2025 IEEE/SICE International Symposium on System Integration (SII)
January 21-24, 2025, Munich, Germany

Dynamic bimanual human-to-robot object handovers using motion prediction deep neural networks

Matija Mavsar, Eiji Uchibe, Jun Morimoto, and Aleš Ude

Abstract— **Facilitating dynamic bimanual handovers between humans and robots is a complex endeavor that requires integrating human pose estimation, motion prediction, and generating appropriate trajectories for receiving robots. This study introduces a method designed to predict required bimanual receiver robot motion during handover tasks, leveraging pose estimation and motion prediction networks. Additionally, we propose a real-time control approach for a dual-arm humanoid robot to dynamically adjust its receiving trajectories. We evaluate the ability of neural networks to accurately predict receiver trajectories and thus improve the handover process. We compare long short-term memory (LSTM) and transformer architectures for motion prediction and also assess prediction accuracy of both absolute and relative receiving trajectories as well trajectories for each separate robot arm, and show that the use of absolute and relative coordinates is beneficial for generating more accurate receiver motions.**

I. INTRODUCTION

Recent advances in robotics have significantly improved human-robot collaboration, which is important for reducing worker stress and increasing productivity. Humanoid robots offer advantages due to their ability to perform bimanual tasks. In scenarios like object handovers, these robots can utilize their arms to facilitate transfer of large objects or objects that would be difficult to grasp with a single gripper.

In collaborative environments, e.g., in manufacturing, object handover is a frequent task where a giver passes an object to a receiver [1]. Realizing natural and efficient human-to-robot handovers requires accurate anticipation of the human intention. Dual-arm robots can additionally improve the handover process due to higher flexibility and have been used for many complex tasks, including handovers of large objects [2] and probabilistic dual-arm handover control [3]. For a safe and dynamic bimanual handover process, the predictions of human motions should be accurate as early as possible so the robot can adapt trajectories of both arms accordingly when newer predictions become available.

Recurrent long short-term memory (LSTM) neural networks [4] are well-suited for processing sequential data. They excel in modeling dynamic and time-dependent human motion, which makes them an attractive option for motion prediction in handover tasks [5]. Another network architecture suitable for processing of human motion data are transformer networks [6]. They use self-attention mechanisms to effectively capture long-range dependencies and are therefore similarly useful for predicting human motion, where understanding the context of movement phases is critical.

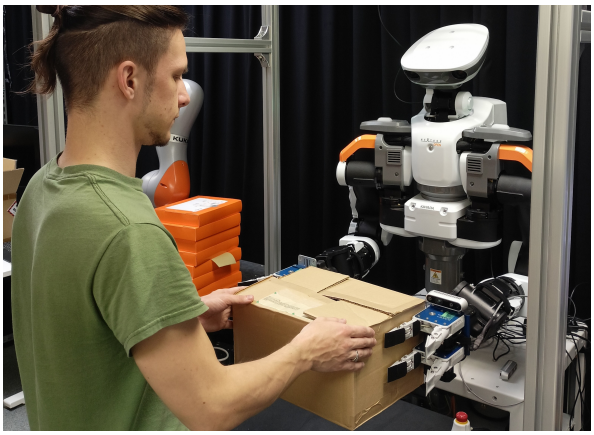

Fig. 1. Experimental setup for the bimanual object handover task. We employ a human pose estimation framework and robot receiver motion prediction neural networks to generate a robot trajectory, which is then executed by a humanoid robot and adapted in real time.

In this paper we propose to apply a state-of-the-art human pose estimation framework to obtain human body landmarks and employ them to predict the required receiver robot trajectories, encoded with Dynamic Movement Primitives (DMPs). LSTM and transformer networks are employed for this purpose. We compare two different approaches; in the first one we predict robot trajectories described with absolute and relative coordinates as proposed in [7], [8], while in the second approach we predict trajectories for each robot arm separately. Furthermore, we design an adaptive robot control system that switches between predicted DMP parameters in real time as the human handover motion is performed.

The main contributions of this paper are: 1) LSTM and transformer neural networks for trajectory prediction of bimanual robot motions based on input 3D human hand trajectories; 2) a database of bimanual human handover motions; and 3) the design and implementation of an adaptive bimanual handover control system using a humanoid robot.

II. RELATED WORK

Recent advancements in human-robot collaboration have focused on improving the fluidity and efficiency of interactions, including object handover scenarios [1]. The goal is to

M. Mavsar and A. Ude are with Jožef Stefan Institute, Dept. of Automatics, Biocybernetics, and Robotics, Ljubljana, Slovenia, also with ATR Computational Neuroscience Laboratories, Dept. of Brain-Robot Interface, Kyoto, Japan. matija.mavsar@ijs.si, ales.ude@ijs.si

E. Uchibe is with ATR Computational Neuroscience Laboratories, Dept. of Brain-Robot Interface, Kyoto, Japan. uchibe@atr.jp

J. Morimoto is with with the Graduate School of Informatics, Kyoto University, Kyoto, Japan, and also with ATR Computational Neuroscience Laboratories, Dept. of Brain-Robot Interface, Kyoto, Japan. xmorimo@atr.jp

enhance user satisfaction and ensure smooth transitions during these exchanges. In a handover, an object is transferred from a giver to a receiver, which can involve any combination of humans and robots. By refining the dynamics of these interactions, research in this context aims to create more intuitive and effective collaborative environments where robots seamlessly integrate into human tasks.

