Information Sciences 283 (2014) 288–303

Contents lists available at ScienceDirect

Information Sciences

journal homepage: www.elsevier.com/locate/ins

A developmental approach to robotic pointing via human–robot interaction

Fei Chao [a], Zhengshuai Wang [a], Changjing Shang [b], Qinggang Meng [c], Min Jiang [a], Changle Zhou [a], Qiang Shen [b,*]

[a] Cognitive Science Department, Fujian Provincial Key Laboratory of Brain-like Intelligent Systems, Xiamen University, Xiamen 361005, China
[b] Department of Computer Science, Institute of Mathematics, Physics and Computer Science, Aberystwyth University, Aberystwyth SY23 3DB, UK
[c] Department of Computer Science, Loughborough University, Loughborough LE11 3TU, UK

ARTICLE INFO

Article history:
Received 12 June 2013
Received in revised form 16 March 2014
Accepted 26 March 2014
Available online 3 April 2014

Keywords:
Developmental robotics
Robotic pointing
Human robot interaction

ABSTRACT

The ability of pointing is recognised as an essential skill of a robot in its communication and social interaction. This paper introduces a developmental learning approach to robotic pointing, by exploiting the interactions between a human and a robot. The approach is inspired through observing the process of human infant development. It works by first applying a reinforcement learning algorithm to guide the robot to create attempt movements towards a salient object that is out of the robot's initial reachable space. Through such movements, a human demonstrator is able to understand the robot desires to touch the target and consequently, to assist the robot to eventually reach the object successfully. The human–robot interaction helps establish the understanding of pointing gestures in the perception of both the human and the robot. From this, the robot can collect the successful pointing gestures in an effort to learn how to interact with humans. Developmental constraints are utilised to drive the entire learning procedure. The work is supported by experimental evaluation, demonstrating that the proposed approach can lead the robot to gradually gain the desirable pointing ability. It also allows that the resulting robot system exhibits similar developmental progress and features as with human infants.

1. Introduction

A major challenge in modern robotics is to liberate robots from controlled industrial settings, allowing them to interact with humans and the changing environments in the real-world. For instance, if we require robots to assist the elderly, or to replace humans for the accomplishment of dangerous tasks, robots should possess certain social interaction and communication abilities. However, robots' internal representations are typically foreknown and are often designed for specific environments, robots supported by traditional artificial intelligent algorithms might not work appropriately under unknown environments. An important issue is therefore that robots should be able to create their own internal representations [31]. For this, robotic scientists have applied developmental psychology and neuroscience to produce a novel research topic, "Developmental Robotics". In the field of developmental robotics, it is assumed that a robot system is not programmed for

* Corresponding author. Tel.: +44 (0) 1970 621825.
E-mail address: qqs@aber.ac.uk (Q. Shen).

http://dx.doi.org/10.1016/j.ins.2014.03.104
0020-0255/

specific and fixed tasks, but rather programmed to develop and learn new behavioural and cognitive competence and skills autonomously [2].

Our particular interest is in the control and coordination of a robotic hand-eye system in order to achieve mastery of its local, egocentric space and to perform pointing tasks. The need for this is just as that in an infant's development where the infant's pointing is the first attempt to build a platform of pre-linguistic communication [41,48]. In addition, an infant's early linguistic skills are built on the pointing platform [3,17,27,44]. Therefore, to implement social interaction skills in a robot, it is necessary to establish a developmental learning approach to facilitate the robot with the pointing ability, and also to understand the internal incremental processes during the robot's development. This paper aims to build a robotic learning system that leads a robot to developing from basic reaching movements to possessing the ability of pointing.

In robotics, learning to point may be implemented by a variety of techniques. Much work focuses on developing a robot's ability to respond to a pointing gesture demonstrated by a human. The objective of this work is to enable a robot to generate its pointing gestures. There have been relevant proposals made in the literature. For example, Marjanovic et al. [32] introduced a robotic system, through the use of a RBF neural network, which learns to point towards visual targets. Hafner and Kaplan [14] applied a multi-layer-perceptron network to train a robot to interpret the pointing gestures of another robot. Doniec et al. emphasised that the task of pointing accelerates the learning of robotic joint attention, and created Jacobian matrices to deal with the pointing gestures [11]. Shademan et al. [39] made use of a locally least squares algorithm, based on a Jacobian estimation method, to build a robotic visual-motor learning system. Sheldon and Lee [40] described a developmental approach for implementing a pointing learning framework by creating a schema mechanism. Qu and Grupen also proposed a novel learning framework, in which their robot can learn pointing gestures and manual skills simultaneously [37]. Liu [28] and Liu et al. [29] proposed a fuzzy qualitative framework to describe the robotic arm's sensory-motor problem. Also, Lemme et al. defined two types of robotic pointing pattern using different learning machines to imitate how to point [26]. In their research, Jamone et al. created an interactive learning strategy to build a reachable map that can handle reaching rather than pointing movements [18,19]. It is their latest work that has inspired us to develop a mechanism that helps a robot to generate both reaching and pointing movements.

