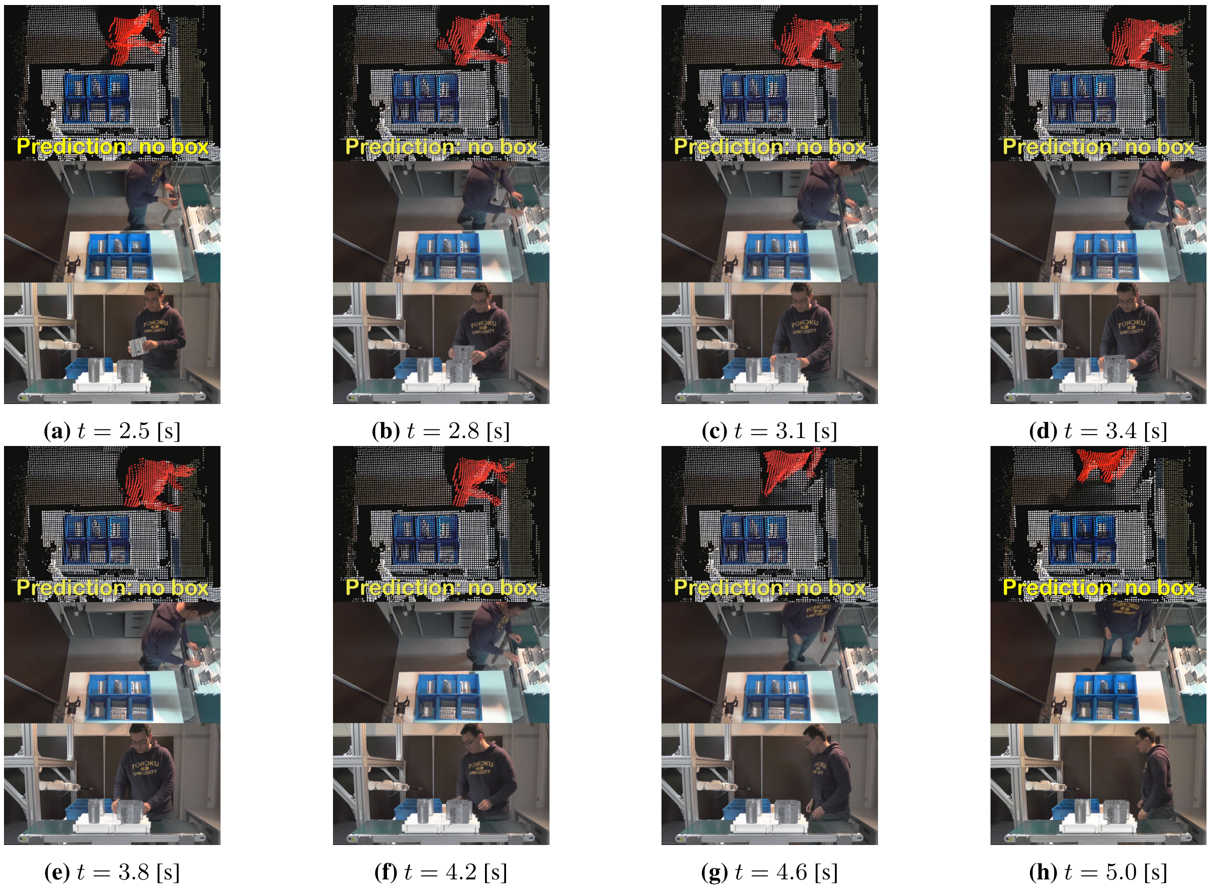


FIGURE 8. Prediction results for the worker's reaching motion in the 65th cycle from time 2.5 [s] to 5.0 [s]. Figures 8a–8e show the hand setting the part in the fixture on the conveyor belt. After the worker picks up the part shown in Fig. 7l, the proposed method also provides the correct prediction of "no box," as shown in Figs. 8a–8h.