Several approaches that aim at controlling human-to-robot handover have been proposed, where they employ motion tracking equipment [9] or wearable sensor systems [10]. In the area of bimanual handovers, Vezzani et al. [11] have presented a pipeline that utilizes visual and tactile feedback to perform handover from one robot arm to another, while authors in [12] focus on enabling comfortable robot-to-human handovers. Research on bimanual handovers of large planar object has also been performed [2]. A method similar to ours was presented by Yan et al. [3], where they employ Gaussian mixture regression to predict handover locations and use kernelized movement primitives to adapt robot motion in real time. However, they do not make use of modern deep learning architectures such as LSTMs and transformer networks for motion prediction. Additionally, we propose the use of third-order DMPs for motion switching and assess both prediction of absolute and relative trajectories for dual arm movements, as well as separate trajectories of each arm.

We previously employed Dynamic Movement Primitives (DMPs) [13] to represent handover trajectories [5], since they allow modulation of the desired trajectory during the handover motion, which is required when the predicted handover location is estimated in real time. DMPs were also used by Widmann and Karayiannidis [14] to estimate handover location using an extended Kalman filter. Alternatively, Probabilistic Movement Primitives (ProMPs) [15] that maintain a distribution of trajectories can be used for predicting the handover trajectories that vary with respect to the final handover location. These methods, however, do not address the prediction of bimanual handover trajectories and their adaptation in real time.

While some methods perform motion recognition directly from input RGB-D videos [16], these approaches suffer from issues such as varying background and clothing of the observed humans. Such issues have been resolved by modern human body trackers such as Nuitrack [17], which was used in our work for 3D motion reconstruction.

In summary, the existing methods either only focus on the prediction of the motion without addressing the issue of adaptive real-time robot control, or employ classical machine learning methods. Conversely, in this paper we focus on enabling bimanual receiving motion prediction using novel deep learning architectures and compare different methods for description of trajectory coordinates.

III. HANDOVER TRAJECTORY PREDICTION

In this section we describe the proposed methods for real-time generation and adaptation of bimanual receiver robot motion and prediction of receiver trajectories based on input human motions, obtained with the Nuitrack pose estimation framework [17]. We explore two scenarios for predicting Dynamic Movement Primitive (DMP) parameters, used to encode receiver trajectories, to facilitate bimanual handover; firstly, we consider the prediction of parameters for absolute and relative trajectories, detailed in Section III-B, and secondly, we investigate the prediction of parameters separately for each robot arm. We compare the performance of both scenarios as well as of LSTM- and transformer-based networks for trajectory prediction.

A. 3D human hand position estimation

Nuitrack utilizes an RGB-D camera to provide 3D positions of 20 body landmarks in the camera coordinate system. In order to obtain a 3D trajectory for the i-th landmark in the robot coordinate system, we use the transformation from robot to camera coordinates, $\mathbf{c}_i = \mathbf{R}^{\mathrm{T}}\mathbf{l}_i - \mathbf{R}^{\mathrm{T}}\mathbf{t}$, where $\mathbf{R}$ and $\mathbf{t}$ are the extrinsic parameters, denoting rotation and translation of the robot coordinate system related to the camera coordinate system.

In our experiments, we employ the hand and wrist landmarks of each human arm to predict the receiving robot motion. These landmarks have indices 7, 8, 12, and 13. Thus we use the following data for neural network training:

$$\mathbf{p}(t) = [\mathbf{c}_7(t)^T, \mathbf{c}_8(t)^T, \mathbf{c}_{12}(t)^T, \mathbf{c}_{13}(t)^T]^T. \tag{1}$$

B. Representation using absolute and relative coordinates

We adopt a bimanual approach for handovers, since it enables the grasping of larger objects. While this approach may reduce the reachable workspace due to the simultaneous movement of both arms, it minimizes the need for specialized grippers. By utilizing both arms to grasp objects from either side, a wider variety of shapes and sizes can be reliably handled.

Bimanual robot motion can be simplified using absolute and relative coordinates [7], [8]. Rather than controlling each arm separately, absolute trajectories describe the motion of the midpoint between end effectors, while relative trajectories describe the pose of one end effector relative to the other. This approach simplifies control, reduces parameters, and aids in training trajectory prediction neural networks.

The absolute position $\mathbf{y}_a$ of the robot system can be defined as the midpoint between end-effector positions $\mathbf{y}_1$ and $\mathbf{y}_2$, while the absolute orientation $\mathbf{R}_a$ is defined as the orientation that is halfway between the orientations of both end effectors:

$$\mathbf{y}_a = \frac{\mathbf{y}_1 + \mathbf{y}_2}{2}, \quad \mathbf{R}_a = \mathbf{R}_1 \mathbf{R}^1_{\mathbf{k}^1_{12}}(\theta_{12}/2). \tag{2}$$

More specifically, $\mathbf{k}^1_{12}$ and θ_{12} are the unit vector and the angle that realize the rotation from the orientation of the first end effector to the orientation of the second one. Therefore, $\mathbf{R}^1_{\mathbf{k}^1_{12}}(\theta_{12}/2)$ corresponds to a rotation about axis $\mathbf{k}^1_{12}$ by half of an angle needed to align both end effectors.

Relative position $\mathbf{y}_r$ and relative orientation $\mathbf{R}_r$ are defined as the positional and rotational difference between one of the end effectors relative to another, $\mathbf{y}_r = \mathbf{y}_2 - \mathbf{y}_1$, $\mathbf{R}_r = \mathbf{R}^2_1$.