In the aforementioned robotic pointing research, human robot interactions are not taken into account. However, a human infant gains the ability to point through interactions with his/her parents or babysitters. Over a course of parent-infant interactions, an infant learns how to point. This observation indicates that the learning procedure developed for robotic pointing may also benefit from a course of human–robot interaction. Indeed, human–robot interaction has been considered as an effective learning method to transfer skills and knowledge from human beings to robots [1,4,9,38,50]. The present work follows this theme of thought.

Developmental approach offers the potential to drive a robot learning from the basic reaching movements to the pointing behaviours. Compared with the "Deep Learning" as described in [13], the developmental approach exhibits staged incremental learning ability, which helps reduce the robot's learning complexity. From this viewpoint, we bring more infant developmental features into our robotic learning system. In particular, this paper presents a novel approach emphasising that the robot should mimic the procedure in which a human infant learns to point. This procedure is summarised as follows: (1) trying to reach objects; (2) failing to reach objects; (3) interacting with an adult human; and (4) ultimately knowing how to point. Based on this procedure, we present an implemented human–robot interactive system that reflects the following three desirable features:

- that pointing gestures are mutually comprehended between the human and the robot;
- that infant brain developmental characteristics are applied to create a robotic developmental learning system; and
- that a developmental learning algorithm – "Lifting Constraints, Act and Saturate (LCAS)" – is employed to drive the robot to learn.

This approach reduces the complexity of building human–robot communications, thereby, injecting more psychological and biological inspiration into robotics, and enabling robots to exhibit higher autonomous capability.

The rest of this paper is organised as follows. Section 2 briefly introduces the configuration of robotic pointing gestures. Section 3 describes an experimental robot system and the main implementation issues involved. Section 4 presents the experimental results and discusses their implications. Section 5 concludes the paper and points out directions for further research.

2. Robotic pointing and related work

Recent robotic pointing research reveals no universally accepted definition of robotic pointing gestures. Much of the current work projects the three-dimensional position of an unreachable object into the robot's two-dimensional visual position. In particular, if a robot's end-effectors block the object within the robot's view, the gesture is generally regarded as a pointing gesture [26,40]. Also, the pointing gesture may be defined as the robot's end-effector in the closest position to the object [7]. Other work only indicates that a robot's gaze is linked to an arm gesture, called a pointing gesture [37]. However, we suggest that a pointing gesture relies on the robot operator's understanding also; therefore, only the robot operator can determine the pointing criteria. In light of this understanding, Fig. 1 defines the pointing gesture as to be used hereafter.

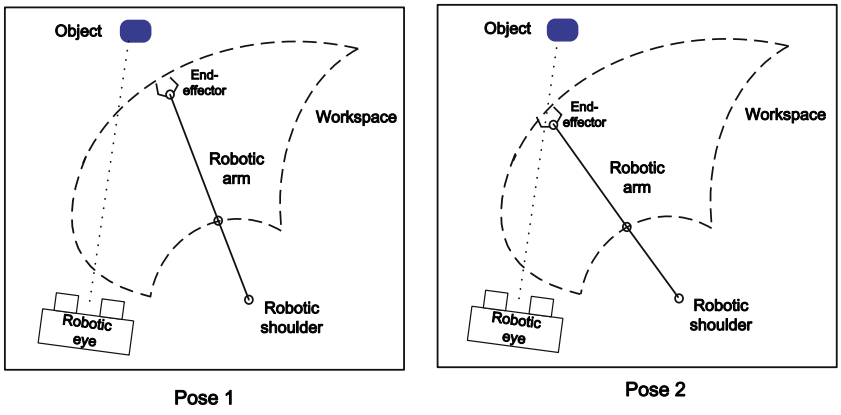

Fig. 1. The robotic pointing gesture.

This figure illustrates two single-arm robot's pointing gestures. The area surrounded by the dashed lines is the arm's workspace, within which the arm can touch any object. A vision system is mounted in the lower left corner as the robotic eye. An object is placed out of the robotic workspace. Labels "Pose 1" and "Pose 2" indicate two gestures of the robot's arm. The end-effector of the arm in Pose 1 is the closest to the object. Many existing approaches define the hand gesture of Pose 1 as pointing at the object. However, the objective of a robot's pointing at an object requires a human or another intelligent agent to understand that the robot is going to capture the object. Therefore, the closest position may supply insufficient information for the human/agent to understand the robot's desire to capture the object. Instead, the human/agent usually makes the judgment through a pointing gesture of the robot.

In this paper, we define the pointing gesture as a linear alignment among the object, the robot's end-effector, and the robot's vision system. The linear alignment means the positions of the object, the end-effector and the vision system, are required to be in a straight line. For instance, in Fig. 1, the "Pose 2" gesture shows such a pointing position. Between the two poses of this figure, we prefer to use "Pose 2" as our robotic pointing gesture because it allows more information to be shared when we implement the robotic learning system. Also, the gesture of the robot's arm needs to be straight between the upper limb and the low limb. This setup is based on the findings of the human infant development. The straight elbow of the infant is the result of the failed reaching movements, which demands the arm to be straight. In summary, the robotic pointing gesture is defined as follows: The robot's arm is straight, with the object, the end-effector, and the vision system all being in a straight line.

