Long-Term Trajectory Prediction of the Human Hand and Duration Estimation of the Human Action

Yujiao Cheng and Masayoshi Tomizuka

***Abstract*—In the framework of human-robot collaborative assembly, it is important to predict the long-term human hand trajectory for collision avoidance and to estimate the durations of the human actions for collaborative task planning. Many existing works predict long-term human trajectory by a preset time horizon, while in this letter, our prediction horizon depends on how far in the future we could predict the human's actions. To be more specific, we predict the human trajectory and estimate the durations for the human's current action and future actions. To address this problem, we propose a recognition-then-prediction framework. First, we present a hierarchical recognition algorithm to infer the human intentions for the current action and the future actions. Next, we propose to use the sigma-lognormal function to model and predict the human movement, and from this model we estimate the action durations. To accommodate different human behaviors, we also propose an online algorithm to adapt the movement model by using the observed trajectory, the human intentions and the scene layout. The effectiveness of the proposed framework is supported by experimental validations on the human trajectory data for conducting a computer assembly task.**

***Index Terms*—Industrial robots, human-centered robotics, assembly, trajectory prediction.**

I. Introduction

ROBOTIC systems are increasingly integrated into human workspace for collaborative tasks, which brings one important challenge as prediction. It is difficult to make accurate predictions of the human movement due to the nonlinearity and stochasticity of the human behavior and the variety of its external and internal stimuli [1]. However, for the human workers in the assembly line, their movement behavior is mainly driven by the task goals and the layout of the parts, and it is possible to make good predictions using such contextual information. For example, as shown in Fig. 1(a), the human worker is assembling a desktop computer. By using the worker's hand trajectory, task information and the layout of the parts, we can infer that the human is going to reach the CPU fan, and we can also anticipate that the human will obtain the CPU fan and install it in the computer case. Therefore, the human's hand trajectory can be predicted in the long term, not only for the current action (reaching for the CPU fan), but also for the predicted actions (obtaining the CPU fan and installing the CPU fan). Besides, from the trajectory prediction, we can also estimate the action duration, which is the time required to complete an action. Therefore, in this letter, we study the following problem: *predict the human hand trajectory and estimate the durations for the human's current action and future actions with the knowledge of historic trajectory, the task information and the scene context.*

Active research explores techniques for human intent recognition and trajectory prediction. For intent recognition, many works focus on deep learning frameworks with RGBD images as inputs [2]–[4]. Typically, the features selected mainly focus on human, for instance, the body pose, hand positions and histogram of oriented gradients (HOG). No information about the objects of interaction are included. However, the objects can provide rich information for inferring what the human is doing via the intrinsic hierarchy among actions, motion types and the objects. Hence, in this letter, we explore such hierarchy to design more robust intent recognition algorithm. For trajectory prediction, two categories of approaches are proposed: 1) learning-based approaches such as recurrent neural networks (RNNs) [5], inverse reinforcement learning [6] and semi-adaptable neural network [7], and 2) model-based approaches such as social force model [8] and minimum jerk model [9]. We are interested in developing model-based methods for their intuitive physical meanings and data efficiency property, and we propose to use sigma-lognormal equations to model the human kinematics. To the best of our knowledge, it is the first time that this model is applied to the human-robot collaborative field. Besides, most works predict the trajectory by a preset horizon, while our prediction horizon is not fixed and it depends on how far in the future we could predict the human's actions, which is the longest prediction horizon given the results of the human intent recognition.

We propose to use a recognition-then-prediction framework. For the recognition, the goal is to recognize the human intentions for the current action and the future actions. We apply a hierarchical approach as in [10], and we improve the method by utilizing the task information. We propose to use a sigma-lognormal function to model and predict the human's velocity profile. The parameters of this model are trained offline, and we propose an adaptive algorithm to change the parameters online to suit different human behaviors. The prediction of the trajectory and the action duration estimation are both based on this model. Experiments validate the effectiveness of the proposed framework and demonstrate the superiority of our prediction method over other four baseline methods in terms

Manuscript received July 11, 2021; accepted October 27, 2021. Date of publication November 2, 2021; date of current version November 10, 2021. This letter was recommended for publication by Associate Editor Gianluca Garofalo. *(Corresponding author: Yujiao Cheng.)*

The authors are with the School of Engineering, Mechanical Engineering Department, University of California, Berkeley, California 94709 USA (e-mail: yujiaocheng@berkeley.edu; tomizuka@me.berkeley.edu).

Digital Object Identifier 10.1109/LRA.2021.3124524

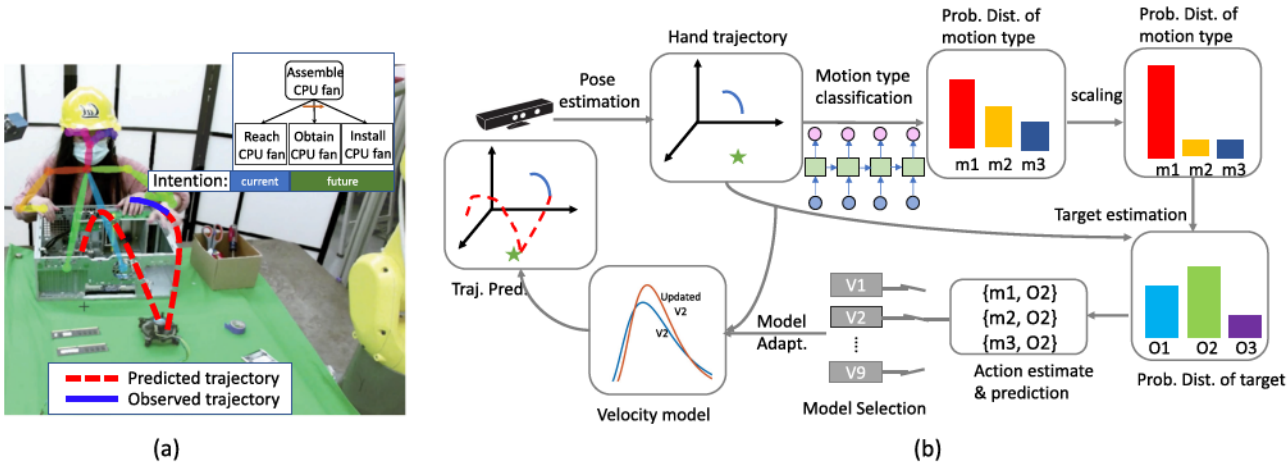

Fig. 1. (a) Long-term human hand trajectory prediction. With the observed trajectory, task information and the layout of the parts, the trajectory of the human hand is predicted over the horizon of the current action and the predicted actions. (b) The framework for the long-term human hand trajectory prediction.

of the trajectory prediction accuracy and the duration estimation accuracy. Besides, experimental results also show that our adaptation algorithm can quickly adapt the model to the human movement with the same motion type but with different starting and ending points.

