Transformer-Based Prediction of Human Motions and Contact Forces for Physical Human-Robot Interaction

Alessia Fusco[1], Valerio Modugno[2], Dimitrios Kanoulas[2], Alessandro Rizzo[1], Marco Cognetti[3,4]

***Abstract*— In this paper, we propose a transformer-based architecture for predicting contact forces during a physical human-robot interaction. Our Neural Network is composed of two main parts: a Multi-Layer Perceptron called Transducer and a Transformer. The former estimates, based on the kinematic data from a motion capture suit, the current contact forces. The latter predicts – taking as input the same kinematic data and the output of the Transducer – the human motions and the contact forces over a time window in the future. We validated our approach by testing the network on directions of motions that were not provided in the training set. We also compared our approach to a purely Transformer-based network, showing a better prediction accuracy of the contact forces.**

I. INTRODUCTION

In the quest to integrate robots into daily life for societal benefit, their interaction with humans has become paramount. Beyond industrial and research settings, robots now engage with humans, prompting safety concerns.

In human-robot interaction, various strategies have been developed to prioritize safety. Initially, the focus was on *safe co-existence*, ensuring robots avoided or halted in human presence to prevent harm. With advancements in robotics, the *safe cooperation* paradigm emerged, where a robot helps a human with a task without any physical contact.

A significant challenge in human-robot interaction today is facilitating direct physical interaction between agents, termed as *safe physical interaction* [1]. This allows operators to guide robots in tasks, while robots gauge human intentions to adjust assistance. To achieve a behavior close to a human-human collaboration, robots are required to *predict human movements* and the forces exchanged. This capability would let robots preemptively address risks, ensuring human safety and reducing their task workload.

In human motion forecasting, numerous studies tackle this intricate issue. For example, a model-based method is presented in [2], where a dynamical system is used for predicting human behavior. However, most research has favored model-free techniques. For example, a study utilizes Hidden Markov Models (HMMs) for motion prediction [3]. Lately, the trend has moved towards Neural Networks (NNs) for this purpose, with [4] using a recurrent network to predict short-term human dynamics.

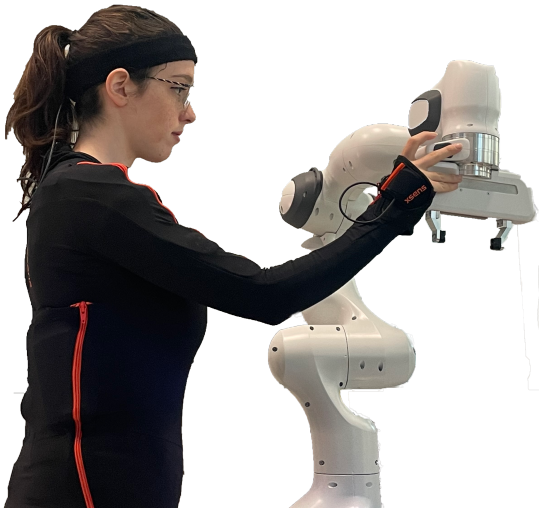

Fig. 1. The experimental setup for training our transformer network. The human is wearing a mocap suit that provides kinematic information about the human. The robot is controlled through an impedance controller, and it estimates, in real time, the contact forces. The user exchanges forces with the robot by moving its end-effector.

Traditionally, model-based techniques, particularly those founded on residual estimation [5], have dominated the landscape of contact force estimation. These methodologies rely on dynamical models to infer forces. Although model-free approaches for force estimation have begun to emerge [6], the concept of predicting the forces during contact remains a relatively underexplored area.

The inherent complexity in estimating and predicting forces exchanged during human-robot interactions emphasizes the pressing need for innovative methodologies that enhance the safety and efficacy of such interactions.

In this paper, we present a framework capable of forecasting both the forces exchanged between a human operator and a robot, and the associated human motions. To this aim, we utilize a Transformer network [7], which has proven its efficacy in time-series forecasting and outperforms traditional Recurrent Neural Network (RNN) models [8].

The principal contributions of this paper are:

- In our research, we introduce a method that forecasts not only the human movements but also the contact forces resulting from physical interactions between humans and robots in a future time horizon. This predictive skill can be leveraged by a controller, such as a model predic-

[1] Department of Electronics and Telecommunications, Politecnico di Torino, Torino, Italy. s303838@studenti.polito.it, alessandro.rizzo@polito.it

[2] RPL Lab, University College London, London, United Kingdom.{v.modugno,d.kanoulas}@ucl.ac.uk

[3] LAAS-CNRS, Université de Toulouse, CNRS, Toulouse, France, mcognetti@laas.fr

[4] Université Toulouse III - Paul Sabatier, Toulouse, France.

tive controller [9], enabling robots to more effectively anticipate human intentions. However, the controller is out of the scope of this paper.
- During the testing phase, our network is able to estimate the contact forces using only kinematic information about the human. This makes our framework more flexible, especially in situations where force/torque sensors might not be available.

The paper is organized as follows. In Section II, we provide an overview of a selection of related works. In Sec. III, we describe the framework setup (Sec. III-A), the data set description (Sec. III-B), the NN architecture (Sec. III-C) used for force and motion prediction, and its training procedure (Sec. III-D). In Sec. IV, we show the procedure for collecting the data (Sec. IV-A), the comparison description with another network (Sec. IV-B), and the performance of the proposed framework on a set of benchmarking motions (Sec. IV-C). Lastly, in Sec. V, we provide and discuss concluding observations and future directions.

