

Research Article

Skill Learning for Intelligent Robot by Perception-Action Integration: A View from Hierarchical Temporal Memory

Xinzheng Zhang,¹ Jianfen Zhang,¹ and Junpei Zhong²

¹*School of Electrical and Information Engineering, Jinan University, Zhuhai, China*

²*National Institute of Advanced Industrial Science and Technology (AIST), Tokyo, Japan*

Correspondence should be addressed to Jianfen Zhang; eejfzhang@gmail.com

Received 3 July 2017; Accepted 7 September 2017; Published 15 November 2017

Academic Editor: Yanan Li

Copyright © 2017 Xinzheng Zhang et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Skill learning autonomously through interactions with the environment is a crucial ability for intelligent robot. A perception-action integration or sensorimotor cycle, as an important issue in imitation learning, is a natural mechanism without the complex program process. Recently, neurocomputing model and developmental intelligence method are considered as a new trend for implementing the robot skill learning. In this paper, based on research of the human brain neocortex model, we present a skill learning method by perception-action integration strategy from the perspective of hierarchical temporal memory (HTM) theory. The sequential sensor data representing a certain skill from a RGB-D camera are received and then encoded as a sequence of Sparse Distributed Representation (SDR) vectors. The sequential SDR vectors are treated as the inputs of the perception-action HTM. The HTM learns sequences of SDRs and makes predictions of what the next input SDR will be. It stores the transitions of the current perceived sensor data and next predicted actions. We evaluated the performance of this proposed framework for learning the shaking hands skill on a humanoid NAO robot. The experimental results manifest that the skill learning method designed in this paper is promising.

1. Introduction

Skill learning autonomously through interactions with the environment is a crucial ability for intelligent robot, and it improves the flexibility and adaptiveness of robots [1]. Imitation learning is a primary method for implementation the skill learning [2]. A perception-action integration or sensorimotor cycle, as an important issue in imitation learning, refers to information flow from the environment to a sensory and motor structure and back to the environment and sensory inputs. It is the processing of sequential sensor information and their transduction to the successive goal-directed behaviors [3]. Skill learning of intelligent robot actually is a paradigm of learning the links between the perceptual environment inputs and the feedback action system. Perception-action integration is a natural mechanism for skill learning without the complex program process. Several research results from cognitive science provide convincing evidence for this statement. Mental simulation [4] stemming from neuropsychology is treated as part of the reactive and

perceptual components because it implements interaction between the action and sensed state of the environment. Perceptual-motor theory [5] states that a neurocomputational framework is used to connect with up-to-date perceptual data on the possible functional role of the motor system. Wolpert et al. [6] reviewed on the computational mechanisms of sensorimotor learning. Also, the computational models of perception-action integration are the dominant technique of skill learning in robotics research field.

1.1. Related Work. Traditional computational model of artificial intelligence methods such as Bayesian modeling [7] and reinforcement learning [8] were first concerned for learning skill. Recently, neurocomputing model and developmental intelligence method are considered as a new trend for implementing the robot skill learning. Do et al. [9] proposed a bootstrapping method for learning the wipe skill. This method bootstrapped from the sensorimotor experience and learned the association between object properties and action parameters. PerAc neural network [10] is applied to learn

the dynamics of coupling of perception of one partner to the action of the other, and the learned association of perception and action is used for recognizing the postures. Neural motor activation [11] that mimics the neuron activation process assigns a weight to each motor component to reveal its degree of activation. The weights are updated by the perception procedure. Neural activation is induced by perception processes with system feedback; therefore, it realizes integration between perception and action. A neural network architecture combining a recurrent neural network with parametric biases (RNNPB) and a horizontal product model was used to predict future percepts and behaviors of a sensorimotor system according to the connections between the development of the ventral/dorsal visual streams and the emergence of conceptualization in the visual streams [12].

Furthermore, some complex cognitive architectures used to simulate the brain pathways of perception-action cycles have been studied. Cutsuridis and Taylor examined several neurocomputational mechanisms of visuomotor brain processes and coordinately integrated them to establish a neural framework for the visual grasping tasks [13]. A cognitive model [14] based on Skinner operant conditioning principle is designed for a robot to master the balancing skill. This model consists of cerebellum, basal ganglia, and cerebral cortex. Each component imitates the basic functions of corresponding parts in human brain. In particular, cerebellum maps sensorimotor states to actions with supervised learning and basal ganglia provide the proper action based on the operant conditioning theory.

1.2. Why Hierarchical Temporal Memory (HTM). As futurist Kurzweil described in his book [15], the neocortex contains a hierarchy of pattern recognition circuits and they are responsible for most aspects of human thought. He also explains that if there exists a design of the digital neocortex, it could be used to create the same capabilities as the human brains. Hierarchical temporal memory (HTM) theory [16], first proposed by Hawkins [17], is an implementation version of Kurzweil’s view of digital neocortex. It attempts to model the brain at a functional level rather than at a neuron or molecular level. HTM is a bioinspired model that captures the predominant characteristics of the neocortex. It mimics the neocortex’s abilities of learning, inference, and prediction from sequential input patterns that are represented in sparse distributed forms, and therefore, it can describe a complex model of the world. Additionally, HTM uses the Sparse Distributed Representations (SDRs) to represent the complex input data and lend the HTM so much flexibility, which is similar to the idea that the brain is a recursive probabilistic fractal whose line of code is represented within the 30–100 million bytes of compressed code in the genome [15].

The cells (neurons) in HTM participate in the sensorimotor integration and learning process, which is supported by biological evidence [18]. In addition, the Cortical Learning Algorithm (CLA) of HTM consists of spatial pooling and temporal memory processes. These are the important components for perception-action integration, which is proved by Lalanne and Lorenceau [19]. Also spatial and temporal capabilities facilitate the acquisition of sensory-motor mappings

with less amount of training data and facilitates the robust behavior [20]. In that research work, they stated that the brain utilizes spatial and temporal coincidence from spatial information when spatial features gathered through different modalities are interconnected.

The core of Kurzweil’s book is the pattern recognition theory of mind. Its main idea is that the hierarchical structure is treated as pattern recognizer and is not just for sensing the world, but for nearly all aspects of thought. It is natural that HTM was first successfully applied for pattern recognition system [21–23].

The reasons and relationships between HTM and neuroscience stated above indicate that HTM can be considered as a promising approach for the implementation of perception-action integration. Therefore, in this study, we applied HTM to design a perception-action integration framework for skill learning. This framework receives the sequential sensor data representing a certain skill from a RGB-D camera. These sensory data are then encoded as a sequence of Sparse Distributed Representation (SDR) vectors. The sequential SDR vectors are treated as the inputs of the perception-action HTM. The HTM learns sequences of SDRs and makes predictions of what the next input SDR will be. It stores the transitions of the current perceived sensor data and next predicted actions. We evaluated the performance of this proposed framework for learning the shaking hands skill on a humanoid NAO robot.

2. HTM Preliminary

The purpose of this study is to implement the skill learning by using HTM based perception-action integration framework, and the process of design follows the general HTM workflow illustrated in Figure 1. The network learns from the time-varying inputs. In this application, the inputs are the captured skeleton joints and depth data. These inputs are encoded by an encoder [16] as a sparse binary string or matrix, which is the necessary input form for an HTM system. In our case, sequential joints data for shaking hands accompanied with depth data are recorded and encoded as a 1,024-bit binary string by an encoder. The format of this string is as

$$\begin{array}{ccccccc} \overbrace{\times \dots \times}^{32 \text{ bits}} & \overbrace{\times \dots \times}^{32 \text{ bits}} & \dots & \overbrace{\times \dots \times}^{32 \text{ bits}} & \overbrace{\times \dots \times}^{32 \text{ bits}} & \overbrace{0 \dots 0} & \\ \text{joint 1} & \text{joint 2} & & \text{joint } n & \text{depth} & \text{reserved} & \end{array} \quad (1)$$

The HTM system learns continuously with the input data. The learning algorithms CLA are designed to work with sensor and motor data that is constantly changing. With each change in the inputs, the memory of the HTM system is updated. The HTM uses the CLA dynamic process to learn the spatial and temporal variability commonly occurring in sequential input data and then to make predictions. The typical CLA is composed of two subprocesses: spatial pooling (SP) and temporal memory (TM) algorithms.

