EEG-Induced Autonomous Game-Teaching to a Robot Arm by Human Trainers Using Reinforcement Learning

Reshma Kar, Lidia Ghosh, Amit Konar, Aruna Chakraborty and Atulya K. Nagar

***Abstract*— This paper deals with a simple indoor game, where the player has to pass a ball through a ring fixed on a variable pan-tilt platform. The motivation of the research is to learn the gaming actions of an experienced player by a robot arm for subsequent training to younger children (trainee) by the robot. The robot learns the gaming actions of the player at different game states, determined by pan-tilt orientations of the ring and its radial distance with respect to the player. The actions of the experienced player/expert are defined by six parameters: three junction-coordinates in the right arm of the player and the 3-dimensional speed of the ball in a given throw. Reinforcement learning is employed here to adapt a state-action probability matrix of a probabilistic learning automation based on the reward (or penalty) scores of the player due to the success (or failure) in passing the ball through a given ring. A hybrid brain-computer interface (BCI) is used to detect the failures in the gaming action of the player by natural arousal of Error-related Potential (ErrP) signal following motor execution, indicated by motor imageries. In absence (presence) of ErrP after a motor imagination, the system considers a success (failure) in the player's trials, and thus adapts the probabilities in the learning automata according to success/failure of individual game instances. After the convergence of the state-action probability matrix, the same is used for planning, where the action corresponding to the highest probability at a given state in the automaton is selected for execution. The robot can autonomously train the game to the children using the learning automaton with converged probability scores. Experiments undertaken confirm that the success rate of the robot arm in the motor execution phase is very high (above 90%) when the ring is placed at a moderate distance of 4 feet from the robot.**

***Index Terms*— Brain-Computer Interfaces, Reinforcement Learning, Gaming, Event-Related Potentials, Event- Related Desynchronization/Synchronization.**

I. Introduction

Brain-Computer Interface (BCI) based gaming is gaining increasing popularity over the last one decade [1-4]. Most of the BCI-based games employ electroencephalography (EEG) for online detection of player's motive, self-assessment about his performance [5], learning skill to improve performance [6], automatic training of external manipulators/robots by BCI-based learning strategies [7, 8], and the like. Existing BCI-gaming applications employ different EEG signals, such as Event Related De-synchronization/Event Related Synchronization (ERD/ERS) [9, 10], P300 [11, 12], Steady-State Visual Evoked Potential (SSVEP) [13, 14], neuro/bio-feedback [1] and hybrid paradigms [15] to address different problems in computer games. In recent times, the authors used left/right motor imagery to control cursors [16], paddles in pinball games [17] and robotic manipulators [8], to improve success-rate in game outcomes. In Chumerin *et al.* [18], the authors aimed at balancing a rod by distributing the player's attention uniformly on two flickering boards to arouse SSVEP. In Martišius and Damaševičius [19], the authors used maze navigation using SSVEP. Another interesting application using SSVEP is automatic target-shooting [20]. Selecting a grid, containing an object of player's interest among a set of grids, using P300, also known as Donchin-Farwell protocol, is popularly used in many virtual games [21, 22]. Early BCI was restricted to multi-trial analysis. However, for real-time gaming applications, single-trial EEG BCI is emphasized [9-15] over its multi-trial counterparts.

Different metrics of performance analysis of BCI-based gaming applications are prevalent in the literature [1-8]. Two well-known metrics that deserve special mention include classification accuracy and success rate. Classification accuracy here refers to the accuracy of the pattern classifiers employed for classification of brain signals to control the gaming actions. Success-rate in connection with BCI-games indicates the number of successes in the winning action among all possible gaming actions. In addition, in maze-type BCI gaming, one parameter, called 'Mission Time Ratio', which is the relative time to reach the goal in a maze [23], is often utilized to measure performance of the system. There are also traces of using difficulty/fun/goal appreciation/motivation of the game-players as performance measure of BCI-based games [24]. In this paper, we introduce precision that measures degree of user's success in the game (for example, deviation of the ball from the target in a ball-throwing game) as an additional metric to examine the level of success of BCI-incorporation in the game.

Existing research on BCI-based gaming attempts to enhance subjective skill of patients suffering from neuro-motor disability [25], locked-in syndromes [26], attention deficit hyperactivity disorder (ADHD) [27], scope of recreation and training of healthy subjects with BCI aids also are reported in a recent work [35]. The last decade has seen significant progress in BCI-based rehabilitative robotics [28, 29]. The motivation of this paper is slightly different from the existing BCI-based gaming applications. Here, the gaming skill of an experienced player is acquired in the form of reward/penalty for each pair of gaming state and subjective action by the player. This is achieved by a novel probabilistic automaton-based reinforcement learning. The learning score/probability of success for each state-action pair is subsequently transferred to a robot arm to copy the action for the highest

reward at a given gaming state. Thus, the robot arm can replace an experienced player in training the game to children. Such practice of training games to children would replace skilled game-experts, particularly when there is a scarcity of such experts. Besides, the boredom of human experts due to repeated training of children can be avoided by the proposed scheme. Questions may be raised whether the machine-centered training is acceptable. The answer is in the affirmative, if the quality of training is at par with that offered by human trainers. The principle adopted to realize machine-centered training includes two phases. In the first phase, the machine (here the robot arm) learns the steps/moves of the game from the experienced players; the knowledge earned thereby is stored in a knowledge-base in the form of a state-action probability matrix (SAPM) for subsequent training of new game learners in the second phase.

In the present context, we consider a simple hypothetical one-person game, where the player has to throw a ball into a distant box through a circular narrow hole, located on top of the box. While undertaking experiments, we considered a number of such boxes with adjustable pan and tilt angles. The player has to stand at a fixed position but can orient his physique in any way required to correctly throw the ball into the target hole. The success or failure in the throw is determined by the player himself and is recorded from the acquired brain signals of the player. Two EEG signals are required to convey the experienced player's opinion about success/failure in a single throw. These signals are event related de-synchronization/synchronization (ERD/ERS) and Error-related Potential (ErrP). The ERD/ERS signal is available at the onset of motor planning/execution during the ball-throw task. In case the subject detects any error in his throw (i.e., the ball does not enter the box through the top hole), it triggers an ErrP signal, which can be measured from the central electrodes, which conveys the computer that the throw was erroneous. The success or failure in individual throws, thus detected by the player, is used to build up a new type of Probabilistic Automaton [34], to record the measure of successive winning score at each state-action indices of the table. Here, state refers to parameters of the box, such as the radial distance d of the box with respect to the player, and the pan (φ) and tilt (θ) angles of the circular ring mounted on the box. The action here stands for the parameters of the player, including the three junction coordinates: J_e, J_s, J_w respectively for elbow, shoulder and wrist of the used upper limb in 3-dimension, and the speed (v) of release of the ball in 3-dimension: $[v_x, v_y, v_z]$, computed from the gesture of the player during ball throws. For convenience of realization of the probabilistic automaton, the parameters included in the state and action space are quantized into non-overlapped intervals, such that union of the intervals for each parameter includes the entire feasible space of the same parameter. For example, if there exist k quantized non-overlapped intervals of $v_x : q_1(v_x), q_2(v_x), ..., q_k(v_x)$ then $\bigcup_{i=1}^{k} q_i(v_x)$ covers the feasible space of v_x.

The experiments are performed with 10 experienced players, each participating in 36 sessions over 36 days (i.e., one session in a day), where each session includes 50 trials (each trial indicating a single throw) over a state-action space of 36×2^{18}. In each session, the pan-tilt orientations of the ring and its radial distance with respect to the player are kept fixed. The dimension of the state-action space is evaluated as follows. Here, a state is defined by 3 parameters: i) distance of the box from the thrower (d), ii) tilt angle (θ), and iii) pan angle (φ) of the box-top, where the above 3 parameters respectively have 4, 3 and 3 variants, which altogether yields $4 \times 3 \times 3 = 36$ states. The action space includes 3 junction coordinates of the right arm of the player and velocities in 3-dimension. For each dimension of junction coordinate or velocity, we consider 2 distinct intervals. Thus for the position/velocity of one of the 3 junctions (shoulder, elbow and wrist), each represented in 3-dimension, we have $2^3 = 8$ possibility action-spaces. Consequently, considering the positions of 3 junctions, the possibility action-space is $8 \times 8 \times 8 = (2^3)^3 = 2^9$. Further, considering variability of 3 dimensional velocities of the 3 junctions, we have an additional 2^9 possibility action-space. Lastly, considering variations of both the junction positions and velocities, both in 3-dimension we have a total possibility action-space of $2^9 \times 2^9 = 2^{18}$. We consider 2 intervals for each component (x-, y- and z-) of junction coordinates and velocities. Finally, for 36 state-spaces and 2^{18} action-spaces, the total state-action space is 36×2^{18}.

