Toward Verifying the User of Motion-Controlled Robotic Arm Systems via the Robot Behavior

Long Huang, Zhen Meng, Zeyu Deng, Chen Wang, *Member, IEEE*, Liying Li, and Guodong Zhao, *Senior Member, IEEE*

***Abstract*—Motion-controlled robotic arms allow a user to interact with a remote real world without physically reaching it. By connecting cyberspace to the physical world, such interactive teleoperations are promising to improve remote education, virtual social interactions, and online participatory activities. In this work, we build up a motion-controlled robotic arm framework comprising a robotic arm end and a user end, which are connected via a network and responsible for manipulator control and motion capture, respectively. To protect the system access, we propose to verify who is controlling the robotic arm by examining the robotic arm's behavior, which adds a second security layer in addition to the system login credentials. We show that a robotic arm's motion inherits its human controller's behavioral biometric in interactive control scenarios. By extracting the angle readings of the robotic arm's all joints, the proposed user authentication approach reconstructs the robotic arm's end-effector movement trajectory that follows the user's hand. Furthermore, we derive the unique robotic motion features to capture the user's behavioral biometric embedded in the robot motions and develop learning-based algorithms to verify the robotic arm user to be one of the enrolled users or a nonuser. Extensive experiments show that our system achieves 94% accuracy to distinguish users while preventing user identity spoofing attacks with 95% accuracy.**

***Index Terms*—Cyber–physical security, interactive control, motion-controlled robot, robot behavior, user authentication.**

I. Introduction

CONSUMER robotic arms have been increasingly used for a multitude of applications for providing augmented interactions, including remote education, healthcare, research, industrial control, and social network [1]–[3]. By controlling a robotic arm, the user is able to conduct mission-critical and high-risk applications in the real world remotely without physically reaching it. Moreover, the fast development of motion capture techniques facilitates using hand motions to directly manipulate the robotic arm, which allows the user to focus more on the real-world operations than control. The outbreak of the COVID-19 pandemic further boosts the motion-controlled robotic arm applications when physical presences or face-to-face interactions might be under a high health threat.

A motion-controlled robotic arm is a type of intelligent cyber–physical system comprising two ends, which are connected by a local or wide area network. The user end is responsible for tracking the user's motions and issuing the corresponding control commands. The robotic arm end receives and executes the commands to perform tasks. In the meanwhile, the user observes the robotic arm's movement feedback and adjusts his/her hand motion accordingly for the interactive control, which facilitates performing fine tasks. But because a motion-controlled robot system involves sensors, actuators, networks, and computing devices at two ends, it suffers from more severe security threats from both cyberspace and the physical world. More specifically, an adversary might intrude and gain the system access by exploiting any of the above interfaces, which makes the defense hard.

The current access control of the robot system is achieved through the user-end authentication, in which the user needs to log into the system by entering correct passwords or physiological biometrics (e.g., fingerprints and faces) [4]–[6]. However, these traditional methods are independent of robot control. They could not guarantee that the robotic arm is consistently under the control of the enrolled user(s). For example, after the system access is granted, an adversary may override or modify the user's control commands anywhere before they are executed to hijack the robot. Moreover, the adversary might fool the authentication at the user end by forging the user's authentication entries.

To secure the access to such cyber–physical systems and defend against various hijacking attacks, we propose to continue to verify the user after the robot system is logged in, and the verification is done by examining the robot's behavior. This work, for the first time, demonstrates that the robotic arm could inherit much of its controller behavioral information in the interactive control environment. This is because the user's arm motions can be uniquely mapped into the robotic arm's movements, though the robotic arm works in a distinct kinematic mechanism with more joints and

Manuscript received 14 June 2021; revised 18 September 2021; accepted 16 October 2021. Date of publication 20 October 2021; date of current version 7 November 2022. This work was supported in part by the Louisiana Board of Regents under Grant LEQSF(2020-23)-RD-A-11, and in part by the Engineering and Physical Sciences Research Council under Grant EPSRC IAA EP/R511705/1. *(Corresponding author: Long Huang.)*

This work involved human subjects or animals in its research. Approval of all ethical and experimental procedures and protocols was granted by the Louisiana State University Institutional Review Board under Application No. IRB #4392.

Long Huang, Zeyu Deng, and Chen Wang are with the School of Electrical Engineering and Computer Science, Louisiana State University, Baton Rouge, LA 70820 USA (e-mail: lhuan45@lsu.edu; zdeng6@lsu.edu; chenwang1@lsu.edu).

Zhen Meng and Guodong Zhao are with the James Watt School of Engineering, University of Glasgow, Glasgow G12 8QQ, U.K. (e-mail: 2227311m@student.gla.ac.uk; guodong.zhao@glasgow.ac.uk).

Liying Li is with the School of Computing Science, University of Glasgow, Glasgow G12 8QQ, U.K. (e-mail: liying.li@glasgow.ac.uk).

Digital Object Identifier 10.1109/JIOT.2021.3121623

different arm lengths. Due to the individually unique human arm structures, strengths, and motion behaviors, the robotic arm exhibits the behaviors highly correlated with the user. We thus design the robotic arm end-user authentication approach, which tracks the robotic arm's movements to verify the robot-inherited human behavior. The proposed approach can detect impersonations and control command forgeries, regardless of where and how the attack originates.

In particular, our approach extracts the angle readings of the robotic arm's all joints to monitor its motions. Based on that, we reconstruct the end-effector movement trajectories using forward kinematics to describe the robotic arm's motion behaviors. Next, we derive the unique robotic motion features to capture the user's behavioral characteristics that are embedded in the robotic arm's motions. Furthermore, we develop a learning-based algorithm to first recognize which task the user is performing and then verify the identity of the user. We build up a real motion-controlled robotic arm framework to evaluate our authentication approach, which consists of a seven degree-of-freedom (DOF) robotic arm for control and six OptiTrack sensors for motion capture. We use a local area network to link the user end with the robotic arm end. We also implement a human–robot kinematic mapping algorithm to enable real-time robotic arm control via hand motions.

Our contributions are summarized as follows.

1) We develop a user authentication approach for motion-controlled robotic arm systems based on examining the robotic arm's movement behavior. The proposed approach ensures that the robotic arm is consistently under the control of the registered user, and the control hijacking can be detected.
2) This work demonstrates that people's motion behaviors in interactive control scenarios are individually unique, though different from their regular behaviors. Moreover, the robotic arm under control inherits such behaviors to show the per-user distinctive robot behaviors.
3) We develop an end-effector trajectory reconstruction scheme to capture the robotic arm's motion behaviors based on the status of robot joints. Moreover, we design the learning-based algorithms to both recognize the type of task the robot perform is performing and verify the identity of the robot's user.
4) We build up a real motion-controlled robotic arm platform based on a 7-DOF robotic arm, OptiTrack sensors, human–robot kinematic mapping algorithms, and network links. Extensive experiments with 30 participants are conducted to evaluate our proposed authentication system in interactive control scenarios. Results show that our system can accurately verify the user while preventing various impersonation and control hijacking attacks.

We arrange the remainder of this article as follows. We discuss the related work in Section II. In Section III, we introduce the framework of our system, the motion-controlled robotic arm platform, as well as the threat models. Then, we discuss the feasibility of robot behavior-based authentication in Section IV. We next present the data preprocessing, robotic motion features extraction, task recognition, and user authentication algorithm in Section V. The performance evaluation is reported in Section VI. Finally, we conclude the work in Section VII.

II. Related Work

The robot control systems are vulnerable to both cyber and physical threats. An attack could be launched by exploiting many attacking interfaces spanning from the controller, the open network, the robotic device, and computing resources. To address the severe cyber–physical security issues, active work is on monitoring the robot behavior to detect attacks at the robot level. Guo *et al.* [7] investigated the robot misbehaviors originated from a variety of sources. They propose to use a robot anomaly detection system to counteract various mobile robot attacks/failures, which exhibit either sensor or actuator misbehaviors. In industrial control systems, learning-based methods are employed to detect abnormal robot behaviors, including water treatment systems [8], chemical processes [9], and autonomous robots [10]. The robot anomalies are learned to further specify which part of the system is under attack. However, these anomaly detection methods did not consider the emerging interactive robot control, where the human is a significant factor in the control process. Moreover, they could not detect the user spoofing attacks, where an adversary takes over the robot to do valid tasks which may not exhibit any anomalies.