II. Related Works

The body of literature on predicting safe human-robot physical interactions is vast. The works in this domain can be broadly classified into three categories: (i) studies concentrated on motion prediction alone; (ii) research delving into predicting interaction forces; and (iii) hybrid approaches that predict both motions and forces.

The largest number of contributions belongs to the first group. Many works focus on deriving a motion model for a human. In this context, the authors in [10] proposed a model based on social forces and environmental constraints. A deep neural network based on the social force model is presented in [11], while a physics-based network is introduced in [12]. In [3], an HMM is employed for motion sequence modeling. A multiple predictor for modeling human motion is proposed in [13], where the predictor is learned directly from the task the human is performing. Given the human motion model, some works focused on the control side. For example, a human-intention-based collision-avoidance algorithm is proposed in [14], while a variable impedance controller – where the intention of the human is modeled as an adaptive neural network – is proposed in [15]. A technique for the real-time estimation of the overloading joint torque for the human is proposed in [16], where the human is modeled as a humanoid robot. Finally, several approaches – named *shared autonomy* – focus on adapting the autonomy level of a robot based on the estimation of human actions/intentions. For example, an adaptive human-robot collaboration scheme is proposed in [17], while a game-theoretical approach is formulated in [18], that is integrated inside an impedance controller. A partially observable Markov decision process is employed in [19] for determining the expertise level of a human operator, in order to provide an assistance level in function of it. Another interesting approach is in [20], where wearable electromyography sensors are used for measuring human fatigue, and a variable impedance controller is employed for minimizing human effort. The reader is referred to [1] for a recent survey about shared autonomy. Our work extends the above-mentioned works that focused on the estimation of human motion, by adding to it the prediction of the contact forces.

To the best of the authors' knowledge, most of the existing works in the literature focus on the estimation of the current contact force, instead of predicting the future one, as we do in this paper: An observer that combines environmental forces and robot velocities is presented in [21]. In [6], the authors propose a vision-based deep-learning method for estimating the interaction forces between a robot and objects during grasping tasks. In [22], the authors introduce a deep learning-based algorithm that is capable of finding a mapping between an electromyography sensor and the one-step-ahead force magnitude that the human operator exchanges with the environment. One exception in the prediction of the contact forces is introduced in [23], a constant force model is used for dyadic cooperative object manipulation tasks.

Lately, several methods have been introduced that jointly predict human movements and contact forces, enhancing prediction accuracy in human-robot interaction scenarios. For example, a Recurrent Neural Network with Long Short-Term Memory units is used in [24] to predict the human position, velocity, and force that are used to estimate the parameters that feed an impedance controller. In this work, differently from our framework, no human data have been recorded and their method requires, at test time, an explicit force measure. Another approach is proposed in [25], where a Bayesian-based method estimates the stiffness and the motion intention of a human that is combined with an impedance controller that uses a neural network that compensates for the uncertainties in robotic dynamics. However, it does not consider the prediction of human future behavior. Finally, a reinforcement learning (RL) approach is presented in [26], where the human is not explicitly modeled but it is considered as an environmental uncertainty in the RL problem formulation. In contrast, our framework explicitly models human motion.

III. Method

Our system is composed of two main actors: a human and a robot, as shown in Fig. 1. The former is wearing a mocap suit, that allows the recording and the real-time streaming of kinematic information (position, velocity, and acceleration) associated with each joint and link of the human body. The latter is a manipulator that is controlled via a Cartesian impedance controller and it estimates the contact forces acting on its end-effector through a residual-based approach (see, e.g., [5]). The choice of the impedance controller is for allowing a human to safely interact with the manipulator, since the system is reduced to a mass-spring-damper system, and the robot is commanded to maintain the initial configuration. Our method encompasses two different stages: a training and a testing stage. During the former, we collect data regarding the human motions and the contact forces. In the latter, we forecast, within a prediction window, the same quantities using only the data from the mocap suit. The learning is performed using a Transformer architecture

which has shown great prediction performance for time series data [7].

A. Framework setup

In this section, we briefly describe the two sources from which we collected the data used for training our Transformer-based network: (i) the mocap suit, and (ii) the contact forces from the manipulator.

1) Mocap suit data: In our case, the mocap suit is an Xsens MVN motion capture system [27]. It is composed of 17 MTx sensors, that are continuously updated through the Xsens bio-mechanical model of the human body. The suit provides the position, velocity, and acceleration of each sensor, whose signal is filtered through a Kalman filter to improve the quality of the data. All the above-mentioned quantities are expressed in a common reference frame, that is defined during the calibration needed for using the suit.