The SAPM is recorded for each player separately and the SAPMs of all the participants' (here, 10 experienced players) response are averaged to teach the robot. After the adaption of SAPM for 10 experienced players over 36 sessions is over, we use the SAPM for game-planning by Jaco robot arm. In the planning stage, the robot is provided with a given state, and it selects the best action at that state. The best action is defined by the action with the highest probability in the selected state of the SAPM. The robot demonstrates the planned action to teach the game to children. Children too enjoy the game-learning from the robot as it is free from human-interaction, which often includes rough voice and/or eyebrow-raising by the game-teacher. In fact, it is observed experimentally that the success-rate of game-learning by children from the robot is higher than the success-rate of learning from experienced teachers.

The paper is divided into five sections. Section II provides a thorough description of the proposed BCI based gaming scheme along with an algorithm for adaptation of the SAPM. It also covers the BCI signals used to control the game actions. In section III, we present EEG based feature extraction and classification along with other experimental details. Performance analysis is undertaken in section IV. Conclusions are listed in section V.

II. Proposed Scheme

Training games to children by traditional human trainers is tedious on part of the trainers. Besides, learning-performance of the trainees is not free from the influence of the human

quality of the trainers. One approach to overcome the above two problems is to design and develop an environment for autonomous training of the children by robotic devices. This paper serves that purpose. Here, the authors attempt to train a robot arm to play a selected game by executing the right action at the right time. After the robot is trained, it can take the initiative to train game to the children.

In the present context, we consider a non-traditional single agent game, where the player has to stand at fixed (radial) distance from a set of boxes with adjusted pan and tilt angles of the circular opening fixed on the top of each. The player has to throw a ball towards a selected target (box). For a given box, the radial distance of the centroid of the circular top opening from the player, and the pan and the tilt angles of the opening are pre-fixed. These parameters jointly represent a state of the game. The 3-dimensional coordinates of the joints (right shoulder, elbow and wrist) of the right arm of the player are captured by Microsoft Kinect machine [31] for the estimation of predicted speed of the ball in 3-dimension. The 3-dimensional junction coordinates of the above mentioned joints at the time-point of release of the ball during ball-throws and 3-dimensional velocity of the ball jointly represent the action of the agent.

We here adopt single agent Reinforcement Learning (RL) algorithm to save the rewards earned by the robot during the correct throws of the ball in the SAPM. The SAPM thus is indexed by states of the box and action of the agent respectively. During a correct throw of the ball, we use a reward-estimating function to determine the reward at a given state-action of the SAPM. Similarly, during failures in placing the ball inside the box, we estimate the penalty and place it at the right cell of the SAPM. It is interesting to note that the occurrence of reward/penalty is determined by the subject himself and is captured naturally using an EEG device. We here use two EEG signals: ERD/ERS (Event Related De-synchronization/synchronization) and ErrP (Error Related Potential). ERD/ERS is captured from the parietal lobe and motor cortex (P3, P4, C3, C4 electrodes) and the ErrP is captured from the *z*-electrodes (Fz, Pz and Cz electrodes) placed on the scalp of the subject. Here, ERD/ERS is used to detect the onset of the motor execution by the subject, just before throwing the ball, while ErrP is used to detect the occurrence of errors. The error here is linked with failures to place the ball inside the box in a ball-throw by the subject (player). The ERD/ERS signal is regarded as the event onset of the ball throw process. After the ERD/ERS is detected, the system waits exactly 800 ms for an ErrP. If no ErrP is detected, a reward is attributed to the present throw. However, if ErrP is detected, it is counted for an occurrence of error in the ball throw, and a negative reward/penalty is attributed for the present throw. Thus, the ErrP signal is used to monitor the occurrence of error and assignment of a penalty at the selected state-action pair. In case no ERD/ERS is detected, subsequent ErrP is not analyzed. We record the reward/penalty scores in the SAPM and for subsequent training of novice players by a robot arm.

After the training of the robot arm is over, it utilizes the SAPM matrix to plan its action at a given state s_i. The planning is performed by selecting (most promising) action a_j, with probability $\Pr(a_j \mid s_i)$. After selecting the action, the robot enacts it based on parameters of the action (including proper orientation of the junctions of the robot, speed and direction of the ball). The motivation of optimal action selection at a given state lies in displaying the gestural action of the robot to help children imitating the best action at the selected state to successfully play the game.

A. Proposed Reinforcement Learning Scheme

Reinforcement learning (RL) refers to learning by reward/penalty. It's a slow and lifelong learning process. In natural RL, we plan any action based on partial learning of our environment. However, in the present scheme, we undertake action planning at a state after convergence of the RL algorithm. The difference between natural and the present RL-based planning is apparent as here the state-action space is finite and small, whereas in natural RL, the state-action space is infinitely large, and so continues life-long for the agents. Let,

$S = \{s_1, s_2, ..., s_n\}$ be a set of *n* states.

$A = \{a_1, a_2, ..., a_m\}$ be a set of *m* actions.

$r(s,a)$ is a positive reward function at state $s \in S$ and action $a \in A$.

$p(s,a)$ is a negative reward function (penalty) at state $s \in S$ and action $a \in A$.

$R(s, a)$ is a cumulative reward at state $s \in S$ and action $a \in A$.

α is a positive constant lying in [0,1], called the learning rate. A small value of α ($\approx$0.05) ensures slow learning without early convergence [32].

We adopt the following learning strategy.

For $s \in S$ *and* $a \in A$

$$Temp(s,a) = \begin{cases} R(s,a) + \alpha \cdot r(s,a), & if\ r(s,a) > 0 \\ R(s,a) + \alpha \cdot p(s,a), & if\ p(s,a) < 0 \end{cases} \quad (1)$$

$$R(s,a) = \frac{1}{1 + \exp(-Temp(s,a))} \quad (2)$$

End-For;

The positive reward function *r(s, a)* and the penalty function $p(s, a)$ are fixed. For example, in the present application, we assign $r(s, a) = +0.01$, and $p(s, a) = -0.01$. The Sigmoid function in (2), restricts $R(s, a)$ in [0, 1]. The occurrence of $r(s,a)$ and $p(s,a)$ are determined from the respective non-occurrence and occurrence of the ErrP signal after 800 ms from the onset of the ERD/ERS.

Probabilistic Learning for Adaptation of SAPM

Input: Occurrence of Error at given State-Action Pairs
Output: Updated SAPM of ($n \times m$) dimension;

Begin

1. **Initialization:** Initialize probability $\Pr(a_j \mid s_i)$ such that

   For each *i*

$\Pr(a_j \mid s_i) \leftarrow \frac{1}{m}, \forall j$ so that $\sum_{\forall j} \Pr(a_j \mid s_i) = 1$

End For;

2. Action Selection: Select an action a_j from the action set *A* at state s_i using Roulette wheel action selection strategy;

3. Adaption in learning space:

If selection of action a_j at state s_i returns a success then increment $\Pr(a_j \mid s_i)$ by a small predefined number $r(s_i, a_j)$, where $0 < r(s_i, a_j) < 1$.

$\Pr(a_j \mid s_i) \leftarrow F(\Pr(a_j \mid s_i) + \alpha \cdot r(s_i, a_j))$ for action *j*, where

$F(x) \leftarrow \frac{1}{1+e^{-x}}$,

$\Pr(a_k \mid s_i) \leftarrow F(\Pr(a_k \mid s_i) - \alpha \cdot \frac{r(s_i, a_j)}{m-1}), \forall action\ k, k \neq j$

where $F(.)$ is Sigmoid function defined above.

If due to the selection of a_j at state s_i there is a failure, then decrease $\Pr(a_j \mid s_i)$ by a small constant penalty $|p(s_i, a_j)|$ for $p(s_i, a_j) < 0$.

$\Pr(a_j \mid s_i) \leftarrow F(\Pr(a_j \mid s_i) + \alpha \cdot p(s_i, a_j))$, for action j

$\Pr(a_k \mid s_i) \leftarrow F(\Pr(a_j \mid s_i) + \alpha \cdot \frac{|p(s_i, a_j)|}{m-1}) \forall k, k \neq j$

Continue through step 2 until $\Pr(a_j \mid s_i)$ converges for all *i*, *j*.

End

B. Proposed Action Selection Strategy

There exist several action selection strategies in RL [33]. Random action selection strategy is often used for its simplicity. However, random action selection does not ensure exploration of the entire action space at the selected state. One approach to overcome this problem is to adopt Roulette wheel selection strategy [32] to select action a_j at state s_i. The Roulette wheel selection is realized by the following 2 steps.

Let $c_{j,i}$ be the cumulative probability sum of *j* probabilities in a sorted array of probabilities, arranged in ascending order. Let *k* be the index of the sorted array. In other words,

$$c_{j,i} = \sum_{k=1}^{j} \Pr(a_k \mid s_i) \quad (3)$$

where $\Pr(a_k \mid s_i) > \Pr(a_{k-1} \mid s_i), \forall k$.

Naturally, from *m* possible actions at state s_i, $c_{m,i} = \sum_{k=1}^{m} \Pr(a_k \mid s_i) = 1$, we generate a random number *r* in [0, 1]. If $c_{j,i} < r < c_{j+1,I}$, then we select action *j*.