Inputs coming from the senses or other parts of the HTM are messy and irregular. The most fundamental function of the SP algorithm is to convert a region’s input into an SDR via overlapping computation, inhibition, and update processes while retaining semantically encoded information. Each SDR

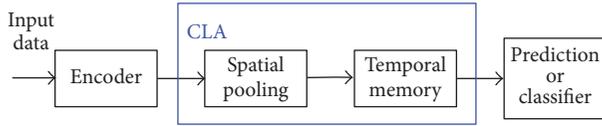


FIGURE 1: Workflow of an HTM application.

has semantic attributes of what is being represented. By determining the overlap between any two SDRs we can immediately see how two representations are semantically similar and how they are semantically different. Because of this semantic overlap property of SDRs, SP associatively connects the input to the HTM cells in a way that they will be able to learn once patterns in the space start to change. TM algorithm is a memory of transitions in a data stream. It learns sequences of SDRs formed by the SP algorithm and makes predictions of what the next input SDR will be. TM is used in both sensory inference and motor generation. It forms a representation of the sparse input that captures the temporal context of previous inputs and then forms a prediction based on the current input in the context of previous inputs. HTM theory postulates that every excitatory neuron in the neocortex is learning transitions of patterns and that the majority of synapses on every neuron are dedicated to learning these transitions.

As a memory system, HTM is essentially a type of neural network. It models the cells, connects and arranges cells in columns, organizes columns in a 2D array to constitute the HTM region, and finally establishes a hierarchical neural network, which is shown in Figure 2. The detailed explanation, properties, and learning pseudocode of HTM can be found in technique reports [16]. In this section, we only describe the crucial contents related to our application.

2.1. Cells. HTM cells extract the most important capabilities of biological neurons, and as shown in Figure 3, they have more complex structures than conventional artificial neurons. A typical HTM cell has three output states: the active state activated from feed-forward input, the predictive state activated from lateral input, and the inactive state. Each HTM cell in one column shares a single proximal dendrite segment (closest to the cell body) and has a list of distal dendrite segments (farther from the cell body). The proximal dendrite segment receives all feed-forward inputs, including the environmental sensory data and outputs of the lower-level region, via active synapses marked by green dots. These active synapses have a linear additive effect at the cell body. Distal dendrite segments receive the lateral inputs from nearby cells through active synapses marked by blue dots. Figure 3 shows that each distal dendrite segment is a threshold detector. The segment will be activated if the number of active synapses on a segment is above a threshold Th_{seg} . An OR operation is executed on all active distal dendrite segments to make the associated cell become the predictive state. Synapses of the HTM cells have binary weights and are formed by a set of potential synapses. The potential synapses are axons that are sufficiently close to a dendrite segment and may become

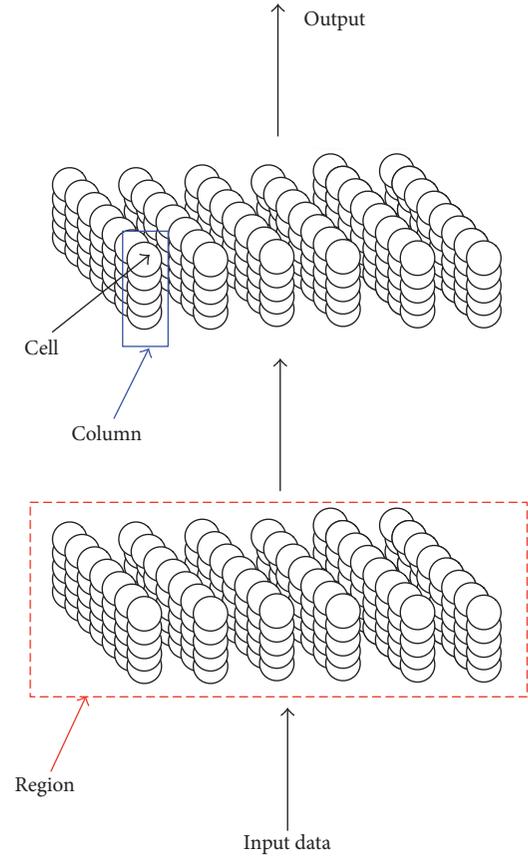


FIGURE 2: The structure of a typical HTM network.

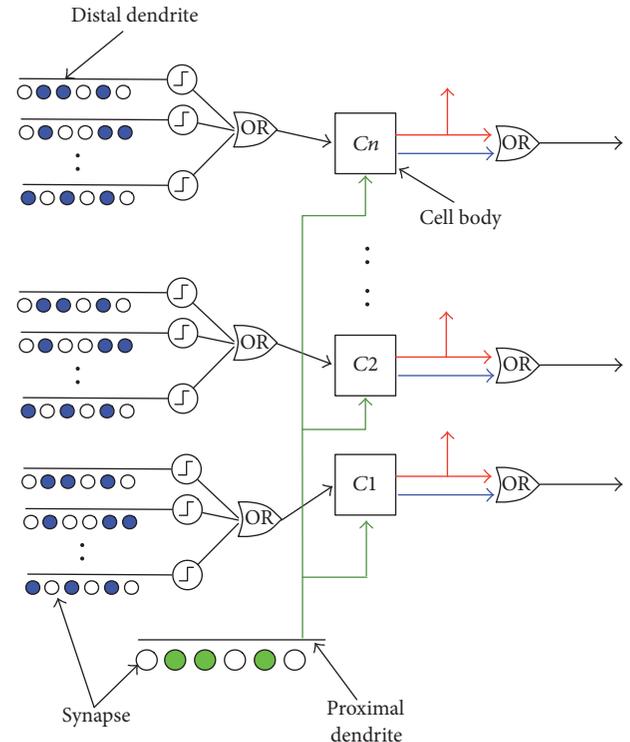


FIGURE 3: The structure of an HTM cell.

synapses. For the proximal dendrite, a potential synapse consists of a subset of all inputs to a region, and for the distal dendrite, the potential synapses are predominantly from the nearby cells in a region. Each potential synapse is assigned a scalar value ranging from 0 to 1. This scalar value is named as permanence, which represents a closeness or connection degree between an axon and dendrite segment. A larger permanence yields a stronger connection. If the permanence is above a threshold Th_{per} , the potential synapse becomes a valid synapse, and the weight of this valid synapse is set as 1. The cell body receives the inputs of synapses from proximal and distal segments and provides two outputs along the axon: one is in an active state (red lines), which is horizontally sent to other adjacent cells, and the other is the OR results of the active and predictive states (blue lines) and sent to the cells of the next region. Because the perception and action are integrated in the HTM network, distal dendritic input can also be the external input. That is, lateral connections between cells will typically be turned off in sensorimotor inference.

2.2. Spatial Pooling (SP). The essential function of spatial pooling is to form an SDR of the inputs. When an input appears on a region, each bit in the input signal will be assigned only to a fixed number of columns. Each column has an associated proximal dendritic segment (shared by all cells of a column, cf. Figure 3), serving as the connection to the input space. Each proximal dendrite segment has a set of potential synapses representing a subset of the input bits. Each potential synapse has a permanence value. These values are randomly initialized around the permanence threshold. Based on their permanence values, when the permanences are greater than the threshold value Th_{syn_per} some of the potential synapses will already be connected.