The key contributions of this letter are:

- We propose a framework to make long-term human hand trajectory prediction and action duration estimation.
- We improve the intent recognition method in [10] by additionally utilizing task information.
- We propose to use the sigma-lognormal function to model and predict the human movement, and we propose an online algorithm for the model adaptation.
- We conduct experiments to validate the effectiveness of our proposed recognition-then-prediction framework and the effectiveness of the trajectory prediction method.

The rest of the letter is organized as follows. Section II demonstrates the proposed framework. Section III proposes the algorithm for human intent recognition, and Section IV proposes the algorithm for trajectory prediction and duration estimation. In Section V, experiments are included. Finally, Section VI discusses some results and Section VII concludes the letter.

II. Framework

Our goal is to predict the human hand trajectory over the horizon of the human's current intended action and the human's future actions. The terminologies we used in this letter are defined as follows:

- *Trajectory*: a time series of the joint positions of an agent (a human) in Cartesian space (3D). It represents the continuous movements of an agent.
- *Motion Type*: A discrete variable/label to represent different types/patterns of trajectories. For instance, typical motions in factory scenarios include "Reaching," "obtaining" and "Installing".
- *Action*: A paired discrete variable/label including a motion type and the target object to act on, i.e., action={motion type, target}. For example, we can define "action 1={Reaching, a CPU fan}".
- *Plan*: A plan is comprised of a sequence of ordered actions. It represents the preferences to finish a *task* (defined below), and we also use **intent** to indicate a human's plan in this letter.
- *Task*: A task represents the work to be conducted by agents. It specifies the initial states, the goal states and the participants.

The pipeline of the proposed approach is schematically illustrated in Fig. 1(b). Three aspects are addressed in this framework: (i) obtaining the human pose from the sensor input, (ii) recognizing the human's current action and predicting the future actions, a.k.a. human intent recognition, and (iii) predicting the trajectory over those actions and estimating the action durations. In the following, we first introduce the method of human intent recognition, and then we propose an approach to estimating the durations and predicting the trajectories over the current and the future actions.

III. Human Intent Recognition

The purpose of human intent recognition is to discover what the human's intended action at the moment is and what the human's future actions can be. To do this, first, we need to recognize the human's current action, which is a combination of the motion type recognition and the target estimation according to the definition of an action. Second, based on the task information and the sequence of actions that the human conducts, we infer human's plan and predict the future actions.

A. Motion Type Recognition

Motion type classification aims to categorize different motions given the task information and the trajectories of the human hand. The recognizer applies a two-step methodology: (i) classifying the trajectories to different motion types using a long-short-term-memory (LSTM) neural network, which has been extensively proved to be an effective approach to model the

dynamics and dependencies in sequential data, and (ii) scaling the probability output of the motion classification by a coefficient vector which is obtained based on the task information. The output of the first step is $P(m_{t_k} = i \mid \mathbf{x}_{t_0:t_k}), i = 1, 2, \ldots, n_m$, the probability of the motion type m being i at time t_k given a series of the hand positions $\mathbf{x}$ from t_0 to t_k, where n_m is the number of motion types. More details can be found in [10]. The second step is a scaling step that weights the output of classifier in the first step, and we will elaborate on it in the following.

The introduction of the scaling step consists in weighting the LSTM probabilities with respect to the soundness of the action transition in a task. The intuition is that when an action is in progress, the motion type stays the same, while when the action is finished and the human continues to do the next action, the motion type will transition to the motion type of the next action. In this sense, we define the i-th component of the coefficient vector $\mathbf{c}_{t_k} \in R^{n_m}$ at time t_k as:

$$c_{t_k}(i) = P\left(m_{t_k} = i \mid a_{t_0:t_{k-1}}\right) = \sum_{q=1}^{n_q} P\left(m_{t_k} = i | a_{t_0:t_{k-1}}, q\right), \quad i = 1, 2, \ldots, n_m$$

$$P\left(m_{t_k} = i | a_{t_0:t_{k-1}}, q\right) = \begin{cases} \frac{1-\sigma}{2}, & m_{t_k} = m_{t_{k-1}} = i \\ \frac{1-\sigma}{2}, & m_{t_k} = m'_{q1} = i \\ \sigma, & otherwise \end{cases}$$

where $q = 1, 2, \ldots, n_q$ indicates different plans that the human might follow for the task, σ is a very small number, $a_{t_0:t_{k-1}}$ is the action history from time t_0 to t_{k-1}, and m'_{q1} is the next motion type predicted from the last time step t_{k-1} for the plan being q, the computation of which is described in Section III-C. For any plan, the human either continues to conduct the previous motion type or switches to the next motion type in the plan after the previous action is finished. The chances that the human gives up the current plan and changes to another are slim. Thus, we assign σ with a very small number to the probability that the human changes the plan, and we make equal the possibilities that the human continues doing the previous motion type or the human conducts the next motion type that is predicted by the previous time step for the plan being q.

Finally, the motion type is identified by the following equation:

$$m_{t_k} = \underset{i \in \{1,2,\ldots,n_m\}}{\arg\max} \; c_{t_k}(i) \cdot P\left(m_{t_k} = i \mid \mathbf{x}_{t_0:t_k}\right).$$

B. Target Estimation

The goal of the target estimation is to estimate the object of interaction or the target point o_{t_k} at time t_k given the classified motion labels $m_{t_0:t_k}$ and a series of observed human hand positions $\mathbf{x}_{t_0:t_k}$.