C. Proposed ERD/ERS and ErrP Based Learning

Automatic detection of reward and penalty based on subject's own judgment is a novel contribution of the present work. Here, two brain signals, called ERD/ERS and ErrP signals are employed to determine the subject's opinion about his/her success in the current trial of the game. The ERD/ERS is associated with motor imagery and/or motor execution [8], whereas ErrP is used to represent subjective error when the subject observes a system/agent committing an error and/or when the subject commits an error himself [44]. The motor execution is performed by the subject at the time-point he releases the ball. The time-point at which ERD/ERS occurs is important, as the system may be programmed to wait for the next 800 ms for possible release of an ErrP signal. The error in the present circumstance indicates a failure in the ball throw, i.e., when the ball fails to reach the subject-defined target position. Thus, in every trial the BCI system looks for an ErrP signal, without noticing whether the ball reaches the target position.

The ERD/ERS and ErrP based error detection is important in order to keep track of the ball position after the ball is thrown. In case, no ErrP is detected, a small positive incremental reward $r(s, a)$ is attributed to the state *s* for the selected action *a* in the SAPM. However, if an ErrP occurs within 800 ms of the ERD/ERS, the SAPM is updated with a negative reward/penalty $p(s, a)$ to the states for the selected action *a*. The SAPM thus obtained for all system state-actions is preserved. The process of SAPM computation is also repeated for all experimental players. Fig. 1 provides a schematic overview to EEG-based game learning. A timing protocol needs to be devised to mark the time points to identify the reward/penalty. This is done by the following two steps. When the subject executes a motor action, an ERD/ERS is released from the parietal lobe and motor cortex. The time point of release of ERD/ERS is marked on the time-line. Next, the BCI system looks for an ErrP within 800 ms from the marked point of ERD/ERS-release on the time-line to update the SAPM due to reward (no ErrP) or penalty (ErrP) in the gaming action. The magnitude of the reward or penalty is determined by a trial-error approach. A small positive reward/penalty takes large convergence time of SAPM, whereas a large value results in a pre-mature convergence. The choice of the incremental probabilistic reward/penalty thus is an important issue.

D. The Proposed Reinforcement Learning based Planning

After the learning phase in SAPM by the proposed algorithm introduced earlier is over, i.e., the probability estimates for the required action at each state has converged, the robot can autonomously generate its plan from the probability estimates. For example, if at state s_i, suppose the action a_j has the highest probability of occurrence. Then the robot will select the action a_j. Now, execution of the selected action requires configuring the robot arm to orient its axes in different angles, which are obtained by inverse kinematics [30]. In Jaco robot arm, the inverse kinematics problem is solved by calling selected library functions that offer the required angular movements of the individual links from the desired coordinate of the end-effectors.

E. Imitating Expert Action by a Robot

Execution of repetitive training by human experts, which generally is tiresome, often causes boredom to the trainers. Here a Jaco humanoid robot arm (Fig. 2 (b)) is trained to replace the human trainer. To imitate all possible actions of the human trainer, a large number of training instances are generated to train the robot arm before it is employed for training games to the children. Human arm kinematics in simplistic form can be approximated as 5 joint movements in fixed directions/orientations, as illustrated in Fig. 2 (a). We used the well-known Denavit-Hartenberg link configuration scheme [30] to describe the turning of individual links, when the person (experienced player/experimental subject) is engaged in the 'ball throwing' experiment. We define the turning of individual robotic links around their *z*-and/or *x*-axis by θ and α respectively.

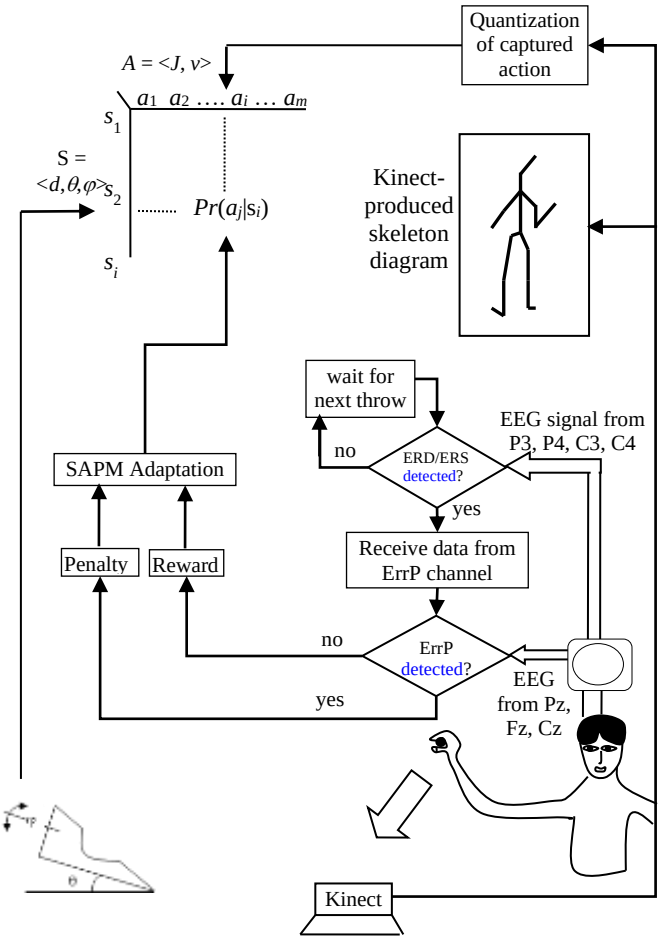


Fig. 1 Reinforcement learning of gaming actions of an expert performer using an EEG system and Microsoft Kinect machine. The expert performer plays the game and his EEG signals are recorded. If an ErrP occurs within 800 ms of the ERD/ERS, the SAPM is updated with penalty or else with a reward.

The overall coordinate transformation matrix, describing the rotations of all the links involved by prescribed angles is given by

$$ {}^{0}T_5 = {}^{0}A_1\,{}^{1}A_2\,{}^{2}A_3\,{}^{3}A_4\,{}^{4}A_5 \qquad (4) $$

where, ${}^{i-1}A_i = T_z(\theta_i) \bullet T_x(\alpha_i)$ *for* $i = 1$ *to* 5.

$$ T_z(\theta_i) = \begin{bmatrix} Cos(\theta_i) & -Sin(\theta_i) & 0 & 0 \\ Sin(\theta_i) & Cos(\theta_i) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, $$

and $$ T_x(\alpha_i) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & Cos(\alpha_i) & -Sin(\alpha_i) & 0 \\ 0 & Sin(\alpha_i) & Cos(\alpha_i) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}. $$

Let the initial position of the palm center be $[x_i\ y_i\ z_i\ 1]^T$, we determine the final coordinate of the same center $[x_f\ y_f\ z_f\ 1]^T$ by

$$ [x_f\ y_f\ z_f\ 1]^T = {}^{0}T_5 \bullet [x_i\ y_i\ z_i\ 1]^T. \qquad (5) $$

It is important to note that Jaco Robot arm possesses only one degree of freedom for the joint J2, representative of human shoulder, to tilt the robot arm, but it has no freedom to change the pan-angle. Naturally, to adopt changes in pan-angle, the robot utilizes its waist joint J1 (Fig. 2(b)). In order to have similarity between the angular movements of the human and the robot arms, we use the coordinate systems: $[x_0–y_0\ –z_0]$, $[x_2–y_2\ –z_2]$, $[x_3–y_3\ –z_3]$, $[x_4\ –y_4\ –z_4]$, $[x_5\ –y_5\ –z_5]$ for the human arm and $[x_0–y_0\ –z_0]$, $[x_1\ –y_1\ –z_1]$, $[x_2\ –y_2\ –z_2]$, $[x_3\ –y_3\ –z_3]$, $[x_4–y_4\ –z_4]$ for the robot arm. Although the above two coordinate systems have differences in the point of application of similar torques to orient the arm to a desired configuration, they suffice to realize human arm movements in the present gaming application using the Jaco robot arm.

The Microsoft Kinect machine is employed to record movements of the body-junctions from the Red-Green-Blue (RGB) color images and the z-coordinate/the depth information from the infrared image. The x-, y- and z-coordinates are saved in the system memory and are used later for subsequent analysis. Fig. 1 provides a schematic diagram of one possible gesture of an experienced game player during ball-throwing and the corresponding gesture captured by a Microsoft Kinect machine. The captured gestures are later imitated by the robot trainer to train children to throw ball at the given position of the box.

