Semi-Adaptable Human Hand Motion Prediction Based on Neural Networks and Kalman Filter

Juxing Wang and Linyong Shen
School of Mechatronic Engineering and Automation, Shanghai University, Shanghai, 200444, China
e-mail: juxing@i.shu.edu.cn

Abstract. This paper focuses on predicting trajectories of the human hand in order to improve the safety for human-robot interactions. In this work, the position and orientation are represented by two curves in the operation space such that the same algorithm can be used for both position and orientation prediction. The motion prediction is achieved in two steps. Firstly, the neural network (NN) model is applied for offline training to model the human hand motion. Secondly, the Kalman filter is added to adjust the weight coefficients of the NN model’s output layer online when a set of new data is measured, such that the NN model is adaptive to new data. An experiment study has been conducted to validate the effectiveness of the proposed algorithm. The result shows that the proposed algorithm achieves a higher prediction accuracy and requires a smaller amount of data to achieve optimal performance compared with the advanced method.

1. Introduction
To guarantee the safety during the human-robot interaction, it is important for the robots to predict the human motion, such that the robots can readjust their trajectories accordingly to avoid impact [1]. Therefore, the prediction of human motion becomes a hot research issue for Human-Robot Collaboration.

Prior works in human motion prediction can be divided into two categories according to whether the prediction results have confidence information: deterministic approach and probabilistic approach. The probabilistic approach captures changes in hand motion by establishing a probabilistic model over the demonstrations. Hidden Markov Model (HMM) [2], Gaussian Process [3] and Gaussian Mixture Model [4] have been effectively used to code the demonstrations and predict human movement. These probabilistic models are directly modelled over the demonstrations, which makes the generalization performance of these algorithms poor in new environments.

The research domains of deterministic approach prominently focus on optimizing the cost function of human motion, latent variable based probabilistic models and various Deep Neural Networks (DNNs) with various structures. Human motion trajectories can be predicted by optimizing the cost function of human movement. The biggest challenge of this type of methods is that it is difficult to obtain the cost function of human motion.[5] used the Inverse Reinforcement Learning (IRL) algorithm to recover the cost function from a set of demonstrations, and then optimized this cost function to predict future human movement. In recent years, Deep Neural Networks (DNNs) have gradually become a popular method in human motion prediction [6,7]. A common method is the Recurrent Neural Networks (RNNs). However, the speed of training the RNN is very slow, especially
when processing a long sequence of training example. Another shortcoming is that RNN models are fixed, making them unable to adapt to random changes and individual differences in human motion.

In this paper, we aim to solve these problems by proposing a semi-adaptable neural network. Specifically, a neural network is trained offline to represent the human motion transition model, and then Kalman Filter parameter adaptation algorithm is adopted for online updating parameters of the neural network model’s output layer. The neural network model has a simple structure, fast training speed, and powerful modeling capabilities. The combination of parameters adaption algorithm and the neural network makes this algorithm adaptive to individual differences in human motions. In the experimental part at the end, this method will be compared with the advanced method: NN-RLS [8].

2. Problem Description and Solution Approach

2.1. Problem Description

The manipulation task in this paper is to pick up and place the object on the desktop, as shown in figure 1. The target object is picked up from target location A and placed to target location B, and then picked up from target location B and placed to target location A. In this process, the trajectories of human hand are shown by the white and black dashed lines in figure 1.

\[ \mathbf{p}_k, \mathbf{w}_k \text{ represent hand’s position vector and orientation vector at time step } k, \text{ respectively.} \]

The problem to be solved is to utilize past information \( \mathbf{X}_k = [\mathbf{p}_k, \mathbf{w}_k, \mathbf{p}_{k+1}, \mathbf{w}_{k+1}, \mathbf{p}_{k+2}, \mathbf{w}_{k+2}]^T \in \mathbb{R}^{3 \times 6} \) to predict the future information \( \mathbf{X}_{k+1} = [\mathbf{p}_{k+3}, \mathbf{w}_{k+3}, \mathbf{p}_{k+4}, \mathbf{w}_{k+4}]^T \in \mathbb{R}^{3 \times 4} \). The past information \( \mathbf{X}_k \) is the position and orientation at time steps \( k, k + 1 \) and \( k + 2 \), while the future information is the position and orientation at time steps \( k + 3 \) and \( k + 4 \).

2.2. Data collection and preprocessing

To determine the position and orientation of the participant’s hand, a 3D-printed tool is fabricated where 3 target balls are be installed, as shown in figure 1.