We propose to estimate the target by the following equation:

$$o_{t_k} = \underset{i \in \{1,2,\ldots,n_o\}}{\arg\max} \; \bar{c}_{t_k}(i) \cdot P\left(\mathbf{x}_{t_k} | \mathbf{x}_{t_0:t_{k-1}}, o_{t_k} = i, m_{t_k}\right),$$

where $P(\mathbf{x}_{t_k} | \mathbf{x}_{t_0:t_{k-1}}, o_{t_k} = i, m_{t_k})$ is the likelihood of the current human hand position given the past trajectory $\mathbf{x}_{t_0:t_{k-1}}$ and the current action $\{m_{t_k}, o_{t_k} = i\}$, and $\bar{c}_{t_k}(i)$ is the coefficient to scale the likelihood. The motivation of introducing the likelihood is that we want to take advantage of the hand trajectory, since the human's action determines the hand trajectory, and vice versa the hand trajectory carries information about the human's action. The motivation of using the scaling factor is that we want to take advantage of the action history similar to that in Section III-A, which will be detailed in the following.

First, to compute the likelihood, we formulate the probability as a Boltzmann policy, which is widely used in many applications [10], [11]:

$$P\left(\mathbf{x}_{t_k} | \mathbf{x}_{t_0:t_{k-1}}, o_{t_k} = i, m_{t_k}\right) \propto \exp\left(V_g\left(\mathbf{x}_{t_k}; \mathbf{x}_{t_0:t_{k-1}}, o_{t_k} = i, m_{t_k}\right)\right), \quad i = 1, 2, \ldots, n_o$$

where n_o is the number of targets and V_g is the value function as in [10], which assumes that humans are optimizing different value functions for taking different actions.

Second, we define the i-th component of the coefficient vector $\bar{\mathbf{c}}_{t_k} \in R^{n_o}$ at time t_k as:

$$\bar{c}_{t_k}(i) = P\left(o_{t_k} = i \mid a_{t_0:t_{k-1}}\right) = \sum_{q=1}^{n_q} P(o_{t_k} = i | a_{t_0:t_{k-1}}, q), \quad i = 1, 2, \ldots, n_o$$

$$P\left(o_{t_k} = i | a_{t_0:t_{k-1}}, q\right) = \begin{cases} \frac{1-\sigma}{2}, & o_{t_k} = o_{t_{k-1}} = i \\ \frac{1-\sigma}{2}, & o_{t_k} = o'_{q1} = i \\ \sigma, & otherwise \end{cases}$$

where o'_{q1} is the predicted next target from the last time step t_{k-1} for the plan being q, the computation of which is also deferred in Section III-C.

C. Action Recognition and Prediction

In [10], the dynamic time warping algorithm is applied to align the observed action sequence $a_{t_0:t_k}$ with the template action sequences for each plan. The human's plan $q*$ is recognized with the minimum alignment distance, and a_{t_k} is updated by the posterior action correction through the alignment results. The future action vector for any plan q is $\mathbf{a}'_q \in R^{np}$, where np is the number of the sequential actions after $a_{t_0:t_k}$ for plan q, and $\mathbf{m}'_q$ $\mathbf{o}'_q$ can be retrieved from $\mathbf{a}'_q$. m'_{q1} and o'_{q1} in Section III-A and III-B are the values of the first components for $\mathbf{m}'_q$ and $\mathbf{o}'_q$, respectively. The future action vector $\mathbf{a}'_{q^*}$ for plan q^* is the best prediction for the future human actions.

IV. Human Trajectory Prediction

This section explains how we predict the trajectory and estimate the duration. We model human movements for each action in the task as the sigma-lognormal equations, and offline learn the parameters from the training data. Online, the model parameters are adapted every time when the measured data comes. We propose an adaptation algorithm, which utilizes the knowledge of the human intention and the target location. Finally, we predict the trajectory by using the updated model and we estimate the duration by the stable zero crossing point of the velocity profile with some threshold.

A. Sigma-Lognormal Model

Many human movement models have been proposed based on various techniques such as neural networks [12], minimization principals [9], and kinematic theories [13]. Among them, the kinematic theory and its sigma-lognormal model explain most of the basic phenomena on human motor control [13], and it is proved that it is ideal for describing the impulse response of a neuromuscular network which is composed of some subsystems controlling the velocity of a movement [13]. It achieves many success in applications like signature verification [14] and handwriting learning [15]. We now apply it to the human-robot collaboration field, which, to the best of our knowledge, is the first application. The model takes the following form:

$$\hat{\mathbf{v}}(t) = \sum_{i=1}^{N} \hat{\mathbf{v}}_i(t) = \sum_{i=1}^{N} \mathbf{D}_i(t)\Lambda_i\left(t; t_{0i}, \mu_i, \sigma_i^2\right); N \geq 3,$$

$$\Lambda_i\left(t; t_{0i}, \mu_i, \sigma_i^2\right) = \frac{1}{\sigma_i\sqrt{2\pi}\,(t - t_{0i})} exp\left(\frac{-\left(\ln(t - t_{0i}) - \mu_i\right)^2}{2\sigma^2}\right) \quad (1)$$

where $\hat{\mathbf{v}}(t)$ is the velocity of the human hand at time t, and $\Lambda(t, t_0, \mu, \sigma^2)$ is a lognormal distribution with the time shift t_0, the expected value of the t's natural logarithm μ and the standard deviation of the t's natural logarithm σ. The velocity $\hat{\mathbf{v}}(t)$ is composed of N lognormal distributions Λ_i, each scaled by variable $\mathbf{D}_i, i = 1, \ldots, N$. Since human's hand is moving in three dimensions, the number of lognormal distribution N is equal to or larger than 3. For example, simple reaching or pointing gestures require three weighted lognormals to describe the speed profile. More complex trajectories require more lognormals.

An empirical way of deciding N is to project the velocity profile to the spaces defined by the x-axis, the y-axis and the z-axis, and sum up the observed number of bell shapes in each file. Indeed, Equation (1) can be decomposed into three independent scalar equations, and they can be learned and adapted independently.