TABLE I
OVERVIEW OF THE EXPERIMENTAL STEPS UNDERTAKEN DURING TRAINING AND TEST PHASES

	Training Phase	**Test Phase for System Validation**
Offline Steps	Feature extraction and classification for ERD/ERS and ErrP.	None
Online Steps	EEG feature extraction and classification for SAPM Updating	Action Planning by Jaco Robot arm using SAPM. Performance Analysis of Jaco in Test Phase. Performance Analysis of Jaco in Training Children

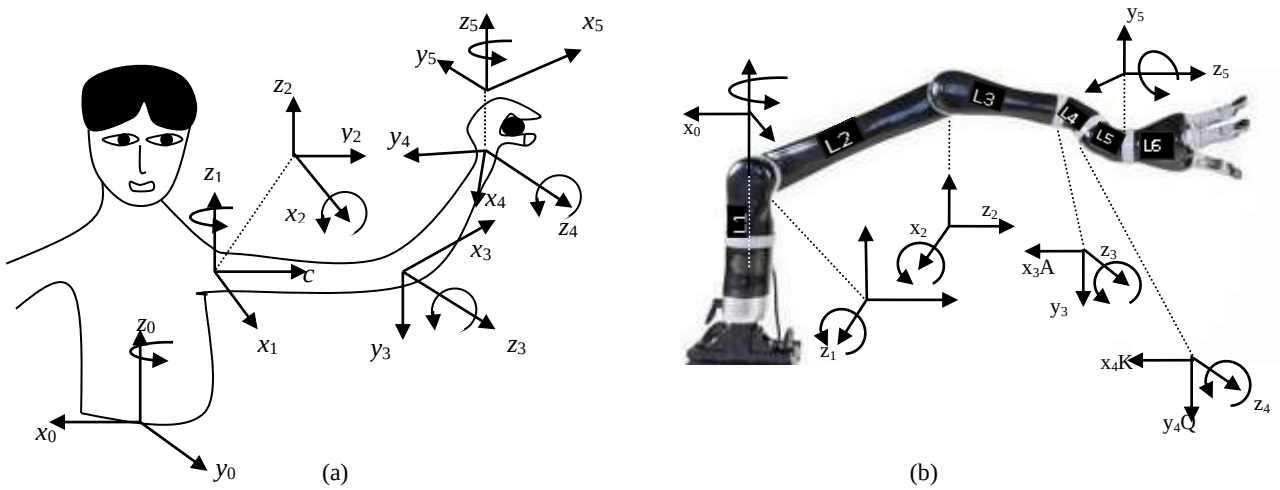


Fig. 2 (a) Human and (b) Robot Kinematics used following Denavit-Hartenberg Link Configuration scheme

F. Capturing Body-Junction Coordinates of the Trainer for Realization by the robot

The Kinect machine has been employed to capture the body-junction coordinates (*J*) of the shoulder, elbow and wrist of the experienced player, during the phase of demonstration of the game by him. The captured movements of the junction coordinates are then realized by the robot arm to imitate the required arm movements to throw the ball to place it in the box located at a given local neighborhood of the robot arm. The Microsoft Kinect machine acquires the visual and infrared spectra of the experimental subject (here, the experienced player) to determine coordinates of the body-joints: right shoulder, elbow and wrist, while releasing the ball, and estimated velocity in 3-dimensions from the last two frames used. Here, the last 2 frames are with respect to ball throw, extracted from the objective movement recorded with the Kinect.

III. EXPERIMENTS

This section deals with experimental protocol, details of experiments undertaken, and main results obtained thereof. The experiments are broadly divided into two phases namely the training phase and the test phase (Table I).

A. Training Phase

The primary objective of the training phase is to update the SAPM such that it represents the best possible action of the robot at the given states. Since SAPM updating is based on the detection of ERD/ERS and ErrP, therefore it is essential to train two classifiers (offline) separately to detect the presence (or absence) of the specific signal in the desired time-window. Thus, the training phase constitutes two main steps: (a) offline analysis of the acquired EEG data to train the classifiers, and (b) online ERD/ERS and ErrP detection followed by SAPM updating.

Offline classifier training instance generation: During the offline training phase, the experimental subject (here, the trainer) is presented with a stimulus, as given in Fig. 3 (a). Each subject underwent 36 sessions with sufficient relaxation time (here, one day) between successive sessions, where the subject has to throw a ball 50 times in a session through a ring mounted on a variable pan-tilt platform. Each session is dedicated for a specific combination of pan-tilt orientations of the ring and its radial distance with respect to the player. The stimulus includes a fixation cross for 1s, followed by motor planning and execution (MPE) session for 1s, and time of flight (TOF) of the ball and ErrP generation together for 800ms and a rest interval of 1 minute. During those sessions, EEG data are acquired from the P3, P4, C3 and C4 electrodes for ERD/ERS detection, and Fz, Cz and Pz electrodes for ErrP detection. The EEG data acquired in the respective time-windows is utilized to manually check the presence of ERD/ERS and/or ErrP signals. A few snapshots describing the experimental set-up during the offline training session are presented in Fig. 4.

Success in the ball-throw here can be determined either by physical examination of the ball in the basket after its release (time-point of ERD/ERS release) or absence of ErrP signal in the right time-window. Here, the second option is attempted, primarily to utilize the trainer's assessment on his own success/failure about his throw towards the pre-defined target box.

Since each of the 10 experimental subjects (human trainers) participates in 36 training sessions, comprising 50 trials, the total number of training instances available for each channel is $10 \times 36 \times 50 = 18{,}000$. Here, ERD/ERS signal is obtained for all the 18,000 instances. The no occurrence instances of ERD/ERS are obtained by considering the EEG data of baseline/rest period acquired from the same electrodes. On the other hand, the presence of ErrP signals is observed in 11,564 training instances, and the rest of the 6436 instances are used as no ErrP trials. It is to be noted that the number of occurrence of errors is comparable across the experienced players. Experimentally, it was found that $(58 \pm 6)\%$ of the throws of every experienced participant had errors.

The signals thus acquired are then processed through 3 steps, including pre-processing, feature extraction (FE) and

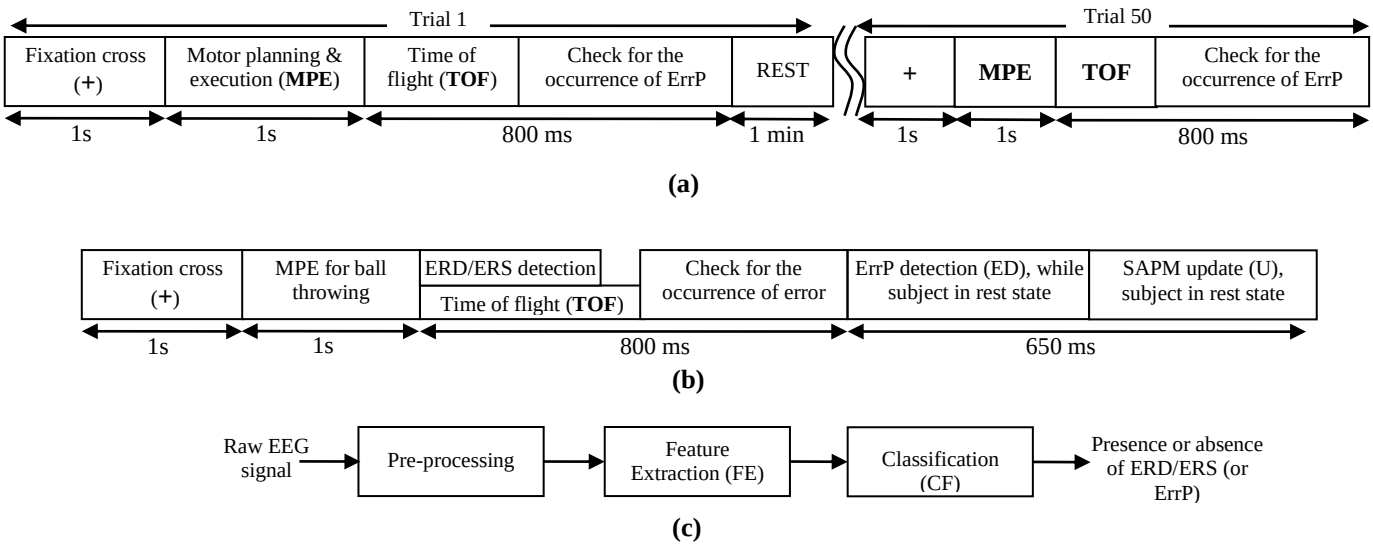


Fig. 3(a) Stimulus presentation for offline training, (b) Stimulus used for online training, and (c) Block diagram for the detection of ERD/ERS or ErrP

classification (CF) with an ultimate aim to train the classifiers used for ERD/ERS and ErrP detections, as indicated in Fig. 3(c). A brief description of each step is given in the subsequent sub-sections.

After acquisition of complete data sets for the 18000 instances, the offline training and test are performed for the 2 class classification between ERD/ERS and no ERD/ERS, and ErrP and no ErrP signals by employing a Support Vector Machine (SVM) with Gaussian and polynomial kernels respectively. A 5-fold cross validation [47] is used to check the performance of the classifier with classification accuracy as the metric.