TABLE 1. parameter setting.

parameter	value	explanation
χ	sphere	space occupied by manipulator: (7)
ϵ_{bs}	2[cm]	background subtraction: (11)
ϵ_{worker}	50 (points)	identifying worker's cluster: (14)
k_0	2	definition of feature vector: (16)
L_{voxel}	0.1[m]	length of a voxel: sec. III-C
κ_0	4	definition of feature vector: (16)
κ_1	4	definition of feature vector: (16)
n_f	1,000	definition of $f(t)$: sec. III-C
n_F	4,000	definition of feature vector $F(t)$: (16)
s^{timer}	0.5 [s]	timer duration for detection: sec. IV
T	51	prediction time: (18)
n_{tree}	16	the number of trees in Random Forest: sec. IV

initial pose of the worker, we define one cycle process to include the worker picking up one part and inserting that part into a fixture located on a conveyor belt. The random forest is constructed using the created training datasets. A method for this machine-learning part is implemented with a machine-learning library called scikit-learn [42].

Table 1 shows the parameter settings for the proposed prediction method. In this experiment, Kinect v2 can measure a 3D space with a volume of 2.2[m] $\times$ 2.2[m] $\times$ 3.1[m]. Here, we set an ROI of 1.0[m] $\times$ 1.0[m] $\times$ 1.0[m] in the observed 3D space. In the ROI, we divide the 3D space by voxels with length $=$ 0.1[m] in each dimension. Thus, the dimension of the binary vector $f(t)$ is equal to 1,000. In addition, the dimension of the feature vector is 4,000 since k_0 is set to be two, as shown in Table 1.

To validate the proposed prediction method, we performed an experiment for 90 cycles (i.e., 15 cycles for each of the six boxes). The number of 3D points obtained by Kinect v2 is almost 217,000. Downsampling is performed at each time point, and the number of obtained 3D points is decreased to nearly 14,000 points.

Figures 7 and 8 shows the obtained 3D point cloud, prediction results, and photos for the 65th cycle. In Figs. 7a–8h, the upper, middle, and lower figures show the 3D point cloud with prediction results, photos captured by the camera located above the conveyor belt, and photos captured by the camera located in front of the conveyer belt, respectively. In Fig. 7 and 8, the processes of stretching out, picking up the part, and putting it on the conveyer belt are shown in 7a–7d, 7i–7l, and 8a–8e, respectively. After picking up the part, the method predicted "no box" (i.e., the worker will not pick up any part in the current phase), as shown in Figs. 8a–8h. This result and

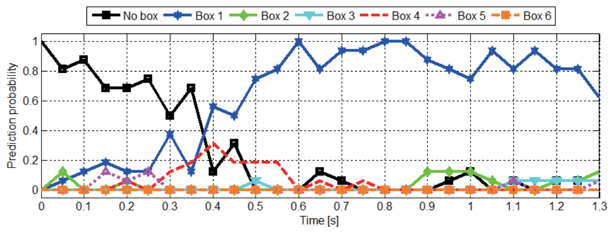


FIGURE 9. Time-series prediction results of the random forest for the cycle shown in Fig.7. The horizontal axis shows time, with $t = 0$ being the beginning of the reaching motion. The vertical axis shows the output of the random forest. The worker can be considered to be in one out of seven states [i.e., reaching box $i \in \{1, \cdots, 6\}$ or not reaching any box ("no box")]. For this cycle, the proposed method can predict the reaching motion correctly at time 0.4[s] since the output of the random forest changes from "no box" to "Box 1" at that time.

Fig.7c show that the proposed method can produce the correct prediction based on slight changes in the worker's posture.

Figure 9 illustrates the result of time-series prediction for the 65th cycle. In this experiment, a total of seven states of the worker can be considered, that is, reaching boxes i, $i \in \{1, 2, \cdots, 6\}$ and not reaching any box ("no box"). Thus, Fig.9 shows the predictions for the seven states. The output of the random forest corresponds to the state with the highest likelihood at each time point. The output of the random forest changes from "no box" to "Box 1" at approximately time $t = 0.4$[s]. Thus, the proposed method achieved the correct prediction based on 0.4[s] of the observation of the reaching motion.

Figure 10 shows the statistical prediction result for the entire validation set of 90 reaching motions. Compared to Fig.9, which illustrates the prediction result over time, this graph shows the prediction result over normalized distance ratio, where 0 and 1 are the initial point of the trajectory of the reaching motion and the point where the worker's hand reached the box, respectively. Figure 10 shows that when the worker has performed 40% to 50% of the reaching motion from the worker's initial position, the proposed method can correctly predict the reaching motion. The placement of the boxes with regard to the initial pose of the worker indicated that the reaching motions for the right arm were longest for box 1, with motion's length diminishing for every box down to box 6. Longer arm reaching motion results in more visual features of worker's posture that can be used to distinguish the reaching motion toward different boxes. This would explain the differences in the early prediction results seen in Fig.10 for different targeted boxes.