We control the robot by sending the desired absolute and relative positions and perform inverse kinematics with the

procedure described in [7]. We construct a Jacobian matrix $\mathbf{J}$ from an absolute Jacobian matrix $\mathbf{J}_a$ and a relative Jacobian matrix $\mathbf{J}_r$, where

$$\mathbf{J}_a = [\tfrac{1}{2}\mathbf{J}_1 \quad \tfrac{1}{2}\mathbf{J}_2], \quad \mathbf{J}_r = [-\mathbf{J}_1 \cdot \mathbf{J}_2], \quad \mathbf{J} = \begin{bmatrix}\mathbf{J}_a \\ \mathbf{J}_r\end{bmatrix}, \quad (3)$$

with $\mathbf{J}_1$ and $\mathbf{J}_2$ being Jacobian matrices for the two robot arms. The resulting matrix $\mathbf{J} \in \mathbb{R}^{12\times 12}$ is then used together with the error $\mathbf{e} = [\mathbf{e}_a^T, \mathbf{e}_r^T]^T$, comprising absolute and relative errors $\mathbf{e}_a$ and $\mathbf{e}_r$, respectively, to calculate inverse kinematics.

C. Receiver trajectory generation

We use Dynamic Movement Primitives (DMPs) [13] in Cartesian space [18] to specify robot motion for handover tasks. DMPs are ideal for HRC environments as they allow smooth transitions between trajectories when the desired motion changes, reducing the risk of collisions. In our setup, the motion prediction network generates a new receiver robot trajectory, encoded with DMPs, after processing each input sample, which requires that the robot is capable of smoothly switching from one trajectory to another. Unlike interpolation methods, which require re-computation for new target positions and may result in abrupt changes, DMPs dynamically adjust trajectories in real time while maintaining smoothness.

We employ the third-order DMP system proposed in [19], [20] for joint space trajectories to describe Cartesian space robot trajectories. In a Cartesian space DMP, the robot's motion is specified by its position $\mathbf{y}(t) \in \mathbb{R}^3$ and orientation trajectory $\mathbf{q}(t) \in \mathbb{R}^4$, where $\mathbf{q}(t)$ denotes the unit quaternions specifying the orientation trajectory at time t. We describe the position trajectory using third-order DMPs in the same way as previously reported in [5] for joint space trajectories.

A different representation is proposed for orientation trajectories. A DMP equation system for standard, i.e., second order DMPs in a unit quaternion space has been proposed in [18]. Based on this, the following equations for third-order unit quaternion DMPs are employed:

$$\tau\dot{\boldsymbol{\eta}} = \mathbf{K}_o 2\log(\mathbf{o} * \overline{\mathbf{q}}) - \mathbf{D}_o\boldsymbol{\eta} + \mathbf{K}_o\mathbf{f}_o(x), \quad (4)$$

$$\tau\dot{\mathbf{q}} = \frac{1}{2}\boldsymbol{\eta} * \mathbf{q}, \quad (5)$$

$$\tau\boldsymbol{\omega}_o = \mathbf{H}_o 2\log(\mathbf{g}_o * \overline{\mathbf{o}}), \quad (6)$$

where $\boldsymbol{\eta} \in \mathbb{R}^3$ and $\mathbf{o} \in \mathbb{R}^4$ are auxiliary variables, with $\boldsymbol{\eta}$ treated as a quaternion with scalar part 0 in Eq. (6), $\boldsymbol{\eta} = \tau\boldsymbol{\omega}$ is the scaled angular velocity of unit quaternion trajectory $\mathbf{q}$, and $\boldsymbol{\omega}_o \in \mathbb{R}^3$ is the angular velocity of the auxiliary unit quaternion trajectory $\mathbf{o}$. Operator $*$ denotes the quaternion product and $\overline{\mathbf{q}}$ is the conjugate of quaternion $\mathbf{q}$. Variables $\mathbf{q}_0$, $\mathbf{g}_o \in \mathbb{R}^4$ are the unit quaternions specifying start and end orientation, respectively, and $\tau > 0$ is a temporal scaling factor, usually set equal to the duration of motion. $\mathbf{K}_o$, $\mathbf{H}_o \in \mathbb{R}^{3\times 3}$ are spring matrices, $\mathbf{D}_o \in \mathbb{R}^{3\times 3}$ is a damping matrix, where we set $\mathbf{K}_o = K_o\mathbf{I}$, $\mathbf{D}_o = D_o\mathbf{I}$, $\mathbf{H}_o = H_o\mathbf{I}$, $D_o = 2\sqrt{K_o}$, $H_o = \sqrt{K_o}$, $K_o > 0$, which provides for the critical damping of the dynamic system. The phase variable x is used to remove direct time dependency, $\tau\dot{x} = -\alpha_x x$, where $\alpha_x > 0$ is a positive constant. The forcing term $\mathbf{f}_o$ from Eq. (4) is defined as a linear combination of M radial basis functions

$$\mathbf{f}_o(x) = \frac{1}{\sum_{i=1}^{M}\Psi_i(x)}\sum_{i=1}^{M} x\Psi_i(x)\mathbf{w}_i^p, \; \Psi_i(x) = \exp\left(-h_i(x-c_i)^2\right), \quad (7)$$

The third-order system (4) – (6) differs from the standard second-order quaternion DMP in [18] by smoothly transitioning the orientation trajectory to a new goal when the goal orientation $\mathbf{g}_o$ changes. Note that in (4) we have omitted the term $-x\mathbf{K}_o 2\log(\mathbf{o} * \overline{\mathbf{q}}_0)$, which causes problems when computing the quaternion DMP initial state.

In our handover experiments, the human worker moves an object using both hands towards the robot. As explained in Section III-A, the human worker motion is observed by an RGB-D camera and the motion prediction networks described in Section III-D continuously output the suggested receiver robot trajectory. For this reason, we must ensure that the robot motion remains smooth even when switching to another receiver trajectory.