Online SAPM Adaptation: Once the classifiers are well-trained to detect the ERD/ERS and ErrP with the highest possible accuracy, the ball-throwing experiment, narrated above, is repeated once again with an additional step of SAPM adaptation. The stimulus used for the online training phase is given in Fig 3(b). It includes 6 tasks for execution over four distinct time-windows. The first time-window is reserved for a fixation cross of 1s duration to make the trainer alert. The second window includes MPE for ball throwing, thereby resulting in the generation of ERD/ERS signal. The third time-window includes engaging the subject to observe the motion of the flying ball for possible occurrence of error, causing liberation of an ErrP, all within a time-interval of 800 ms as shown in Fig. 3(b). The fourth window of 650 ms is reserved for ErrP detection and SAPM updating, while the subject simply is having a rest state.

Pre-processing, FE and CF of ERD/ERS are undertaken just after the ball is set in motion. The detection of the 'ball in motion' is carried out by analyzing the color images captured by the camera of the Kinect. An audio feedback regarding the ERD/ERS and ErrP detection is provided to the subject during the online training session.

During the online training phase, EEG feature extraction and classification for ERD/ERS and ErrP detection are performed online. For ERD/ERS detection, the data acquired during the 1s slot, allocated to MPE and ERD/ERS generation session, is used for subsequent feature extraction and classification (FE + CF). The FE + CF for ERD/ERS detection is done in parallel with the TOF of the ball, as indicated in Fig. 3(b). The EEG data acquired during entire the time-slot of 800 ms, allocated for TOF and ErrP generation, is then processed through the necessary steps (FE + CF) in the next 650 ms time-window to detect ErrP The presence or absence of ErrP is then utilized to update the SAPM in the same time-span.

B. Test Phase

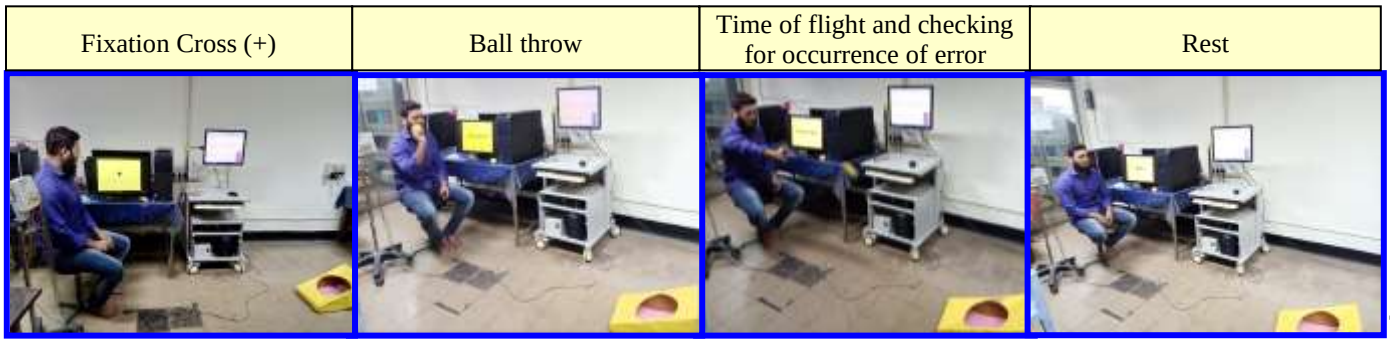


Fig. 4 Snapshots of experimental set-up during offline training session

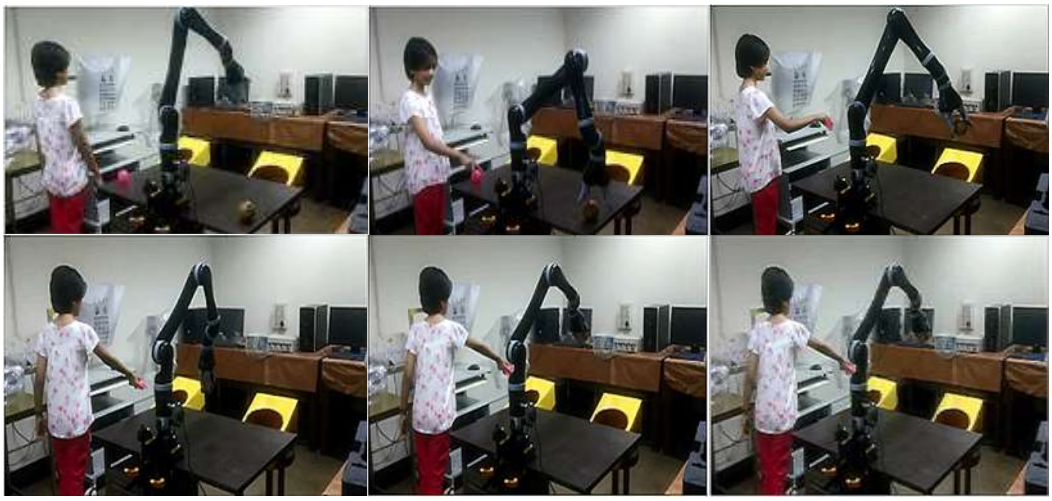


Fig.5 A child imitating robotic action while learning the game from the pre-trained robot arm

In the test phase, action planning by Jaco robot arm is performed by employing the updated SAPM. Also, in the test phase, the performance of children trained by Jaco is evaluated and compared to the performance of children trained by human trainers. Fig. 5 shows the imitating action of the robot by a child.

C. EEG Electrodes and Signal Acquisition

Electrodes are placed on the scalp of each subject according to the standard 10-20 electrode placement scheme [34]. For the detection of ERD/ERS, 4 electrodes: C3, C4 in the motor cortex and P3, P4 in the parietal lobe are selected. To detect the ErrP signals, data is acquired from the three electrodes: Cz, Pz and Fz. Therefore, altogether 7 EEG electrodes are used to acquire the data. Data acquisition was performed using the Nihon Kohden device at a sampling rate of 200 Hz. Two databases [49] acquired from 2 different groups of experts, one from South 24-pargana district, and the other from South Kolkata, West Bengal, India are prepared to undertake the experiment. Each database contains the ERD/ERS and ErrP signals of 10 subjects (experts/trainers), where each subject undergoes altogether 36 sessions and 50 throws/trials per session as explained in the previous subsections. Both the databases are prepared in Artificial Intelligence Laboratory of Jadavpur University. However, we consider only one database (South 24-parganas database) as reference in the rest of the paper.

D. Pre-processing

The pre-processing includes Common Average Referencing (CAR). The CAR operator subtracts the average of the instantaneous EEG signal values acquired from all the used channels. It has the merit to reduce the effect of artifacts on filtered signals. We also employed a Chebyshev band pass filter of order 5 to ensure sharp cutoff at undesired frequency-bands. Frequency roll-offs are also reduced due to introduction of Chebyshev band-pass filter. The cut-off frequency of the filter was 8-30Hz (such that beta band could be included in analysis). The signal was not down-sampled. Eye-blink artifacts were removed using independent component analysis [48].

E. Feature Extraction

During feature extraction, we select the wavelet coefficient (WC) with dB4 as the mother wavelet. We consider the percentage of energy of the fourth and the fifth detail wavelet coefficient, following [29]. We also compute the 7th order Adaptive Autoregressive (AAR) parameters during feature extraction. For ERD/ERS detection, the dimension of WC obtained from 4 electrodes (P3, P4, C3, C4) for a single trial is 114 (coefficients) × 4 = 456, and the dimension of AAR from the said 4 electrodes is 7 × 4 = 28. For ErrP detection, the dimension of WC obtained from 3 electrodes (Pz, Cz, Fz) for a single trial is 114 × 3 = 342, and that for AAR is 7 × 3 = 21. Each type of feature is tested separately to find out the best performing feature-classifier pair.

F. Classification

For the present application we need 2 classifiers, one to classify motor execution and the other to classify error-related potential. We selected the following off-the-shelf classifiers:

TABLE II

COMPARATIVE PERFORMANCE OF OFF-THE SHELF CLASSIFIERS FOR ERD/ERS CLASSIFICATION

Classifier	Average Classification Accuracy for ERD/ERS		Average Classification Accuracy for ErrP	
	WC	AAR	WC	AAR
LDA	78.98	79.09	78.55	87.56
QDA	86.99	89.91	89.77	83.81
Feed-Forward BPNN	79.78	81.68	76.92	73.14
Cascade-Forward BPNN	81.20	85.30	87.88	76.19
Naïve Bayes	85.66	89.66	77.81	89.99
SVM*	**96.15**	**95.22**	**89.99**	**96.78**

TABLE III

OPTIMAL SELECTION OF KSVM PARAMETERS FOR BOTH (A) ERD/ERS AND (B) ERRP CLASSIFICATION

(a)

SVM Classifier: **ERD/ERS DETECTION**

RBF Kernel

AAR Parameters

c	*σ*			
	0.01	*0.75*	*1.00*	*2.0*
0.5	*71.65*	*83.56*	*80.22*	*77.65*
1	*76.45*	***95.22***	*88.44*	*86.44*
5	*66.55*	*78.11*	*73.33*	*69.11*