Note that as the pointing gesture definitions vary, the recent methods to obtain pointing gestures also vary. When an artificial neural network is used for implementation, its generalisation property can help to produce the reaching gestures, which may point at the object [7]. However, in certain situations, the trained network may not produce very accurate pointing gestures. Several research used visual servo technology to reduce the distance between the robot's end-effector and the object. The visual servo repeatedly works until the end-effector covers the object [26]. Nevertheless, we believe that the robot should use different types of trial movement in its attempt to reach an object that is out of the arm's reachable space, and use these trial movements to allow a human demonstrator to realise that it intends to capture the object. To support this work, we review the typical infant development process in order to extract important features that may assist our robot to obtain pointing gestures.

2.1. Inspirations from infant development

Developmental psychological findings indicate that pointing emergence has been recognised as the first step of a human infant's social cognitive development and the foundation of an infant's early linguistic skills [44]. Therefore, in order to transfer a human pointing pattern to robots, it is necessary to know how a human infant develops his/her pointing ability. Vygotsky pointed out that pointing motions develop out of unsuccessful capture behaviours in which the infant attempts to reach for a certain object which is beyond his/her hand reach range [46]. Although an infant's fingers may make grasping movements, his/her hand still remains in the air. The infant's parent realises this as the infant pointing at a desired object, and then fetches the object for the infant. Thus, with the act of reaching for a distant object, a new meaning is given to the attempted gesture. In this act, the parent infers the entire social and behavioural meaning.

As a general observation from the above scenario, an infant often makes a real attempt to reach the object, but fails. Through the parent's actions, the infant learns to link the unsuccessful reaching movement with the objective situation, eventually understanding how to use the movement to get salient objects. Therefore, the failure of reaching for a target far away is an important process of learning to point; without the interaction with the infant's parents, the infant cannot establish the shared attention between the infant and the parents. We summarise such an infant development process of pointing in the following four sequential steps, which will be applied to our robot system: (1) trying to reach objects; (2)

failing to reach objects; (3) interacting with an adult human; and (4) ultimately knowing how to point. What is the driving force that leads our robot to shift from one step to the next? We use our previous work [25], introduced in the following subsection, as the driving force.

2.2. Developmental constraints and LCAS algorithm

The infant's 4-step procedure of learning to point outlined above is inspirational. However, the internal forces that drive the infant's development from Step 1 to Step 4 need to be identified. The intention is to implement a robotic developmental driving mechanism which mimics the infant's internal developmental force. Our robotic system uses such a mechanism to decide when to develop new abilities.

Developmental psychology presents evidence that lifting constraints can lead infants to progressing from a certain competence level to a new, even more complicated competence level [35]. This is because, under constrained environments, infants cannot immediately sense information that is too complex. Infants sense and learn to process simple information only. Thus, developmental constraints reduce the learning complexity [2,31]. Over the course of their development, infants overcome more constraints and gradually become capable of handling more complicated work. Based on this finding, our previous research [25] has led to a type of (implemented) driving mechanism that has been referred to as the "Lifting Constraints, Act and Saturate" (LCAS) algorithm. This algorithm follows the procedure in which a robot starts to search any novel stimuli to learn under fully constrained conditions. When the robot's learning system saturation rate (Sat) becomes stable, a new constraint is assigned to the robot; after that, the robot repeats the search for novel stimuli in an attempt to learn under this new condition. When all of the constrained conditions have been experienced, the robot will then have learned a number of new abilities.

To facilitate the understanding of LCAS, Algorithm 1 outlines the constraint lifting procedure in pseudo code. In this algorithm, i is the current learning time under the constrained environment. $Sat(i)$ is the saturation value during the ith learning iteration. The LCAS algorithm has been applied in several developmental models successfully [5,6,8,20–22]. In this work, again, we apply the LCAS algorithm to implement a developmental mechanism that drives our robot to progress from Step 1 to Step 4.

Algorithm 1. Integrated Learning Algorithm

```
1: while not all constraints are released do
2:   for i = 0 to n do
3:     if Sat(i) is TRUE then
4:       quit this for-loop and release a new constraint;
5:     else
6:       repeat doing the learning process within this for-loop (Lines 2–8);
7:     end if
8:   end for
9: end while
```

In summary, in this section we have (1) defined a robotic pointing gesture, (2) reviewed the infant developmental procedure, (3) created the "trying to reach objects-failing to reach objects-interacting with an adult human-knowing how to point" procedure, and (4) introduced a developmental approach for our robotic system to learn to point. The next section introduces the implementation of this system.