2) Contact forces from the manipulator: The manipulator used in this paper is a Franka Emika Panda robot [28]. The contact forces are estimated through a residual-based approach [29], [30]. In particular, given the dynamic model of the robot

$$\boldsymbol{M}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + \boldsymbol{g}(\boldsymbol{q}) = \boldsymbol{\tau} + \boldsymbol{\tau}_{\text{ext}}$$

where $\boldsymbol{q} \in \mathbb{R}^N$ is the N-dimensional vector of the robot joints, $\boldsymbol{M}(\boldsymbol{q})$ is the positive-definite inertia matrix, $\boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\boldsymbol{q}}$ encompasses the Coriolis and centrifugal terms, $\boldsymbol{g}(\boldsymbol{q})$ represents the gravity term, $\boldsymbol{\tau}$ is the control torque, and $\boldsymbol{\tau}_{\text{ext}}$ is the resulting torque due to generalized contact forces acting on the robot from the environment. The residual vector can be defined as

$$\boldsymbol{r}(t) = \boldsymbol{G}_I \left(\boldsymbol{m} - \int_0^t \left(\boldsymbol{\tau} + \boldsymbol{C}^T(\boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\boldsymbol{q}} - \boldsymbol{g}(\boldsymbol{q}) + \boldsymbol{r} \right) ds \right)$$

where $\boldsymbol{m} = \boldsymbol{M}(\boldsymbol{q})\dot{\boldsymbol{q}}$ is the generalized momentum of the robot, and $\boldsymbol{G}_I > 0$ is a gain matrix. Computing the residual dynamics using the dynamic model of the robot results in

$$\dot{\boldsymbol{r}}(t) = \boldsymbol{G}_I(\boldsymbol{\tau}_{\text{ext}} - \boldsymbol{r})$$

Thus, for sufficiently large $\boldsymbol{G}_I$, we have $\boldsymbol{r} \approx \boldsymbol{\tau}_{\text{ext}}$, meaning that the residual vector approximates the external torques acting on the robot. The external forces can be easily computed from statics through

$$\boldsymbol{f}_{\text{ext}} = \boldsymbol{J}_c^{\dagger}(\boldsymbol{q})\boldsymbol{r} \tag{1}$$

where $\boldsymbol{J}_c(\boldsymbol{q})$ is the Jacobian at the contact point, and $\boldsymbol{J}_c^{\dagger}(\boldsymbol{q})$ is its pseudoinverse. In our case, since we assume that the contact is always at the robot end-effector, $\boldsymbol{J}_c(\boldsymbol{q})$ is the geometric Jacobian of the robot.

B. Data Set Description

In our framework, we record the Cartesian velocities and accelerations of three segments of the human arm interacting with the manipulator: the shoulder, the upper arm, and the forearm. These Cartesian data are expressed in an earth-fixed reference frame, defined by the mocap suit. Concurrently, we collect the joint positions of the shoulder and the elbow during the interaction. In the remainder of the paper, we will refer to the kinematic quantities recorded at a time instant t_i as $\boldsymbol{k}(t_i) = (\boldsymbol{q}^h(t_i), \dot{\boldsymbol{p}}^h(t_i), \ddot{\boldsymbol{p}}^h(t_i))^T$, where $\boldsymbol{q}^h(t_i) = (q^h_{s\phi}(t_i), q^h_{s\theta}(t_i), q^h_{s\psi}(t_i), q^h_{e\psi}(t_i), q^h_{e\theta}(t_i), q^h_{e\psi}(t_i))^T$, $\dot{\boldsymbol{p}}^h(t_i) = (\dot{\boldsymbol{p}}^h_s(t_i), \dot{\boldsymbol{p}}^h_u(t_i), \dot{\boldsymbol{p}}^h_f(t_i))^T$ and $\ddot{\boldsymbol{p}}^h(t_i)$ is the time derivative of $\dot{\boldsymbol{p}}^h(t_i)$ at t_i. In particular, the symbol $q^h_{xy}, x = \{s, e\}, y = \{\phi, \theta, \psi\}$ denotes the roll (ϕ), pitch (θ), yaw (ψ) angle of the shoulder (s) or elbow (e) joint. Moreover, $\dot{\boldsymbol{p}}^h_x, x = \{s, u, f\}$ (resp. $\ddot{\boldsymbol{p}}^h_x$) denotes the Cartesian velocity (resp. acceleration) of the shoulder (s), upper arm (u) and forearm (f) of the arm interacting with the robot.

On the manipulator side, we collect the three-dimensional force $\boldsymbol{f}_{\text{ext}}(t_i)$ from eq. (1) at each time instant t_i. These forces are expressed in the "stiffness frame" of the Panda robot which is positioned right before the robot end-effector.

All the above-mentioned data are indexed with $i \; s.t. \; t_i = i \cdot \delta t$ – with δt being the time step. Since Transformers were originally developed for natural language processing tasks, their inherent parallel processing can pose challenges in recognizing sequential dependencies, potentially limiting their ability to learn temporal patterns. To address this limitation, we incorporated timestamps into the input data using a method called *positional encoding*, enhancing the integration of spatial context into the sequence.

A summary of all the data forming the data set for training our model is given in Table III-C.

The input data undergoes a normalization step to ensure that every signal is on a consistent scale, preventing any disproportionate influences on the model's learning process. The chosen technique is the MinMax scaler, which maps the data to a fixed range (from 0 to 1): $x_{\text{MinMax}} = (x - x_{\min})/(x_{\max} - x_{\min})$, where x is the data to be normalized, and $x_{\min}, x_{\max}$ is its minimum and maximum value, respectively. This approach is often favored because it results in a smaller standard deviation, effectively mitigating the impact of outliers (in contrast to other popular normalization methods like Z-score). It is worth mentioning that we applied the normalization only to the input series, while the target series were not normalized. In fact, the latter could unintentionally provide the model with information about future events, which is a critical consideration to avoid during training.