B. Offline Model Fitting

To use the sigma-lognormal model for analyzing the human movement, we need to choose the values for parameters $\alpha = \{\mathbf{D}_i, t_{0i}, \mu_i, \sigma_i^2\}, i = 1, \ldots, N$ such that the model can fit the training data well. Two categories of parameter extractors have been proposed in the literature: the Xzero-based extractor [16] and the prototype-based extractor [17]. The Xzero-based extractor has closed-form solutions by analyzing several characteristic points located in the original velocity profile, but it lacks the knowledge of the nature of the movement and it is easily influenced by the noise in the data. The prorotype-based extractor takes advantage of a priori information and works well for stereotypical movements. It first synthesizes the prototype, then it adjusts the parameters by scaling and offsetting. This is a good philosophy for online parameter extraction, but not suitable for offline model fitting.

We propose to use the Levenberg-Marquardt algorithm [18], also known as the damped least-squares method, to solve the following problems: given a set of m data points $(t_j, \mathbf{v}_j)$ from the collected trajectories, find α such that the sum of the squares of the deviations $S(\alpha)$ is minimized:

$$\alpha^* = \arg\min_{\alpha} S(\alpha) = \arg\min_{\alpha} \sum_{j=1}^{m} ||\mathbf{v}_j - \hat{\mathbf{v}}\left(t_j, \alpha\right)||_2^2.$$

C. Online Parameter Extraction

Since human behavior is time-varying and different people have different motion styles, it is necessary to adapt the offline learned model to accommodate such variations. [19] concludes that the time scaling and shifting account for a large variability of human movement, and only the modification of μ_i and the t_{0i} is required through scaling factor C_{si} and t_{si} in Equation (2) and Equation (3). In addition to C_{si} and t_{si}, we add one more scaling factor D_{si} for D_i to account for more variations as in Equation (4).

$$t_{0is} = C_{si}t_{0i} + t_{si} \quad (2)$$

$$\mu_{is} = \mu_i + ln(C_s) \quad (3)$$

$$D_{is} = D_{si}D_i \quad (4)$$

Therefore, the sigma-lognormal model becomes $\hat{\mathbf{v}}(t; \alpha^*, \beta)$, where $\beta = \{C_{si}, t_{si}, D_{si}\}, i = 1, \ldots, N$. The problem of adapting the sigma-lognormal model is to learn the values of β. The initialization of β requires that the sigma-lognormal model starts from the offline learned model in Section IV-B. Thus, the scaling factors C_{si} and D_{si} are set to 1, and the time shifting factor t_{si} is set to 0, for $i = 1, \ldots, N$. As indicated in Section IV-A, the sigma-lognormal equation in Equation (1) comprises three scalar equations which can be adapted independently. Hence, we decouple the parameter β into three parts $\{\beta_x, \beta_y, \beta_z\}$ and adapt them separately.

To adapt β when a new data point is available, two clues are essential: 1) how the new data point deviates from the model prediction can direct the modification of the model parameters, and 2) the integral of the velocity file should be aligned with the human's intentions and the target locations.

By the first clue, β is updated using Levenberg-Marquardt algorithm as in Equation (5) (6) and (7). Note that the operator "." indicates an element-wise operation throughout this letter. $\hat{v}_{x,t_k}, \hat{v}_{y,t_k}, \hat{v}_{z,t_k}$ and $v_{x,t_k}, v_{y,t_k}, v_{z,t_k}$ are the model predictions and the measurements of the velocities in x, y and z directions at time t_k. $\nabla_{\beta_x}, \nabla_{\beta_y}, \nabla_{\beta_z}$ are the gradients of $\hat{v}_{x,t_k}, \hat{v}_{y,t_k}, \hat{v}_{z,t_k}$ with respect to $\beta_x, \beta_y, \beta_z$, and $\lambda_x, \lambda_y, \lambda_z$ are the non-negative damping factors, which make the algorithm interpolate between the Gauss-Newton algorithm and the method of gradient descent [20]. If values of λ's components are large, the step goes in the direction for the gradient descent algorithm, and if values are small, the step will be close to the step in

the Gauss-Newton algorithm.

$$\beta_x = \beta_x - (\hat{v}_{x,t_k} - v_{x,t_k}) \nabla_{\beta_x} \hat{v}_{x,t_k} ./ \left((\nabla_{\beta_x} \hat{v}_{x,t_k})^{.2} + \lambda_x \right) \tag{5}$$

$$\beta_y = \beta_y - (\hat{v}_{y,t_k} - v_{y,t_k}) \nabla_{\beta_y} \hat{v}_{y,t_k} ./ \left((\nabla_{\beta_y} \hat{v}_{y,t_k})^{.2} + \lambda_y \right) \tag{6}$$

$$\beta_z = \beta_z - (\hat{v}_{z,t_k} - v_{z,t_k}) \nabla_{\beta_z} \hat{v}_{z,t_k} ./ \left((\nabla_{\beta_z} \hat{v}_{z,t_k})^{.2} + \lambda_z \right) \tag{7}$$

Second, to take advantage of the human intentions and the scene information, we further update β to minimize the following three objectives.

$$J_1(\beta) = || \int_{t_k}^{t_f} \hat{\mathrm{v}}(s; \alpha^*, \beta) ds - \mathrm{d}_{Togo} ||_2^2 \tag{8}$$

$$J_2(\beta) = || \mathrm{v}_{t_k} - \hat{\mathrm{v}}\,(t_k; \alpha^*, \beta)\,||_2^2 \tag{9}$$

$$J_3(\beta) = || \hat{\mathrm{v}}\,(t_f; \alpha^*, \beta)\,||_2^2 \tag{10}$$

The objective function (8) implies the predicted displacement from the current time t_k to the final time t_f should be close to the displacement to the goal d_{togo}, where the final time t_f is obtained from a stable zero-crossing of the model $\hat{\mathrm{v}}(t; \alpha^*, \beta)$ with a threshold, and d_{togo} is obtained by $\mathrm{p}_O - \mathrm{p}_{t_k}$. p_{t_k} is the current position of the human hand and p_O is the position of the target. In addition, the objective function (9) and (10) ensure the model precision at the current time t_k and the final time t_f, respectively. Therefore, to optimize the three objective functions is to fix the two ends of the future velocity file and to modify velocities in between so that the human's hand can reach the goal at the anticipated final time. Since (8), (9), (10) may not necessarily be minimized for the same value of β, we make weighted sum of the three objectives as $K(t_f, \beta) = \gamma_1 J_1(\beta) + \gamma_2 J_2(\beta) + \gamma_3 J_3(\beta)$, and update β_x, β_y and β_z by:

$$\beta_x = \beta_x + K\,(t_f, \beta) \nabla_{\beta_x} K ./ \left((\nabla_{\beta_x} K)^{.2} + \lambda'_x \right) \tag{11}$$

$$\beta_y = \beta_y + K\,(t_f, \beta) \nabla_{\beta_y} K ./ \left((\nabla_{\beta_y} K)^{.2} + \lambda'_y \right) \tag{12}$$

$$\beta_z = \beta_z + K\,(t_f, \beta) \nabla_{\beta_z} K ./ \left((\nabla_{\beta_z} K)^{.2} + \lambda'_z \right) \tag{13}$$

The outline of the algorithm at time t_k is shown in Algorithm 1. Lines 4-6 receive the position of the human hand x_{t_k}. Lines 5 represents the intent recognition process, and it determines the current action a_{t_k} and the future actions $\mathrm{a'}_{q^*}$. Line 3 obtains the current velocity where δT is the time difference between the current time step and the last time step. Lines 7-10 retrieve the current action model and the future action models if the action recognition changes. Lines 11-18 illustrate the model adaptation process. Finally, Lines 19-25 predict the trajectory for the current action and the predicted actions.

Algorithm 1: Long-term motion prediction algorithm at time t_k.

Input $\mathrm{v}_{1:t_{k-1}}$, $\mathrm{x}_{1:t_{k-1}}$, $a_{t_{k-1}}$, $\hat{\mathrm{v}}$, $\hat{\mathrm{v}}_{1,2,\ldots,np}$
Output t_f, $\hat{\mathrm{x}}_{t_{k+1},\ldots,t_f}$
1: x_{t_k} = PoseEstimation()
2: $a_{t_k}, \mathrm{a'}_{q^*}$ = IntentRecognition($\mathrm{x}_{1:t_k}$)
3: $\mathrm{v}_{t_k} = (\mathrm{x}_{t_k} - \mathrm{x}_{t_{k-1}})/\delta T$
4: **if** a_{t_k} changed **then**
5: $\quad \hat{\mathrm{v}}(t; \alpha^*, \beta)$ = RetrieveMotionModel(a_{t_k})
6: $\quad \hat{\mathrm{v}}_{1,2,\ldots,np}(t; \alpha^*, \beta)$ = RetrieveMotionModel($\mathrm{a'}_{q^*}$)
7: **end if**
8: **for** Iteration = 1,2,... **do**
9: $\quad$ Update β using Equation (5)(6)(7)
10: **end for**
11: t_f = ZeroCrossing($\hat{\mathrm{v}}(t; \alpha^*, \beta)$)
12: **for** Iteration = 1,2,... **do**
13: $\quad$ Update β using Equation (11)(12)(13)
14: $\quad t_f$ = Zero-Crossing($\hat{\mathrm{v}}(t; \alpha^*, \beta)$, threshold)
15: **end for**
16: **for** t = $t_{k+1}, \ldots, t_f$ **do**
17: $\quad \hat{\mathrm{x}}(t) = \mathrm{x}(t_k) + \int_{t_k}^{t} \hat{\mathrm{v}}(s; \alpha^*, \beta) ds$
18: **end for**
19: t_f' = Zero-Crossing($\hat{\mathrm{v}}_{1,2,\ldots,np}(t; \alpha^*, \beta)$,threshold)
20: **for** $i = 1, 2, \ldots, np$ **do**
21: $\quad$ Predict $\hat{\mathrm{x}}(t)$, for $t = t'_{f\,i-1} : t'_{f\,i}$
22: **end for**

V. Experiments

A. Scenario

A computer assembly task is considered. The task includes three subtasks, "Assemble CPU fan," "Assemble memory" and "Assemble system fan," where human should complete "Assemble CPU fan" first, and then complete either "Assemble memory" or "Assemble system fan". Therefore there are two plans in total. For all subtasks, the action sequence is reaching the part, obtaining the part and installing the part.

B. Hypothesis

We evaluate the effectiveness of the proposed method through experiments by verifying the following hypotheses.

- H1: Our method makes better trajectory prediction and duration estimation than other baseline methods when the human's intentions are known.
- H2: Our method makes good trajectory prediction and duration estimation when the human's intentions are unknown.
- H3: Our proposed adaption algorithm can quickly adapt the movement model to the motions with the same motion type but different starting and ending positions.

The conditions that the human's intentions are known or unknown reflect whether the human intent recognition module in Section III is involved or not. The first hypothesis shows the effectiveness of our trajectory prediction module in Section IV only, and the second hypothesis shows the effectiveness of the

overall structure. For the third hypothesis, different starting and ending positions implies different human behaviors, and this hypothesis shows the adaptiveness of our method to different human behavior.

C. Manipulated Variables

We manipulated three variables to evaluate the effectiveness of our method. The first variable is different methods. Our algorithm is compared against the following baseline methods:

1) *Vector field method [21]:* This method performs long-term predictions of human motion by employing a map of vector fields. The final time of an action for this method can be estimated by setting a velocity threshold.
2) *Minimum jerk model [9]:* This model states that human arms move by minimizing mean-square jerk for any unconstrained point-to-point movement. As proposed in [22], the value of the final time can be obtained by a search algorithm.
3) *DTW-based method [23]:* This method utilizes dynamic time warping (DTW) algorithm to align the online activity with a reference template, then it estimates the expected duration by proportionally manipulating the alignment time.
4) *Our method without task context:* This is our method without the adaptation steps in Lines 12-15 of Algorithm 1.