3. Proposed method

3.1. The robotic system

The laboratory robot that is employed to illustrate the proposed work is shown in Fig. 2A, consisting of one manipulator and a camera system configured in a manner similar to the spatial arrangement of an infant's arms and head: the arm is mounted on a vertical backplane and operates in the horizontal plane a few centimetres above the work space. The manipulator contains 6 degrees-of-freedom, but only 2 joints are used. The rest of the joints are set to be fixed in order to keep the robotic arm moving stably and safely. The upper and lower limbs are labeled as L_1 and L_2, respectively. The length of L_1 is 14 mm and that of L_2 is 20 mm. The arm contains two joints labeled J_1 and J_2 (see Fig. 2B). The limitation of each joint is given by $J_1 \in [30, 110], J_2 \in [60, 180]$.

A colour camera (see Fig. 2C) is mounted above and looks down on the work area. In the experiment, the camera is fixed. Fig. 2B and D illustrate a vertical view of the workspace, a white table. An orange ball, representing a fingertip, is mounted on the arm's end-point. The rest of the arm is ignored by the image processing software. Only the fingertip's movement can be

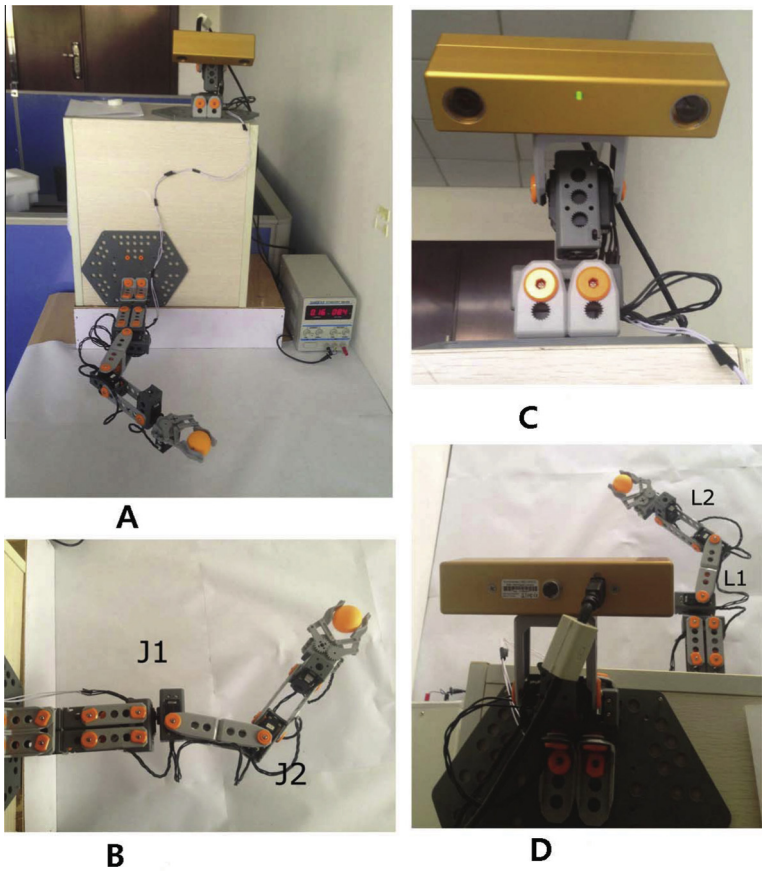

Fig. 2. Experimental robotic system.

detected by the vision system. The setup is arranged in this way to simplify the subsequent image processing task, especially the task of object detection.

3.2. The human–robot interaction procedure

This work focuses on building the robot's learning procedure, by which the robot learns to transfer visual stimuli to its arm in joint space. The stimuli may be caused by the robot's hand or objects. Based on the 4-step procedure outlined earlier, which is inspired by human infant development in Section 2.1, the robot's learning scheme is devised as follows:

1. The robot gathers image input from its Robotic Vision Module. When a target appears within the vision system's view, the robot sends the target position $x_o,\ y_o$ to the Hand Sensorimotor Module.
2. The Hand Sensorimotor Module uses $x_o,\ y_o$ to drive the arm to capture the object.
3. After the movement, the Coverable Checking Module starts to check whether the object is captured. If "yes," the robot completes the grasping movement. If "no," the target is out of the arm's reaching range, and the robot proceeds to the Attempt Module.
4. The Attempt Module generates a new motor value $j_1,\ j_2$ to the Hand Sensorimotor Module.
5. A new gesture is then generated by the Hand Sensorimotor Module to reach the target.
6. At this moment, a human demonstrator will join in the experiment, attempting to understand the robot's intentions. When the demonstrator realises that the robot wants to capture the object, the demonstrator will move the object into the robot's reaching range.
7. After the human's interaction, the robot invokes the Robotic Vision Module to detect the object's position.
8. The Coverable Checking Module checks whether it can capture the target. If not, the robot (and the procedure) returns to Step 4, the Attempt Module generates another new gesture. If yes, the robot completes the capturing movement.
9. The robot gathers the target's previous position $x_o,\ y_o$ and the arm's joint values $j_1,\ j_2$ as a training pattern, and pushes the pattern into a temporal memory. Then, the robot invokes the Network Training Module to develop successful pointing gestures. Note that the demonstrator's interaction is the supervisor signal to decide when to start the training.