Figure 11 shows the computation time for the prediction process. Each prediction process, including the processing of the 3D point cloud, takes 5.18 [ms]. This result indicates that real-time prediction can be performed by the proposed method.

The changes in prediction accuracy and computation time with the number of decision trees constructing the random forest are shown in Fig.12. Figure 12 illustrates "Prediction using Random Forest" corresponding to Fig.11 since the

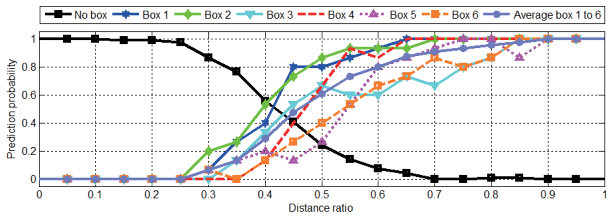


FIGURE 10. Statistical result of the prediction of 90 reaching motions. The horizontal axis shows the distance ratio of trajectories beginning from the initial position to the worker's hand reaching the boxes. The average prediction result for each reaching motion is illustrated in this graph. Based on the 40% to 50% initial trajectory of the reaching motion, the proposed method can correctly predict the reaching motion.

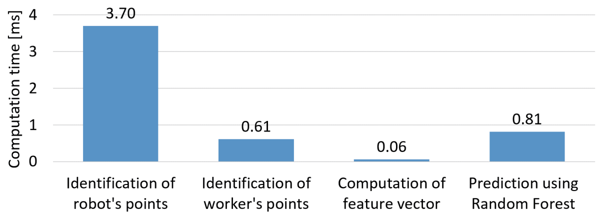


FIGURE 11. Computation time for a prediction. The average number of points obtained by the Kinect v2 camera in the experiment is approximately 14,000. The proposed method takes 5.18 [ms] to predict the worker's reaching motion. The prediction process mainly comprises four steps, as shown in this plot. The detection of the robot and worker described in Sec. III-B, computation of the proposed feature vector for the worker's reaching motion described in Sec. III-C, and prediction using a random forest described in Sec. IV take 3.70[ms](71%), 0.61[ms](12%), 0.06[ms](1%), and 0.81[ms](16%), respectively.

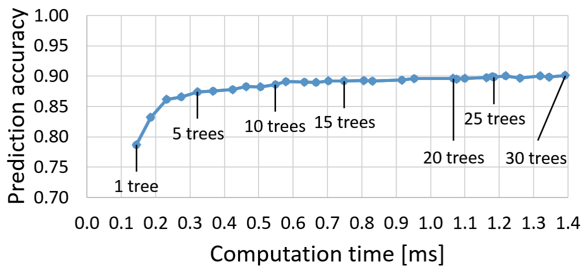


FIGURE 12. Relationship between prediction accuracy and prediction time corresponding to "Prediction using Random Forest" in Fig.11. The prediction time depends mainly on the number of trees in the random forest. The data shown in this graph are obtained by changing the number of trees.

computation time for processing the 3D point cloud and feature vector do not depend on the number of decision trees. This figure shows that the prediction accuracy exceeds more than 85% when the computation time is more than 0.2 [ms], and the accuracy gradually increases up to 90%. This result indicates that the proposed method can perform high-accuracy prediction in real-time.

VI. CONCLUSION

This paper proposed a prediction method for reaching motions used to achieve efficient collaborative tasks by a worker and an industrial robot (F-PREMO). The proposed method has the following three advantages: First, the proposed method does not require the posture of each joint of the worker to be estimated or a skeleton model of the worker. In other words, the method does not impose any restrictions such as attaching markers to the worker or placing a 3D sensor in front of the worker. The second advantage relates to the computation time and prediction accuracy of the proposed method. The experimental results show that the performance of the proposed method is similar to or better than existing methods, without imposing any restrictions for the placement of the 3D sensor. Finally, the proposed method does not require much time and cost to generate training data. In general, the training data utilized for supervised learning methods are created manually, requiring considerable time and cost. To solve this problem, the training data are generated semi-automatically in the proposed method using a random forest.

In future work, we will confirm that the proposed prediction method can be applied to worker motions other than the reaching motion. It is also necessary to consider the method that generates the trajectory of the manipulator by using the prediction obtained by the proposed method.