For any given input, determine how many connected synapses on each column are connected to active input bits (bit 1). The connected synapses become active synapses. The number of active synapses is multiplied by a boost factor bf , which is dynamically determined by how often a column is active relative to its neighbors. The columns with the highest activation after boosting disable a fixed percentage of the columns within an inhibition radius. The result of the inhibition is to form a sparse set of active columns that are treated as the inputs of the TM subprocess in the same region. A Hebbian-like learning procedure is implemented for each of the active columns. Permanence values of synapses aligned with active input bits are increased, and those aligned with inactive input bits are decreased. The changes of permanence values make some synapses become valid or invalid accordingly. Simultaneously, the boost factor and inhibition radius are both updated according to

$$\begin{aligned} bf_c &= f_{bf}(ADC_{avg}^c, ADC_{min}^c) \\ r_{inh} &= \frac{CS_{avg} \times PI_{col} - 1}{2}, \end{aligned} \quad (2)$$

where ADC_{avg} (active duty cycle) is a sliding average that represents how often column c has been active after inhibition, for example, over the last 500 iterations. ADC_{min}

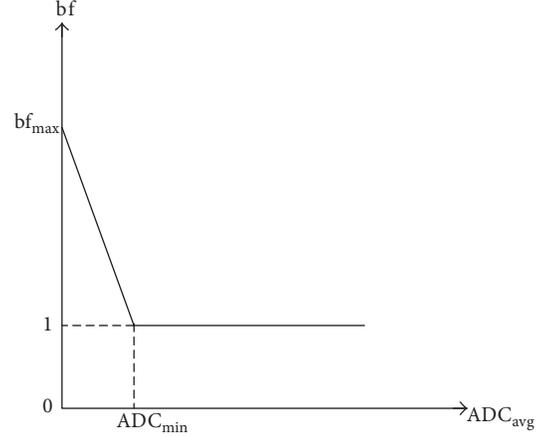


FIGURE 4: Function for updating the boost factor.

represents the minimum desired firing rate for column c . f_{bf} is the update function, which linearly interpolates the boost factor between the points $(0, bf_{max})$ and $(DC_{min}, 1)$, as shown in Figure 4. In general, the boost factors for all columns are updated simultaneously. For the inhibition radius updating, the number of inputs to which a column is connected (denoted by CS_{avg}) should first be determined, and then, this number is multiplied by the total number of columns that exist for each input (denoted by PI_{col}). For multiple dimensions, the aforementioned calculations are averaged over all dimensions of inputs and columns.

2.3. Temporal Memory (TM). TM is more complex than SP because it combines the learning and prediction procedures. It learns SDRs formed by the SP algorithm and makes predictions. TM consists of three phases.

2.3.1. Phase 1: Forming a Representation of the Input in the Context of Previous Inputs (Determining the Active State of Cells). After spatial pooling, the TM algorithm converts the columnar representation of the input into a new representation that includes state, or context, from the past. The new representation is formed by activating a subset of the cells within each column, typically only one cell per column.

For each active column obtained in SP, the cells that are fired to a predictive state from a previous time are activated (referring to (3)). Simultaneously, the distal dendrite segment on each of these cells is marked as active when the number of synapses is over a threshold Th_{act} . The learning cells are chosen by (6). Additionally, if a segment is activated from the learning cells during the previous time, the cell to which this segment connects is set as the learning cell (see (4)). If no cell is in a predictive state, all of the cells in the column are activated, which is defined in (5). For this case, the segment that has the largest number of active synapses is found in column c cell i at time $t - 1$, and then, the related cell to which this segment connects is chosen as the learning cell. If no cell has such a segment, we select the cell that has the fewest number of segments as the learning cell (see (6)). In

Phase 1, the resulting set of active cells consists of the current input in the context of prior inputs.

For the perception-action integration case, there is an optional “Learn-On-One-Cell (LOOC) (Available at <https://github.com/numenta/htmresearch/wiki/Sensorimotor-Inference-Algorithm>)” hysteresis mode. This mode is switched in the following situation. When a column is not predicted but activated by the sensory input, cells that were previously selected as the learning cell would still act as the learning cell at the current time. If no such cell exists, the learning cell is also determined by (6). If the LOOC mode is triggered, a copy of the motor signal is added to the input of the distal dendrites.

$$\begin{aligned} na_i^c(t) &= 1, \\ c \in C_{act}(t) \end{aligned}$$

$$\begin{aligned} nl_i^c(t) &= 0, \\ c \in C_{act}(t) \end{aligned} \quad (3)$$

$$\text{if } np_i^c(t-1) = 1, \quad sga_i^c(t-1) = 1$$

$$\begin{aligned} na_i^c(t) &= 1, \\ c \in C_{act}(t) \end{aligned}$$

$$\begin{aligned} nl_i^c(t) &= 1, \\ c \in C_{act}(t) \end{aligned} \quad (4)$$

$$\text{if } np_i^c(t-1) = 1, \quad sga_i^c(t-1) = 1, \quad sgl_i^c(t-1) = 1$$

$$\begin{aligned} na_i^c(t) &= 1, \quad i = 1, \dots, n_c \\ c \in C_{act}(t) \end{aligned} \quad (5)$$

$$\begin{aligned} nl_i^c(t) &= 1, \\ c \in C_{act}(t) \end{aligned} \quad (6)$$

Equation (6) is subject to the condition that “(cell i has the segment with the largest number of active synapses at time $t-1$) || (cell i with the fewest number of segments if at time $t-1$).” $na_i^c(t)$ represents the active state of cell i in column c at time t given the current feed-forward input and previous temporal context; $nl_i^c(t)$ and $np_i^c(t-1)$ are the learning and predictive state of cell i in column c at time t and $t-1$, respectively; and $sga_i^c(t-1)$ represents the active segment on cell i in column c at time $t-1$. Similarly, $sgl_i^c(t-1)$ is the segment activated by the learning cell at time $t-1$. If multiple segments are active, sequence segments are given preference. n_c is the number of cells in column c . $C_{act}(t)$ is the set of the active column index at time t .

2.3.2. Phase 2: Forming a Prediction Based on the Input in the Context of Prior Inputs. Following *Phase 1*, according to (7), the cells with active segments are admitted to the predictive state unless they are already active due to feed-forward input. $np_i^c(t)$ represents the predictive state of cell i in column c at time t . All of the predictive cells form the prediction of the region.

$$\begin{aligned} np_i^c(t) &= 1, \quad \text{if } sga_i^c(t) = 1. \\ c \in C_{act}(t) \quad c \in C_{act}(t) \end{aligned} \quad (7)$$

On column c cell i , the current active segment is added to the update list $SU_i^c(t)$, which will be used in *Phase 3*. To extend

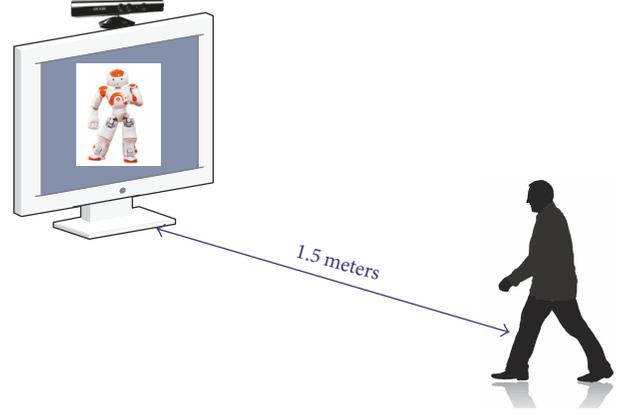


FIGURE 5: Settings for performance examination.

the prediction back in time, another distal dendrite segment that has the largest number of active synapses at the previous time is also considered to add to the update list.

2.3.3. Phase 3: Updating Synapses. Similar to the synapse updates of the proximal dendrite in the SP algorithm, whenever a distal dendrite segment becomes active, the permanence values of its associated potential synapses are modified by the Hebbian rule only if the cell correctly predicted the feed-forward input. Thus, the synapse permanence values for the segments in update list $SU_i^c(t)$ will be reinforced positively or negatively.

Finally, a vector representing the OR results of the active and predictive states of all cells in a region becomes the input to the next region in the hierarchy. Rather than storing a set of predicted cells, TM algorithm stores a set of active distal dendritic segments, that is, the segments related to predicted skeleton positions for the shaking hands. With the prediction, the HTM network can estimate approximately when the inputs will likely arrive next and invoke and separate the motor information.

3. Results

3.1. Experimental Setup. We applied the HTM based perception-action integration for learning the shaking hands skill on a NAO robot. Since there is no practical NAO robot in our lab, the Webots NAO simulator combined with the Kinect V1 RGB-D cameras is considered as the experimental configuration for examining the skill learning performance. As is shown in Figure 5, the RGB-D camera installed on the top of LCD is for simulating the camera and sonar sensors of the real NAO robot. The RGB-D camera captured sequential human’s motion skeletons and depth data between the NAO simulator and a human. It should be noted in Figure 5 that the distance between camera and the object has to be configured within the effective detection range of RGB-D camera, that is, 2 meters for Kinect V1. Here, we set 1.5 meters.