Wavelet Coefficients

c	*σ*			
	0.01	*0.75*	*1.00*	*2.0*
0.5	*78.95*	***96.15***	*85.12*	*87.15*
1	*86.45*	90.22	*88.44*	*71.44*
5	*61.55*	*68.11*	*89.33*	*80.11*

Polynomial Kernel

AAR Parameters

c	*d*			
	1	*2*	*3*	*4*
0.5	*73.65*	*81.56*	*70.21*	*79.45*
1	*71.41*	75.22	*78.44*	*76.14*
5	*69.55*	*71.12*	*71.33*	***93.11***

Wavelet Coefficients

c	*d*			
	1	*2*	*3*	*4*
0.5	*78.90*	*86.98*	*87.14*	*76.98*
1	*70.95*	*93.57*	*81.82*	*79.65*
5	***91.56***	70.22	*88.44*	*77.44*

(b)

SVM Classifier: **ERRP DETECTION**

RBF Kernel

AAR Parameters

c	*σ*			
	0.01	*0.75*	*1.00*	*2.0*
0.5	*82.67*	***90.18***	*78.97*	*87.99*
1	*78.97*	*67.88*	*78.99*	*87.65*
5	*67.57*	*89.66*	*56.86*	*78.90*

Wavelet Coefficients

c	*σ*			
	0.01	*0.75*	*1.00*	*2.0*
0.5	*45.87*	*78.87*	*68.98*	***89.99***
1	*67.98*	*89.06*	*78.99*	*78.99*
5	*89.80*	*78.95*	*67.87*	*87.66*

Polynomial Kernel

AAR Parameters

c	*d*			
	1	*2*	*3*	*4*
0.5	*87..65*	*81.56*	*70.89*	*69.95*
1	*76.89*	*75.22*	***96.78***	*89.18*
5	*78.85*	*87.19*	*76.39*	*67.88*

Wavelet Coefficients

c	*d*			
	1	*2*	*3*	*4*
0.5	***89.79***	*87.98*	*78.89*	*78.98*
1	*70.95*	*87.67*	*87.76*	*79.65*
5	*89.15*	*71.23*	*88.44*	*79.49*

Linear Discriminant Analysis (LDA) [35], Quadratic Discriminant Analysis (QDA)[36], Feed-forward and Cascade-Forward Back-Propagation Neural Net (BPNN)[37],Naïve Bayes Classifier [39], Kernelized Support Vector machine (KSVM) [38] with linear, polynomial and Radial basis Function (RBF) kernel to examine the classifier performance for the present data set. The comparative performance of different classifiers for ERD/ERS and ErrP classification is given in Table II using the average of the classification accuracies obtained from the 5 fold cross-validation results. We also varied the parameters (c, σ, d) of the KSVM algorithms to select the optimal parameters of the selected KSVM for both for the ERD/ERS and ErrP classification. This is given in Table III (a) and (b) above. It is apparent that the performance of a classifier depends greatly on the choice of features [45]. Therefore it is necessary to select the right feature-classifier pair for ERD/ERS and ErrP classification. The results of this study are given in Table II and III. It is apparent from the tables that wavelet coefficient and KSVM with Gaussian Kernel together yields the best performance for ERD/ERS classification, while AAR parameters and KSVM with polynomial kernel together has the highest classification accuracy for the ErrP classification, given specific values of kernel parameters. Thus, the KSVM with Gaussian Kernel is chosen as the classifier for the online detection of the ERD/ERS, and the KSVM with polynomial kernel function for the online detection of ErrP. Furthermore, sensitivity and specificity [46] analysis of the best performing classifier-feature pair is performed and the results are given in Table IV, for both the ErrP and ERD/ERS classification.

TABLE IV

ANALYSIS OF BEST PERFORMING FEATURE-CLASSIFIER PAIR FOR ERRP AND ERD/ERS DETECTION

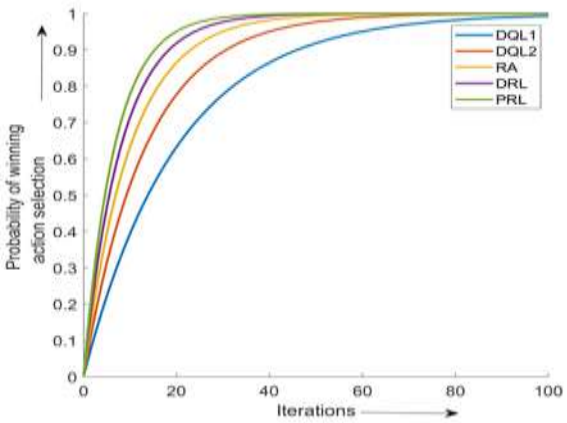
	Sensitivity	**Specificity**
ErrP	0.89	0.97
ERD/ERS	0.96	0.85

IV. PERFORMANCE ANALYSIS

This section deals with the evaluation of experimental results. The first part analyzes the convergence of the proposed RL scheme. The second part deals with the performance analysis of Jaco robot arm in imitating actions of the expert and the last part deals with analyzing the performance of children in learning the game from humans versus Jaco.

A. Performance Analysis of SAPM Updating by Reinforcement Learning

It is apparent that the probability of selecting a state-action pair is increased when an ErrP is absent following 800 ms of occurrence of an ERD/ERS. The proposed work on probabilistic reinforcement learning (PRL), is compared with two variants of Double Q-Learning namely, DQL1 [40] and DQL2 [41], Rainbow Algorithm [42], and Deep Reinforcement Learning (DRL) [43]. Fig. 6 provides a plot of probability of winning action selection versus iteration/learning epoch. It is observed from Fig. 6 that the proposed RL algorithm converges faster than the above mentioned algorithms, and thus justifies its importance over the existing techniques in the present application.



FFig. 6 Convergence of onFe Q-Ftable element over the iterations

B. Performance Analysis of SAPM Updating by Reinforcement Learning

After updating the SAPM, the action corresponding to the highest probability for a given state is selected. In order to verify that correct updating of actions has been performed, we test the performance of Jaco robot in the ball throwing game. The Jaco robot is allowed to perform the ball-throwing task for 50 times in each of the 36 experimental test sessions. Each time the configuration of the box is supplied, a state corresponding to the box position is selected as the row index. In the selected row, a column containing the highest reward is used to obtain the column index, which indicates the action to be performed by the Jaco robot. The robot is able to imitate the action represented by the human trainer, by utilizing the Denavit-Hartenberg scheme for robot-kinematics, as explained in section II E. Subjective Kinect data, representing the body junction coordinates used in the ball-throw task along with velocity of the junctions are transferred to a humanoid Jaco robot arm for testing the performance of the subject. The performance is measured using the success in the robot's aim in reaching the target position. We measure the radial distance between the fixed target point and the current location of the ball in 2-dimension (ignoring the *z*-dimension) to measure the separating distance between the centroid of the ball and the target position. Table V provides the measured radial distance between the ball-centroid and the target position in 2D for a given state. Considering 9 possible combinations of pan-tilt orientation of the target ring for a given radial distance and 50 ball throws for each combination, altogether the robot has to perform 450 throws for different distances. It is apparent from the Table that in about 78% cases, the ball was placed inside the box. Additionally, it is apparent from Table VI that the ball is placed within small vicinity (6 inches) around the target position in 81% cases. This ensures that the Jaco robot arm could accurately configure its structure to mimic the human subject. The results in Table V are also plotted in Fig. 7, showing the approximated performance of the Jaco robot arm in placing the ball within a radial distance (RD) of 6″, 8″ and 10″ from the box. Here, the experiment is carried out for 4 different distances of the box from the thrower: 4′, 5′, 6′ and 7′, 3 different ranges of tilt angle: [0 10), [10 20), [20 30) and 3 different ranges of pan angle: [-15 0), [0 15) and [15 30).

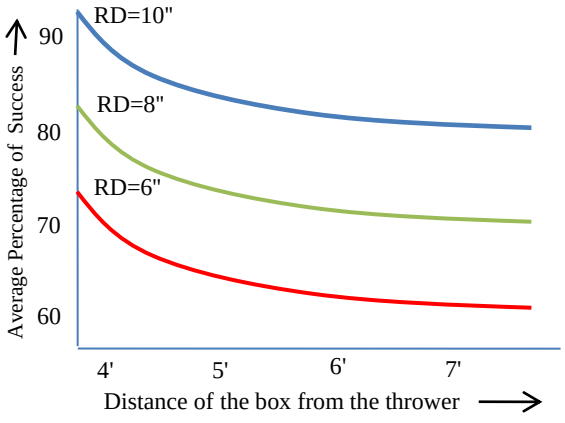


Fig. 7 Percentage success of ball-throwing by the Jaco arm for θ of the box within [0°, 10°) and ϕ within [-15°, 0°) with RD (radial distance) of below 6, 8 and 10 inches from the target box

C. Performance of Jaco in Training Children

20 right-handed children aged 8-10 years having normal or corrected vision, were asked to play the ball throwing game. Here, we arranged 36 training sessions for each child where each session consists of 50 trials/throws. In each experimental trial, two distinct phases of training are conducted. In the first phase, the Jaco robot arm selects the best action from the SAPM table and enacts the gesture to throw the ball to reach the target. In the latter phase, a human trainer does the same thing as Jaco did in the last phase. The children in either case have to mimic the trainer. Thus, a score-sheet like the one in Table VI is prepared. It is apparent from the Table that children's success to place the ball within 6″ of the target is higher by approximately (3 – 4)% when they are trained by the robot rather than by the human subject.