C. Neural Network Architecture

As depicted in Fig. 2, our neural network is composed of two main parts: a Transducer and a Transformer. The former is a fully-connected Multi-Layer Perceptron (MLP) [31], that takes as input the kinematic data $\boldsymbol{k}(t_i)$ from the suit at the current time instant t_i, and estimates the force $\boldsymbol{f}_{\text{ext}}(t_i)$ exerted at t_i. The latter is a Transformer that takes a multivariate time series as input at time t_i. In particular, a sliding time window of width t_w selects a portion of the data, defining the input as $(\boldsymbol{k}(t), \tilde{\boldsymbol{f}}_{\text{ext}}(t), t)$ with $t \in [t_i - t_w, t_i]$ where $\tilde{\boldsymbol{f}}_{\text{ext}}(t)$ is the sequence of the estimated contact forces coming from the Transducer (since we assume that the Transformer has never access to the measured forces as input). In order to provide the sequence of estimated forces

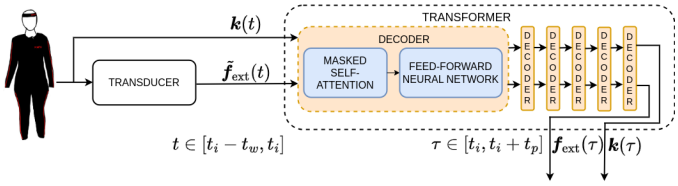

Fig. 2. The architecture of our neural network. A Transducer continuously estimates the contact force $\boldsymbol{f}_{\text{ext}}(t)$ based on the kinematic data $\boldsymbol{k}(t)$ within a time window $t \in [t_i - t_w, t_i]$, with t_i the current time instant. The output of the Transducer is sent to the Transformer that, fusing it with the kinematic data, produces a prediction of the contact forces $\boldsymbol{f}_{\text{ext}}(t)$ and of the kinematic data $\boldsymbol{k}(t)$, $t \in [t_i, t_i + t_p]$, with t_p the prediction time.

Data type	Description	Symbol	Units	Device	Input/Target
Joints	shoulder roll	$q^h_{s\phi}$	rad	mocap suit	Input&Target
	shoulder pitch	$q^h_{s\theta}$			
	shoulder yaw	$q^h_{s\psi}$			
	elbow roll	$q^h_{e\phi}$			
	elbow pitch	$q^h_{e\theta}$			
	elbow yaw	$q^h_{e\psi}$			
Cartesian vel.	shoulder	$\dot{\boldsymbol{p}}^h_s$	m/s	mocap suit	Input&Target
	upper arm	$\dot{\boldsymbol{p}}^h_u$			
	forearm	$\dot{\boldsymbol{p}}^h_f$			
Cartesian acc.	shoulder	$\ddot{\boldsymbol{p}}^h_s$	m/s^2	mocap suit	Input&Target
	upper arm	$\ddot{\boldsymbol{p}}^h_u$			
	forearm	$\ddot{\boldsymbol{p}}^h_f$			
Contact force		$\boldsymbol{f}_{\text{ext}}$	N	robot+eq. (1)	Target

TABLE I
THE DATA USED FOR THE TRAINING OF THE PROPOSED NEURAL NETWORK.

$\tilde{\boldsymbol{f}}_{\text{ext}}$ to the Transformer, we use the Transducer to preprocess the sequence of kinematic data as shown in Fig. 2. The Transformer target (i.e., the output) is $(\boldsymbol{k}(t), \boldsymbol{f}_{\text{ext}}(t))$ with $t \in [t_i, t_i + t_p]$, where t_p is the prediction time of the target. The time window slides by increasing the current window time t_i until the final time of the collected data is met.

Transformers are a class of deep-learning models that have gained immense popularity due to their ability to handle sequential data efficiently. Unlike traditional sequence models like Recurrent Neural Networks (RNNs) [32], Transformers do not rely on sequential processing. Instead, they employ a mechanism called *attention* [7] to capture dependencies between different elements in a sequence simultaneously. This parallelism is key to mitigating the issue related to short-term memory and vanishing gradient that commonly affects RNN such as Long-Short Term Memory (LSTM) networks and for dealing with long sequences of data.

The architecture that we have adopted for the Transformer is an adaptation of GPT-2 [33] for time series. Traditionally, a transformer model has two main components: an encoder and a decoder. In the encoder, there is a self-attention mechanism and a feedforward neural network. The former allows each element in the input to find the connection on different parts of the entire sequence, depending on 3 matrices called Queries ($\boldsymbol{Q}$), Keys ($\boldsymbol{K}$), and Values ($\boldsymbol{V}$). These matrices depend on the entire input sequence $\boldsymbol{X}$, and they are defined as $\boldsymbol{Q} = \boldsymbol{X} \cdot \boldsymbol{W}_Q$, $\boldsymbol{K} = \boldsymbol{X} \cdot \boldsymbol{W}_K$, and $\boldsymbol{V} = \boldsymbol{X} \cdot \boldsymbol{W}_V$, where $\boldsymbol{W}_Q, \boldsymbol{W}_K, \boldsymbol{W}_V$ are weight matrices that are learned during the Transformer training phase. Therefore, the self-attention output $\boldsymbol{Y}$ can be computed as

$$\boldsymbol{A}_w = \text{softmax}\left(\frac{\boldsymbol{Q} \cdot \boldsymbol{K}^T}{\sqrt{d}}\right)$$
$$\boldsymbol{Y} = \boldsymbol{A}_w \cdot \boldsymbol{V}$$

where $\boldsymbol{A}_w$ denotes the attention weights, and $\boldsymbol{d}$ is a scaling factor.