To learn the skill for NAO robot, we need the perception and action data. Therefore, we collected the training data

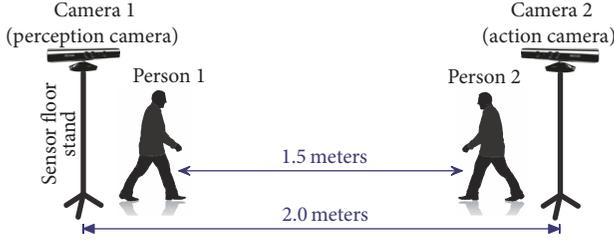


FIGURE 6: Settings for training data collection.

from two persons. These training data are used to swarm the best HTM model parameters. The setting for training data collection is shown in Figure 6. One person stood 1.5 meters far from another one, and two RGB-D cameras were installed on top of their heads. The distance between two sensor floor stands is 2 meters. Camera 1, named as perception camera, is used to acquire the skeleton data of person 2 and the depth from person 2. The purpose of camera 2 (i.e., action camera) is the same as camera 1 except that the data are from person 1. Note that the depth data from the separate cameras have to be converted to the distance between two persons. Combining the skeleton data from two cameras and the converted distance, we build up the training dataset. Two groups of training data were recorded, and each group consists of ten sets of shaking hands skeleton data and depth data. Group one is for the case that one person (person 2) walked towards and stopped 0.5 meters far from another one (person 1) and then shook hands. Group two is for the case that two persons walked towards and stopped 0.5 meters far from each other and then began to shake hands. Camera 1 captured skeleton data of person 2 which will be treated as the perception data for NAO simulator; camera 2 recorded skeleton data of person 1 which is to be taken as the action data for NAO simulator. Because these perception-action skeleton motion data are from different cameras, it is necessary to consider the synchronization issue of data acquisition time. In this paper, we applied asynchronous mechanism to address this problem. The perception camera acquisition thread first started

and then triggered the action camera acquisition thread. These two threads are alternate. Furthermore, when two data acquisition threads started, persons stood statically 5 seconds to maintain the skeleton data stable before recording. This asynchronous manner imitates the perception-action cycle. When the HTM network is trained, we used the experimental setup in Figure 5 to examine the performance of skill learning in online form. The RGB-D camera captured the first frame of person's skeleton and measured the distance between the simulator and the person. These perception data were sent to HTM network, and HTM network provided the predicted skeleton actions. The predicted skeleton actions are converted from the skeleton coordinates system of RGB-D camera to the joint coordinate framework of NAO simulator and then executed on the joints. This is a perception (sensor data acquisition) action (prediction) cycle in online evaluation. This cycle is performed frame by frame until the hand shaking is completed.

The data structure of the recording file is as (8). The data are ordered following the time stamp. "ID" depicts the RGB-D camera ID which captured the skeleton data; depth data are always acquired only by camera 1 (ID = 1) and for the depth bits of camera 2 we copied the depth data of camera 1 directly. It can be found that the perception and action data are recorded alternately. Each joint data consist of 3D Cartesian coordinates and 20 joints include 60 coordinate values (referring to https://msdn.microsoft.com/en-us/library/nui-skeleton.nui_skeleton_position_index.aspx). Practically, for NAO simulator, there are only 12 joints that can be controlled and each joint comprises several Euler angle values (referring to Table 1). It is necessary to address this issue before the swarming process. We selected the corresponding skeleton joints and converted their 3D Cartesian coordinates to Euler angles and then reorganized the converted data following the format of (9) for the optimal parameters swarming. Therefore, 12 joints of NAO simulator cover 20 Euler angle values. Since the size of image is 640×480 , to make the computation efficient, the depth information within a region of interest (ROI) was extracted. The ROIs were selected as a 64×48 rectangle around the image center. The sampling time is set as 100 milliseconds.

$$\begin{array}{ccccccc}
 \text{TimeStamp} & \text{ID} & \text{Joint}_1 & \text{Joint}_2 & \cdots & \text{Joint}_{20} & \text{depth} \\
 15 : 26 : 46 : 151 & 1 & (x, y, z)_1 & (x, y, z)_2 & \cdots & (x, y, z)_{20} & d_1 \\
 15 : 26 : 46 : 198 & 2 & (x, y, z)_1 & (x, y, z)_2 & \cdots & (x, y, z)_{20} & d_1 \\
 \vdots & \vdots & \vdots & \vdots & & \vdots & \vdots
 \end{array} \tag{8}$$

$$\begin{array}{ccccccc}
 \text{TimeStamp} & \text{ID} & \text{Joint}_1 & \text{Joint}_2 & \cdots & \text{Joint}_{12} & \text{depth} \\
 15 : 26 : 46 : 151 & 1 & \text{Euler}_1 & \text{Euler}_2 & \cdots & \text{Euler}_{12} & d_1 \\
 15 : 26 : 46 : 198 & 2 & \text{Euler}_1 & \text{Euler}_2 & \cdots & \text{Euler}_{12} & d_1 \\
 \vdots & \vdots & \vdots & \vdots & & \vdots & \vdots
 \end{array} \tag{9}$$

The HTM was designed based on the open source NuPIC (available at <https://github.com/numenta/nupic>), and

its settings were identical for both of the two cases above. The HTM model is one-region network. The size of the columns

TABLE 1: The Euler angles of NAO simulator joints.

Joint	Euler angle
Left shoulder	Pitch & roll
Left elbow	Yaw & roll
Left wrist	Yaw
Right shoulder	Pitch & roll
Right elbow	Yaw & roll
Right wrist	Yaw
Left hip	Pitch & roll
Left knee	Pitch
Left ankle	Pitch & roll
Right hip	Pitch & roll
Right knee	Pitch
Right ankle	Pitch & roll

in this region is set to 2,048 (arranged as 64×32 in a 2D plane), and the number of cells in each column was set to 32. This configuration maintains the diversity of SDRs and a low probability of a false match between any two SDRs. As is shown in (1), the converted skeleton data are encoded as a binary string by a scalar encoder and each joint data occupied 32 bit. The depth data are also encoded as a 32-bit binary string by a category encoder. Here we defined two categories: Close and Far for depth data. “Close” means that persons are close enough to begin to shake hands, and “Far” means that persons keep walking. The encoding mechanism is determined by the minimal distance extracted within the ROI. If the minimal distance is less than a threshold, that is, 50 cm in our experiments, the category is “Close” and vice versa. The reserved bits are designed for the additional sensor information in the future work.

3.2. Results Analysis. We chose the first five sets of training data in each group to swarm the optimal HTM network parameters. The final swarming results of main parameters for SP and TM algorithms described in the previous section are listed in Table 2. With these optimal parameters and rest of training data, we examined the skill learning performance by offline and online form, respectively. Offline validation is a paradigm of batch testing; that is, the skeleton and depth data collected by camera 1 (perception camera) at sampling time t_i ($i = 1, \dots, n - 1$) were first encoded as the sequential binary strings and then sent to the HTM network to obtain a batch of one-step ahead predicted action skeleton data for sampling time t_j ($j = 2, \dots, n$). We transferred the predicted skeleton to the joint coordinate system of NAO simulator and NAO simulator retrieved the shaking hands in batch form. The predictions are compared with the original skeleton data recorded by camera 2 (action camera) at t_j ($j = 2, \dots, n$), and the compared joints trajectories and statistical results are shown in Figure 7 and Table 3, respectively. Since in Case 1 NAO simulator stood statically and only shook right hand, the joints data of right arm are recorded only in Figure 7(a), where the robot completed the task within

20th to 100th sampling time. Figures 7(b)–7(e) illustrate all joints trajectories, where robot shakes hand within 100th to 180th sampling time and it walks 0.5 meters from 0 to 100th sampling time. It can also be found in Figure 7 that the predicted skeletons are consistent with the practical skeleton actions captured by camera 2. Table 3 shows the statistics for action predictions compared with the training data. It can be seen that the mean and variance for each prediction are close to zero, which manifests that the actions are predicted correctly and successfully. Figure 8 shows grabbed frames (the complete video clip can be found in attached media 1), where the left columns are for Case 1 and the right column is for Case 2. These offline examination results demonstrate that our proposed perception-action integration provides the correct action predictions according to the different perceived input data.