Active researches have been done on the robust interactive control for robots. Levine *et al.* [11] proposed an end-to-end deep learning method to coordinate vision and control, which maps the visual observations of raw camera images into the motor torque instructions for the robotic arm to perform tasks including grasping, pushing, screwing, etc. In [12], a convolutional neural networks (CNNs)-based approach is proposed for hand–eye coordination when the user controls a robotic arm to grasp objects. The approach predicts the robotic arm moving path based on the historical monocular images and achieves the continuous servoing mechanism. Finn and Levine [13] further developed a robot motion control method that combines deep action-conditioned video prediction models with model-predictive control. The method enables a robot to successfully push the new objects, which have not been seen in the training process. In [14], a human–machine–human control loop is established by a digital twin (DT) framework, which can achieve the low-latency visual feedback and short system reaction times between the operator and the robot. In [15], a novel DT prototype is developed, which explores the potential of deploying human–computer interaction technology to key scenarios, such as remote surgery. Lv *et al.* [16] combined the remote motion capture and the robotic arm control to build a remote healthcare system, which helps people with disabilities to pick up bottles easily. However, the access control of these systems still relies on verifying passwords or biometrics during the system login, which is limited to address severe cyber–physical threats.

This work proposes to leverage behavioral biometrics to verify who is controlling the robotic arm. There have been many studies on verifying the users via their motion behaviors.

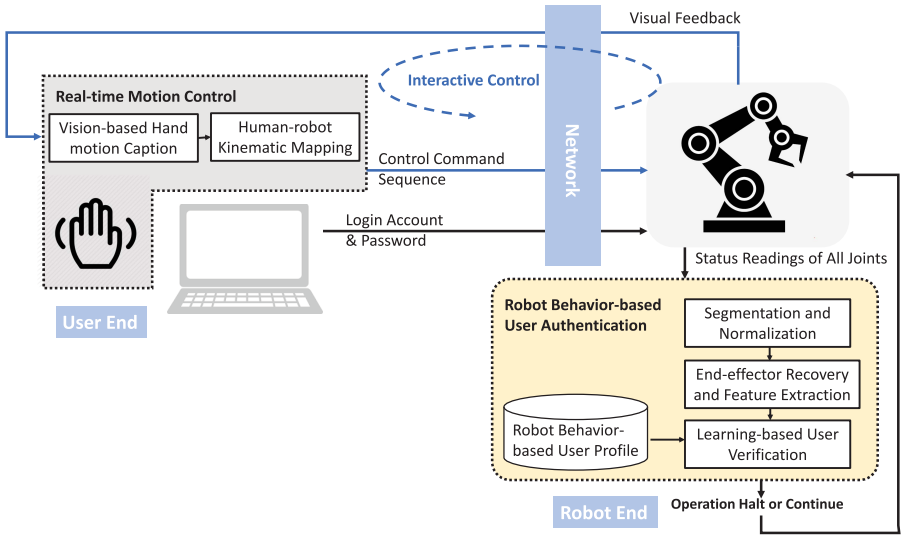


Fig. 1. Secure motion-controlled robotic arm framework with the robot behavior authentication deployed at the robot end.

Kinwrite uses a Kinect sensor to track the user's fingertip movements and utilizes the captured 3-D handwriting as a password to verify the user's identity [17]. Similarly, the Leap Motion camera tracks the user's hand motions in the 3-D space, and the hand gesture pattern is learned and classified by using machine learning methods [18], [19]. In addition to the vision-based approaches, the diverse sensors on wearable devices can be utilized to verify the user's hand gestures in the air, including inertial sensors [19]–[21] and electromyography (EMG) sensors [20]. Additionally, the recent wireless sensing approaches examine how the wireless signals (e.g., WiFi and mm-Wave) are interfered by the user's hand motions, and they extract behavioral characteristics from the signal interference for authentication [22]–[25]. However, these methods are based on the user's motion data captured at the user end, which are hard to secure the robotic arm at the other end of the network. It is still hard to know whether the control commands originate from the user and are not overridden or altered before they are executed.

III. Framework Overview and Attack Model

A. Framework Architecture

The objective of this work is to build a secure motion-controlled robotic arm framework, which provides the robot behavior-based user authentication in interactive control scenarios. In the logical layer, the framework has two components: 1) a motion-controlled robotic arm platform and 2) a robot behavior-based user authentication module. The complete framework is shown in Fig. 1, which has two physical components: 1) a user end for the real-time motion capture and 2) a robotic arm end for executing control commands. The robot behavior-based user authentication is deployed at the robot end. When accessing the robotic arm, the user enters the login account and password at the user end. Once the credentials are accepted, the user can manipulate the robotic arm with hand motions in real time. The user's hand motion is captured by a camera system and further transformed into control command sequences based on the human–robot kinematic mapping, which is sent to the robot end to execute. In the meanwhile, the user receives the visual feedback of the robotic arm's movement to perform the interactive control, which forms a control loop.

When the robotic arm receives and executes the control commands, the sampled angle values of its all joints are logged and input to our authentication approach. The joint readings are first segmented and normalized to represent the results of the user's every hand movement. Next, we reconstruct the moving trajectory of the robotic arm's end effector in the 3-D space based on forward kinematics. We then derive unique robotic motion features to capture the robotic arm's unique behavior associated with the user's behavioral biometric, which are fed into our learning-based algorithms, where task recognition and user identification are performed successively. When registering the system and using the robotic arm for the first time, the user's profile is created based on the robotic arm's motion behavior. When the user accesses the robotic arm later, the current robotic arm behavior is compared with the user's profile to verify whether the user's identity is as claimed. Based on the verification result, the robotic arm would continue to operate or reject the access and halt. If the access is rejected, the framework requires the relogin with correct credentials, and challenge/security questions and message authentication codes could be further required. The traditional login credentials and robot-behavior verification form the two security layers for the motion-controlled robotic arm framework.

B. Motion-Controlled Robotic Arm Platform Prototype

We first build the motion-controlled robotic arm platform by mapping human motions to robotic movements in real time, which allows users to interactively control a robotic arm by hand motions. As shown in Fig. 2, the platform consists of user-end devices and robotic arm end devices, which

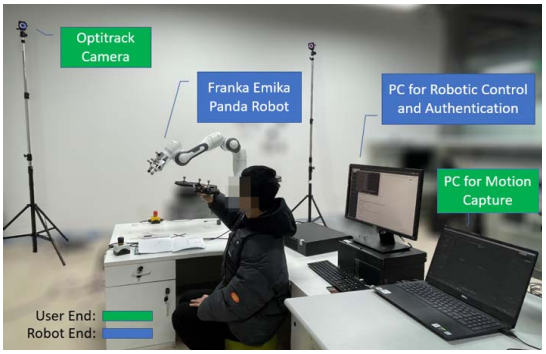


Fig. 2. Motion-controlled robotic arm platform to facilitate interactive control.

are connected by a local area network. The user-end devices capture the position and orientation information of the user's hand and map it to the corresponding target's motion in the robot's workspace. The robotic arm end devices perform the path-planning method, as well as executing the commands to achieve the goal for the end effector of the robot arm to follow the user's hand in real time. In the meanwhile, the user observes the robotic arm movements and adjusts his/her hand motions to achieve interactive control.