Switching to a new positional DMP is performed in the same way as described in [5]. For the initialization of the variables of the orientational part of the trajectory, let's denote the current Cartesian DMP integration state as $\mathbf{q}_\mathrm{p}, \boldsymbol{\eta}_\mathrm{p}, \mathbf{o}_\mathrm{p}$ and the terms defined by the current and next Cartesian DMP (temporal scaling factor, forcing term, end and initial orientation) as $\tau_\mathrm{p}, \mathbf{f}_{o,\mathrm{p}}, \mathbf{g}_{o,\mathrm{p}}, \mathbf{q}_{0,\mathrm{p}}$ and $\tau_\mathrm{n}, \mathbf{f}_{o,\mathrm{n}}, \mathbf{g}_{o,\mathrm{n}}, \mathbf{q}_{0,\mathrm{n}}$, respectively. We should initialize the next DMP integration state $\mathbf{q}_\mathrm{n}, \boldsymbol{\eta}_\mathrm{n}, \mathbf{o}_\mathrm{n}$ so that the position, angular velocity and angular acceleration of the robot motion remain smooth, i.e., $\mathbf{q}_\mathrm{p} = \mathbf{q}_\mathrm{n}, \boldsymbol{\omega}_\mathrm{p} = \boldsymbol{\omega}_\mathrm{n}, \dot{\boldsymbol{\omega}}_\mathrm{p} = \dot{\boldsymbol{\omega}}_\mathrm{n}$. By taking into account that $\mathbf{q}_\mathrm{n}$ is a unit quaternion, we can use Eqs. (4) – (6) to compute the following initialization values for the integration of the next Cartesian DMP, starting at the current phase x:

$$\mathbf{q}_\mathrm{n} = \mathbf{q}_\mathrm{p}, \quad \boldsymbol{\eta}_\mathrm{n} = \frac{\tau_\mathrm{n}}{\tau_\mathrm{p}}\boldsymbol{\eta}_\mathrm{p} \quad (8)$$

$$\mathbf{o}_\mathrm{n} = \exp\left(\left(1 - \frac{\tau_\mathrm{n}}{\tau_\mathrm{p}}\right)\frac{\tau_\mathrm{n}\mathbf{D}_o\boldsymbol{\eta}_\mathrm{p}}{2\tau_\mathrm{p}\mathbf{K}_o} + \frac{\tau_\mathrm{n}^2}{\tau_\mathrm{p}^2}\left(\log(\mathbf{o}_\mathrm{p} * \overline{\mathbf{q}_\mathrm{p}}) + \frac{\mathbf{f}_{o,\mathrm{p}}}{2}\right) - \frac{\mathbf{f}_{o,\mathrm{n}}}{2}\right) * \mathbf{q}_\mathrm{n}. \quad (9)$$

D. Deep neural networks for handover prediction

Several architectures for sequential input processing have emerged recently. LSTM networks are effective for motion prediction due to their ability to store and use previous states, while transformer networks leverage attention mechanisms to highlight key parts of input sequences.

We propose two deep neural architectures, LSTM- and transformer-based, for predicting entire trajectories of the receiver robot during bimanual handover tasks. The outputs are the predicted trajectory parameters, which guide the robot in synchronizing its movements with the human worker's actions. We use 3D hand position data obtained through the Nuitrack system (see Section III-A) to train our models, which predict the receiver robot trajectories, either using absolute and relative coordinates or coordinates for each robot arm separately.

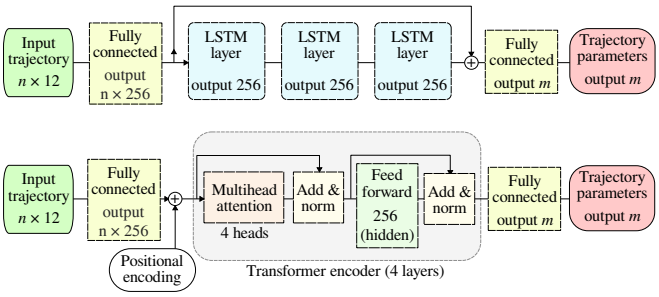

Fig. 2. The proposed LSTM (top) and transformer (bottom) networks for receiver trajectory prediction based on input trajectory of human hands. All layers except output layers are followed by a nonlinear activation function. The networks predict either DMP parameters of the absolute receiving robot trajectory along with the final arm distance and angle, or DMP parameters for each robot arm trajectory separately (see Section III-E).

The two proposed networks are shown in Fig. 2. Inputs are sequences of wrist and hand landmarks of both human arms, flattened into a combined landmark vector $\mathbf{p} \in \mathbb{R}^{12}$ (see Eq. (1)). Both networks are trained using partial and full sequences, which enables them to make predictions before the entire hand trajectory is available. The input data, i.e., each combined landmark vector $\mathbf{p}$, is first processed by the same fully connected network with 256 output neurons. For each subsequence of n landmark vectors, this results in an overall output of size $n \times 256$. The data preprocessed this way is then fed into either a transformer network or an LSTM network, as shown in Fig. 2.

When using the LSTM network (top architecture in Fig. 2), the outputs of the first fully connected layer are fed through three consecutive LSTM layers, which store information across different time steps. The elements of the input sequence are processed sequentially, with the internal states of the LSTM being updated at each step. The output of the final LSTM layer is added to the input of the first LSTM layer at each time step, creating a residual connection. The result is passed through a fully connected network to generate m parameters for the handover trajectory.