The trajectories of the 3 target balls with respect to the world frame \( \mathcal{W} \) are recorded by the Qualisys brand of motion capture system during the motion with a recording frequency of 20Hz. Then, the trajectory of the local frame \( \mathcal{A} \) attached on the tool can be determined by the trajectories of the 3 target balls.

The origin point of the local frame \( \mathcal{A} \) is given by

\[ \mathbf{p} = \frac{\mathbf{b}_1 + \mathbf{b}_2}{2} \]

The \( x, y, z \) axes of the local frame \( \mathcal{A} \) are given by

Figure 1. The pick-and-place operation motion.
\[ n_x = \frac{b_2 - b_1}{\left\| b_2 - b_1 \right\|}, \quad n_y = \frac{b_3 - p}{\left\| b_3 - p \right\|}, \quad n_z = n_x \times n_y \]  
\[ n_\widehat{x} = n_x, \quad n_\widehat{y} = n_y, \quad n_\widehat{z} = n_\widehat{x} \times n_\widehat{y} \]  
(2)

Therefore, the homogeneous transformation matrix of frame \( \{A\} \) with respect to frame \( \{W\} \) can be represented by

\[ T_{W,A} = \begin{bmatrix} R_{W,A} & p \\ 0_{1 \times 3} & 1 \end{bmatrix} \]  
(3)

where \( R_{W,A} = [n_x, n_y, n_z] \) is the orientation of frame \( \{A\} \) with respect to frame \( \{W\} \).

However, to describe the position and orientation of frame \( \{A\} \), it is more convenient to use a vector \( [9] \) instead of \( T_{W,A} \). In this paper, the pose of frame \( \{A\} \) with respect to frame \( \{W\} \) is given by

\[ a = [p \ w] \]  
(4)

where \( w = p + \lambda \theta \omega \), in which

\[ 2 \cos \theta + 1 = \text{trace}(R_{W,A}) \]  
(5)

and

\[ \omega = \frac{1}{2 \sin \theta} (R_{W,A} - R_{W,A}^T) \]  
(6)

Then, the continuous motion of a rigid body can be represented by two continuous spatial curves, which is more convenient for trajectory smoothing with Savitzky-Golay Filter algorithm in subsequent steps. In this work, the 3D-printed tool and the participant’s hand are attached and have limited relative motion. Therefore, the position and the orientation of the hand at step \( k \) can be represented by a vector \( a_k = [p_k \ w_k] \).

2.3. Trajectory Prediction Method

Considering that the human hand movement is highly nonlinear, we choose a three-layer neural network with the ReLU activation function (NN model) to construct the state transition function \( f^*(X_k) \) from \( X_k \) to \( \tilde{X}_{k+1} \). Human motion is highly time-varying, and different people complete the same task in significantly different ways. Therefore, for adapting to the new environment, the neural network needs to be combined with an adaptive algorithm to update the model online. This paper adjusts the weight coefficients of the neural network’s output layer online to improve the generalization performance of the model.

- Training the neural network

In this three-layer neural network, the number of neurons in the input layer, hidden layer and output layer are set to 12, 20 and 18, respectively. The learning rate is set to be 0.0001. The loss function is set to be L2, and the number of epochs is set to 600. The neural network model is described by

\[ f^*(X_k) = W^T \max(0, g(U, s_k)) + e(s_k) \]  
(7)

in which \( s_k = [X_k^T, 1]^T \in \mathbb{R}^{3 \times 6+1} \) is the input vector, \( X_k \) defined in the 2.1 section of problem description, \( g(U, s_k) \) denotes the first two layers neural network, whose weights are packed in \( U \). \( e(s_k) \in \mathbb{R}^{3 \times 4} \) is the function reconstruction error. \( W \in \mathbb{R}^{21 \times 12} \) is the weights of the last layer. \( W \) (a total of 252 parameters) will be iteratively updated in the next section using Kalman Filter algorithm.

- Kalman filtering for parameters adaptation

The purpose of Kalman filtering is to adjust the parameters of the prediction model online such that the model is adaptive to new data. To apply this algorithm, the weight coefficients of the neural network’s output layer are regarded as the state variables, denoted as \( x \). Each measured the position and orientation vector of the hand can be regarded as an observation to the state variables, denoted as \( y \). The state equation is

\[ x_{k+1} = Ax_k + e_{k+1} \]  
(8)

The observation equation is given by

\[ y_{k+1} = Cx_{k+1} + f_{k+1} \]  
(9)
where \( x_k \) is the weight coefficients of the neural network’s output layer (\( W \) defined in (7)) in the \( k^{th} \) step. \( y_{k+1} \) is the actual measured value of the position and orientation at time steps \( k + 3 \) and \( k + 4 \) (\( \tilde{X}_{k+1} \) defined in the 2.1 section of problem description). \( e_{k+1} \) and \( f_{k+1} \) are the noise single in the \( k + 1^{th} \) step, \( e_{k+1} \sim \mathcal{N}(0,Q) \) and \( f_{k+1} \sim \mathcal{N}(0,R) \). \( A \) represents the state transition matrix which is identity matrix in this model.