The second manipulated variable considers task types. Two types are considered: (1) the task that only includes a simple reaching task ("reaching the CPU fan" as in Section V-A), where the human's intention is determined and known beforehand, and (2) the whole task as described in Section V-A, where the human's intention is uncertain and needs to be recognized. Our method and all the 4 baseline methods are applied to the first task type, while for the second task type, we only test our method.

The third manipulated variable is a set of progress percentages for the test timings. They are 10%, 30%, 40% 50%, 60%, 70%, 80%, 85% and 90%. For the reaching task, at these progress percentages of the reaching motion, we will study trajectory prediction and duration estimation, while for the whole computer assembly task, we will study trajectory prediction and duration estimation at these progress percentages of the reaching motion of the second subtask, when the human intent recognition starts to take an important role to decide on which subtask the human intends to do.

For data collection, we instructed 6 human subjects to complete the computer assembly task for both plans, then each human subjects were asked to conduct the task for several times using either plan randomly. In total, we collected 51 trials, but 2 of them were too noisy to be used, thus 49 trials were utilized in the experiments. Among them, 42 trials were for offline training and 7 trials were for testing. The training and the testing sets were chosen through a random selection to avoid bias. For the simple reaching task, we only used segments of each trial, and for the whole task, we used the whole trajectory of each trial.

D. Dependent Measures

To quantify the accuracy of trajectory prediction, we measure the mean Euclidean distance between the predictions and the observed data, where the Euclidean distance error for each trial at time t_k is the average of $||\hat{\mathbf{x}}(t) - \mathbf{x}(t)||_2, t = t_{k+1}, \ldots, t_f$. For duration estimation, we compute the mean absolute percentage error. The absolute percentage error is $|\frac{t'_f - T}{T}| \times 100\%$, where T is the groundtruth duration.

E. Results

H1: Table I shows the duration estimation results when the human is conducting a simple reaching task. Our method has significantly smaller average percentage errors than other baseline methods at all the test timings except for the timing at the 80% progress, where ours is the second best. By doing paired t-test, we find that the results of our method are significantly better than any other baseline method ($P < 0.05$). Besides, after the 30% progress, the average percentage error of our method drops below 5%, while for other baseline methods except for our method without task context, the errors are always above 5%. For our method and the minimum jerk method, the average percentage error drops as time elapses, while for the other methods, the average percentage error oscillates or even increases at the latter part. A possible reason is that our method and minimum jerk method explicitly utilize the knowledge of the target positions, while the others do not. Among the baseline methods, the DTW-based method is the best in the early phase before the 30% progress, the vector field method is the best in the middle phase from the 30% to the 60% progress, and our method without task context is the best in the latter phase after the 60% progress.

Table II shows the trajectory prediction results when the human is conducting a simple reaching task. Our method has the least average distance errors at all the test timings except for the timings at 85% and 90% progress, where ours is the second best and the third best respectively. By doing paired t-test, we find that the results of our method are significantly better than any other baseline method ($P < 0.05$). Besides, all the average distance errors of our method remain within 20 mm, and it drops below 10 mm after the 60% progress. For our method, the vector field method and the minimum jerk method, the average percentage errors show a trend of decreasing as time elapses. However, the DTW-based method and our method without task context show a slight increase in the early phase before the 50% progress. Among the baseline methods, the minimum jerk method is the best throughout all the test timings except for the timing at 90% progress, and the DTW-based method is the worst throughout all the test timings except for the timing at 10% progress.

H2: Table III displays the results of the trajectory prediction and the duration estimation for the computer assembly task. For the current action, the average percentage error of the duration estimation varies between 1.3% and 14.9%, and it drops below 5% after the 30% progress. The average distance error for trajectory prediction varies between 3.6 mm to 19.7 mm, and it drops below 10 mm after the 60% progress. For the predicted actions, different from the results for the current action, the errors do not feature a decreasing trend, which means β is not the same among different motions for one task. The average percentage error of the duration estimation varies between 6.9% and 13.2%, and the average distance error of the trajectory prediction varies

TABLE I
AVERAGE ABSOLUTE PERCENTAGE ERROR OF DURATION ESTIMATION FOR A SIMPLE REACHING TASK (UNIT: %)

Time Percentage	10%	30%	40%	50%	60%	70%	80%	85%	90%
Ours	**13.6**±12.0	**3.9**± 9.7	**2.0**±8.4	**4.0**±7.0	**1.9**±3.5	**3.2**±3.1	1.7±3.6	**1.2**±6.1	**0.1**±5.2
Vector field	25.8±10.3	14.1±7.6	8.7±6.4	8.2±4.5	6.9±3.9	6.7±4.9	10.0±3.8	10.6±5.8	8.4±4.7
Minimum jerk	25.5±5.5	22.1± 6.8	22.4± 7.7	21.5±8.3	18.3± 7.4	14.4±6.2	10.0±5.2	7.7± 4.1	6.1± 3.6
DTW	15.3±18.1	10.9± 12.2	12.1± 11.7	11.7±10.7	10.8±9.8	13.7±7.1	11.5±10.6	14.9± 16.7	12.4± 14.9
Ours w/o task context	16.1±11.6	14.1± 8.8	12.8± 8.2	9.6±6.2	9.9±5.4	6.2±2.3	**1.1**±2.9	3.9±4.9	2.4±5.1

TABLE II
AVERAGE DISTANCE ERROR OF TRAJECTORY PREDICTION FOR A SIMPLE REACHING TASK (UNIT: MM)

Time Percentage	10%	30%	40%	50%	60%	70%	80%	85%	90%
Ours	**16.4**±8.7	**17.3**±4.8	**13.0**±3.3	**12.2**±3.4	**10.3**±3.9	**6.3**±2.0	**5.1**±1.7	4.5±1.0	3.2±0.7
Vector field	69.0±14.5	38.9±10.0	27.4±10.3	21.5±9.2	20.2±9.3	22.4±14.9	13.4±14.3	10.4±11.8	7.4 ±8.3
Minimum jerk	23.8±6.5	18.0±4.2	17.3±4.2	15.6±4.2	12.8±3.7	9.4±3.0	5.9±2.4	**4.3**±1.9	3.1±1.5
DTW	42.7±24.6	55.1±16.4	59.5±14.6	60.6±15.8	59.3±16.5	49.9± 17.8	43.9±19.1	40.3±21.9	32.7±16.8
Ours w/o task context	28.1±8.4	28.9±6.0	31.2±6.5	31.5±6.1	29.5±5.4	20.5±4.5	10.1±1.7	5.1±1.2	**2.3**±0.5