1) User-End Devices: At the user end, six OptiTrack cameras are deployed in a 5 m by 5 m circular area as the system input to capture the user's hand movements. Since this is the first work of authenticating a user based on robot behavior, we start with a high-precision OptiTrack to build the system. OptiTrack is one of the most precise motion capture cameras in the world, which can achieve 240 FPS capture rate and with positional errors less than $+/-0.20$ mm and rotational errors less than 0.5°. It captures the positions of passive markers attached on the object being detected rather than deploying the traditional way of using images process and extracting features of human motions. The device is widely used in a variety of scenarios requiring high-resolution motion capture, from real-time 3-D modeling in video games and filmmaking to remote surgery. Based on this high-performance motion capture module, we can explore the extent of using robot behavior to authenticate the user without suffering much from the motion capture errors. The user is required to wear a 3-D-printed glove attached with seven passive reflective markers, which allows the motion capture system to construct a rigid body of the user's hand and obtain high accuracy measurements of its position and orientation in six *degree of freedom* (6-DOF) with less than 0.2-mm errors [26]. When the user moves the hand in the air, the 3-D hand pose position represented by Cartesian coordinates and the orientation represented by quaternion are sampled at 120 Hz. Then, the personal computer (PC) performs the robotic arm inverse kinematics algorithm based on the sampled motion information and the configuration of the robot which is described by Denavit–Hartenberg parameters, converting the sampled information to seven target joint angles of robots [27]. In addition, to enable the robotic arm to follow human hand movements, we set the robotic arm and the motion capture system in the same coordination system by directly mapping the measured pose position and orientation of the hand in the robot's workspace. Here, they both use the right-hand coordination system, i.e., the x-axis faces to the left, the y-axis faces to the top, and the z-axis faces forward.

2) Robotic Arm-End Devices: At the robotic arm end, a Franka Emika Panda robot is used in the experiments [28]. It consists of a robotic arm and a control cabinet (Panda's control), which parses and executes the instructions from the PC. The robot arm has 7 DOF achieving up to 2 m/s end-effector speed and $+/-$ 0.1 mm repeatability, which ensured the applicability of industrial scenes and the accurate replication of human hand movements. The PC acts as the robotic arm controller for path planning. Considering the significant differences between robots and the human arm of physiology, kinematic character, joint, and velocity limits, it is challenging to directly mapping human movements to the robot arm. Here, to generate smoothly, continuously mimicry-based control trajectories, we design a path-planning strategy to relax the mapping constraints between the human's hand and end effector of robots by introducing the proportional–integral–derivative (PID) control algorithm, converting the received joint angle values into a series of robotic control angular velocity commands within trajectory limitations. At the robotic arm end, once the PC for robot control received the new target joint angle from the motion capture system, it will generate feasible joint angular velocity commands in each control loop based on the deviation between target joint value and current joint state. Then, the commands can be directly executed by the application programming interface (API) functions provided by libfranka, which is the C++ implementation of Franka control interface (FCI) [29]. The interface streams the joint angular velocity commands to the low-level engine driving the robot arm and reads the robot state to get the sensor data at 1 kHz. Since the updating rate of the target joint value is 120 Hz, a new target path planning is implemented at approximately every 8 ms.

3) Local Area Network: The flexible network architecture can be deployed in our system. For simplicity, the commercial Ethernet is used to connect the two ends and facilitate the data transmission from the user-end PC to the robot-end PC. We utilize the user datagram protocol (UDP) to support the high packet delivery rate. This protocol also reduces the queuing delay at the transmitter side and improves real-time tracking performance. The system can also be flexibly altered to deploy the TCP/IP protocol for higher reliability based on the specific application scenarios and performance requirements. Other high-speed networks, such as LTE and 5G, can also be used in our prototype.

C. Threat Model

This work aims to secure access to motion-controlled robotic arms. We focus on the adversaries, whose goal is to hijack the robotic arm or impersonate the target user to control it for gaining higher control privileges, causing physical damages, and achieving the repudiation of performing malicious tasks. These attacks could evade the existing anomaly detection methods [7], because an adversary could control the robotic arm to do valid tasks (e.g., grasping), which may

not show any anomalies at the robotic arm. As most robot systems require login authentication, we assume the adversary has obtained the user's login credentials and gained access to the robotic arm to perform various control tasks. Our robot behavior-based user authentication then works as the second security layer to verify who is controlling the robotic arm. In particular, we consider the following attacks.

1) *Zero-Knowledge Attack:* We consider an adversary who has no knowledge of our robot behavior-based authentication so that the adversary applies the program-generated control commands to control the robotic arm, where no human behavior is presented. Such attacks could be launched at the user end by compromising the user-end devices, in the network by packet spoofing or at the robot end by hacking the CAN bus.
2) *Random Impersonation Attack:* We consider an adversary who realizes the existence of our authentication approach and but has no knowledge of the user's control behavior. Then, the adversary could launch attacks at the user end and attempt to control the robotic arm with his/her motion behavior. Moreover, a stronger adversary could attack the network and the robotic arm end by substituting the user's control commands with his/her own to take over the robotic arm.
3) *Imitation Attack:* A more skilled impersonation attacker may obtain the opportunity to observe how the target user controls the robotic arm. Based on this prior knowledge, the adversary imitates the user's motion behavior to impersonate the user while controlling the robotic arm to perform tasks. By presenting the user's control behavior, the adversary attempts to cheat our authentication.

IV. Feasibility of Robot Behavior-Based Authentication

Before using the robot behaviors to verify users, it is significant to first answer the three questions.

1) Are people's behaviors under the interactive control environment still the same as their regular behaviors?
2) If no, are the user's interactive control behaviors sufficiently consistent and distinctive for user authentication?
3) Is a robotic arm able to inherit the user's behavioral information?

A. Interactive Control Behaviors Versus Regular Behaviors

To answer the first two questions, we conduct a feasibility study to investigate the user's regular behaviors and that in the interactive control scenarios. In particular, we ask one participant to repeatedly draw an *S* in the air 40 times in the two scenarios, respectively. In the noncontrol scenario, the participants freely move one hand in the air to write the letter just as they regularly do. In the interactive control scenario, the participants operate a 7-DOF robotic arm to write the same letter using our platform. We utilize the OptiTrack camera system to precisely capture the users' hand motion trajectory in the time series of 3-D coordinates. To figure out whether the interactive control and noncontrol scenarios contribute to differences in human behaviors, we compare the participant's motions both within each scenario and across the two scenarios, where their Euclidean distances are calculated after normalization.

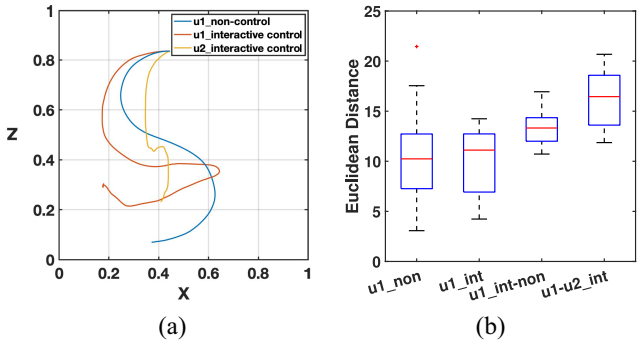


Fig. 3. Comparison of the user's motion behaviors under the interactive control and the noncontrol scenarios (illustrated with user drawing a 3-D *S* in the air). (a) User hand trajectories. (b) Distance distribution.

Fig. 3 presents the trajectories comparisons of a user's behavior in noncontrol and interactive control scenarios and that of two user behaviors in the interactive control scenario. Fig. 3(a) shows that when drawing an *S* in the air, the trajectories of User 1 in the two scenarios (i.e., "u1_noncontrol" and "u1_interactive control") are different. Moreover, the trajectories of the two users both in the interactive control scenario (i.e., "u1_interactive control" and "u2_interactive control") are also different. Fig. 3(b) presents the statistical analysis when each user is asked to repeat the experiment 20 times for each scenario, respectively. We find that the interclass Euclidean distances of *u1_int-non* are much higher than that of the intraclass comparison *u1_non*. This result indicates the user's motion behaviors under the interactive control scenario differ from their regular behaviors. Moreover, we find that the intraclass comparison *u1_int* shows the low Euclidean distances comparable to *u1_non*, which indicates that the user's motion behaviors under the interactive control scenario are consistent. We take one step further to compare two participants' interactive control behaviors when they draw the same curve (e.g., *S*) and find that their Euclidean distances *u1-u2_int* are very high. The results show that each user has an individual way of operating the robotic arm. The behavioral uniqueness and consistency demonstrated above confirm the feasibility to distinguish people through their interactive control behaviors.