\( C \) is a diagonal composition of the hidden layer’s output, i.e.

\[
C = \begin{bmatrix}
    c & 0 & \cdots & 0 \\
    0 & c & \cdots & 0 \\
    \vdots & \vdots & \ddots & \vdots \\
    0 & 0 & \cdots & c
\end{bmatrix} \in \mathbb{R}^{12 \times (21 \times 12)}
\]

in which \( c \in \mathbb{R}^{1 \times 21} \) is the output of the hidden layer, and 12 is the number of neurons of the output layer in the NN model.

Each time when a new measured datum is added, the following steps are executed.

Firstly, estimating the weight coefficients of the last layer at step \( k + 1 \) as well as it’s covariance matrix

\[
\bar{x}_{k+1} = A x_k \tag{10}
\]

and

\[
\bar{P}_{k+1} = A P_k A^T + Q \tag{11}
\]

where \( Q \) represents the covariance of the process noise (\( e_{k+1} \)). \( P_k \) is the covariance matrix of the state variable (\( x_k \)) at step \( k \).

Secondly, revising the weight coefficients of the last layer at step \( k + 1 \)

\[
x_{k+1} = \bar{x}_{k+1} + K_{k+1} (y_{k+1} - C \bar{x}_{k+1}) \tag{12}
\]

where \( K_{k+1} = \bar{P}_{k+1} C^T (C \bar{P}_{k+1} C^T + R)^{-1} \), and \( R \) represents the covariance of the measurement noise (\( f_{k+1} \)).

Thirdly, updating the covariance matrix by

\[
P_{k+1} = (I - K_{k+1} C) \bar{P}_{k+1} \tag{13}
\]

3. Experimental Validation

To validate the effectiveness of the proposed method in this paper, an experimental study has been conducted. In the experiment, 3 participants performed a pick-and-place operation as described in the 2.1 section of problem description. In total, 4 sets of data are obtained in our experiment, the sizes of which are 5260, 540, 2055, 1722, respectively. The first three sets of data (data size:5260, 540, 2055) obtained from the first and second participants are used as the training data for the NN model, while the rest set of data (data size:1722) obtained from the third participant is used for the online prediction.

In Kalman filter algorithm, the initial covariance matrix \( P_1 \) is set to be an identity matrix. In addition, the values on the diagonal of the diagonal matrix \( Q \) and \( R \) are set to 1 and 0.5, respectively. Like Kalman filter, the recursive least squares (RLS) algorithm also is a common online parameter estimation method. Chen et al. [8] proposed the method which combined NN and RLS to predict the position of the wrist joint in future movement. A comparative study is conducted between the proposed NN-Kalman algorithm and NN-RLS algorithm proposed in [8]. The prediction errors are the mean square errors of the online prediction values with respect to the actual measured values.

Figure 2 and figure 3 are the comparison between NN-Kalman method and NN-RLS model. In two figures, the blue and brown curves are always beneath the yellow and purple curves, respectively, which shows that the prediction accuracy of the NN-Kalman algorithm is higher than the NN-RLS algorithm. In addition, the blue and brown curves become stable after about 500 steps, while the yellow and purple curves become stable after about 700 steps. The proposed NN-Kalman algorithm requires fewer data than the NN-RLS algorithm. Therefore, the NN-Kalman algorithm is more adaptive than the NN-RLS algorithm.
Figure 2. The prediction errors of the hand position. The blue curve and the brown curve represent the prediction errors of the position at time step $k+3$ and time step $k+4$, respectively, using the NN-Kalman method. The yellow curve and the purple curve represent the prediction errors of the position at time step $k+3$ and time step $k+4$, respectively, using the NN-RLS method.

Figure 3. The prediction errors of the hand orientation. The blue curve and the brown curve represent the prediction errors of the orientation at time step $k+3$ and time step $k+4$, respectively, using the NN-Kalman method. The yellow curve and the purple curve represent the prediction errors of the orientation at time step $k+3$ and time step $k+4$, respectively, using the NN-RLS method.