REFERENCES

[1] L. Rozo, S. Calinon, D. G. Caldwell, P. Jimenez, and C. Torras, "Learning physical collaborative robot behaviors from human demonstrations," *IEEE Trans. Robot.*, vol. 32, no. 3, pp. 513–527, Jun. 2016.

[2] M. Saida, Y. Hirata, and K. Kosuge, "Motion control of caster-type passive mobile robot with servo brakes," *Adv. Robot.*, vol. 26, nos. 11–12, pp. 1271–1290, Jul. 2012.

[3] T. Mukai, S. Hirano, H. Nakashima, Y. Kato, Y. Sakaida, S. Guo, and S. Hosoe, "Development of a nursing-care assistant robot RIBA that can lift a human in its arms," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Oct. 2010, pp. 5996–6001.

[4] A. Sharkey and N. Sharkey, "Granny and the robots: Ethical issues in robot care for the elderly," *Ethics Inf. Technol.*, vol. 14, no. 1, pp. 27–40, Mar. 2012.

[5] 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, Jan. 2017.

[6] B. Matthias, S. Kock, H. Jerregard, M. Kallman, and I. Lundberg, "Safety of collaborative industrial robots: Certification possibilities for a collaborative assembly robot concept," in *Proc. IEEE Int. Symp. Assem. Manuf. (ISAM)*, May 2011, pp. 1–6.

[7] FANUC Corporation. *Collaborative Robot CR-35iA*. Accessed: Apr. 28, 2020. [Online]. Available: https://www.fanuc.eu/be/en/robots/robot-filter-page/collaborative-robots/collaborative-cr35ia

[8] Kawasaki Heavy Industries. *DuAro1 Robot*. Accessed: Apr. 28, 2020. [Online]. Available: https://robotics.kawasaki.com/en1/products/robots/dual-arm-scara/duAro1/

[9] KAWADA Robotics Corporation. *Next Generation Industrial Robot NEXTAGE*. Accessed: Apr. 28, 2020. [Online]. Available: http://nextage.kawada.jp/en/

[10] ABB. *YuMi-Creating an Automated Future Together*. Accessed: Apr. 28, 2020. [Online]. Available: https://new.abb.com/products/robotics/industrial-robots/irb-14000-yumi

[11] *KUKA Roboter GmbH, LBR IIWA*. Accessed: Apr. 28, 2020. [Online]. Available: https://www.kuka.com/en-de/products/robot-systems/industrial-robots/lbr-iiwa

[12] *Rethink Robotics, BAXTER With INTERA 3*. Accessed: Apr. 28, 2020. [Online]. Available: https://www.rethinkrobotics.com/sawyer

[13] FUNUC. *Green is the New Yellow*. Accessed: Apr. 28, 2020. [Online]. Available: http://www.fanuc.eu/fi/en/who-we-are/news/green-is-the-new-yellow

[14] S. Haddadin, R. Belder, and A. Albu-Schäffer, "Dynamic motion planning for robots in partially unknown Environments," *IFAC Proc. Volumes*, vol. 44, no. 1, pp. 6842–6850, Jan. 2011.

[15] K. B. Kaldestad, S. Haddadin, R. Belder, G. Hovland, and D. A. Anisi, "Collision avoidance with potential fields based on parallel processing of 3D-point cloud data on the GPU," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2014, pp. 3250–3257.

[16] F. Flacco, T. Kroger, A. De Luca, and O. Khatib, "A depth space approach to human-robot collision avoidance," in *Proc. IEEE Int. Conf. Robot. Autom.*, May 2012, pp. 338–345.

[17] L. Balan and G. Bone, "Real-time 3D collision avoidance method for safe human and robot coexistence," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Oct. 2006, pp. 276–282.

[18] J. Kinugawa, A. Kanazawa, S. Arai, and K. Kosuge, "Adaptive task scheduling for an assembly task coworker robot based on incremental learning of Human's motion patterns," *IEEE Robot. Autom. Lett.*, vol. 2, no. 2, pp. 856–863, Apr. 2017.

[19] J. Elfring, R. van de Molengraft, and M. Steinbuch, "Learning intentions for improved human motion prediction," *Robot. Auto. Syst.*, vol. 62, no. 4, pp. 591–602, Apr. 2014.

[20] N. Chiba, S. Arai, and K. Hashimoto, "Feedback projection for 3D measurements under complex lighting conditions," in *Proc. Amer. Control Conf. (ACC)*, May 2017, pp. 4649–4656.