B. Human Behavioral Biometrics Embedded in Robot Motions

We next answer the third question based on similar experiments in the interactive control scenario, where the same OptiTrack cameras are utilized to record both the user's hand movement curve and the robotic arm end-effector trajectory as shown in Fig. 4. We find that the robotic arm follows the path of the user's hand and draws a similar "triangle" curve. The reason is that both the user and the robotic arm try to adapt to each other's motion in the interactive control scenario. The robotic arm then carries a portion of the user's behavioral biometrics when replicating the user's hand motions. This

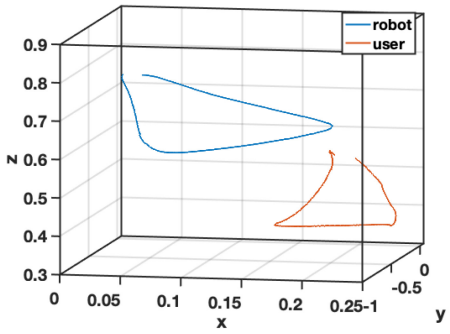


Fig. 4. Comparison of the user's hand and the robotic arm end-effector raw trajectories of drawing a triangle. Both are obtained by OptiTrack cameras.

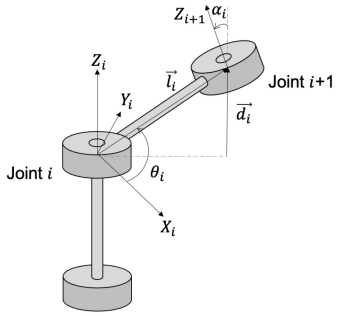


Fig. 5. Coordinates and robot kinematics parameters for a general manipulator model.

serves as the basis for us to design the robot behavior-based authentication approach.

However, we also notice multiple challenges when extracting human behavioral biometrics from the robot motions. We find that the robotic arm does not completely replicate the movement trajectory of the user's hand. It does not move in the same coordinate as the user's hand, and the movement scale is different. This is because the 7-DOF robotic arm has a different kinematic mechanism compared to the human arm. Furthermore, when following the user's hand motions, the sampling errors of the motion capture devices, the network traffic delays, and the user's sudden hand motion changes all cause the robotic arm to exhibit many additional or redundant movements. In addition, in practical scenarios, we leverage the robotic arm joint states to derive its motions, which exempts the overhead of additional camera systems at the robot end, but the derived robotic arm trajectory suffers from more noises. Therefore, distinguishing users via the robot behaviors is more difficult than directly using human motions.

V. User Authentication Design

A. Reconstructing the Robotic Arm's End-Effector Trajectory

Our authentication approach utilizes the sampled angle values of the robotic arm's joints to verify the user. To capture how the robotic arm follows the user's motion and inherits the user's behavior, we reconstruct the end-effector trajectory of the robotic arm from these joint states based on the forward kinematics. The robotic arm's motion mechanism is modeled as shown in Fig. 5. For each joint such as Joint i, Z_i is its rotation axis, and X_i and Y_i are on its rotation plane. X_i is defined to be in the same direction of the arm link $\vec{l_i}$ (toward Joint $i+1$) at the beginning of the robotic arm control. When the robotic arm starts to move, the coordinate of Joint i can be determined by all of its $i-1$ prior joints. We use the Denavit–Hartenberg [30] parameters $|\vec{l}|$, $\vec{d}$, α, and θ, namely, the link length, link offset, link twist, and joint angle to describe the robot kinematics. In particular, the link length $|\vec{l_i}|$ is the distance between Joint i and Joint $i+1$, while the link offset $\vec{d_i}$ is measured against the Z_i-axis. Fig. 5 illustrates the joint status at time t, when the angle between Z_i and Z_{i+1} is α_i, and the arm link $\vec{l_i}$ rotates with the angle θ_i. The transformation matrix from Joint i to Joint $i+1$ can thus be expressed as

$$
\begin{aligned}
{}^{i}T_{i+1} &= R_x(\alpha_i)D_x(a_i)R_z(\theta_i)Q_i(d_i) \\
&= \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\alpha_i & -\sin\alpha_i & 0 \\ 0 & \sin\alpha_i & \cos\alpha_i & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & a_i \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \\
&\quad \begin{bmatrix} \cos\theta_i & -\sin\theta_i & 0 & 0 \\ \sin\theta_i & \cos\theta_i & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \\
&= \begin{bmatrix} \cos\theta_i & -\sin\theta_i & 0 & a_i \\ \sin\theta_i\cos\alpha_i & \cos\theta_i\cos\alpha_i & -\sin\alpha_i & -\sin\alpha_i d_i \\ \sin\theta_i\sin\alpha_i & \cos\theta_i\sin\alpha_i & \cos\alpha_i & \cos\alpha_i d_i \\ 0 & 0 & 0 & 1 \end{bmatrix}
\end{aligned}
$$

where R represents the rotation matrix, and D and Q denote the translation. The end effector $joint_n$ is obtained by multiplying all the transformation matrix ${}^{i}T_{i+1}$ starting from $joint_0$

$$
{}^{0}T_n = {}^{0}T_1 {}^{1}T_2 2T_3 \cdots {}^{n-1}T_n. \tag{1}
$$

The 3-D coordinate of the end effector ${}^{n}pos$ is calculated based on the reference point ${}^{0}pos = (x_0, y_0, z_0)^T$, which is set as

$$
{}^{n}pos = \begin{bmatrix} x_n \\ y_n \\ z_n \\ 1 \end{bmatrix} = {}^{0}T_n \begin{bmatrix} x_0 \\ y_0 \\ z_0 \\ 1 \end{bmatrix} = {}^{0}T_n \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}. \tag{2}
$$

The derived 3-D coordinate time series $[{}^{n}pos_0, {}^{n}pos_1, \ldots, {}^{n}pos_t]$ describe the movements of the robotic arm, based on which we further extract the robotic arm's unique behaviors.

B. Data Calibration and Normalization

The user's hand movement scales, varying speeds, and different orientations and distances to the motion capture device all affect the states of the robotic arm joints. The derived robotic arm movement trajectories must be calibrated and normalized to reduce these impacts so that the minute behavioral differences among users could be captured from robot behaviors. For this purpose, we apply the rescaling and the axis alignment schemes to calibrate the reconstructed end-effector trajectories. In particular, the end-effector trajectory

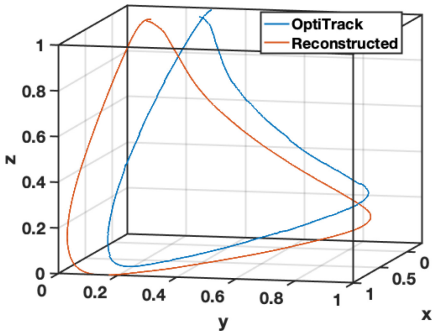


Fig. 6. OptiTrack recorded robotic arm movement and reconstructed movement, both are calibrated and normalized. The task is drawing a triangle.