In online evaluation, a person stood in front of the RGB-D camera (refer to the camera in Figure 5). When the camera captures the person’s skeleton, the HTM network predicts the corresponding actions and then the actions are transferred to the joint positions for NAO simulator so that the NAO simulator can shake hands with the person interactively. Figure 9 displays the shaking hands interaction between NAO simulator and the person (the complete video clip can be found in attached media 2), where grabbed frames in the left column are for Case 1 and the right column is for Case 2. The joints trajectories of NAO simulator are shown in Figure 10. In comparison with the training data curves in Figure 7, it can be seen that the shapes of predicted skeleton data curves are similar to those of training data, which manifests that our proposed approach can also be used for online skill learning. In comparison with the trajectories in offline examination, it should be noted that in online evaluation the NAO simulator has a default initial motion of which the related trajectories are the data sampled from 0 to 50th time in Figures 10(a)–10(e). We do not consider these parts of joint trajectory in our proposed skill learning framework and just simulated the initial actions of the real NAO robot. Additionally, the learned action skeleton data in the training process are remembered in the HTM, and they are treated as the reference for the predicted actions. If the prediction is abnormal, these stored actions can be used for anomaly detection, which is discussed in Section 4.

The computational platform is a Corel i7-6500U 2.50 GHz, with a 12 G RAM laptop. The time for swarming optimal parameters of HTM network is around 60 minutes (the number of training data lines is around 1000). The online evaluation process, which consists of loading optimal parameters of HTM network, grabbing a frame of skeleton and depth data, encoding these perception data, implementing SP and TM algorithms, and output predictions, consumes 2.35 seconds. The cost time of online validation is considerably less than that of the training because the training is an optimal searching processing which is usually time-consuming. Additionally, only one frame of RGB-D data has to be processed; hence, the computational time is reduced considerably. Considering the results in terms of time cost, it is reasonable to use the proposed perception-action integration for real-time skill learning tasks.

TABLE 2: Main optimal parameters for SP and TM algorithms.

Parameters	Description	Value
Th_{seg}	Threshold for the number of active synapses on a segment	15
Th_{per}	Threshold for the permanence of potential synapse	0.2
bf_{ini}	Initial value of the boost factor	1.0
bf_{max}	Maximal boost factor	2.0
$r_{inh,ini}$	Initial value of the inhibition radius	0
LA_{min}	Minimal number of winning columns	1
$pm_{syn,inc}$	Incremented permanence value in spatial pooling	0.05
$pm_{syn,dec}$	Decremental permanence value in spatial pooling	0.05
$Th_{syn,per}$	Any synapse whose permanence value is above this threshold will become an active synapse	0.1
ADC_{min}	Minimum active duty cycle	0.001
Th_{act}	Threshold used to determine whether a distal segment is activated	14
pm_{inc}	Incremented permanence value in temporal pooling	0.1
pm_{dec}	Decremental permanence value in temporal pooling	0.1

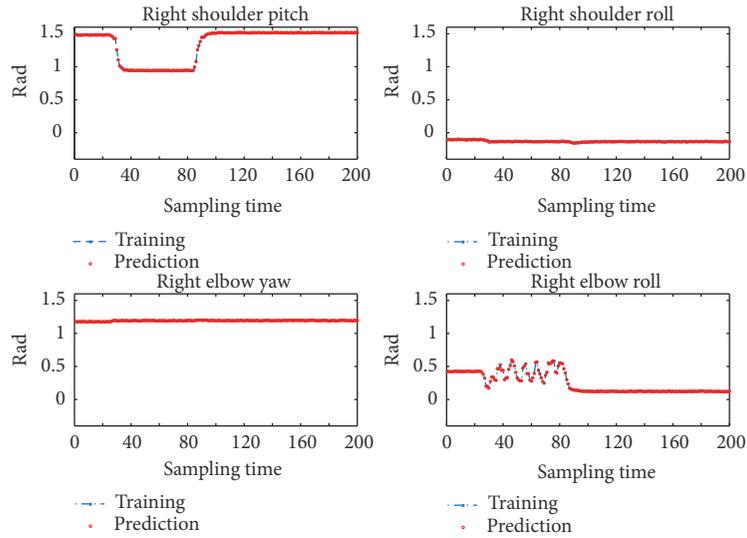
TABLE 3: Statistics for offline evaluation.

Case	LShoulder Pitch		LShoulder Roll		LElbow Yaw		LElbow Roll		LWrist Yaw	
	μ	σ^2	μ	σ^2	μ	σ^2	μ	σ^2	μ	σ^2
1	—	—	—	—	—	—	—	—	—	—
2	-0.0051	$0.819e-5$	-0.0015	$0.068e-5$	-0.0048	$0.786e-5$	-0.0049	$0.814e-5$	-0.0015	$0.073e-5$
Case	LHip Roll		LHip Pitch		LKnee Pitch		LAnkle Pitch		LAnkle Roll	
	μ	σ^2	μ	σ^2	μ	σ^2	μ	σ^2	μ	σ^2
1	—	—	—	—	—	—	—	—	—	—
2	-0.0016	$0.078e-5$	-0.0052	$0.81e-5$	-0.0050	$0.827e-5$	-0.0051	$0.814e-5$	-0.0014	$0.075e-5$
Case	RShoulder Pitch		RShoulder Roll		RElbow Yaw		RElbow Roll		RWrist Yaw	
	μ	σ^2	μ	σ^2	μ	σ^2	μ	σ^2	μ	σ^2
1	-0.0051	$0.822e-5$	-0.0048	$0.814e-5$	-0.0050	$0.775e-5$	-0.0048	$0.906e-5$	—	—
2	-0.0048	$0.862e-5$	-0.0015	$0.076e-5$	-0.0050	$0.86e-5$	-0.0051	$0.824e-5$	-0.0015	$0.081e-5$
Case	RHip Roll		RHip Pitch		RKnee Pitch		Rankle Pitch		Rankle Roll	
	μ	σ^2	μ	σ^2	μ	σ^2	μ	σ^2	μ	σ^2
1	—	—	—	—	—	—	—	—	—	—
2	-0.0015	$0.072e-5$	-0.0051	$0.797e-5$	-0.0049	$0.881e-5$	-0.0049	$0.893e-5$	-0.0016	$0.077e-5$

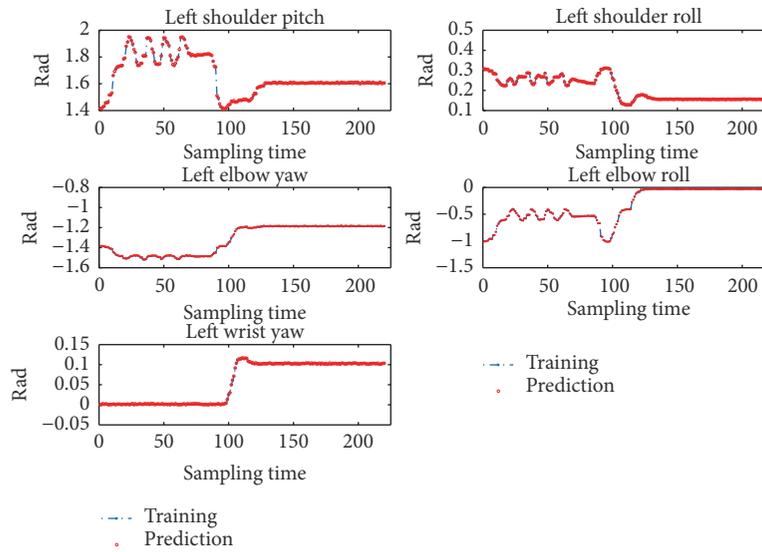
4. Discussion

4.1. Anomaly Detection. There is an important issue to be considered in the online evaluation. If the predicted actions deviate from those expected, the robot likely fails in the tasks of shaking hands. This situation is referred to in the terms of NuPIC as an *anomaly*. It is valuable to detect anomalies in real-time for many applications. CLA takes the *anomaly likelihood* computed from an anomaly score, a powerful anomaly detection analysis approach, to address this problem [24]. The anomaly likelihood enables the CLA to provide a metric representing the degree to which each record of the input sequence is predictable. It is relative to the data stream rather than an absolute measurement of abnormal behavior and is thus a critical reference to detect whether the pattern with a high anomaly score is actually