As the robot completes a successful capture movement, the demonstrator places an object in a different position; thus, the robot will start the learning procedure again in an effort to point at the object. If the demonstrator places the object inside the robot's workspace, the robot will be able to directly reach the object, rather than to point to the object. In this case, the robot must develop the reaching ability very precisely before the robot starts to establish the pointing behaviour; otherwise, the robot will exhibit incorrect reaching movements as it learns to point. The detailed implementation of each module as outline above is described in the following subsections.

3.2.1. Robotic Vision Module

A colour camera is used to capture workspace images, and an image processing algorithm is utilised to implement the vision system. The colour space applicable to this system is HSI, but only the hue component is used. A pink ball is the target towards which our robotic system needs to point. The robot's grabber is orange. Such a setup simplifies the image processing work. In order to detect the fingertip in the images captured by the cameras, we first need to convert a given image from the RGB colour space to the HSV colour space, and then, we use the popular foreground generation method to generate the finger's histogram. In the system, if a pixel's hue value falls into the range of $Hue[245, 253]$, we set this pixel as a candidate target, and assign 255 to this pixel; otherwise, we assign 0. The pink ball's colour range is $Hue[30, 42]$. After the colour conversion, a binary image is generated. All detected target pixels are clustered into a number of regions, and the largest region is then regarded as the fingertip or the target object.

Fig. 3 shows examples of the original captured images and the processed images. The upper left hand picture contains the robotic arm only. We convert it into a binary picture and use a square to label the hand's position $P_{hand}(x, y)$ as shown in the upper right hand picture. The lower left and lower right hand pictures show the situation where an object is placed into the workspace. Both the hand and the object are highlighted; another square is used to label the object to tell the robot that it is the target rather than is own hand; in addition, the centre of the square is used as the position of the target $P_{target}(x, y)$.

3.2.2. Coverable Checking Module

This module works simultaneously with the Robotic Vision Module, receiving $P_{target}(x, y)$ and $P_{hand}(x, y)$ values from the latter. Our robotic system may encounter three situations after each movement: (1) the hand captures the object; (2) the hand is very close to the object; and (3) the hand is not close to the object. In this paper, Situations 1 and 2 regard the object as coverable; Situation 3 regards the object as un-coverable. This module gives "TRUE" output when the object is coverable, and gives "FALSE" when the object is un-coverable. For Situations 2 and 3, we calculate the Cartesian distance d between $P_{target}(x, y)$ and $P_{hand}(x, y)$, and set a threshold δ_{cover}. If $d < \delta_{cover}$, the object is coverable; otherwise, it is un-coverable. However, for Situation 1, the hand has captured the object; in terms of our robot's setup, the object cannot be detected by the vision module. In this case, we compare the vision module's outputs before and after a movement. If only $P_{hand}(x, y)$ remains after a movement, the object is captured, and the Checking Module also gives a "TRUE" output. In this work, δ_{cover} is empirically set to 10 pixels.

3.2.3. Attempt Module

The Attempt Module guides the robot's arm in an attempt to reach the object, which is out of the arm's reachable area. When the Hand Sensorimotor Module receives the object's position in the robot's visual space, the Hand Sensorimotor

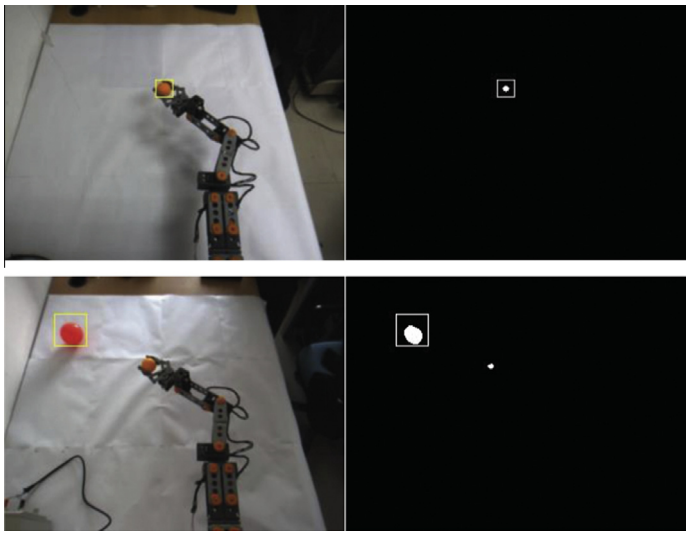

Fig. 3. Output of image processing module.