In the transformer model (bottom architecture in Fig. 2) we use only the encoder part of the transformer architecture, which is customary for non-sequence-to-sequence tasks. The input sequence, augmented with positional encodings, is processed by 4 layers of a transformer encoder with a multihead attention module, which handles variable-length sequences and captures dependencies. The last row of the output matrix is then passed through a fully connected network to produce the parameters describing the receiver robot trajectory.

E. Representation of bimanual handover trajectories

When predicting bimanual trajectories, we use either absolute and relative coordinates or we treat the trajectories of each robot arm separately.

In the first case, we only train the network that outputs DMP parameters of absolute position trajectory. All other motions, i.e., absolute and relative orientation motion and relative position motion, are generated by construction. This is done by first computing two additional parameters, namely the angle of the object orientation β and the final end-effector distance s. The final object orientation is obtained by assuming that the human hands are aligned with the table and rotated by an angle equal to the rotation of the object placed on the table, i.e., the rotation of the object around the z-axis in the robot coordinate system. In practice, we obtain this value by computing the angle of rotation about the z-axis for the difference vector of two hand positions. To also allow for cases where the worker tilts the box, an additional angular parameter would have to be included into the network's output to ensure that the receiving robot trajectories are adequate. The end-effector distance s is obtained by computing the distance between the two human hands at the end of motion plus a constant offset to ensure that the two robot arms are sufficiently apart to bimanually grasp the object. We obtain the following data for network training:

$$\mathbf{D}_a = \{\{\mathbf{p}_{i,j}\}_{i=1}^{L_j}, \mathbf{d}_{a,j}\}_{j=1}^{M}, \tag{10}$$

$$\mathbf{d}_{a,j} = \{\{\mathbf{w}_{a,k,j}\}_{k=1}^{N}, \mathbf{g}_{a,j}, \mathbf{y}_{0,a,j}, \tau_{a,j}, \beta_j, s_j\}. \tag{11}$$

M is the number of example sequences in the training dataset and $\mathbf{p}_{i,j}$ are the hand landmarks as defined in Eq. (1). $\{\mathbf{w}_{a,k,j}\}_{k=1}^{N}$, $\mathbf{g}_{a,j}$, $\mathbf{y}_{0,a,j}, \tau_{a,j}$ are the DMP parameters of absolute position trajectory, which are obtained by generating a minimum jerk trajectory from the fixed initial absolute position of the humanoid robot to the final absolute position of the humanoid robot. This final absolute position is computed from the final position of the human giver hands on each recorded trajectory, with an added offset to prevent collisions. N is the number of weights in the forcing term (7). β_j and s_j are the final object orientation and the distance between both end effectors as described above. Thus our networks are trained to output DMP parameters of the absolute position trajectory and the two values for the final angle and robot hand distance.

Next we compute a generic absolute orientation trajectory from the initial absolute orientation to the final absolute orientation, which is calculated from the predicted final angle and the assumption of the alignment of the grippers with the table. This trajectory is computed by SLERP (spherical linear interpolation) and encoded as a quaternion DMP. During the real object handover, the final absolute orientation is constantly updated using the predicted final angle parameter β. The goal quaternion DMP parameter $\mathbf{g}_o$ is set to this orientation and the absolute orientation DMP is adapted during motion to smoothly correct the trajectory towards the new goal.

For the relative position trajectory, we first compute a generic minimum jerk polynomial from the initial end-effector distance to an average final distance between the two human hands plus a constant offset. This polynomial defines the trajectory of the relative coordinate y, which is then encoded as a DMP. The other two relative coordinates are always equal to zero. During the real object handover, the final relative position is constantly updated using the predicted final distance between the two hands plus a constant

offset. The relative orientation of both end effectors is kept constant and can be set to an identity matrix.

When treating each robot hand trajectory separately, we compute the minimum jerk trajectory for each receiver robot arm from the fixed initial position to the final position of each human hand, with an added offset to prevent collisions with the transferred objects. The training data pairs for the prediction network are defined as

$$\mathbf{D}_s = \{\{\mathbf{p}_{i,j}\}_{i=1}^{L_j}, \{\mathbf{d}_{e,j}\}_{e=1}^{2}\}_{j=1}^{M}, \quad (12)$$

$$\mathbf{d}_{e,j} = \{\{\mathbf{w}_{e,k,j}\}_{k=1}^{N}, \mathbf{g}_{e,j}, \mathbf{y}_{0,e,j}, \tau_{e,j}\}, \quad (13)$$

where e is the index of the end effector (left or right), and $\{\mathbf{w}_{e,k,j}\}_{k=1}^{N}$, $\mathbf{g}_{e,j}$, $\mathbf{y}_{0,e,j}, \tau_{e,j}$ are DMP parameters of the position trajectory for the e-th robot arm.

The orientation trajectories are computed in a similar way as in the case when robot motion is defined by absolute and relative coordinates. We first compute generic orientation trajectories for both arms using SLERP and encode them as DMPs. Note that the final orientations of both robot hands are the same as the absolute orientation described above. Thus during the real object handover, we update the generic orientation DMP with the final goal orientation, which is the same as the absolute orientation, but is computed from the predicted final position of both robot hands.