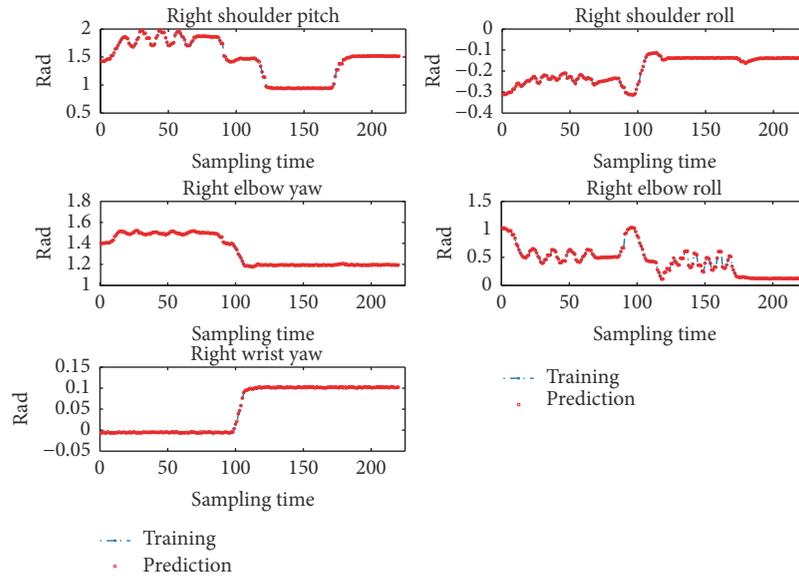
anomalous. Anomaly likelihood creates an average of the error score and then compares the current average error to a distribution of what the average error has been over the past data stream. This allows us to identify anomalies based on probability. As shown in Figure 11, if the anomaly likelihood is in the green section, this suggests that the record is normal. If it is in the red section, the record shows an abnormal value, which indicates that the pattern is a novel one not seen in any sequence. The yellow section indicates the pattern is somewhat unusual and that we do not have high confidence. In our application, we consider a pattern anomalous if its likelihood is in the yellow section. Based on the concept of anomaly detection, we calculated the anomaly likelihood for each predicted action in the shaking hands learning task. If the anomaly likelihood of any action is above a predefined probability threshold $P_{Th,ano}$ (0.90 in



(a) Joints trajectories in Case 1

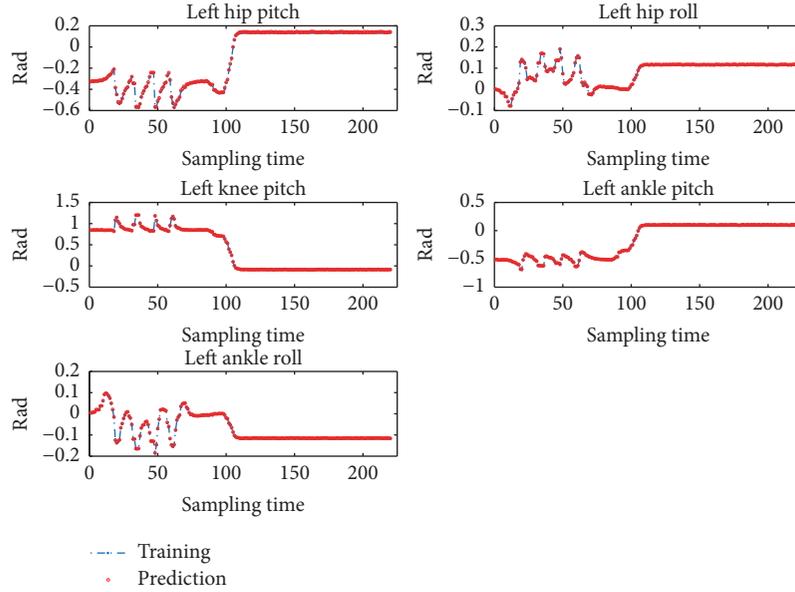


(b) Joints trajectories in Case 2

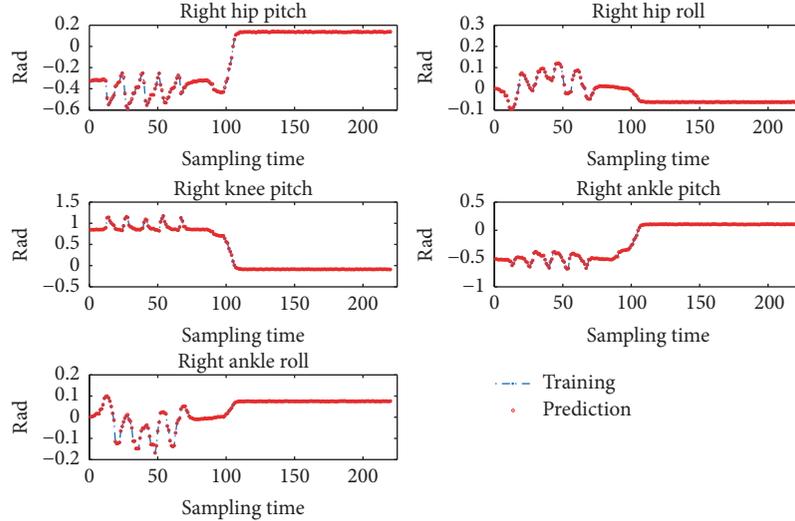


(c) Joints trajectories in Case 2 (cont.)

FIGURE 7: Continued.



(d) Joints trajectories in Case 2 (cont.)



(e) Joints trajectories in Case 2 (cont.)

FIGURE 7: Joints trajectories compared with the training data in offline evaluation.

our experiment, that is, the probability or accuracy of the green section is 90%, which is equivalent to a 1.65σ tolerance interval for a normal distribution), we designed a simple action retrieval strategy, that is, recalling the stored cells' active distal dendritic segments corresponding to the action sequence of training data to replace the predicted action which has a higher anomaly likelihood. The retrieved action is treated as the prediction for the next time.

It can be found in Figure 8 that the person's hand skeleton data in the grabbed frame of Case 1 video clip are deviation. The anomaly likelihood of predicted action corresponding to this perceived hand skeleton is 0.954, which is over 0.90. We replaced this anomalous action with the stored action during the training and sent it back to HTM as the prediction for

the next time. With this replacement procedure, the following predicted actions were correctly maintained and the NAO simulator continued to shake hands correctly. Because the CLA prediction mechanism in our experiment is one step ahead, we only retrieved one predicted action. If a multistep ahead prediction mechanism is adopted, the number of action retrievals is determined by the number of prediction steps and anomaly likelihoods.

4.2. Biological Evidence for Action Prediction. Learning the incorporated actions from different persons is an important cognitive function in the perception-action integration system, which has been examined by Knoblich and Flach [25]. They also proved that this type of prediction becomes more



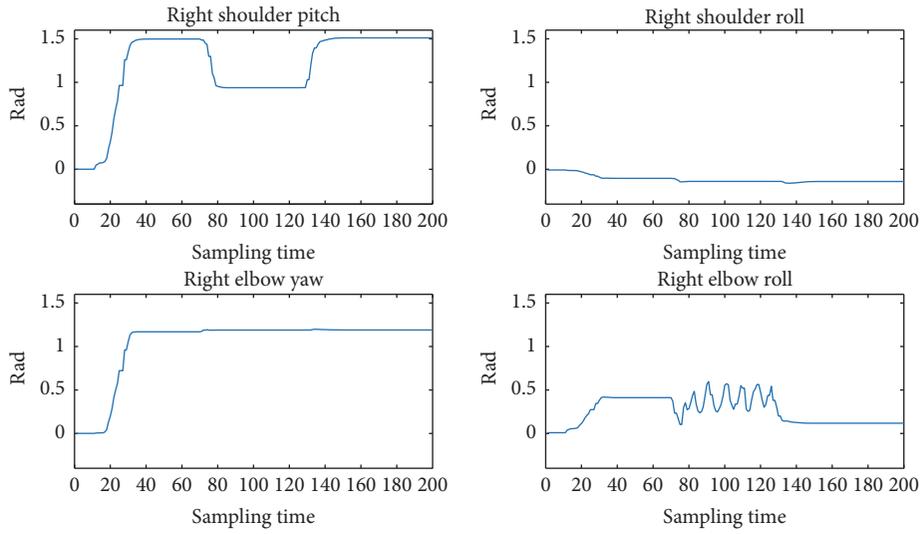
FIGURE 8: Grabbed frames of shaking hands learning in offline evaluation.