TABLE III
THE AVERAGE PERCENTAGE ERROR OF THE DURATION ESTIMATION AND THE AVERAGE DISTANCE ERROR OF THE TRAJECTORY PREDICTION FOR THE COMPUTER ASSEMBLY TASK

Time percentage		10%	30%	40%	50%	60%	70%	80%	85%	90%
time error (%)	current action	14.9±11.8	4.9± 8.6	4.5±4.4	4.4±7.5	2.3±3.7	2.4±2.9	3.1±3.9	2.2±4.7	1.3±4.6
	future actions	11.2±10.6	10.5±13.3	8.7±9.5	9.2±10.6	10.3±11.9	7.4±10.7	6.9±11.6	13.2±9.1	11.2±10.5
trajectory error (mm)	current action	19.7±6.8	14.6±7.6	13.0±5.6	13.4±7.6	9.1±5.9	7.3±3.5	6.3±2.6	5.4±1.0	3.6±1.1
	future actions	21.4±13.6	9.4±8.8	12.3±8.4	14.7±11.4	8.5±6.7	10.4±13.7	8.9±9.6	7.7±8.1	12.7±9.8

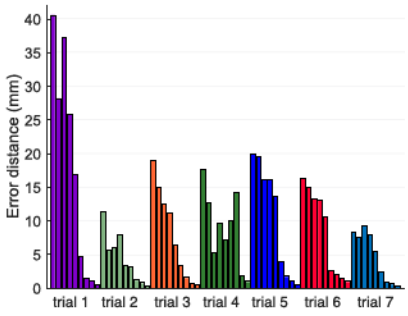

Fig. 2. The distance errors of the model predictions for adapting to a new task.

between 7.7 mm to 21.4 mm. The largest average distance error of the trajectory prediction 21.4 mm appears at the 10% progress, when the human intent recognition module has a difficulty distinguishing the human's plan due to the lack of observations. After the 30% progress, the distance error drops exceedingly, because the human's intentions are correctly recognized. Overall, the duration estimation is good, since the error is within 5% for the current action and within 14% for the future actions after the 30% progress; the trajectory prediction is good, since errors are less than 25 mm, which is within the safety distance as in [10].

H3: We adapt the sigma-lognormal model which is trained on the "reaching for the CPU fan" task to a new task "reaching for the memory," where the motion type is the same, but the starting and the ending positions of the human hand are different. Fig. 2 shows the distance errors of the model predictions for 7 trials. The model parameters are adapted through the first trial to the seventh trial, and the distance errors are calculated at 10%, 30%, 40% 50%, 60%, 70%,80%, 85% and 90% progress of each trial. As shown in the figure, the distance errors are large for the first trial, which can reach up to 40 mm. However, the error drops drastically after the first trial, which means the model can adapt to the new reaching task within one trial. Table IV summarizes the statistics for the results of the trajectory prediction and the duration estimation, and it shows that the results are comparable with the results of our method in Table I and Table II, where the model is trained and tested on the same reaching task. For the duration estimation, the percentage error is below 10% after the 30% progress and below 5% after the 50% progress. For the trajectory prediction, all the average distance errors remain within 20 mm, and it drops below 10 mm after the 50% progress.

VI. DISCUSSION

The sigma-lognormal model fits the human motion data better than the minimum jerk model. Fig. 3 shows the residual diagrams for the minimum jerk model and the sigma lognormal model of one trial for the simple reaching task. Both diagrams are unskewed and include no outliers, and both residuals are normally distributed with probability $P = 0.09 > 0.05$ ($\chi^2 = 4.81$, $df = 2$) and probability $P = 0.18 > 0.05$ ($\chi^2 = 3.41$, $df = 2$) respectively for D'Agostino-Pearson's test. Actually, this normality conclusion holds for all the other 6 test trials, which indicates that we can trust the results of the regression analysis. Note that although normality assumption is not needed for nonlinear regression, if it is clearly violated, we may not trust the model. That is why we do the normality test. Two things to note about the normality: 1) Normality assumption is not needed for nonlinear regression, but if it is clearly violated, we may not trust the model; 2) We trained one minimum jerk model and one sigma lognormal model on all the test data, and the D'Agostino-Pearson's test shows that the residuals for both models are not normally distributed. This again shows that one model for all trials is not sufficient to cover all the trends and

TABLE IV
THE AVERAGE PERCENTAGE ERROR OF THE DURATION ESTIMATION AND THE AVERAGE DISTANCE ERROR OF THE TRAJECTORY PREDICTION FOR ADAPTING TO ANOTHER TASK

Time Percentage	10%	30%	40%	50%	60%	70%	80%	85%	90%
time error (%)	15.1±16.2	12.6 ±10.2	5.4±6.3	5.1±6.2	4.2±5.7	3.5±4.9	3.8±5.2	1.8±2.5	1.2±2.3
trajectory error(mm)	19.0±10.3	14.8±7.5	14.2±10.9	13.1±6.3	9.1±4.8	4.3±2.7	3.3±4.8	1.1±0.5	0.6±0.3

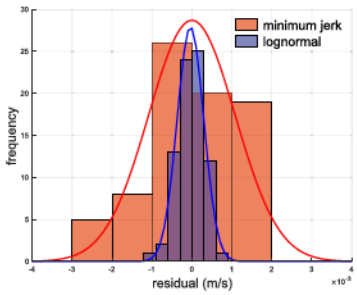

Fig. 3. Histograms and normal fits for the residuals of the minimum jerk model and the lognormal model.

there are different patterns in human's movement when doing the same action. Next, we check the mean absolute residual to see which model fits the data better. The mean absolute residual for the minimum jerk models is 1.20 ± 1.05 mm, and the mean absolute residual for the sigma lognormal models is 0.52 ± 0.42 mm. The sigma-lognormal model has 56.5% less error. This shows that the sigma-lognormal model fits the data better than the minimum jerk model.