Further, children were asked to rate the motivation difficulty and fun on a scale of 1 to 3 for both the human and robot trainers. It was seen that learning from the Jaco robot was found slightly more difficult by the children. However, the fun and motivation of learning from a robot was higher and hence the children trained by robots experienced higher success (Table VII). The percentage of children who rated robot/human trainer experience for the three levels of fun/motivation/difficulty is given in Table VII.

TABLE V

PERCENTAGE OF BALL THROWS REACHED INTO TARGET FOR JACO ROBOT ARM

Distance of box from thrower	Tilt Angle Θ	Pan Angle Φ	Average % Success of throw /Ball entering the box, guided by Robot (Human Trainer)
4’	[0 10)	[-15 0)	**93%** (90%)
	[10 20)	[-15 0)	**93%** (90%)
	[20 30)	[-15 0)	92% (**93%**)
	[0 10)	[0 15)	**91%** (87%)
	[10 20)	[0 15)	**92%** (91%)
	[20 30)	[0 15)	**90%** (87%)
	[0 10)	[15 30)	**88%** (86%)
	[10 20)	[15 30)	**87%** (83%)
	[20 30)	[15 30)	**85%** (84%)
5’	[0 10)	[-15 0)	**86%** (83%)
	[10 20)	[-15 0)	**87%** (86%)
	[20 30)	[-15 0)	85% (**87%**)
	[0 10)	[0 15)	**84%** (83%)
	[10 20)	[0 15)	**86%** (82%)
	[20 30)	[0 15)	83% (**87%**)
	[0 10)	[15 30)	**84%** (81%)
	[10 20)	[15 30)	**87%** (85%)
	[20 30)	[15 30)	**82%** (81%)
6’	[0 10)	[-15 0)	**81%** (80%)
	[10 20)	[-15 0)	**83%** (82%)
	[20 30)	[-15 0)	80% (**85%**)
	[0 10)	[0 15)	**79%** (76%)
	[10 20)	[0 15)	**78%** (77%)
	[20 30)	[0 15)	73% (**74%**)
	[0 10)	[15 30)	**76%** (71%)
	[10 20)	[15 30)	**75%** (70%)
	[20 30)	[15 30)	**76%** (75%)
7’	[0 10)	[-15 0)	**75%** (60%)
	[10 20)	[-15 0)	**77%** (73%)
	[20 30)	[-15 0)	**75%** (74%)
	[0 10)	[0 15)	73% (**76%**)
	[10 20)	[0 15)	**72%** (68%)
	[20 30)	[0 15)	**75%** (69%)
	[0 10)	[15 30)	**74%** (73%)
	[10 20)	[15 30)	72% (**75%**)
	[20 30)	[15 30)	**73%** (71%)

TABLE VI

PERCENTAGE OF SUCCESS (TO PLACE THE BALL WITHIN $6''$ OF TARGET) OF CHILDREN TRAINED BY ROBOT VERSUS HUMAN TRAINER

Distance of box from the Robot arm	***Θ* in [0º, 10⁰), *Φ* in [-15º, 0⁰),**	
	Ball within $6''$ from target	
	Robot	**Human trainer**
4’	92%	91%
5’	86%	82%
6’	79%	76%
7’	75%	69%

TABLE VII

PERCENTAGE OF CHILDREN WHO RATED THE EXPERIENCE OF LEARNING FROM A ROBOT FROM LOW (1) TO HIGH (3) IN TERMS OF MOTIVATION/FUN/DIFFICULTY

	Human Trainer			**Robot Trainer**		
Rating	**1**	**2**	**3**	**1**	**2**	**3**
Motivation	18.86	**60.13**	21.01	15.03	13.96	**71.01**
Difficulty	**51.97**	31.01	17.07	30.98	**51.03**	17.99
Fun	**49.33**	30.12	20.55	4.05	8.8	**87.15**

V. CONCLUSIONS

The paper proposes a novel approach to train children autonomous ball-throwing towards a given target with the help of a pre-trained robotic manipulator. The most important aspect of the paper is to develop automatic learning skill of the robot from the acquired junction coordinates of the expert during ball throwing experiments. A learning automaton is used to acquire parameters of the successful ball-throws for given position and orientation of the goals. After the automaton converges, the acquired learning skill is transferred to a robot arm for throwing balls to a given bin at a fixed distance with adjusted pan and tilt angles of the top surface. This is undertaken in the planning phase of the robot arm.

Experiments undertaken reveal that the robotic manipulator offers better success rate with reference to human trainers, when the performance is measured at the children end. The ERD/ERS and ErrP classifier performance was also analyzed to test their consistency using specificity and sensitivity analysis. The analysis reveals a reasonably good classifier performance.

REFERENCES

[1] D. Marshall, D. Coyle, S. Wilson, and M. Callaghan, “Games, Game Play, and BCI: The State of The Art,” *IEEE Transactions on Computational Intelligence and AI in Games*, vol. 5, no. 2, pp. 82–99, 2013.

[2] C. Guger, B. Z. Allison, and M. A. Lebedev, “Recent Advances in Brain-Computer Interface Research—A Summary of the BCI Award 2016 and BCI Research Trends,” in *Brain-Computer Interface Research*. Springer, 2017, pp. 127–134.

[3] H. Gurkok, A. Nijholt, and M. Poel, “Brain-Computer Interface Games:¨ Towards a Framework,” in *International Conference on Entertainment Computing*. Springer, 2012, pp. 373–380.

[4] B. Kerous, F. Skola, and F. Liarokapis, “EEG-Based BCI and Video Games: A Progress Report,” *Virtual Reality*, vol. 22, no. 2, pp. 119–135, 2018.

[5] L. Bonnet, F. Lotte, and A. Lecuyer, “Two Brains, One Game: Design and Evaluation of a Multiuser BCI Video Game based on Motor Imagery,” *IEEE Transactions on Computational Intelligence and AI in games*, vol. 5, no. 2, pp. 185–198, 2013.

[6] D. Coyle, J. Garcia, A. R. Satti, and T. M. McGinnity, “EEG-based Continuous Control of a Game Using a 3 Channel Motor Imagery BCI: BCI Game,” in *2011 IEEE Symposium on Computational Intelligence, Cognitive Algorithms, Mind, and Brain (CCMB)*. IEEE, 2011, pp. 1–7.

[7] L. Bi, X.-A. Fan, and Y. Liu, “EEG-based Brain-Controlled Mobile Robots: A Survey,” *IEEE transactions on human-machine systems*, vol. 43, no. 2, pp. 161–176, 2013.

[8] G. Pfurtscheller, “Functional Brain Imaging Based on ERD/ERS,” *Vision research*, vol. 41, no. 10-11, pp. 1257–1260, 2001.

[9] G. Pfurtscheller, C. Neuper, D. Flotzinger, and M. Pregenzer, “EEG-based Discrimination Between Imagination of Right and Left Hand Movement,” *Electroencephalography and clinical Neurophysiology*, vol. 103, no. 6, pp. 642–651, 1997.

[10] Y. Li, J. Long, T. Yu, Z. Yu, C. Wang, H. Zhang, and C. Guan, “An EEG based BCI system for 2-d cursor Control by Combining Mu/Beta Rhythm and P300 Potential,” *IEEE Transactions on Biomedical Engineering*, vol. 57, no. 10, pp. 2495–2505, 2010.

[11] W. Caesarendra, "A Method to Extract P300 EEG Signal Feature Using Independent Component Analysis (ICA) for Lie Detection," *JEMMME (Journal of Energy, Mechanical, Material, and Manufacturing Engineering)*, vol. 2, no. 1, pp. 9–16, 2017.

[12] G. R. Muller-Putz, R. Scherer, C. Brauneis, and G. Pfurtscheller,¨ "Steady-State Visual Evoked Potential (SSVEP)-based Communication: Impact of Harmonic Frequency Components," *Journal of Neural Engineering*, vol. 2, no. 4, p. 123, 2005.

[13] O. Friman, I. Volosyak, and A. Graser, "Multiple Channel Detection of Steady-State Visual Evoked Potentials for Brain-Computer Interfaces," *IEEE Transactions on Biomedical Engineering*, vol. 54, no. 4, pp. 742–750, 2007.

[14] G. Pfurtscheller, B. Z. Allison, G. Bauernfeind, C. Brunner, T. Solis Escalante, R. Scherer, T. O. Zander, G. Mueller-Putz, C. Neuper, and N. Birbaumer, "The Hybrid BCI," *Frontiers in Neuroscience*, vol. 4, p. 3, 2010.