Conversely, the decoder incorporates an extra layer before the feedforward neural network known as Masked Self-Attention. The "masking" ensures that an element in a certain position in the sequence can only attend to past input data, preserving the autoregressive nature of the decoder.

The major difference between the original Transformer architecture and GPT-2 is that the latter relies on a decoder-only structure. Differently from the traditional Transformer which was originally designed for sequence-to-sequence tasks like language translation, GPT-2 is more versatile and it can be used for time series forecasting thanks to its generative capabilities. Thus, our architecture utilizes only decoder layers. In particular, we use six decoder blocks, arranged sequentially.

D. Neural Network Training

In this section, since our architecture is composed of two networks – the Transducer and the Transformer – we describe the two training procedures. In the initial phase, the Transducer network undergoes standalone training. Subsequently, during the Transformer's training process, the Transducer is connected to the Transformer as shown in Fig. 2 while maintaining its weight parameters constant.

1) Transducer training: The transducer is trained on a multivariate input composed entirely of kinematic quantities $\boldsymbol{k}$ from the mocap suit. The data set that we used for the training contains 86000 data points. Prior to feeding into the model, all data is shuffled and normalized using a MinMax scaler. We assess the model's performance during the training phase by comparing its predicted outputs to the values estimated by the Franka Emika manipulator, employing the Mean Square Error criterion.

2) Transformer training: The main issue affecting any NN during training regards the discrepancy between the training and the inference phases. During training, the model has typically access to the ground truth (i.e., the correct) output sequence. However, during inference, when the model is applied to unseen data, the prediction error is always larger. To mitigate this issue and to enhance the Transformer performance, we employed a technique called *scheduled sampling* [34]. This technique allows the Transformer to use, during the initial phases of the training, the ground-truth (i.e., true) data. However, as the training advances, the Transformer progressively integrates more of its own

predicted outputs into its input sequence, gradually reducing the usage of true values. This transition begins with a high probability of using true values as inputs and gradually shifts towards a greater reliance on the model's predictions. This strategy finds an equilibrium between equipping the model with valuable training insights and preparing it for real-world situations where actual values might not be accessible. The transition function we employed is a sigmoid with an inverse decay pattern

$$v = \frac{l}{l + \exp\left(\frac{n_{ep}}{l}\right)}$$

where v is the probability of selecting the ground-truth data, n_{ep} is the epoch number in the training set, $l \geq 1$ is a user-defined parameter for the speed of convergence.

Since our training set encompasses both kinematic and dynamic (i.e., forces) values, we opted for a weighted mean square error as a loss function. The decision is influenced by the consistent access the entire network has to the measured kinematic values, making them simpler to forecast. Furthermore, given the greater importance of force prediction for physical-human-robot interaction, we opt for a higher emphasis on the force prediction error.

$$loss = \frac{1}{n_t} \sum_{i=1}^{n_t} w_i \cdot (y_i - \hat{y}_i)^2$$

In the last equation, n_t represents the number of data in the training set, w_i represents the user-defined weight assigned to the i-th data point, y_i represents the actual value of the i-th data point, and $\hat{y}_i$ represents the predicted value of the i-th data point.

IV. EXPERIMENTS

In this section, we explain how the data are collected both for training and testing and we show some experimental results of using our framework for predicting physical human-robot interaction. In order to prove the architecture prediction capabilities and to show the effectiveness of the Transducer, we compare our network with a pure Transformer-based architecture.

A. Data Collection Protocol

The human – wearing the mocap suit – is positioned close to the robot and intentionally applies forces to the end-effector using their right arm. As mentioned before, the robot is controlled by means of an impedance controller, keeping constant the stiffness, the inertial, and the damping matrices for the entire data collection phase. The robot is commanded to maintain its initial configuration, waiting for a human to apply a force to its end-effector.

Different training motions were collected. In each of them, the robot starts from its resting configuration, and the human moves it along one of 8 equally spaced planar directions shown in Fig. 3. We gathered 40 samples for each direction of motion, encompassing a wide range of force intensities, spanning from 0 N to $\pm$40 N. This results in a robot end-effector displacement ranging from 0 cm to $\pm$25 cm. The duration of each individual sample averages 2.5 seconds.

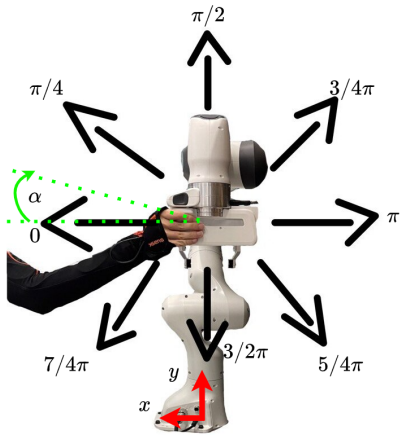

Fig. 3. Visualization of the 8 directions of motions – all in the manuscript plane – where our neural network was trained. The robot starts from its resting configuration and moves along one of the directions denoted with $\alpha = \{0, \pi/4, \pi/2, 3/4\pi, \pi, 5/4\pi, 3/2\pi, 7/4\pi\}$.

Throughout these interactions, once a force is detected, the kinematic human information described in Sec. III-A.1 and the estimated contact forces from Sec. III-A.2 are collected at 125 Hz, enabling the capture of rapid movements. In all our experiments, we maintained constant stiffness in the robotic arm during data collection. After the collection phase, the data, since they come from two different sources, were synchronized and filtered using a moving average filter. This process was employed to enhance the overall data quality and to mitigate the impact of noise interference.