[21] S. Zhang, D. Van Der Weide, and J. Oliver, "Superfast phase-shifting method for 3-D shape measurement," *Opt. Express*, vol. 18, no. 9, p. 9684, Apr. 2010, doi: 10.1364/OE.18.009684.

[22] C. Kingkan, S. Ito, S. Arai, T. Nammoto, and K. Hashimoto, "Model-based virtual visual servoing with point cloud data," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Oct. 2016, pp. 5549–5555.

[23] Y. Yokokohji, Y. Kawai, M. Shibata, Y. Aiyama, S. Kotosaka, W. Uemura, A. Noda, H. Dobashi, T. Sakaguchi, and K. Yokoi, "Assembly challenge: A robot competition of the industrial robotics category, world robot summit—Summary of the pre-competition in 2018," *Adv. Robot.*, vol. 33, no. 17, pp. 876–899, Sep. 2019, doi: 10.1080/01691864.2019.1663609.

[24] D. Liu, S. Arai, Z. Feng, J. Miao, Y. Xu, J. Kinugawa, and K. Kosuge, "2D object localization based point pair feature for pose estimation," in *Proc. IEEE Int. Conf. Robot. Biomimetics (ROBIO)*, Dec. 2018, pp. 1119–1124.

[25] Y. Xu, S. Arai, F. Tokuda, and K. Kosuge, "A convolutional neural network for point cloud instance segmentation in cluttered scene trained by synthetic data without color," *IEEE Access*, vol. 8, pp. 70262–70269, 2020, doi: 10.1109/ACCESS.2020.2978506.

[26] H.-Y. Kuo, H.-R. Su, S.-H. Lai, and C.-C. Wu, "3D object detection and pose estimation from depth image for robotic bin picking," in *Proc. IEEE Int. Conf. Autom. Sci. Eng. (CASE)*, Taipei, Taiwan, Aug. 2014, pp. 1264–1269.

[27] M. Fujita, Y. Domae, A. Noda, G. A. G. Ricardez, T. Nagatani, A. Zeng, S. Song, A. Rodriguez, A. Causo, I. M. Chen, and T. Ogasawara, "What are the important technologies for bin picking? Technology analysis of robots in competitions based on a set of performance metrics," *Adv. Robot.*, vol. 34, nos. 7–8, pp. 560–574, 2020, doi: 10.1080/01691864.2019.1698463.2019.

[28] D. Liu, S. Arai, J. Miao, J. Kinugawa, Z. Wang, and K. Kosuge, "Point pair feature-based pose estimation with multiple edge appearance models (PPF-MEAM) for robotic bin picking," *Sensors*, vol. 18, no. 8, p. 2719, Aug. 2018, doi: 10.3390/s18082719.

[29] R. Luo and D. Berenson, "A framework for unsupervised online human reaching motion recognition and early prediction," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Sep. 2015, pp. 2426–2433.

[30] J. Mainprice, R. Hayne, and D. Berenson, "Predicting human reaching motion in collaborative tasks using inverse optimal control and iterative re-planning," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2015, pp. 885–892.

[31] C. Perez-D'Arpino and J. A. Shah, "Fast target prediction of human reaching motion for cooperative human-robot manipulation tasks using time series classification," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2015, pp. 6175–6182.

[32] H. C. Ravichandar and A. Dani, "Human intention inference and motion modeling using approximate E-M with online learning," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Sep. 2015, pp. 1819–1824.

[33] D. Kulić, W. Takano, and Y. Nakamura, "Incremental learning, clustering and hierarchy formation of whole body motion patterns using adaptive hidden Markov chains," *Int. J. Robot. Res.*, vol. 27, no. 7, pp. 761–784, Jul. 2008.