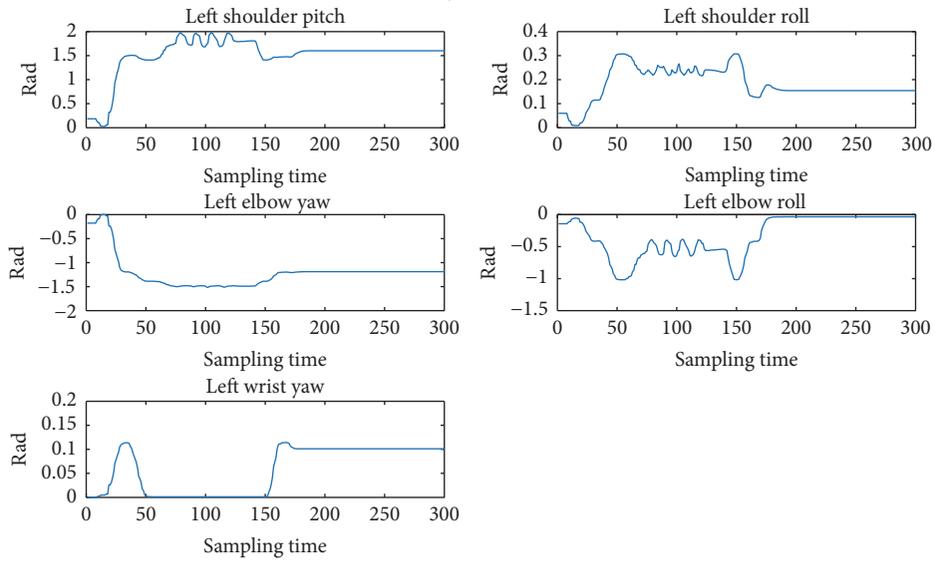
FIGURE 9: Grabbed frames of shaking hands learning in online evaluation.

accurate when one obtains the knowledge from one's own actions rather than those of others. Their research provides the biological evidence to support the action prediction mechanism of HTM and its application for skill learning tasks. However, the current HTM only implements a simple consequence prediction. It provides a sequence of predicted actions, including one-step or multistep predictions, but does not consider the potential information behind these

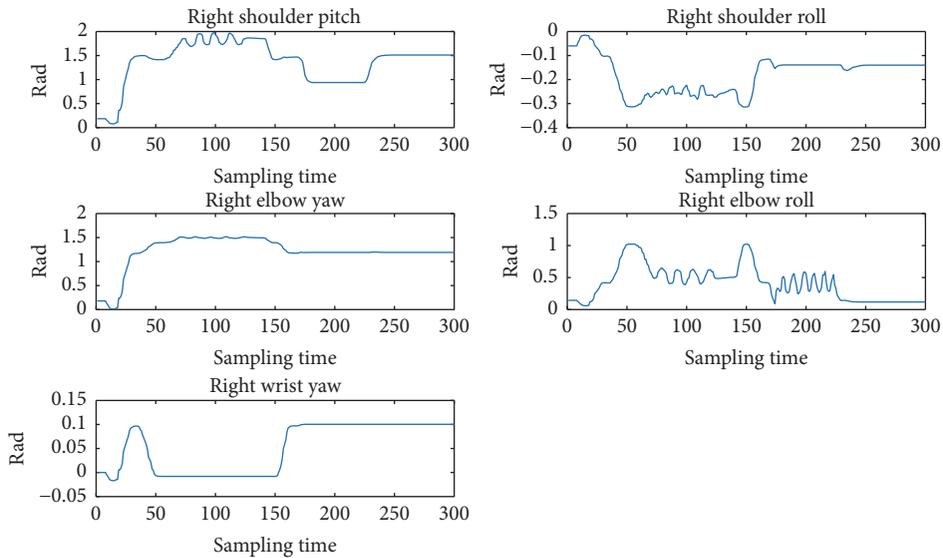
predictions. From a biological viewpoint, the present version of HTM does not link the perceptual input with the action system to predict the future outcome of actions [25]; that is, it does not explain the perception of intentionality for goal-related actions [26] or implement the understanding of the intention hidden in the sequential predicted actions [27] and how to learn to perceive something new [28]. Additionally, how the predicted actions guide the future perception process



(a) Joints trajectories in Case 1

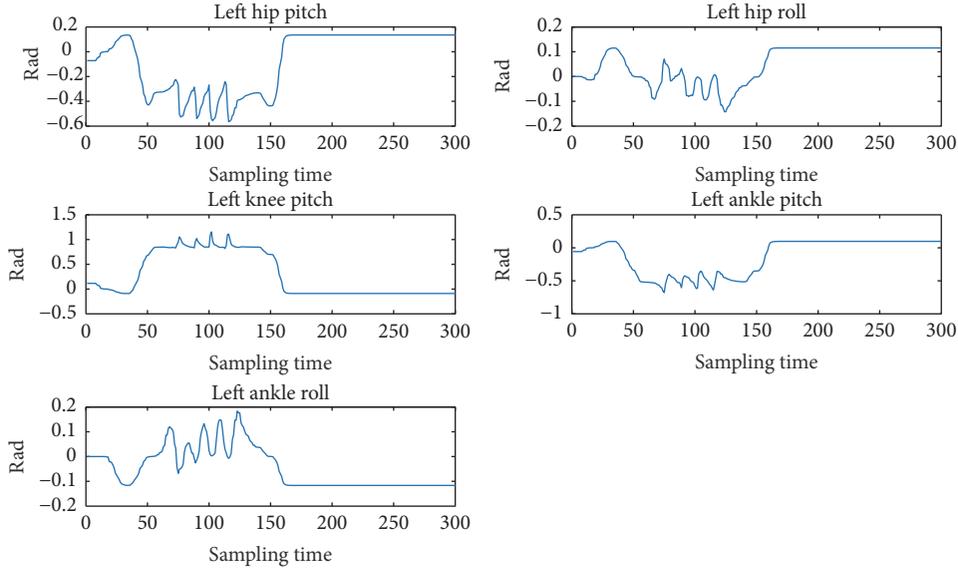


(b) Joints trajectories in Case 2

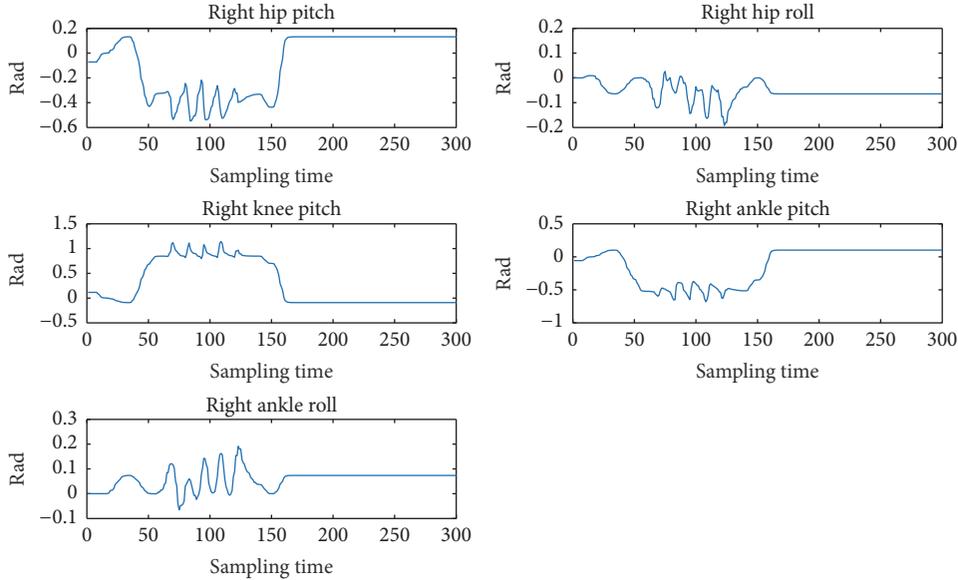


(c) Joints trajectories in Case 2 (cont.)

FIGURE 10: Continued.



(d) Joints trajectories in Case 2 (cont.)



(e) Joints trajectories in Case 2 (cont.)

FIGURE 10: Joints trajectories in online evaluation.

is not considered. Therefore, both of these two issues above will be the topics of our future work.