We use a weighted variant of the mean squared error (MSE) for training the networks, as the input data are sequences. The loss for the j-th training sample in predicting absolute trajectories is computed as

$$\mathscr{L}_a(j) = \frac{1}{L_j}\sum_{i=1}^{L_j}\gamma_i\left(\sum_{k=1}^{N}||\mathbf{w}_{a,k,j}-\mathbf{w}^o_{a,k,i}||^2+\alpha_g\ ||\mathbf{g}_{a,j}-\mathbf{g}^o_{a,i}||^2+\right.$$
$$\alpha_y||\mathbf{y}_{0,a,j}-\mathbf{y}^o_{0,a,i}||^2+\alpha_\tau||\tau_{a,j}-\tau^o_{a,i}||^2+$$
$$\left.\alpha_s((\beta_j-\beta^o_i)^2+(s_j-s^o_i)^2)\right). \quad (14)$$

The parameters $\mathbf{w}_{a,k,j}, \mathbf{g}_{a,j}, \mathbf{y}_{0,a,j}, \tau_{a,j}, \beta_j$ and s_j are the ground-truth parameters from the training set (11), while $\mathbf{w}^o_{a,k,i}$, $\mathbf{g}^o_{a,i}, \mathbf{y}^o_{0,a,i}, \tau^o_{a,i}, \beta^o_i$ and s^o_i are the output parameters computed by the neural network for the i-th input $\mathbf{p}$. Additional weights $\alpha_g, \alpha_y, \alpha_\tau, \alpha_s > 0$ are used to balance the importance and scaling of parameters with different units. A logistic function is used to compute the weights γ_i to decrease the significance of early video frames and increase the significance of later frames.

The loss for separate arm trajectory prediction is calculated in a similar way, utilizing DMP parameters for both arm trajectories, and not including β and s parameters.

The output size m of the networks when predicting absolute trajectories, final angle and final end-effector distance is 84, consisting of 6 values for start and goal positions, 75 values for 25 basis functions, 1 value for τ and 2 values for final angle and end-effector distance. For separate trajectory prediction, the output size m is 164, due to predicting two DMP-encoded trajectories.

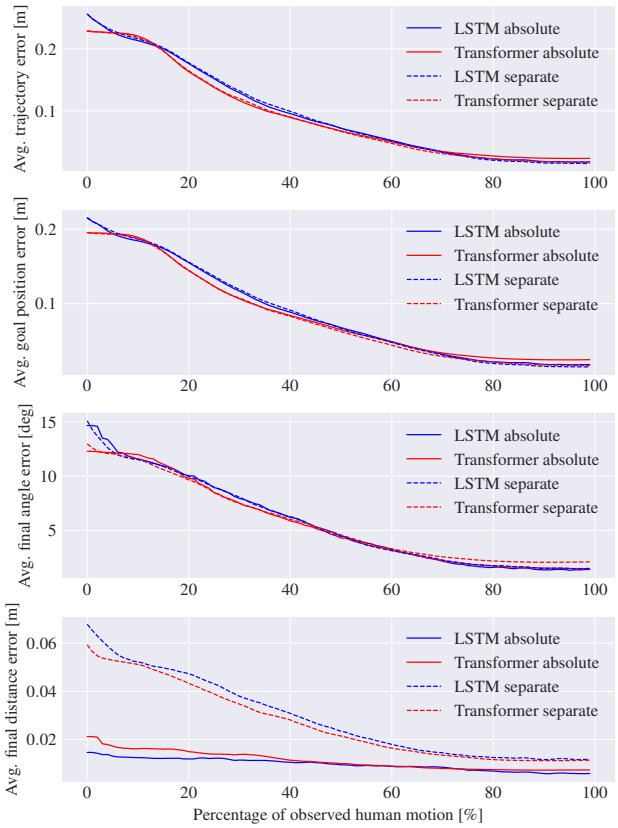

Fig. 3. Prediction errors of LSTM and transformer networks trained to predict either absolute and relative dual arm trajectories or two separate arm trajectories. Note that separate trajectory predictions were transformed into absolute positions, final angles and final hand distances for comparison.

IV. EXPERIMENTS

A. Data acquisition and processing

The experimental setup is shown in Fig. 1, where a human worker is passing an object to a humanoid robot. During data acquisition, the human was performing handover motions from a random initial position to a random location in the vicinity of the robot. At the start of the motion, a signal is used to trigger capturing of 3D body landmarks at 15 Hz, provided by the Nuitrack human pose estimation framework and an Intel RealSense depth camera D435. When the worker's hands reach a position near the robot and stop moving, the capture process is stopped. A database of 1151 samples was acquired in this way, consisting of sequences of estimated body landmarks $\mathbf{p}_{i,j} = \mathbf{p}_j(t_i), i = 1,\ldots,L_j$, $j = 1,\ldots,1151$. The rest of the training data in (10) and (12) was computed as described in Section III-E.

B. Results of the handover prediction networks

The prediction accuracy of LSTM and transformer motion prediction networks was evaluated across five dataset splits based on input motion percentage for absolute, relative, and separate arm trajectories. Fig. 3 shows the prediction errors for LSTM and transformer models and for both types of predicted trajectories. While average trajectory and average

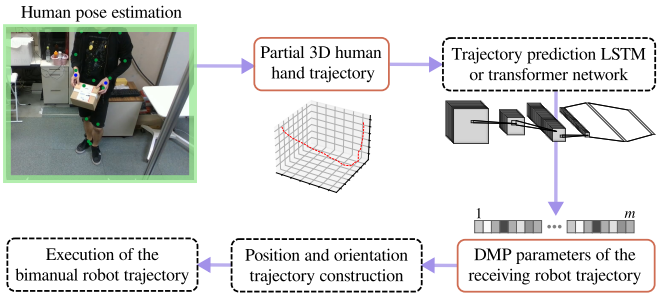

Fig. 4. Workflow of the handover process. The human hand landmarks are processed by a neural network to obtain DMP parameters, which are then used to construct receiving robot trajectories.