[34] J. Mainprice and D. Berenson, "Human-robot collaborative manipulation planning using early prediction of human motion," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Nov. 2013, pp. 299–306.
[35] Y. Jiang and A. Saxena, "Modeling high-dimensional humans for activity anticipation using Gaussian process latent CRFs," in *Proc. 10th Robotics: Sci. Syst.*, Jul. 2014, pp. 1–8.
[36] *Kinect for Windows v2 Sensor*. Accessed: Apr. 28, 2020. [Online]. Available: https://developer.microsoft.com/en-us/windows/kinect/
[37] R. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*, 2nd ed. Cambridge, U.K.: Cambridge Univ. Press, 2004, pp. 153–176.
[38] R. B. Rusu, "Semantic 3D object maps for everyday manipulation in human living environments," *KI Künstliche Intelligenz*, vol. 24, no. 4, pp. 345–348, Nov. 2010.
[39] A. AbuBaker, R. Qahwaji, S. Ipson, and M. Saleh, "One scan connected component labeling technique," in *Proc. IEEE Int. Conf. Signal Process. Commun.*, Nov. 2007, pp. 1283–1286.
[40] L. Breiman, "Random forests," *Mach. Learn.*, vol. 45, no. 1, pp. 5–32, 2001.
[41] A. Aldoma, Z.-C. Marton, F. Tombari, W. Wohlkinger, C. Potthast, B. Zeisl, R. B. Rusu, S. Gedikli, and M. Vincze, "Tutorial: Point cloud library: Three-dimensional object recognition and 6 DOF pose estimation," *IEEE Robot. Autom. Mag.*, vol. 19, no. 3, pp. 80–91, Sep. 2012.
[42] *Scikit-Learn: Machine Learning in Python*. Accessed: Apr. 28, 2020. [Online]. Available: http://scikit-learn.org/stable/

SHOGO ARAI (Member, IEEE) received the B.S. degree in aerospace engineering and the M.S. and Ph.D. degrees in information sciences from Tohoku University, Sendai, Japan, in 2005, 2007, and 2010, respectively.

From 2010 to 2016, he was an Assistant Professor with the Intelligent Control Systems Laboratory, Tohoku University. In 2016, he joined the System Robotics Laboratory, Department of Robotics, Tohoku University, as an Associate Professor, where he is currently an Associate Professor. His researches focus on the fields of robot vision, machine vision, 3D measurement, production robotics, networked control systems, and multiagent systems.

Dr. Arai received the 32th Best Paper Award from The Robotics Society of Japan, in 2019, the Certificate of Merit for Best Presentation from The Japan Society of Mechanical Engineers, in 2019, the Excellent Paper Award from The Institute of Systems from Control and Information Engineers, in 2010, the Best Paper Award Finalist at IEEE International Conference on Mechatronics and Automation, in 2012, the SI2019 Excellent Presentation Award from The Society of Instrument and Control Engineers, in 2019, the SI2018 Excellent Presentation Award from The Society of Instrument and Control Engineers, in 2018, the SI2017 Excellent Presentation Award from The Society of Instrument and Control Engineers, in 2017, the Graduate School Research Award from Society of Automotive Engineers of Japan, Inc., in 2007, and the Best Paper Award from FA Foundation, in 2019.

ANDREAS LENNART PETTERSSON received the B.S. degree in computer science and engineering from the KTH Royal Institute of Technology, Stockholm, Sweden, in 2013, the M.S. degree in information sciences from Tohoku University, Sendai, Japan, in 2016, and the M.S. degree in computer science and engineering with KTH, in 2018. From 2014 to 2016, he was a member of the Intelligent Control Systems Laboratory, Tohoku University. From 2017 to 2019, he was an IT Teacher with AW Academy Sweden. In 2019, he joined FRISQ AB, where he currently works within eHealth as a Software Developer. His research interests include computer vision, 3D sensors, point cloud processing, machine learning, and robotics.

KOICHI HASHIMOTO (Member, IEEE) received the B.S., M.S., and Ph.D. degrees in engineering from Osaka University, Osaka, Japan, in 1985, 1987, and 1990, respectively.

From 2000 to 2004, he was an Associate Professor with the Department of Mathematical Engineering and Information Physics, Graduate School of Engineering, The University of Tokyo. He is currently a Professor with the Intelligent Control Systems Laboratory, Graduate School of Information Sciences, Tohoku University. His major research interests include visual servoing, parallel processing, and biological systems.

Dr. Hashimoto is a member of RSJ, SICE, ISCIE, IPSJ, and JSME. He received the Best Mechatronics Paper Award at the IEEE International Conference on Mechatronics and Information Technology, in 2005, the Best Biomimetics Paper Award at IEEE International Conference on Robotics and Biomimetics, in 2006, and the Best Paper Award at IEEE International Conference on Mechatronics and Automation, in 2009. He is the Editor of the books *Visual Servoing: Real-Time Control of Robot Manipulators Based on Visual Sensory Feedback* (World Scientific) and *Control and Modeling of Complex Systems* (Birkhauser).