TABLE I
FEATURES EXTRACTED FOR DTW

Position	$pos(t) = (pos_x(t), pos_y(t), pos_z(t))^T$
Position Difference	$pos\_diff(t) = \|pos(t+1) - pos(t)\|$
Velocity	$\dot{pos}(t) = (\dot{pos}_x(t), \dot{pos}_y(t), \dot{pos}_z(t))^T$
Acceleration	$\ddot{pos}(t) = (\ddot{pos}_x(t), \ddot{pos}_y(t), \ddot{pos}_z(t))^T$
Acceleration Norm	$\|\ddot{pos}(t)\|$
Jerk	$\dddot{pos}(t) = (\dddot{pos}_x(t), \dddot{pos}_y(t), \dddot{pos}_z(t))^T$
Slope Angle	$\theta_{xy}(t) = arctan\frac{\dot{pos}_y(t)}{\dot{pos}_x(t)}$, $\theta_{zx}(t) = arctan\frac{\dot{pos}_x(t)}{\dot{pos}_z(t)}$
Curvature	$\kappa(t) = \frac{\sqrt{c_{zy}^2(t)+c_{xz}^2(t)+c_{yx}^2(t)}}{(\dot{pos}_x^2(t)+\dot{pos}_y^2(t)+\dot{pos}_z^2(t))^{3/2}}$, $where\ c_{zy}(t) = \ddot{pos}_z(t) \times \dot{pos}_y(t) - \ddot{pos}_y(t) \times \dot{pos}_z(t)$

of each movement is first segmented based on the short-time energy. The 3-D space that encloses a trajectory is scaled to a $1 \times 1 \times 1$ bounding box. In addition, to unify the orientations of different trajectories, we predefine a reference direction and rotate all the trajectories to align with it. Fig. 6 illustrates the reconstructed end-effector trajectory of drawing an "Δ" after calibration and normalization, which has a very high similarity with that obtained by the OptiTrack camera system. We next derive unique features to analyze the robot behavior.

C. Extraction of Robotic Motion Features

For each end-effector trajectory, we derive eight types of robotic motion features to describe its behavior, including 3-D coordinate, position difference, 3-D velocity, 3-D acceleration, acceleration norm, jerk, slope angle, and curvature. These features at the sampling time index $t, t = 1, \ldots, N$, can be denoted as $pos(t)$, $pos\_diff(t)$, $\dot{pos}(t)$, $\ddot{pos}(t)$, $\|\ddot{pos}(t)\|$, $\dddot{pos}(t)$, $\theta(t)$, and $\kappa(t)$. The detailed derivations of these features are presented in Table I. In particular, we obtain three feature sequences, respectively, for the 3-D coordinates, velocity, acceleration, and jerk by leveraging their three axes. Moreover, we derive two slope angle sequences and one feature sequence for position difference, acceleration norm, and curvature, respectively. The resulted feature sequences $f_k = \{f_k(1), f_k(2), \ldots, f_k(N)\}, k = 1, \ldots, 17$ describe the inherent user behavior embedded in the robot motions in seventeen dimensions, including the large scale hand movements, the habits of accelerating and making turns, and the hand vibrations. For example, when different users perform the same task, the 3-D coordinate sequences of their hand movements might show some similarities, but the detailed hand speeds, accelerations, and jerks during the task could be distinct, owing to their unique arm sizes, strength, habits, and the individual ways to interact with the robotic arm. Based on these robotic motion features, we first recognize which task the robotic arm is performing and then verify who is controlling the robotic arm under this task.

D. Dynamic Time Warping

To achieve the goal of task recognition and user authentication, we resort to machine learning algorithms and propose to use dynamic time warping (DTW). Compared to deep learning, machine learning methods do not require high training efforts and computational resources, and thus are more easily to deploy in practice. Furthermore, the time-warping capability of DTW expands or shrinks the feature sequences to find their minimum Euclidean distances during the template matching, which can tolerate the impacts of unstable network latency, the user's varying hand movement speeds, behavioral inconsistency, and other environmental noises. For example, given two end-effector trajectories, we denote their feature sequences as $f_{k,tra_1} = \{f_{k,tra_1}(1), f_{k,tra_1}(2), \ldots, f_{k,tra_1}(N_1)\}$ and $f_{k,tra_2} = \{f_{k,tra_2}(1), f_{k,tra_2}(2), \ldots, f_{k,tra_2}(N_2)\}$. Then, we build an $N_1 \times N_2$ distance matrix M where the element $M_{ij} = \|f_{k,tra_1}(i) - f_{k,tra_2}(j)\|$, $i = 1, 2, \ldots, N_1$, $j = 1, 2, \ldots, N_2$. A nondecreasing path is found between M_{11} and $M_{N_1N_2}$, such that the sum value of the elements along this path is minimum, which is defined as the DTW distance between the two trajectories, denoted as $DTW(f_{k,tra_1}, f_{k,tra_2})$.

E. Task Recognition

As the user could operate the robotic arm to perform different tasks, the robotic arm needs to recognize what task is being performed rather than only executing a sequence of commands. This is critical for detecting prohibited tasks or auditing the user's entire access duration based on his/her privilege level. Moreover, we need to exclude the impact of the task differences to focus better on the robot's minute behavioral differences. Thus, we first recognize which task the robotic arm is performing before utilizing the user's template of the identified task to verify the user.

We apply DTW to learn the task templates and classify the testing task. In particular, we develop a weighted DTW-based method to recognize tasks based on the robotic motion features, which consists of a training phase to create templates and a testing phase for task classification. The reason why we use robotic motion features for task recognition is that the platform we developed is a robotic arm that follows the user's motions, and inherits the user's detailed behaviors. Though the tasks are defined based on the user's hand movement trajectory, we find that the other motion characteristics, such as the speeds, acceleration/de-acceleration, and the hand vibrations when making turns can also be utilized to distinguish movement tasks. We thus derive the motion features of the robotic arm (e.g., position, velocity, acceleration, slope angle, etc.) to recognize the robotic arm's tasks in the interactive control environment.

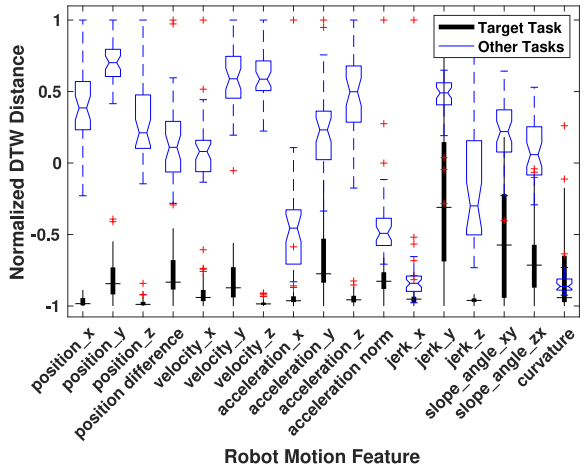


Fig. 7. Illustration of using robotic motion features to distinguish tasks.

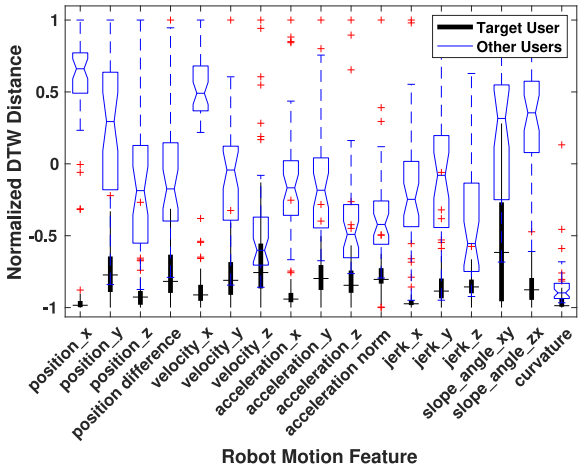


Fig. 8. Illustration of using robotic motion features to distinguish users.

Template Construction: In the training phase, we construct the task templates $f_{k,\text{task}}$ based on the task instances from a number of people. Specifically, with m instances collected for a task, we select the top n instances, which have the smallest DTW feature distances to all the m instances. The task template set is then formed by these n instances as $\{f_{k,\text{task\_train}_i} | i = 1, \ldots, n$ and $k = 1, \ldots, 17\}$.