Analysing the data in Table 1 shows that compared with NN-RLS, NN-Kalman has reduced $E_i$ by about 65%-80%, $E_l$ by about 30%-50%, and $E_a$ by about 55%-65%. The result shows that the NN-Kalman method has higher prediction accuracy compared with the NN-RLS method. Especially when the data size is small, the NN-Kalman method has more significant advantages.

| Method  | Average error of initial 200 steps: $E_i$ | Average error of last 200 steps: $E_l$ | Average error of all steps: $E_a$ |
|---------|------------------------------------------|---------------------------------------|----------------------------------|
| NN-KLM  | $p_{k+3}$ $5.18 \times 10^{-6}$  | $2.72\times 10^{-6}$  | $3.25\times 10^{-6}$ |
|         | $p_{k+4}$ $1.43\times 10^{-5}$  | $6.91\times 10^{-6}$  | $8.67\times 10^{-6}$ |
|         | $w_{k+3}$ $1.23\times 10^{-5}$  | $5.35\times 10^{-6}$  | $5.99\times 10^{-6}$ |
|         | $w_{k+4}$ $3.49\times 10^{-5}$  | $1.22\times 10^{-6}$  | $1.49\times 10^{-6}$ |
| NN-RLS  | $p_{k+3}$ $2.36\times 10^{-5}$  | $4.01\times 10^{-6}$  | $7.38\times 10^{-6}$ |
|         | $p_{k+4}$ $7.10\times 10^{-5}$  | $1.29\times 10^{-6}$  | $2.28\times 10^{-6}$ |
|         | $w_{k+3}$ $3.84\times 10^{-5}$  | $8.93\times 10^{-6}$  | $1.34\times 10^{-6}$ |
|         | $w_{k+4}$ $1.02\times 10^{-4}$  | $2.32\times 10^{-6}$  | $3.58\times 10^{-6}$ |

Generally, the proposed NN-Kalman method has higher prediction accuracy and better adaptivity than the NN-RLS method. The NN-Kalman method performs better than the NN-RLS method because the process noise and the measurement noise are taken into account in this model.
4. Conclusion
In this work, a semi-adaptable neural network for human hand motion prediction is proposed, in which the position and orientation information of the human hand are represented by two curves in the operation space. The neural network model is established to predict the hand motion of the next two time steps. Then the Kalman filter algorithm is added to update the weight coefficients of the neural network model’s output layer online when a new datum is measured, such that the NN model is adaptive to new data. An experiment study has been conducted to validate the effectiveness and adaptivity of the proposed algorithm.

References
[1] R. C. Luo and L. Mai (2019) Human Intention Inference and On-Line Human Hand Motion Prediction for Human-Robot Collaboration. In: IEEE/RSJ International Conference on Intelligent Robots and Systems. (IROS). Macau. pp. 5958-5964.
[2] H. Ding, G. Reißig, and K. Wijaya (2011) Human arm motion modeling and long-term prediction for safe and efficient Human-Robot-Interaction. In: IEEE International Conference on Robotics and Automation, Shanghai, pp. 5875-5880.
[3] J. M. Wang, D. J. Fleet and A. Hertzmann (2008) Gaussian Process Dynamical Models for Human Motion. In: IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 30, no. 2, pp. 283-298.
[4] J. Mainprice and D. Berenson (2013) Human-robot collaborative manipulation planning using early prediction of human motion. In: IEEE/RSJ International Conference on Intelligent Robots and Systems. Tokyo. pp. 299-306.
[5] J. Mainprice, R. Hayne and D. Berenson (2015) Predicting human reaching motion in collaborative tasks using Inverse Optimal Control and iterative re-planning. In: IEEE International Conference on Robotics and Automation (ICRA), Seattle. pp. 885-892.
[6] J. Butepage, M. J. Black, D. Kragic and H. Kjellström (2017) Deep Representation Learning for Human Motion Prediction and Classification. In: the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI. pp. 1591-1599.
[7] P. Ghosh, and J. Song, E. Aksan (2017) Learning Human Motion Models for Long-Term Predictions. In: the International Conference on 3D Vision (3DV), Qingdao. pp. 458-466.
[8] Y. Cheng, W. Zhao, C. Liu, and M. Tomizuka (2019) Human motion prediction using semi-adaptable neural networks”. In: IEEE Amer. Control Conf., pp. 4884–4890.
[9] X. F. Zha (2002) Optimal pose trajectory planning for robot manipulators. Mechanism and Machine Theory, 37(10): 1063–1086.