Module is able to produce a pair of motor values based on the module's generalisation property. Unfortunately, the motor values cannot normally drive the arm to the correct pointing gesture. When the robot observes no human interaction, it will consider various strategies to move its hand as close as possible to the object. Hence, a type of the reinforcement algorithm (Q-learning) is used to guide the robot to reach the target. The reasons for using the reinforcement learning algorithm are due to the recognition of its biological plausibility and its wide use in robotic learning systems and other intelligent agents (e.g., [10,12,15,16,45,47,49]).

An agent acting in an environment is described as a tuple $\langle S, A, P, R\rangle$, where (1) S is the agent's state set; (2) A is the finite action set; (3) $P(s_{t+1}|s_t,\ a)$ is the transition model that describes the probability of converting s_t to s_{t+1}, after an action a, with $s_t,\ s_{t+1} \in S$ and $a \in A$; and (4) $R(s_t, a, s_{t+1})$ is the reward of action a when the agent transits from s_t to s_{t+1}. The goal of the Q-learning algorithm is to find a policy $\pi : S \mapsto A$, so that the agent's cumulative rewards are maximised. If the P and R are predefined, the optimal control policy for the agent can be determined efficiently using techniques such as value iteration [12]. Eq. (1) presents the regular Q-learning algorithm:

$$Q_{t+1}(s_t, a_t) = (1-\eta)Q_t(s_t, a_t) + \eta[r_t + \gamma \max_{a_{t+1}} Q_t(s_{t+1}, a_{t+1})] \tag{1}$$

where s_t is the hand position relative to the target; a_t is an executable action of the robotic arm; r_t is a reward value obtained from Eq. (3); η is the learning speed control parameter; and γ is the discount parameter.

Early created Q values cannot be used as further movement candidates because these Q values are not optimised. The Boltzmann exploration [43] is applied to the Q-learning algorithm. At each state, the robot has a list of actions $(a_{p1}, a_{p2}, \ldots, a_{pn})$ to choose from. Eq. (2) shows the probability for action a to be chosen at s_t:

$$p(s_t, a_t, s_{t+1}, a_{t+1}) = \frac{e^{\frac{Q(s_0,a_t)}{T}}}{\sum_{n=0}^{t} e^{\frac{Q(s_0,a_n)}{T}}} \tag{2}$$

where T is a positive parameter called temperature. With a high temperature ($T \to \infty$), all actions have almost the same probability for selection. If $T \to 0$, with this exploration mechanism, it is still possible to select actions with smaller Q values so that the action space can be explored. Eq. (3) is the reward function based on the distance between the object and the robot's hand:

$$r = \frac{1}{\sqrt{(x_{hand} - x_{target})^2 + (y_{hand} - y_{target})^2}} \tag{3}$$

where x_{hand} and y_{hand} denote the hand position within the camera view after each movement; and x_{target} and y_{target} denote the target position within the camera view. The significance of this equation is that if a movement brings the hand closer to the target, this movement will receive a higher reward. Note that Eq. (3) is only applied before the demonstrator's interaction movements; when the robot is touching the object, this equation will not be applied.

$Q(s_0, a_t)$ in Eq. (2) (aka. Q table) contains the robotic movement strategies. Each motor has two types of action $(+, -) : +$ means the arm joint turns clockwise; and $-$ means the arm joint turns anticlockwise. In Eq. (1), η denotes the Q-learning systems's learning speed. In the experiment, we follow the popular approach in the literature, by empirically choosing $\eta = 0.9$ and $\gamma = 0.1$.

Because the reward function relies merely on the distance between the hand and the object, the Q-learning algorithm guides the robot's arm to the closest position to the object. However, this position may not reflect the pointing gesture. In this case, after the robot has attempted several movements, a large value (say 100) is assigned to T in Eq. (2) by trial and error. Thus, the robot tends to generate random movements, with the temperature recognised as a developmental constraint. First the robot performs under low temperature until it cannot find any novel stimuli; then, the robot receives a high temperature and must act again under this new condition. The decision to change the value of T guides the robot's developmental mechanism, which is to be described in Section 3.3.

3.2.4. Robotic Hand-eye Sensorimotor Module

This module drives the robot's arm to respond to the visual stimuli caused by novel objects being placed within the arm's range. Thus, the module mainly handles the robotic reaching movements. The robotic reaching is also an important topic in developmental robotics [24]. A robot can gradually gain this reaching ability through the cognitive course of the robot's hand-eye coordination, without human-intervention during the learning process of reaching. In this work, the robot's reaching ability is assumed to be able to be perfectly trained before it learns to point, reflecting the nature of human infant behaviour.

A minimal resource allocating network (MRAN) is applied to implement the transformation from the visual space into the hand motor space. During the training phase for learning the reaching ability, the input of the Robotic Hand-eye Sensorimotor Module is the hand's position $P_{hand}(x, y)$ within the images captured (by the Robotic Vision Module). It generates related motor commands $M_{hand}(j_1, j_2)$. The learning algorithm of reaching has been implemented in our previous work, please refer to [8] for details. However, when the robot attempts to point at the object, the input of this module becomes a number of random motor values generated from the Attempt Module $RAN(j_1, j_2)$. Therefore, the arm moves to the new positions.