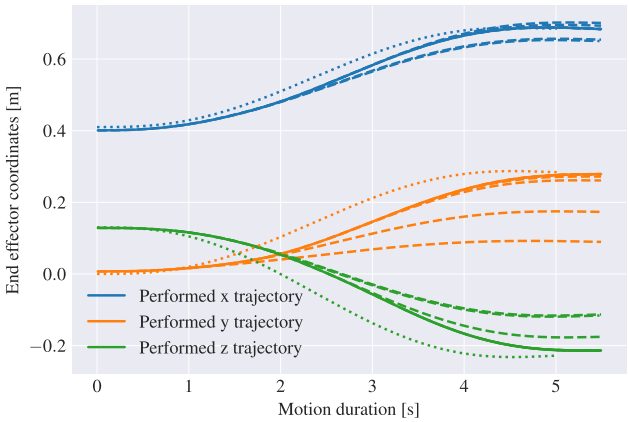

Fig. 5. Example positional absolute robot trajectory during a handover task. The solid lines represent the actual performed trajectory, the dashed lines represent the planned trajectory after each switch, while the dotted lines are the ground-truth trajectories.

goal errors are similar for all four models, LSTM networks performed slightly better towards the end of motion, while transformers achieved better accuracy in the middle part of motion. The use of absolute and relative coordinates instead of separate hand trajectories proved beneficial for prediction of the final distance error, which is crucial for ensuring that the robot does not collide with the object and grasps it successfully. The results consistently showed that as more of the input motion became available for analysis, the average accuracy of motion prediction networks improved for all evaluated architectures.

C. Bimanual handover experiment with a dual-arm robot

The LSTM network for absolute and relative trajectory prediction was employed to conduct several bimanual human-to-robot handover experiments using Nextage humanoid robot. The robot acted as the receiver, utilizing predictions from the LSTM network using absolute and relative coordinates. Fig. 4 shows the process workflow during the handover. A video of these experiments is included as supplementary material to this paper.

The robot was controlled by adapting predicted DMP-encoded trajectories in real time and also taking into account the predicted final angle and end-effector distance, as described in Section III-C. During the experiment, the giver, i.e., the human worker, passed the object with both hands from a random distant location towards a random position closer to the robot. The robot used both arms to perform the receiving motion and grasp the object passed by the human. An example of switching between different absolute positional DMPs is shown in Fig. 5. The resulting trajectory was smooth even though the predicted receiver trajectory changed several times. Additional examples of ground-truth and integrated trajectories to different goal locations are shown in Fig. 6. The LSTM processed the input trajectory points at a rate of 15 Hz, showing that it is viable for applications that require real-time performance.

Publishing of LSTM predictions started when the observed velocity of the human hand surpassed 0.02 m/s and halted when the velocity dropped below 0.02 m/s. These threshold values were empirically chosen to minimize noise effects.

The advantage of the proposed approach is that it allows the robot to begin moving as soon as it detects human motion, without having to wait for the human to bring the object within close reach. This allows for more fluid and efficient handover interactions.

V. CONCLUSIONS

We proposed a method to enhance bimanual human-to-robot handover tasks by predicting and adapting receiver robot motion. Two architectures, an LSTM with residual connections and a transformer network, were compared for predicting absolute and relative trajectories, as well as separate arm trajectories. While position errors were similar across models, using absolute and relative coordinates improved accuracy in predicting the final end-effector distance, crucial for successful handovers.

For future work we plan to analyze additional variants of LSTM- and transformer-based deep neural network architectures to find an optimal setup. For a more accurate object handover, we plan to integrate object localization to refine the final desired position and orientation for object handover. For increased safety it would also be beneficial to implement the detection of human finger placement on the object.

ACKNOWLEDGMENT

This work has received funding from the program group Automation, robotics, and biocybernetics (P2-0076) and young researcher grant PR-09781, both supported by the Slovenian Research Agency, from the DIGITOP Project funded by the Ministry of Higher Education, Science and Innovation of Slovenia, Slovenian Research and Innovation Agency and European Union—NextGenerationEU, and from EU's Horizon Europe grant euRobin (GA no. 101070596). It was further supported by project JPNP20006, commissioned by NEDO, by JSPS KAKENHI JP22H03669 and JP22H04998, by JST Mirai Program under grants JPMJMI21B1, by JST Moonshot R&D Program (Grant Number JPMJMS223B-3), and by Tateishi Science and Technology Foundation.