It is important to stress that we assume that the human actually produces a motion during an experiment, trying to keep one of the above-mentioned directions. This is needed to ensure a correlation between kinematic and force data in our training set.

B. Comparison description

To perform a fair comparison, the two networks share the same input and output. In particular, exclusively kinematic data from the mocap suit are used as input, while the output is, in both cases, the kinematic quantities and contact forces within a time horizon. The pure-transformer-based network has the same structure as the one used in our framework (a GPT-2-based model with 6 decoder layers). We train the network using the same data set (as described in Sec. IV-A), using the same loss function and the same weights. In particular, we weight the error on the force values four times more than the error on the kinematic components. Moreover, we use a prediction window of 1.6 seconds and we employ an Early Stopping with a dropout rate equal to 0.35 to reduce overfitting. During the testing phase, we reduced the forecast window to 800 ms to optimize performance. The two networks are trained on the same computer (64 GB of RAM, Intel(R) i7 Core running at 3.20 GHz). We test the two networks on a dataset that comprises motion directions that were not present in the training set, as will be described in the subsequent section.

α	$\overline{q}_s^h$ [deg]	$\overline{q}_e^h$ [deg]	$\overline{\dot{p}}_s^h$ [m/s]	$\overline{\dot{p}}_u^h$ [m/s]	$\overline{\dot{p}}_f^h$ [m/s]	$\overline{\ddot{p}}_s^h$ [m/s^2]	$\overline{\ddot{p}}_u^h$ [m/s^2]	$\overline{\ddot{p}}_f^h$ [m/s^2]	$f_{ext,x}$ [N]	$f_{ext,y}$ [N]	$f_{ext,z}$ [N]
$4/3\pi$	16.404	5.465	0.013	0.021	0.063	0.197	0.082	0.223	0.016	0.226	0.421
$\pi/3$	1.508	3.315	0.010	0.013	0.017	0.042	0.100	0.322	1.959	1.919	2.503
$5/3\pi$	15.322	11.554	0.019	0.019	0.054	0.223	0.138	0.358	0.876	0.844	1.104
$8/9\pi$	12.482	5.244	0.034	0.043	0.077	0.337	0.136	0.367	0.020	0.328	0.229
$10/9\pi$	11.174	8.304	0.017	0.022	0.049	0.179	0.070	0.228	0.143	0.834	1.099
$\pi/6$	6.227	12.052	0.014	0.018	0.055	0.072	0.115	0.281	0.228	0.570	2.151
$11/6\pi$	0.973	5.128	0.023	0.028	0.040	0.050	0.057	0.185	0.097	0.316	1.425
avg	**9.156**	**7.295**	0.019	0.023	**0.051**	0.156	**0.099**	0.290	**0.477**	**0.720**	**1.276**

TABLE II

NN PREDICTION ERROR FOR THE ARCHITECTURE THAT COMPRISES THE TRANSDUCER AND THE TRANSFORMER. THE FIRST COLUMN INDICATES THE DIRECTION ALONG WHICH THE MOTION WAS PERFORMED (ANGLE α IN FIG. 3). THE BAR OVER $\overline{q}_s^h$ AND $\overline{q}_e^h$ INDICATES THE ERRORS AVERAGED ALONG THE ROLL, PITCH, AND YAW ANGLES LOCATED AT THE SHOULDER AND ELBOW, RESPECTIVELY. SIMILARLY, $\overline{\dot{p}}_s^h$, $\overline{\dot{p}}_u^h$, $\overline{\dot{p}}_f^h$ (RESP. $\overline{\ddot{p}}_s^h$, $\overline{\ddot{p}}_u^h$, $\overline{\ddot{p}}_f^h$) INDICATES THE ERRORS AVERAGED ALONG THE x, y, z COMPONENTS OF THE VELOCITY (RESP. ACCELERATION) FOR THE SHOULDER, UPPER ARM, AND FOREARM. THE LAST ROW IS THE AVERAGE OF THE QUANTITIES IN THE PREVIOUS ROWS. THE BOLD VALUES ON THE LAST ROWS ARE THE ERRORS THAT ARE LOWER FOR THIS ARCHITECTURE COMPARED TO TAB. III.

α	$\overline{q}_s^h$ [deg]	$\overline{q}_e^h$ [deg]	$\overline{\dot{p}}_s^h$ [m/s]	$\overline{\dot{p}}_u^h$ [m/s]	$\overline{\dot{p}}_f^h$ [m/s]	$\overline{\ddot{p}}_s^h$ [m/s^2]	$\overline{\ddot{p}}_u^h$ [m/s^2]	$\overline{\ddot{p}}_f^h$ [m/s^2]	$f_{ext,x}$ [N]	$f_{ext,y}$ [N]	$f_{ext,z}$ [N]
$4/3\pi$	15.621	6.435	0.013	0.019	0.058	0.188	0.082	0.195	1.091	11.231	4.403
$\pi/3$	10.724	8.762	0.012	0.031	0.110	0.057	0.319	0.565	5.039	9.851	10.729
$5/3\pi$	2.610	11.586	0.016	0.021	0.035	0.059	0.087	0.346	5.485	6.860	15.795
$8/9\pi$	21.478	8.599	0.014	0.018	0.062	0.242	0.205	0.305	7.155	12.634	3.926
$10/9\pi$	12.885	10.120	0.019	0.022	0.049	0.158	0.072	0.228	1.485	14.166	3.814
$\pi/6$	2.012	8.988	0.012	0.016	0.023	0.075	0.080	0.193	1.007	5.995	1.471
$11/6\pi$	1.202	8.027	0.011	0.017	0.030	0.050	0.071	0.184	1.903	7.853	4.137
avg	9.505	8.931	0.014	0.021	0.052	0.118	0.131	0.288	3.309	9.799	6.325

TABLE III

NN PREDICTION ERROR FOR THE ARCHITECTURE THAT COMPRISES ONLY THE TRANSFORMER. THE SYMBOLS ARE DESCRIBED IN TAB. II.