5. Conclusion

This study is the first attempt to explore the perception-action integration from the view of HTM for skill learning issue in intelligent robot. The main concept is that sequential perceptual information contributes to predicting one-step future actions. We selected the shaking hands as an example to evaluate the skill learning performance of our proposed framework on the NAO simulator. The perceived skeleton

of the target person and depth data from the target person are grabbed from a RGB-D camera. The perception data are first encoded as a sequential binary string. By using the SP algorithm, the binary string sequences are organized as a 2D SDRs. With the SDRs, TM algorithm makes predicted skeleton data for NAO simulator via storing the transitions between the current perceived skeleton data and predictions for the next future time. The prediction data are transformed to the joint coordinates framework so that the NAO simulator can implement the hands shaking actions with a real person. The experimental results manifest that the proposed method in this paper is promising for the skill learning of intelligent robots.

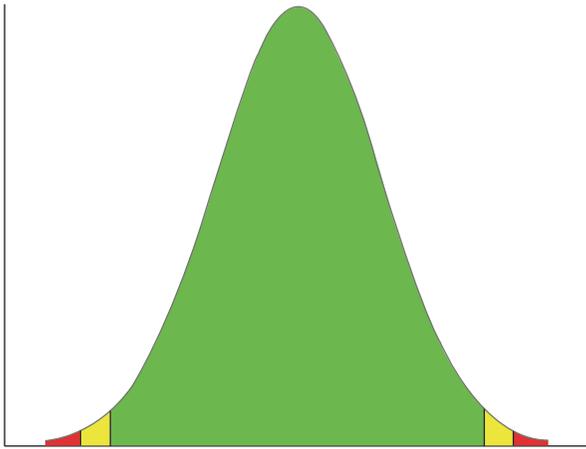


FIGURE 11: Anomaly likelihood curve.

Conflicts of Interest

The authors declared no potential conflicts of interest with respect to the research, authorship, and/or publication of this article.

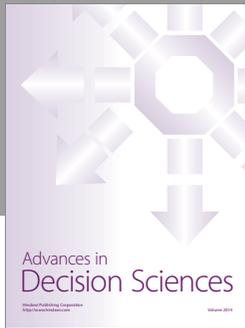
Acknowledgments

This work was supported by National Natural Science Foundation of China under Grant no. 61203338. The authors thank the NuPIC Open Source Project and all the contributors of the NuPIC codes.

References

- [1] O. Sigaud and J. Peters, *From Motor Learning to Interaction Learning in Robots*, Springer, Berlin, Germany, 2010.
- [2] S. Schaal, "Is imitation learning the route to humanoid robots?" *Trends in Cognitive Sciences*, vol. 3, no. 6, pp. 233–242, 1999.
- [3] J. M. Fuster, "Upper processing stages of the perception-action cycle," *Trends in Cognitive Sciences*, vol. 8, no. 4, pp. 143–145, 2004.
- [4] N. L. Cassimatis, J. G. Trafton, M. D. Bugajska, and A. C. Schultz, "Integrating cognition, perception and action through mental simulation in robots," *Robotics and Autonomous Systems*, vol. 49, no. 1-2, pp. 13–23, 2004.
- [5] J.-L. Schwartz, A. Basirat, L. Ménard, and M. Sato, "The Perception-for-Action-Control Theory (PACT): A perceptuo-motor theory of speech perception," *Journal of Neurolinguistics*, vol. 25, no. 5, pp. 336–354, 2012.
- [6] D. M. Wolpert, J. Diedrichsen, and J. R. Flanagan, "Principles of sensorimotor learning," *Nature Reviews Neuroscience*, vol. 12, no. 12, pp. 739–751, 2011.
- [7] K. P. Körding and D. M. Wolpert, "Bayesian integration in sensorimotor learning," *Nature*, vol. 427, no. 6971, pp. 244–247, 2004.
- [8] J. Piater, S. Jodogne, R. Detry et al., "Learning visual representations for perception-action systems," *International Journal of Robotics Research*, vol. 30, no. 3, pp. 294–307, 2011.
- [9] M. Do, J. Schill, J. Ernesti, and T. Asfour, "Learn to wipe: a case study of structural bootstrapping from sensorimotor experience," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA '14)*, pp. 1858–1864, Hong Kong, May–June 2014.
- [10] H. Guedjou, S. Boucenna, and M. Chetouani, "Posture recognition analysis during human-robot imitation learning," in *Proceedings of the Joint IEEE International Conference on Development and Learning and Epigenetic Robotics (ICDL-EpiRob '16)*, pp. 193–194, Cergy-Pontoise, France, September 2016.
- [11] A. Sadeghipour and S. Kopp, "Embodied gesture processing: motor-based integration of perception and action in social artificial agents," *Cognitive Computation*, vol. 3, no. 3, pp. 419–435, 2011.
- [12] J. Zhong, A. Cangelosi, and S. Wermter, "Toward a self-organizing pre-symbolic neural model representing sensorimotor primitives," *Frontiers in Behavioral Neuroscience*, vol. 8, article 22, 2014.
- [13] V. Cutsuridis and J. G. Taylor, "A cognitive control architecture for the perception-action cycle in robots and agents," *Cognitive Computation*, vol. 5, no. 3, pp. 383–395, 2013.
- [14] T. Shi, W. Yang, and H. Ren, "A study on the cognitive model of robot sensorimotor system," *Journal of Intelligent and Fuzzy Systems*, vol. 28, no. 5, pp. 1955–1968, 2015.
- [15] R. Kurzweil, *How to Create a Mind: The Secret of Human thought Revealed*, Viking, 2013.
- [16] J. Hawkins, S. Ahmad, S. Purdy, and A. Lavin, *Biological and Machine Intelligence (BAMI)*, 2016, <http://numenta.com/biological-and-machine-intelligence/>.
- [17] J. Hawkins, *On Intelligence*, Times Books, New York, NY, USA, 2004.
- [18] D. Huber, D. A. Gutnisky, S. Peron et al., "Multiple dynamic representations in the motor cortex during sensorimotor learning," *Nature*, vol. 484, no. 7395, pp. 473–478, 2012.
- [19] C. Lalanne and J. Lorenceau, "Crossmodal integration for perception and action," *Journal of Physiology—Paris*, vol. 98, no. 1-3, pp. 265–279, 2004.
- [20] E. Wieser and G. Cheng, "Progressive learning of sensory-motor maps through spatiotemporal predictors," in *Proceedings of the Joint IEEE International Conference on Development and Learning and Epigenetic Robotics (ICDL-EpiRob '16)*, pp. 43–48, Cergy-Pontoise, France, September 2016.
- [21] Y.-S. Huang and Y.-J. Wang, "A hierarchical temporal memory based hand posture recognition method," *IAENG International Journal of Computer Science (IJCS)*, vol. 40, no. 2, pp. 87–93, 2013.
- [22] Y. Du, W. Wang, and L. Wang, "Hierarchical recurrent neural network for skeleton based action recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR '15)*, pp. 1110–1118, June 2015.
- [23] D. Rozado, F. B. Rodriguez, and P. Varona, "Extending the bioinspired hierarchical temporal memory paradigm for sign language recognition," *Neurocomputing*, vol. 79, pp. 75–86, 2012.
- [24] S. Ahmad, A. Lavin, S. Purdy, and Z. Agha, "Unsupervised real-time anomaly detection for streaming data," *Neurocomputing*, vol. 262, pp. 134–147, 2017.
- [25] G. Knoblich and R. Flach, "Predicting the effects of actions: interactions of perception and action," *Psychological Science*, vol. 12, no. 6, pp. 467–472, 2001.
- [26] A. E. Monroe, G. D. Reeder, and L. James, "Perceptions of intentionality for goal-related action: Behavioral description matters," *PLoS ONE*, vol. 10, no. 3, Article ID e0119841, 2015.

- [27] S.-J. Blakemore and J. Decety, "From the perception of action to the understanding of intention," *Nature Reviews Neuroscience*, vol. 2, no. 8, pp. 561–567, 2001.
- [28] E. A. Di Paolo, X. E. Barandiaran, M. Beaton, and T. Buhrmann, "Learning to perceive in the sensorimotor approach: Piaget's theory of equilibration interpreted dynamically," *Frontiers in Human Neuroscience*, vol. 8, article 551, 2014.



Hindawi

Submit your manuscripts at
<https://www.hindawi.com>