Weight Assignment: As different features have different value ranges and different capabilities to distinguish various robotic arm tasks, we assign a weight $w_{k,\text{task}}$ between 0 and 1 to each task feature sequence by computing DTW distances. The weight of a feature sequence is determined by its ability to distinguish different tasks. Features that separate the target task from other tasks with greater DTW distances should be assigned with higher weights. Fig. 7 shows the DTW distance distributions of tasks in regard to a target task based on each feature. We find that most features clearly separate the target task from others, especially the 3-D position and velocity, which confirms the effectiveness of capturing the robotic motion features for recognizing the type of tasks. The reason is that the different tasks show different paths of robotic arm movement. The trajectories and velocities associated with moving and making turns capture such differences well, which should be assigned with high weights. In comparison, the curvature feature is not good at separating different tasks and should be assigned with a low weight. By selecting proper weights for these different robotic motion features, the developed DTW-based algorithm is able to effectively recognize different tasks.

Task Classification: The calculated weights are used in the testing phase to compute the weighted DTW distance sum between the testing end-effector trajectory to each of the task template sets, which is the sum of the weighted DTW distance between the testing end-effector trajectory to each template in the set, expressed as $\sum_{i=1}^{n} \sum_{k=1}^{17} \text{DTW}(f_{k,\text{test}}, f_{k,\text{task\_train}_i}) \times w_{k,\text{task}}$. The robotic task is recognized to be the one whose template exhibits the minimum weighted DTW distance sum to the testing task.

F. User Authentication

As the user must log into the robotic arm platform to operate the robotic arm, the robot behavior-based user authentication becomes to verify whether the user's identity is as claimed by the login credentials. After the robotic arm task is identified, we resort to the user's template under the task and compute the weighted DTW distance to verify the user.

Template Construction: The user's template is created based on the user's multiple instances of performing tasks. Similar to the task template construction, we choose n top instances that show the minimum DTW feature distances to all of the m collected instances. The user template set is then formed by these n instances as $\{f_{k,\text{user\_train}_i} | i = 1, \ldots, n$ and $k = 1, \ldots, 17\}$.

Weight Assignment: Although we use the same sets of features as in task classification, the weights $w_{k,\text{user}}$ used for user verification are derived based on their diverse abilities to distinguish users. Fig. 8 illustrates using the robotic motion features to distinguish users when they perform the same robotic arm task. We find it is more difficult to distinguish users than classifying tasks, as some feature distances between users are much smaller, such as the 3-D coordinates. The reason is that when the users perform the same task, similar end-effector trajectories are resulted in. However, the other features, such as accelerations, jerks, and slope angles perform well to separate the target user from other users. These features capture the user's unique small-scale hand movements and vibrations, the patterns to move or make turns and the interaction habits when operating the robotic arm. The result indicates the feasibility to distinguish users via robot behaviors even when they operate the robotic arm to do the same task, using the DTW-based method which leverages the derived robotic motion features. Thus, we compute the weights for the user verification features based on the DTW distances.

User Verification: During the user verification phase, we compute the weighted DTW distance sum between the testing end-effector trajectory and the user's template set as $\sum_{i=1}^{n} \sum_{k=1}^{17} \text{DTW}(f_{k,\text{test}}, f_{k,\text{user\_train}_i}) \times w_{k,\text{user}}$. If the resulting Euclidean distance is less than a threshold, the robotic arm is believed to be under the control of the user as claimed.

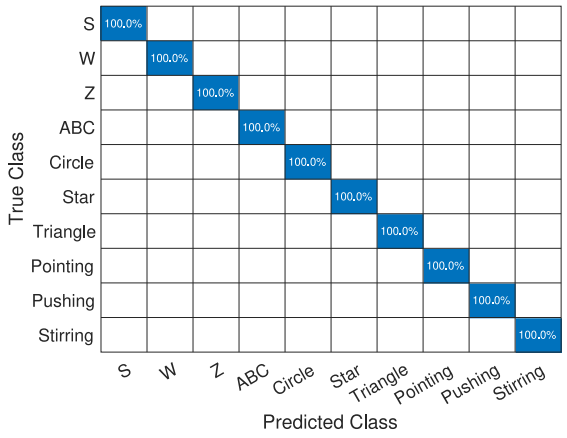


Fig. 9. Confusion matrix of recognizing robotic arm tasks.

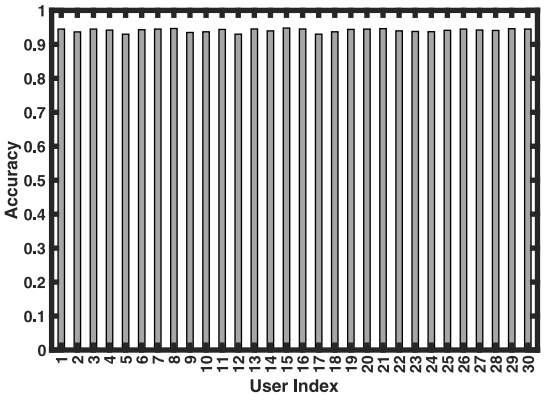


Fig. 10. Performance of user verification.

Otherwise, the verification fails, the robotic arm operation is aborted. The system requires a relogin.

VI. Performance Evaluation

A. Experimental Setup

We evaluate our robot behavior-based user authentication approach based on the motion-controlled robotic arm platform we build in Section III-B. We recruit 30 participants for the interactive control experiments, who are all first-time users of the platform. The IRB approval is obtained with IRB #4392. Before experiments, the users have 5 min to be familiar with the platform by operating the robotic arm. During the data collection, the participants are asked to control the robotic arm to perform various tasks involving three basic categories: 1) writing; 2) drawing; and 3) operations. In particular, the participants control the robotic arm to write the letters "S," "W," "Z," and "ABC," draw the curves, including "Circle," "Five-pointed Star" and "Triangle" in the air and perform the basic operations, such as "Pointing," "Pushing," and "Stirring." For each task, the participants repeat 30–40 times, and the angle readings of the robotic arm joints are logged. In addition, to prevent the robotic arm from causing any harm to people and objects, we set the excessive force on the robotic arm so that it stops if it exceeds an area boundary or touches a piece of furniture. The data collection lasts for six months, and each participant completes the experiments in around an hour. The resulted data set is split by half for training and testing, respectively.

B. Task Recognition

We first present the performance of our approach to recognizing robotic arm tasks. Fig. 9 shows the confusion matrix of the task classification. We find that our approach achieves 100% accuracy to differentiate all these tasks, regardless of whether they are letter-writing tasks or drawing or operation tasks. The result indicates that our robotic motion features capture the differences among tasks well and the weighted DTW-based classifier tolerates the network delays and the user's varying hand movement speeds when recognizing a task. The accurate task classification not only helps audit the user's control tasks but also enables us to further focus on the minute user behavioral differences embedded in robot motions.

C. User Verification

We now evaluate our approach to verify users. In particular, we alternatively select each participant to be the target legitimate user and use the corresponding user template for verification. We compute the verification accuracy of each user for all types of tasks. Fig. 10 presents the average accuracy for each of the 30 participants. We observe that our user verification approach achieves high accuracy for all the participants. In particular, all the participants achieve over 93% accuracy, while the median accuracy is 94.1%. Some participants reach around 95% accuracies. The results confirm that our approach verifies users via robot behaviors accurately.

D. Impact of Training Data Size

Next, we study the impact of training data size on user verification performance. In particular, the user verification results are obtained based on the user templates derived with different numbers of training instances. Our approach turns out to be able to verify users accurately within a limited number of training instances. In particular, when the training data size is 6, the verification accuracy is over 80%. The performance is improved to 93.5% when 13 instances are used for training. After that, increasing the training data size does not show significant performance improvement. For example, the accuracy is 94.1% when 15 instances are used for training.

E. Algorithms Comparison

This work presents the first method to verify a user via the robot behavior. To further demonstrate the efficiency of our DTW-based method, we develop three other machine learning-based methods for comparison, including hidden Markov models (HMMs), support vector machine (SVM), and K-nearest neighbors (KNNs), and the same features are used. The results of our four methods are shown in Fig. 11. We observe that for both the task recognition and the user verification, our DTW-based method performs better than the three methods. Specifically, HMM, SVM, and KNN achieve 91.4%, 94.1%, and 93.7% accuracy for task recognition, and 84.7%,

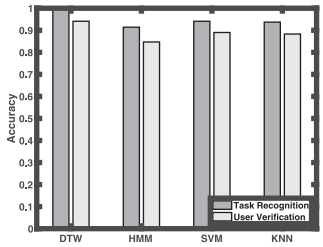


Fig. 11. Comparison with other algorithms.