C. Results and discussion

The two networks are tested on 7 different motions which are not included in the training set. The testing motions are collected following the same procedure described in Sec. IV-A with the only difference that we gather only one motion for each testing direction. The reader is referred to the first column of Tab. II and Tab. III for the definition of the α values of Fig. 3 that characterize the testing motion.

In these tables, the prediction errors for the Transducer and Transformer architecture (Tab. II) and for the pure Transformer one (Tab. III) are reported. In the tables, the overbars on $\overline{q}_s^h$ and $\overline{q}_e^h$ represent the average errors computed across the roll, pitch, and yaw angles, for the shoulder and the elbow joints respectively. Likewise, $\overline{\dot{p}}_s^h$, $\overline{\dot{p}}_u^h$, and $\overline{\dot{p}}_f^h$ (and correspondingly, $\overline{\ddot{p}}_s^h$, $\overline{\ddot{p}}_u^h$, and $\overline{\ddot{p}}_f^h$) represent the mean errors computed along the spatial components (x, y, z) of velocity and acceleration pertaining to human shoulder, upper arm, and forearm, respectively[1]. The concluding row presents the average values derived from the aforementioned rows. The bold terms in Tab. II indicate errors where our presented model outperforms the purely transformer-based one.

By comparing Tab. II and Tab. III, it is clear that, when predicting the kinematic components, our model exhibits a slight error reduction. However, the improvement is not really significant. On the other hand, we can notice a significant improvement in the prediction of the contact forces when a transducer is added to the architecture.

It is important to notice that our framework is able to accurately predict the contact forces thanks to the constant robot stiffness, which allows to create a unique mapping between the human kinematics and the contact forces.

[1]Due to space constraints, we reported the averages in place of the individual vector components. However, their evolution is similar.

V. CONCLUSIONS

In this paper, we proposed a transformed-based architecture for predicting kinematic human data and interaction forces within a physical human-robot interaction context. Our approach consists of two main components: an MLP Transducer that estimates the contact forces based on the kinematic data from a mocap suit, and a Transformer that predicts the kinematic and force quantities within a future time horizon. The comparison with a pure Transformer-based network showed a similar behavior in predicting the kinematic data and an improvement in the prediction accuracy of the contact forces. Future works will focus on (i) considering a variable stiffness for the robot; (ii) developing other relevant directions of motion (e.g., out-of-plane motions) (iii) coupling the proposed approach with a control strategy in order to take into account future human behavior when planning the robot motion.

REFERENCES

[1] M. Selvaggio, M. Cognetti, S. Nikolaidis, S. Ivaldi, and B. Siciliano, "Autonomy in physical human-robot interaction: A brief survey," *IEEE Robotics and Automation Letters*, vol. 6, no. 4, pp. 7989–7996, 2021.

[2] M. Ackermann and A. J. Van den Bogert, "Optimality principles for model-based prediction of human gait," *Journal of biomechanics*, vol. 43, no. 6, pp. 1055–1060, 2010.

[3] H. Liu and L. Wang, "Human motion prediction for human-robot collaboration," *Journal of Manufacturing Systems*, vol. 44, pp. 287–294, 2017, special Issue on Latest advancements in manufacturing systems at NAMRC 45.

[4] P. Kratzer, M. Toussaint, and J. Mainprice, "Prediction of human full-body movements with motion optimization and recurrent neural networks," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*, 2020, pp. 1792–1798.

[5] A. De Luca and R. Mattone, "Sensorless robot collision detection and hybrid force/motion control," in *Proceedings of the 2005 IEEE International Conference on Robotics and Automation*, 2005, pp. 999–1004.

[6] D.-K. Ko, K.-W. Lee, D. H. Lee, and S.-C. Lim, "Vision-based interaction force estimation for robot grip motion without tactile/force sensor," *Expert Syst. Appl.*, vol. 211, no. C, jan 2023. [Online]. Available: https://doi.org/10.1016/j.eswa.2022.118441

[7] 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.

[8] Q. Wen, T. Zhou, C. Zhang, W. Chen, Z. Ma, J. Yan, and L. Sun, "Transformers in time series: A survey," in *International Joint Conference on Artificial Intelligence (IJCAI)*, 2023.

[9] C. E. Garcia, D. M. Prett, and M. Morari, "Model predictive control: Theory and practice—a survey," *Automatica*, vol. 25, no. 3, pp. 335–348, 1989.

[10] M. Luber, J. A. Stork, G. D. Tipaldi, and K. O. Arras, "People tracking with human motion predictions from social forces," in *2010 IEEE International Conference on Robotics and Automation*, 2010, pp. 464–469.