VII. CONCLUSION

We proposed a recognition-then-prediction framework for long-term human hand prediction and action duration estimation, where the prediction horizon relies on how far in the future we can predict the human actions rather than a preset time. We improved the recognition method by introducing scaling vectors, which takes advantage of the task knowledge of action transitions. The sigma-lognormal equation was proposed to model the human's movement, and an adaptation algorithm was proposed to accommodate different human behaviors. We conducted experiments for a computer assembly task. The results showed the effectiveness of the overall framework and the effectiveness of the trajectory prediction module only. The proposed method had significantly smaller average percentage errors as well as average distance errors. This is a definite advantage in predicting the long-term human hand trajectory for collaborative task planning. The experiments also showed that our adaptation algorithm can quickly adapt the movement model to the trajectories with the same motion type but different starting and ending positions.

REFERENCES

[1] A. Rudenko, L. Palmieri, M. Herman, K. M. Kitani, D. M. Gavrila, and K. O. Arras, "Human motion trajectory prediction: A survey," *Int. J. Robot. Res.*, vol. 39, no. 8, pp. 895–935, 2020.
[2] J. Sung, C. Ponce, B. Selman, and A. Saxena, "Human activity detection from rgbd images," in *Proc. Workshops 25th AAAI Conf. Artif. Intell.*, 2011, pp. 47–55.
[3] S. Mukherjee, L. Anvitha, and T. M. Lahari, "Human activity recognition in rgb-d videos by dynamic images," *Multimedia Tools Appl.*, vol. 79, no. 27, pp. 19 787–19 801, 2020.
[4] H. Xue, D. Q. Huynh, and M. Reynolds, "Bi-prediction: Pedestrian trajectory prediction based on bidirectional lstm classification," in *Proc. Int. Conf. Digit. Image Comput.: Techn. Appl.*, 2017, pp. 1–8.
[5] J. Martinez, M. J. Black, and J. Romero, "On human motion prediction using recurrent neural networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 2891–2900.
[6] L. Sun, W. Zhan, and M. Tomizuka, "Probabilistic prediction of interactive driving behavior via hierarchical inverse reinforcement learning," in *Proc. 21st Int. Conf. Intell. Transp. Syst.*, 2018, pp. 2111–2117.
[7] Y. Cheng, W. Zhao, C. Liu, and M. Tomizuka, "Human motion prediction using semi-adaptable neural networks," in *Proc. Amer. Control Conf.*, 2019, pp. 4884–4890.
[8] D. Helbing and P. Molnar, "Social force model for pedestrian dynamics," *Phys. Rev. E*, vol. 51, no. 5, 1995, Art. no. 4282.
[9] T. Flash and N. Hogan, "The coordination of arm movements: An experimentally confirmed mathematical model," *J. Neurosci.*, vol. 5, no. 7, pp. 1688–1703, 1985.
[10] Y. Cheng, L. Sun, C. Liu, and M. Tomizuka, "Towards efficient human-robot collaboration with robust plan recognition and trajectory prediction," *IEEE Robot. Automat. Lett.*, vol. 5, no. 2, pp. 2602–2609, Apr. 2020.
[11] Z. Wang, Z. Shi, Y. Li, and J. Tu, "The optimization of path planning for multi-robot system using boltzmann policy based q-learning algorithm," in *Proc. IEEE Int. Conf. Robot. Biomimetics*, 2013, pp. 1199–1204.
[12] G. Gangadhar, D. Joseph, and V. S. Chakravarthy, "An oscillatory neuromotor model of handwriting generation," *Int. J. Document Anal. Recognit.*, vol. 10, no. 2, pp. 69–84, 2007.
[13] R. Plamondon, C. Feng, and A. Woch, "A kinematic theory of rapid human movement. part iv: A formal mathematical proof and new insights," *Biol. Cybern.*, vol. 89, no. 2, pp. 126–138, 2003.
[14] R. Plamondon, "The design of an on-line signature verification system: From theory to practice," *Int. J. Pattern Recognit. Artif. Intell.*, vol. 8, no. 03, pp. 795–811, 1994.
[15] S. Djeziri, W. Guerfali, R. Plamondon, and J. Robert, "Learning handwriting with pen-based systems: Computational issues," *Pattern Recognit.*, vol. 35, no. 5, pp. 1049–1057, 2002.
[16] C. O'Reilly and R. Plamondon, "Development of a sigma-lognormal representation for on-line signatures," *Pattern Recognit.*, vol. 42, no. 12, pp. 3324–3337, 2009.
[17] R. Plamondon, C. O'Reilly, J. Galbally, A. Almaksour, and É. Anquetil, "Recent developments in the study of rapid human movements with the kinematic theory: Applications to handwriting and signature synthesis," *Pattern Recognit. Lett.*, vol. 35, pp. 225–235, 2014.
[18] K. Levenberg, "A method for the solution of certain non-linear problems in least squares," *Quarterly Appl. Mathematics*, vol. 2, no. 2, pp. 164–168, doi: 10.1090/qam/10666.
[19] A. Woch, R. Plamondon, and C. O'Reilly, "Kinematic characteristics of successful movement primitives in young and older subjects: A delta-lognormal comparison," *Hum. Mov. Sci*, vol. 30, no. 1, pp. 1–17, 2011.
[20] K. Levenberg, "A method for the solution of certain non-linear problems in least squares," *Quart. Appl. Math.*, vol. 2, no. 2, pp. 164–168, 1944.
[21] J. F. Carvalho, M. Vejdemo-Johansson, F. T. Pokorny, and D. Kragic, "Long-term prediction of motion trajectories using path homology clusters," in *IROS*, 2019, pp. 765–772.
[22] C. T. Landi, Y. Cheng, F. Ferraguti, M. Bonfè, C. Secchi, and M. Tomizuka, "Prediction of human arm target for robot reaching movements," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2019, pp. 5950–5957.
[23] R. Maderna, M. Ciliberto, A. M. Zanchettin, and P. Rocco, "Robust real-time monitoring of human task advancement for collaborative robotics applications," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2020, pp. 11094–11100.