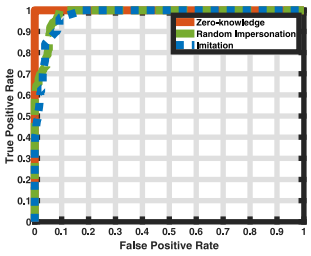


Fig. 12. ROC curves of our authentication approach under different attacks.

89.0%, and 88.3% accuracy for user verification, respectively. The reason is that our method based on DTW is more robust to the unstable network latencies and the user's behavioral inconsistency.

F. Under Zero-Knowledge Attack

Setup: In this attacking scenario, we assume the adversary has no idea of our authentication approach. Thus, the adversary only focuses on operating the robotic arm to complete malicious tasks without considering the robot behaviors during the control, which is a common control hijacking scenario. We simulate such attacks by using programs to generate command sequences, which control the robotic arm to perform all three categories of tasks.

Results: The performance of defending the zero-knowledge attack is shown in Fig. 12. We find that our system achieves 100% true positive rate (TPR) and 0% false positive rate (FPR) to verify the legitimate user. The reason is that the zero-knowledge attacks fail to present any human control behaviors in the control commands, which can be easily prevented by our behavior-based authentication approach. The results confirm the strong capability of our approach to handling the existing robot hijacking attacks.

G. Under Random Impersonation Attack

Setup: In this attacking scenario, the adversary is assumed to know the deployment of our approach but has no prior knowledge of the target user's control behavior. Thus, the adversary attempts to cheat our approach with his/her own behavior. To simulate this attack, we alternatively select each participant to be the target user and the others to be the random impersonation attackers.

Results: The performance of our approach under the random impersonation attack is shown in Fig. 12. Our system achieves a 94.6% TPR and a 6.9% FPR to verify the legitimate user under the random impersonations. The equal error rate (EER) is 6.9%. The results confirm the strong capability of our approach to distinguish users via the robot behaviors.

H. Under Imitation Attack

Setup: In this attacking scenario, an adversary is assumed to have prior knowledge of the target user's control behaviors. Thus, the adversary imitates the user's behavior to cheat our approach when controlling the robotic arm to perform malicious tasks. To simulate this attack, the authors and three skilled participants assume the roles of adversaries. They watch the video of the target users' control processes to imitate the behavior, including the trajectories, speeds, and accelerations when moving or making turns and pauses. After practicing, the adversaries control the robotic arm to spoof the target users.

Results: The performance of defending the imitation attacks is shown in Fig. 12. Our system achieves a 96.5% TPR and a 9.2% FPR to prevent imitation attacks, and the EER is 8.5%. The ROC curve under the imitation attacks is only slightly lower than the random impersonations. The results indicate that it is hard to imitate the user's interactive control behaviors, which integrates the user's regular behaviors with the interaction habits, and our approach greatly improves the security of the motion-controlled robotic arm.

I. Attack Detection Latency

The attack detection latency is a very important evaluation metric for our system, which determines how much time we need to detect and stop an attack. The attack detection latency of our system mainly depends on the observation window, which is the duration. According to our experiments, the median latency is 3.7 s, while the other types of latency time (network latency, computation latency, etc.) are ms level. In our future work, we will further reduce the latency by exploring the potential of using partial tasks for authentication.

VII. Conclusion

In this work, we demonstrate that when a user interactively controls a robot to perform tasks, the robotic arm could inherit a portion of the user's motion behaviors, which can be leveraged to distinguish users. We propose a robot behavior-based user authentication approach for the motion-controlled robotic arm. The proposed approach ensures the robotic arm is consistently under the user's control but not hijacked or overtaken by an adversary. In particular, we reconstruct the robotic arm's end-effector trajectories from the angle readings of its all joints to describe the robot motion, which reflects how it follows the user's hand. We then derive unique robotic motion features to capture the robot's behaviors where the user's behavioral information is embedded. Based on that, we develop a DTW-based algorithm to first recognize what type of task the robot is performing and then verify who is controlling the robotic arm. We build a real motion-controlled robotic arm platform to evaluate our approach, which consists of the motion capture devices, a 7-DOF robotic arm, and a local area network. Extensive experiments show that our authentication approach

verifies the user with 94% accuracy and prevents hijacking and impersonation attacks with over 95% accuracy.

REFERENCES

[1] J. Rekimoto and K. Nagao, "The world through the computer: Computer augmented interaction with real world environments," in *Proc. 8th Annu. ACM Symp. User Interface Softw. Technol.*, 1995, pp. 29–36.

[2] D. Mourtzis, E. Vlachou, G. Dimitrakopoulos, and V. Zogopoulos, "Cyber-physical systems and education 4.0—The teaching factory 4.0 concept," *Procedia Manuf.*, vol. 23, pp. 129–134, Apr. 2018.

[3] A. Grau, M. Indri, L. L. Bello, and T. Sauter, "Industrial robotics in factory automation: From the early stage to the Internet of Things," in *Proc. 43rd Annu. Conf. IEEE Ind. Electron. Soc.*, 2017, pp. 6159–6164.

[4] V. M. Vilches *et al.*, "Introducing the robot security framework (RSF), a standardized methodology to perform security assessments in robotics," 2018, *arXiv:1806.04042*.

[5] R. Hirata, "Robot system," U.S. Patent 14 028 543, May 15, 2014.

[6] S. T. Kim and Y. M. Ro, "Attended relation feature representation of facial dynamics for facial authentication," *IEEE Trans. Inf. Forensics Security*, vol. 14, no. 7, pp. 1768–1778, Jul. 2019.

[7] P. Guo, H. Kim, N. Virani, J. Xu, M. Zhu, and P. Liu, "RoboADS: Anomaly detection against sensor and actuator misbehaviors in mobile robots," in *Proc. 48th Annu. IEEE/IFIP Int. Conf. Dependable Syst. Netw. (DSN)*, 2018, pp. 574–585.

[8] J. Inoue, Y. Yamagata, Y. Chen, C. M. Poskitt, and J. Sun, "Anomaly detection for a water treatment system using unsupervised machine learning," in *Proc. IEEE Int. Conf. Data Min. Workshops (ICDMW)*, 2017, pp. 1058–1065.

[9] A. Keliris, H. Salehghaffari, B. Cairl, P. Krishnamurthy, M. Maniatakos, and F. Khorrami, "Machine learning-based defense against process-aware attacks on industrial control systems," in *Proc. IEEE Int. Test Conf. (ITC)*, 2016, pp. 1–10.

[10] Á. M. Guerrero-Higueras, N. DeCastro-García, and V. Matellán, "Detection of cyber-attacks to indoor real time localization systems for autonomous robots," *Robot. Auton. Syst.*, vol. 99, pp. 75–83, Jan. 2018.

[11] S. Levine, C. Finn, T. Darrell, and P. Abbeel, "End-to-end training of deep visuomotor policies," *J. Mach. Learn. Res.*, vol. 17, no. 1, pp. 1334–1373, 2016.

[12] S. Levine, P. Pastor, A. Krizhevsky, J. Ibarz, and D. Quillen, "Learning hand–eye coordination for robotic grasping with deep learning and large-scale data collection," *Int. J. Robot. Res.*, vol. 37, nos. 4–5, pp. 421–436, 2018.

[13] C. Finn and S. Levine, "Deep visual foresight for planning robot motion," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, 2017, pp. 2786–2793.

[14] I. A. Tsokalo, D. Kuss, I. Kharabet, F. H. Fitzek, and M. Reisslein, "Remote robot control with human-in-the-loop over long distances using digital twins," in *Proc. IEEE Global Commun. Conf. (GLOBECOM)*, 2019, pp. 1–6.