[11] A. Antonucci, G. P. R. Papini, P. Bevilacqua, L. Palopoli, and D. Fontanelli, "Efficient prediction of human motion for real-time robotics applications with physics-inspired neural networks," *IEEE Access*, vol. 10, pp. 144–157, 2022.

[12] Z. Zhang, Y. Zhu, R. Rai, and D. Doermann, "Pimnet: Physics-infused neural network for human motion prediction," *IEEE Robotics and Automation Letters*, vol. 7, no. 4, pp. 8949–8955, 2022.

[13] P. A. Lasota and J. A. Shah, "A multiple-predictor approach to human motion prediction," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*, 2017, pp. 2300–2307.

[14] L. Balan and G. M. Bone, "Real-time 3d collision avoidance method for safe human and robot coexistence," in *2006 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2006, pp. 276–282.

[15] Y. Huo, X. Li, X. Zhang, and D. Sun, "Intention-driven variable impedance control for physical human-robot interaction," in *2021 IEEE/ASME International Conference on Advanced Intelligent Mechatronics (AIM)*. IEEE, 2021, pp. 1220–1225.

[16] W. Kim, J. Lee, L. Peternel, N. Tsagarakis, and A. Ajoudani, "Anticipatory robot assistance for the prevention of human static joint overloading in human–robot collaboration," *IEEE Robotics and Automation Letters*, vol. 3, no. 1, pp. 68–75, 2018.

[17] S. Nikolaidis, D. Hsu, and S. Srinivasa, "Human-robot mutual adaptation in collaborative tasks: Models and experiments," *The International Journal of Robotics Research*, vol. 36, no. 5-7, pp. 618–634, 2017.

[18] Y. Li, K. P. Tee, R. Yan, W. L. Chan, and Y. Wu, "A framework of human–robot coordination based on game theory and policy iteration," *IEEE Transactions on Robotics*, vol. 32, no. 6, pp. 1408–1418, 2016.

[19] L. Milliken and G. A. Hollinger, "Modeling user expertise for choosing levels of shared autonomy," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2017, pp. 2285–2291.

[20] L. Peternel, N. Tsagarakis, D. Caldwell, and A. Ajoudani, "Robot adaptation to human physical fatigue in human–robot co-manipulation," *Autonomous Robots*, vol. 42, pp. 1011–1021, 2018.

[21] P. Hacksel and S. E. Salcudean, "Estimation of environment forces and rigid-body velocities using observers," in *Proceedings of the 1994 IEEE International Conference on Robotics and Automation*. IEEE, 1994, pp. 931–936.

[22] H. Su, W. Qi, Z. Li, Z. Chen, G. Ferrigno, and E. D. Momi, "Deep neural network approach in emg-based force estimation for human–robot interaction," *IEEE Transactions on Artificial Intelligence*, vol. 2, no. 05, pp. 404–412, jan 2021.

[23] E. Noohi, M. Žefran, and J. L. Patton, "A model for human–human collaborative object manipulation and its application to human–robot interaction," *IEEE transactions on robotics*, vol. 32, no. 4, pp. 880–896, 2016.

[24] H. Maithani, J. A. C. Ramon, and Y. Mezouar, "Predicting human intent for cooperative physical human-robot interaction tasks," in *2019 IEEE 15th International Conference on Control and Automation (ICCA)*. IEEE, 2019, pp. 1523–1528.

[25] X. Yu, W. He, Y. Li, C. Xue, J. Li, J. Zou, and C. Yang, "Bayesian estimation of human impedance and motion intention for human–robot collaboration," *IEEE Transactions on Cybernetics*, vol. 51, no. 4, pp. 1822–1834, 2021.

[26] A. Ghadirzadeh, J. Bütepage, A. Maki, D. Kragic, and M. Björkman, "A sensorimotor reinforcement learning framework for physical human-robot interaction," in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2016, pp. 2682–2688.

[27] M. Schepers, M. Giuberti, G. Bellusci *et al.*, "Xsens mvn: Consistent tracking of human motion using inertial sensing," *Xsens Technol*, vol. 1, no. 8, pp. 1–8, 2018.

[28] C. Gaz, M. Cognetti, A. Oliva, P. Robuffo Giordano, and A. De Luca, "Dynamic identification of the franka emika panda robot with retrieval of feasible parameters using penalty-based optimization," *IEEE Robotics and Automation Letters*, vol. 4, no. 4, pp. 4147–4154, 2019.

[29] E. Magrini, F. Flacco, and A. De Luca, "Estimation of contact forces using a virtual force sensor," in *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2014, pp. 2126–2133.

[30] F. Flacco, A. Paolillo, and A. Kheddar, "Residual-based contacts estimation for humanoid robots," in *2016 IEEE-RAS 16th International Conference on Humanoid Robots (Humanoids)*. IEEE, 2016, pp. 409–415.

[31] M. W. Gardner and S. Dorling, "Artificial neural networks (the multilayer perceptron)—a review of applications in the atmospheric sciences," *Atmospheric environment*, vol. 32, no. 14-15, pp. 2627–2636, 1998.

[32] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016, http://www.deeplearningbook.org.

[33] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, I. Sutskever *et al.*, "Language models are unsupervised multitask learners," *OpenAI blog*, vol. 1, no. 8, p. 9, 2019.

[34] S. Bengio, O. Vinyals, N. Jaitly, and N. Shazeer, "Scheduled sampling for sequence prediction with recurrent neural networks," *Advances in neural information processing systems*, vol. 28, 2015.