3.2.5. Interaction of human demonstrator

A human demonstrator plays the role of the robot's babysitter. When the demonstrator realises that the robot is attempting to capture a far away object, the demonstrator moves the object into the robot's reachable space to assist the robot in capturing it. The standard of a pointing movement, which can be understood by the demonstrator, is "Pose 2" in Fig. 1. Other poses, which do not correspond to the standard, are not regarded as pointing gestures.

The complete proposed human robot interaction approach is outlined in Algorithm 2.

Algorithm 2. Human Robot Interaction

```
1: Find a salient object and try to touch it
2: if object-captured is TRUE then
3:    finish reaching
4: else
5:    repeat
6:       choose one action via Eq. (2)
7:       invoke the MRAN network to execute the action
8:       waiting for the interaction of the demonstrator
9:       if no interaction of the demonstrator then
10:         go to Step 6
11:      else
12:         scan the workspace and try to touch the object
13:      end if
14:   until reach the object
15: end if
16: collect pointing data for training the MRAN network
```

3.2.6. Network Training Module

Using the gathered pointing gesture data, the Network Training Module trains the neural network that implements the Hand Sensorimotor Module. The MRAN network starts with no hidden units, and with each learning step, grows or shrinks as necessary by adjusting the network parameters appropriately. These features fit the findings and the theories from the area of developmental psychology [34].

A typical MRAN network [30] is expressed as:

$$f(x) = \alpha_0 + \sum_{k=1}^{N} \alpha_k \phi_{k(x)} \tag{4}$$

$$\phi_{k(x)} = \exp\left(-\frac{1}{\sigma_k^2}\|x - \mu_k\|^2\right) \tag{5}$$

where $f(x) = (f_1(x), f_2(x), \ldots, f_{N_O}(x),)^T$ is the network output vector; x is the network input; α_k is the weight vector from the hidden unit $\phi_k(x)$; N_O is the number of radial basis function units, and μ_k and σ_k are the kth hidden unit's centre and width, respectively. In learning hand-eye coordination, the training data is of the form $(P_{hand}(x,y), j_1, j_2)$. In learning the pointing ability, the pointing gestures are formed by the Attempt Module, thereby having the training pattern $(P_{target}(x,y), j_1, j_2)$.

The network growth criteria are derive from the following two observations: (1) whether the network prediction error for the current learning observation is bigger than a threshold, and (2) whether the node to be added is far enough from the existing nodes in the network, as shown in Eqs. (6) and (8). Eq. (7) checks the prediction error within a sliding window, of length m, to ensure that growth is smooth

$$\|e(t)\| = \|y(t) - f(x(t))\| > e_1 \tag{6}$$

$$\sqrt{\sum_{j=t-(m-1)}^{t} \frac{\|e(j)\|^2}{m}} > e_2 \tag{7}$$

$$\|x(t) - \mu_r(t)\| > e_3 \tag{8}$$

where $x(t)$ and $y(t)$ jointly form the tth learning data, m is the sliding window size, $\mu_r(t)$ is the centre vector of the nearest node to $x(t)$, and e_1, e_2, and e_3 are the three corresponding thresholds. In the experimentation, there are empirically set such that $e_1 = 0.05$, $e_2 = 0.005$, and $e_3 = \max\{0.4 \times 0.999^i, 0.07\}$ with i being the learning step. If the conditions in the above three equations are not met, the Extended Kalman Filter algorithm [30] is applied to modify the network's weights in order to decrease the network's error. Otherwise, a new node is inserted into the network. The benefit of using the MRAN is that

the network supports on-line learning, so as to enable the robot system to conduct incremental learning feature that is of a natural appeal in humans and animal world.

3.3. Developmental mechanism with LCAS algorithm

Our developmental mechanism is employed to determine when to remove the constraints. First, we identify all possible and available constraints and decide which should be initially applied to the system. Next, the robot executes the motor actions to discover any stimuli in the sensorimotor modalities. As the times of arm movement increase, the system tends to become more saturated. When the saturation becomes stable and is less than a fixed value ψ, we regard the robotic system as being saturated. Therefore, we change the constraint, and the robot acquires new training data to retrain the network, until the saturation value of the whole system is stabilised again. This mechanism is formally captured by the following equation:

$$Sat(G(t),t) = \begin{cases} true; & \text{if} |G(t) - G(t-\varphi)| \\ & < \varepsilon \quad \text{and} \quad G(t) < \psi; t = \varphi \cdots n \\ false; & \text{else} \end{cases} \tag{9}$$

where t is the number of training epoches; $G(t)$ is the system's global excitation value at epoch t; φ is for the sampling rate; and ψ is a fixed value that controls the global excitation's amplitude of variation. This formula expresses the following saturation rule: If $G(t)$ is less than ψ for a certain rate φ, the situation is regarded as saturated and hence, the constraint is lifted. In other words, Eq. (9) indicates that: if the learning system remains stable for a fixed term, a new constraint is assigned to the system. In the experiment, the parameters φ, ψ and ε are empirically set to 5, 0.1 and 0.02, respectively.