[15] H. Laaki, Y. Miche, and K. Tammi, "Prototyping a digital twin for real time remote control over mobile networks: Application of remote surgery," *IEEE Access*, vol. 7, pp. 20325–20336, 2019.

[16] H. Lv, G. Yang, H. Zhou, X. Huang, H. Yang, and Z. Pang, "Teleoperation of collaborative robot for remote dementia care in home environments," *IEEE J. Transl. Eng. Health Med.*, vol. 8, pp. 1–10, 2020.

[17] J. Tian, C. Qu, W. Xu, and S. Wang, "KinWrite: Handwriting-based authentication using kinect," in *Proc. NDSS*, vol. 93, 2013, pp. 1–18.

[18] W. Xu *et al.*, "Which one to go: Security and usability evaluation of mid-air gestures," 2018, *arXiv:1811.10168*.

[19] D. Lu and D. Huang, "FmCode: A 3D in-the-air finger motion based user login framework for gesture interface," 2018, *arXiv:1808.00130*.

[20] X. Zhang, X. Chen, Y. Li, V. Lantz, K. Wang, and J. Yang, "A framework for hand gesture recognition based on accelerometer and EMG sensors," *IEEE Trans. Syst., Man, Cybern. A, Syst., Humans*, vol. 41, no. 6, pp. 1064–1076, Nov. 2011.

[21] R. Xu, S. Zhou, and W. J. Li, "MEMS accelerometer based nonspecific-user hand gesture recognition," *IEEE Sensors J.*, vol. 12, no. 5, pp. 1166–1173, May 2012.

[22] B. Kellogg, V. Talla, and S. Gollakota, "Bringing gesture recognition to all devices," in *Proc. 11th USENIX Symp. Netw. Syst. Design Implement. (NSDI)*, 2014, pp. 303–316.

[23] P. Melgarejo, X. Zhang, P. Ramanathan, and D. Chu, "Leveraging directional antenna capabilities for fine-grained gesture recognition," in *Proc. ACM Int. Joint Conf. Pervasive Ubiquitous Comput.*, 2014, pp. 541–551.

[24] Q. Pu, S. Gupta, S. Gollakota, and S. Patel, "Whole-home gesture recognition using wireless signals," in *Proc. 19th Annu. Int. Conf. Mobile Comput. Netw.*, 2013, pp. 27–38.

[25] H. Abdelnasser, M. Youssef, and K. A. Harras, "WiGest: A ubiquitous wifi-based gesture recognition system," in *Proc. IEEE Conf. Comput. Commun. (INFOCOM)*, 2015, pp. 1472–1480.

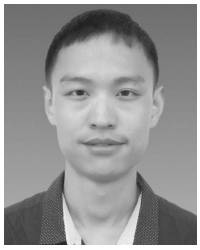
[26] "Optitrack Prime 13." I.D.O. NaturalPoint. May 2020. [Online]. Available: https://www.optitrack.com/cameras/primex-13.html

[27] L. Sciavicco and B. Siciliano, "Modelling and control of robot manipulators," *Meas. Sci. Technol.*, vol. 11, no. 12, p. 1828, 2000.

[28] Franka Emika GmbH. *Franka Control Interface Documentation*. (Dec. 2020). [Online]. Available: https://frankaemika.github.io/docs/overview.html

[29] Franka Emika GmbH. *Requirements on Communication Interface for Panda Research Robot*. (Dec. 2020). [Online]. Available: https://frankaemika.github.io/docs/requirements.html

[30] J. Denavit and R. S. Hartenberg, "A kinematic notation for lower-pair mechanisms based on matrices," *J. Appl. Mech.*, vol. 22, no. 2, pp. 215–221, Jun. 1955.



Long Huang received the B.Eng. degree in electrical and electronic engineering from the University of Electronic Science and Technology of China, Chengdu, China, in 2015, and the M.Sc. degree in electrical and electronic engineering from the Stevens Institute of Technology, Hoboken, NJ, USA, in 2019. He is currently pursuing the Ph.D. degree with Louisiana State University, Baton Rouge, LA, USA.

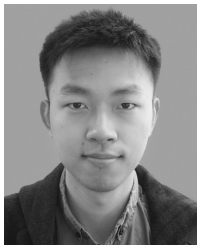
His research interests include signal processing and Internet of Things.

Mr. Huang received the Best Paper Award from the 14th International Workshop on Wireless Sensor, Robot and UAV Networks, IEEE WISARN 2021.

Zhen Meng received the B.Eng. degree from the Department of Engineering, University of Glasgow, Glasgow, U.K., in 2019, where he is currently pursuing the Ph.D. degree.

His research interests include the co-design of wireless communications and control applications.

Mr. Meng received the Best Paper Award from the 14th International Workshop on Wireless Sensor, Robot and UAV Networks, IEEE WISARN 2021.



Zeyu Deng received the B.Eng. degree in electrical and electronic engineering from the University of Birmingham, Birmingham, U.K., and Huazhong University of Science and Technology, Wuhan, China, in 2018, and the M.Sc. degree in Internet engineering from University College London, London, U.K., in 2019. He is currently pursuing the Ph.D. degree with Louisiana State University, Baton Rouge, LA, USA.

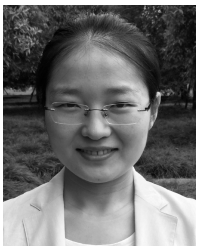
His research interests include smart devices and the Internet of Things.

Mr. Deng received the Best Paper Award from the 14th International Workshop on Wireless Sensor, Robot and UAV Networks, IEEE WISARN 2021.

Chen Wang (Member, IEEE) received the Ph.D. degree from Rutgers University, New Brunswick, NJ, USA, in 2019.

He is currently an Assistant Professor with Louisiana State University, Baton Rouge, LA, USA. His research interests include cyber security and privacy, smart healthcare, mobile sensing, and computing.

Dr. Wang received the five best paper awards from the IEEE Conference on Communications and Network Security (CNS 2018), the IEEE CNS 2014, the ACM Conference on Information, Computer and Communications Security (ASIACCS 2016), the EAI International Conference on IoT Technologies for HealthCare (EAI HealthyIoT 2019), and the IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS 2021). His four research studies have been reported by over 150 media outlets, including IEEE Spectrum, NSF Science 360, CBS TV, BBC News, NBC, IEEE Engineering 360, Fortune, ABC News, and MIT Technology Review.



Liying (Emma) Li received the Ph.D. degree in electrical engineering from the University of Electronic Science and Technology of China (UESTC), Chengdu, China, in 2011.

From 2011 to 2019, she worked as an Assistant/Associate Professor with the School of Automation Engineering, UESTC. From 2020 to 2021, she was a Senior Lecturer with Northumbria University at Newcastle, Newcastle upon Tyne, U.K. In 2021, she started her Lecturer position with the School of Computing Science, University of Glasgow, Glasgow, U.K. Her current research interest is interactive and trustworthy AI in the context of robotics and communication systems.

Guodong (Philip) Zhao (Senior Member, IEEE) received the Ph.D. degree in electrical engineering from Beihang University, Beijing, China, in 2011.

From 2011 to 2018, he was an Associate Professor with the University of Electronic Science and Technology of China, Chengdu, China. In 2018, he joined the University of Glasgow, Glasgow, U.K., as a Lecturer (Assistant Professor) and was promoted as a Senior Lecturer in 2021. He published more than 100 peer-reviewed research papers (including more than 20 IEEE journals) and had more than 2200 citations at Google Scholar. His current research interest is the joint design of wireless communication and robotics systems for many emerging use cases, such as healthcare, Industry 4.0, and autonomous driving.

Dr. Zhao won the University Teaching Excellence Award at the University of Glasgow in 2021, the Best Workshop Paper Award (WISARN Workshop) at IEEE INFOCOM 2021, the Best Poster Award at IEEE WCNC 2018, and the Best Paper Award at IEEE GLOBECOM 2012.