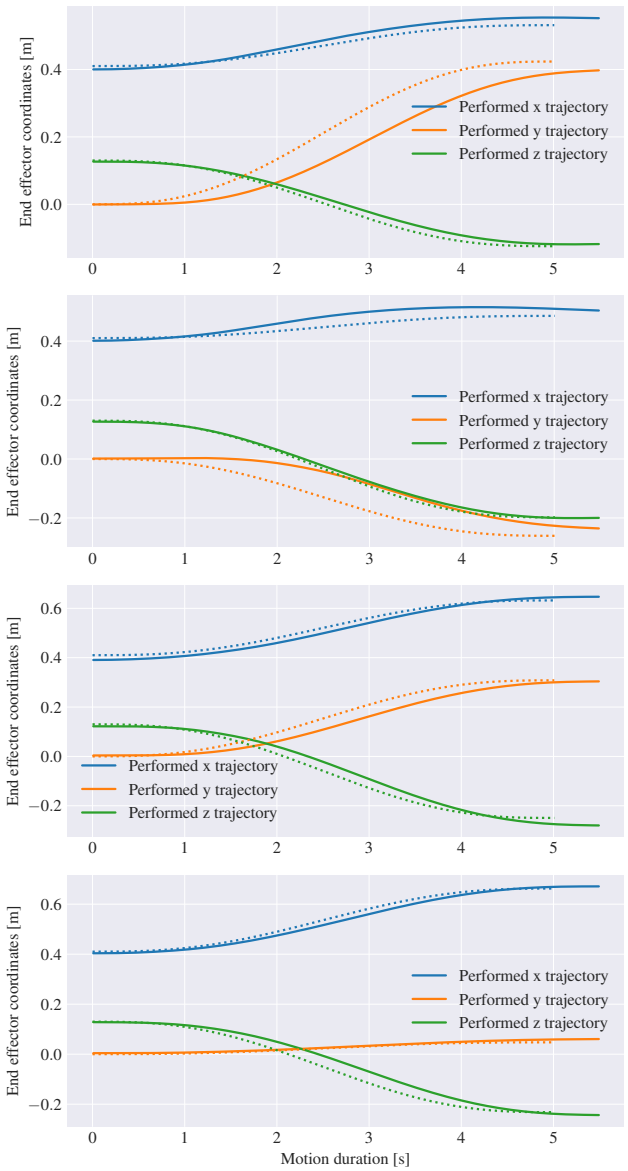

Fig. 6. Resulting positional absolute robot trajectories (solid lines) to four different goal handover locations. Dashed lines show the ground-truth trajectories.

REFERENCES

[1] V. Ortenzi, A. Cosgun, T. Pardi, W. P. Chan, E. Croft, and D. Kulić, "Object handovers: a review for robotics," *IEEE Trans. Robotics*, vol. 37, no. 6, pp. 1855–1873, 2021.

[2] W. He, J. Li, Z. Yan, and F. Chen, "Bidirectional human–robot bimanual handover of big planar object with vertical posture," *IEEE Transactions on Automation Science and Engineering*, vol. 19, no. 2, pp. 1180–1191, 2022.

[3] Z. Yan, W. He, Y. Wang, L. Sun, and X. Yu, "Probabilistic motion prediction and skill learning for human-to-cobot dual-arm handover control," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 35, no. 1, pp. 1192–1204, 2024.

[4] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, 1997.

[5] M. Mavsar, B. Ridge, R. Pahič, J. Morimoto, and A. Ude, "Simulation-aided handover prediction from video using recurrent image-to-motion networks," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 35, no. 1, pp. 494–506, 2024.

[6] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," *Advances in Neural Information Processing Systems*, vol. 30, 2017.

[7] P. Chiacchio, S. Chiaverini, and B. Siciliano, "Direct and inverse kinematics for coordinated motion tasks of a two-manipulator system," *Journal of Dynamic Systems, Measurement, and Control*, vol. 118, no. 4, pp. 691–697, 1996.

[8] B. Nemec, N. Likar, A. Gams, and A. Ude, "Bimanual human robot cooperation with adaptive stiffness control," in *IEEE-RAS International Conference on Humanoid Robots (Humanoids)*, Cancun, Mexico, 2016, pp. 607–613.

[9] J. Zhang, H. Liu, Q. Chang, L. Wang, and R. X. Gao, "Recurrent neural network for motion trajectory prediction in human-robot collaborative assembly," *CIRP Annals*, 2020.

[10] W. Wang, R. Li, Z. M. Diekel, Y. Chen, Z. Zhang, and Y. Jia, "Controlling object hand-over in human–robot collaboration via natural wearable sensing," *IEEE Transactions on Human-Machine Systems*, vol. 49, no. 1, pp. 59–71, 2019.

[11] U. P. G. Vezzani, M. Regoli and L. Natale, "A novel pipeline for bimanual handover task," *Advanced Robotics*, vol. 31, no. 23-24, pp. 1267–1280, 2017.

[12] S. E. Ovur and Y. Demiris, "Naturalistic robot-to-human bimanual handover in complex environments through multi-sensor fusion," *IEEE Transactions on Automation Science and Engineering*, pp. 1–12, 2023.

[13] A. J. Ijspeert, J. Nakanishi, H. Hoffmann, P. Pastor, and S. Schaal, "Dynamical movement primitives: learning attractor models for motor behaviors," *Neural Comput.*, vol. 25, no. 2, pp. 328–373, 2013.

[14] D. Widmann and Y. Karayiannidis, "Human motion prediction in human-robot handovers based on dynamic movement primitives," in *European Control Conf.*, 2018, pp. 2781–2787.

[15] G. J. Maeda, G. Neumann, M. Ewerton, R. Lioutikov, O. Kroemer, and J. Peters, "Probabilistic movement primitives for coordination of multiple human–robot collaborative tasks," *Autonomous Robots*, vol. 41, no. 3, pp. 593–612, 2017.

[16] W. Che and S. Peng, "Convolutional lstm networks and rgb-d video for human motion recognition," in *2018 IEEE 4th Information Technology and Mechatronics Engineering Conf. (ITOEC)*, 2018, pp. 951–955.

[17] 3DiVi Company, "Nuitrack." [Online]. Available: https://nuitrack.com/

[18] A. Ude, T. Petrič, B. Nemec, and J. Morimoto, "Orientation in Cartesian space dynamic movement primitives," in *Int. Conf. on Robotics and Automation (ICRA)*, China, 2014, pp. 2997–3004.

[19] S. Schaal, J. Peters, J. Nakanishi, and A. J. Ijspeert, "Learning movement primitives," in *Robotics Research, The Eleventh International Symposium*, 2005, pp. 561–572.

[20] B. Nemec and A. Ude, "Action sequencing using dynamic movement primitives," *Robotica*, vol. 30, no. 5, pp. 837–846, 2012.