[15] T. Kayagil, O. Bai, P. Lin, S. Furlani, S. Vorbach, and M. Hallett, "Binary EEG Control for Two-Dimensional Cursor Movement: An online Approach," *IEEE/ICME International Conference on Complex Medical Engineering*. IEEE, 2007, pp. 1542–1545.

[16] M. Krauledat, K. Grzeska, M. Sagebaum, B. Blankertz, C. Vidaurre, K.-R. Muller, and M. Schr¨oder, "Playing Pinball with Non-Invasive BCI,"" in *Advances in neural information processing systems*, 2009, pp. 1641–1648.

[17] E. C. Lalor, S. P. Kelly, C. Finucane, R. Burke, R. Smith, R. B. Reilly, and G. Mcdarby, "Steady-state VEP-Based Brain-Computer Interface Control in an Immersive 3d Gaming Environment," *EURASIP Journal on Advances in Signal Processing*, vol. 2005, no. 19, p. 706906, 2005.

[18] N. Chumerin, N. V. Manyakov, M. van Vliet, A. Robben, A. Combaz, and M. M. Van Hulle, "Steady-State Visual Evoked Potential-Based Computer Gaming on a Consumer-Grade EEG Device," *IEEE transactions on computational intelligence and ai in games*, vol. 5, no. 2, pp. 100– 110, 2012.

[19] I. Martisius and R. Damaˇ seviˇ cius, "A Prototype SSVEP based Real-Time BCI gaming system," *Computational intelligence and neuroscience*, vol. 2016, 2016.

[20] C. Muhl, H. Gurkok, D. P.-O. Bos, M. Thurlings, L. Scherffig,¨ M. Duvinage, A. Elbakyan, S. Kang, M. Poel, and D. Heylen, "Bacteria Hunt: A multimodal, multi-paradigm BCI game," in *Fifth International Summer Workshop on Multimodal Interfaces*, 2009, pp. 41–62.

[21] M. Congedo, M. Goyat, N. Tarrin, G. Ionescu, L. Varnet, B. Rivet, R. Phlypo, N. Jrad, M. Acquadro, and C. Jutten, " Brain Invaders: A Prototype of an Open-Source P300-based Video Game Working with the Openvibe Platform," 2011.

[22] J. E. Munoz, R. Chavarriaga, and D. S. Lopez, "Application of Hybrid-BCI and Exergames for Balance Rehabilitation after Stroke," in *Proceedings of the 11th Conference on Advances in Computer Entertainment Technology*, 2014, pp. 1–4.

[23] E. M. Holz, J. Hohne, P. Staiger-S¨ alzer, M. Tangermann, and A. K¨ ubler,¨ "Brain–Computer Interface controlled gaming: Evaluation of Usability by Severely Motor Restricted End-Users," *Artificial intelligence in medicine*, vol. 59, no. 2, pp. 111–120, 2013.

[24] D. A. Rohani, H. B. Sorensen, and S. Puthusserypady, "Brain-Computer Interface Using P300 and Virtual Reality: A Gaming Approach for Treating ADHD," in *2014 36th Annual International Conference of the IEEE Engineering in Medicine and Biology Society*. IEEE, 2014, pp. 3606– 3609.

[25] S. Bhattacharyya, A. Konar, and D. Tibarewala, "Motor Imagery, P300 and Error-Related EEG-based Robot Arm Movement Control for Rehabilitation Purpose," *Medical & biological engineering & computing*, vol. 52, no. 12, pp. 1007–1017, 2014.

[26] A. Khasnobish, A. Konar, D. N. Tibarewala, and A. K. Nagar, "Bypassing the Natural Visual-Motor Pathway to Execute Complex Movement Related Tasks using interval Type-2 Fuzzy Sets," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 25, no. 1, pp. 91– 105, 2016.

[27] K. S. Fu, R. Gonzalez, and C. Lee, "G. robotics: Control, sensing, vision and intelligence," *New York: McGraw-Hill*, pp. 78–82, 1987.

[28] S. Saha, S. Datta, A. Konar, and R. Janarthanan, "A Study on Emotion Recognition from Body Gestures using Kinect Sensor," in *2014 International Conference on Communication and Signal Processing*. IEEE, 2014, pp. 056–060.

[29] S. Bhattacharyya, A. Konar, and D. Tibarewala, "Motor Imagery and Error-related Potential Induced Position Control of a Robotic Arm," *IEEE/CAA Journal of AutomaticaSinica*, vol. 4, no. 4, pp. 639–650, 2017.

[30] K. S. Narendra and M. A. Thathachar, *Learning automata: an introduction*. Courier corporation, 2012.

[31] C. Liang, "Recreational Devices Controlled using an SSVEP-based Brain Computer Interface (BCI)," *Innovation, Communication and Engineering*, p. 175, 2013.

[32] A. Konar, *Computational intelligence: Principles, Techniques and Applications*. Springer Science & Business Media, 2006.

[33] T. M. Mitchell *et al.*, "Machine learning. 1997," *Burr Ridge, IL: McGraw Hill*, vol. 45, no. 37, pp. 870–877, 1997.

[34] R. W. Homan, J. Herman, and P. Purdy, "Cerebral Location of International 10–20 System Electrode Placement," *Electroencephalography and Clinical Neurophysiology*, vol. 66, no. 4, pp. 376–382, 1987.

[35] S. Balakrishnama and A. Ganapathiraju, "Linear Discriminant Analysis-A Brief Tutorial," *Institute for Signal and information Processing*, vol. 18, pp. 1–8, 1998.

[36] K. S. Kim, H. H. Choi, C. S. Moon, and C. W. Mun, "Comparison of k-Nearest Neighbor, Quadratic Discriminant and Linear Discriminant Analysis in Classification of Electromyogram Signals based on the Wrist-Motion Directions," *Current Applied Physics*, vol. 11, no. 3, pp. 740–745, 2011.

[37] R. Kar, A. Konar, and A. Chakraborty, "EEG-Analysis for the Detection of True Emotion or Pretension," in *Handbook of Research on Synthesizing Human Emotion in Intelligent Systems and Robotics*. IGI Global, 2015, pp. 283–298.

[38] V. Jakkula, "Tutorial on support vector machine (SVM)," *School of EECS, Washington State University*, vol. 37, 2006.

[39] I. Rish*et al.*, "An Empirical Study of the Naive Bayes Classifier," in *IJCAI 2001 Workshop on Empirical Methods in Artificial Intelligence*, vol. 3, no. 22, 2001, pp. 41–46.

[40] Y. Zhang, P. Sun, Y. Yin, L. Lin, and X. Wang, "Human-like Autonomous Vehicle Speed Control by Deep Reinforcement Learning with Double Q-learning," in *2018 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 2018, pp. 1251–1256.

[41] Z. Zhang, Z. Pan, and M. J. Kochenderfer, "Weighted Double Q-learning." in *IJCAI*, 2017, pp. 3455–3461.

[42] M. Hessel, J. Modayil, H. Van Hasselt, T. Schaul, G. Ostrovski, W. Dabney, D. Horgan, B. Piot, M. Azar, and D. Silver, "Rainbow: Combining improvements in deep reinforcement learning," in *Thirty Second AAAI Conference on Artificial Intelligence*, 2018.

[43] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski*et al.*, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, pp. 529–533, 2015.

[44] P. W. Ferrez and J. d. R. Millan, "Error-related EEG potentials generated´ during simulated brain–computer interaction," *IEEE Transactions on Biomedical Engineering*, vol. 55, no. 3, pp. 923–929, 2008.

[45] H. Zhao and H. Liu. "Multiple classifiers fusion and CNN feature extraction for handwritten digits recognition." *Granular Computing*, vol. 5, no. 3, pp. 411-418, 2020.

[46] J. Maroco, D. Silva, A. Rodrigues, M. Guerreiro, I. Santana, and A. de Mendonça. "Data mining methods in the prediction of Dementia: A real-data comparison of the accuracy, sensitivity and specificity of linear discriminant analysis, logistic regression, neural networks, support vector machines, classification trees and random forests." *BMC research notes*, vol. 4, no. 1,pp. 1-14, 2011.

[47] K. Tanaka, T. Kurita, F. Meyer, L. Berthouze and T. Kawabe, "Tepwise Feature Selection by Cross Validation for EEG-based Brain Computer Interface", in *Int. Jt. Conf. on Neural Network*, 2006.

[48] A. Hyvärinen, J. Hurri, and P. O. Hoyer. "Independent component analysis." In *Natural Image Statistics*, pp. 151-175. Springer, London, 2009.

[49] Databases used to test the performance of the paper: EEG-Induced Autonomous Game-Teaching of a Robot Arm by Human Trainers using Reinforcement Learning., AI lab., ETCE Dept., Jadavpur University. See: https://drive.google.com/drive/folders/1AzgZqH8tgmWYHlMNUIYU3J4IajLtII_i?usp=sharing.