Note that $G(t)$ is an important parameter, which should be defined carefully. Because the output error of the network cannot reveal whether it is stable, the error values are not used as $G(t)$ directly. Instead, we apply a habitation formula to define $G(t)$. At each step of learning to point, novelty and habituation play a significant role in driving the learning process. Here, novelty refers to new or particularly salient sensory stimuli, while habituation is defined as a decrease in the strength of a behavioural response to repeated stimuli. A habituated stimulus may evoke a further response after the presentation of an intervening novel stimulus.

Novelty and habituation mechanisms help a system to explore new places or events, while monitoring the current situation. Therefore, the system gleans experience over its entire environment. In our system, we use a biologically plausible habituation model which was created in our previous work [23,33] to describe how global excitation, G, varies with time:

$$G(t+1) = G(t) + \frac{\alpha}{\tau}[G_0 - G(t)] - \frac{S}{\tau} \tag{10}$$

where G_0 is the original value of G, and τ and α are time constants governing the rate of habituation and recovery. In the experiment, we set τ, α and G_0 to be 5, 0.9 and 1.0, respectively. This setup of the parameters τ and α is in order to change the global excitation value smoothly, as opposed to abruptly. If the robot always receives the same stimulus $S = 0$, the robot is regarded to be in a habituation phase. In this case, the value of $G(t)$ will fall gradually. If the robot receives a salient stimulus $S \neq 0$, the robot is said to be in a recovery phase, and the value of $G(t)$ will increase.

During the training phase of the pointing gesture, an unreachable object is recognised as a salient stimulus. The estimation of such salience needs to invoke the Coverable Check Module, which is described in Section 3.2.2. After each attempt movement, if the human demonstrator does not react to move the object, a salient stimulus is sent to the robot. Otherwise, the robot will receive a non-salient stimulus.

The robot system is imposed with two types of constraint: (1) Object appearance, and (2) temperature in Q-learning. The sequence of change in these two constraints is as follows:

(a) Instead of placing objects into the robot's workspace, visual stimuli are produced near by the robot's hand.
(b) An object is placed inside the range of the robot' arm, where the arm can reach it.
(c) An object is placed outside the arm's range, forcing the robot to apply Q-learning in attempting to reach the object.
(d) The temperature T in Q-learning is changed, and the robot performs random movements.
(e) With human interactions, the robot successfully points at the object.
(f) The robot then returns to Step (c) above and starts the loop anew.

Throughout, the LCAS algorithm controls when to change from one step to the next. In the robot's workspace, a "rest area" is set. When the robot's saturation rate is low, the robot's arm will move to and stay at the rest area. This action tells the human demonstrator: the robot has been familiar with current stimuli, wanting to have more interesting things. Then, the demonstrator regards this action as a signal to change constraints. Note that in Step (d) of the sequence of change, the temperature T is set to be changed by the robot's internal developmental mechanism (rather than by the human demonstrator).

4. Experimental results

This experimental study is designed to simulate the developmental procedure of infant's pointing by the robot system. The procedure is as follows: A human demonstrator stands nearby the robot watching the robot's movements. The robot's arm moves randomly above the workspace in which there is no target object. The developmental learning algorithm begins to build the hand-eye coordination. After the robot's arm enters the rest area, the demonstrator places a target object randomly in the workspace, but inside the arm's working range. When the robot's arm again returns to the rest area, the demonstrator places an object outside the range of the robot's arm. The robot cannot reach the object; thus, the Attempt Module begins to drive the arm in an attempt to touch the object. However, the demonstrator cannot understand the robot's pointing gesture; the robot, then, performs random movements around the object. Upon realising the robot's intention, the demonstrator moves the object into the robot's reachable workspace.

The reason for the sequence of object placing is to simulate human infant development. At the beginning of the robot's development, it can only detect close-range objects with poor vision. The objects placed far away will not invoke the robot's attention. After a number of movements, the robot's reaching capability is gradually developed, so that the robot may notice the objects that are placed far away from the robot's arm and attempt to touch the objects. From this viewpoint, the sequence of placing objects forms an important developmental step in this work.

4.1. Results

Fig. 4 illustrates two unsuccessful pointing actions before the robotic system starts the human–robot interactions. Although learning to point has not been involved at this stage, the system can generate a gesture. However, the robot cannot reach or point at the object yet because the MRAN network has not sufficiently developed its generalisation ability [36]. That is, the MRAN network is yet unable to generate accurate pointing behaviour.

Fig. 5 shows the duty of the human demonstrator. The demonstrator will not interrupt the robot's movements until he recognises that the current gesture belongs to our defined pointing gesture. Then, the demonstrator pushes the object to where the robot's hand is. Thus, a successful human–robot interaction loop is completed. The robot uses this gesture information to train its learning system.

Fig. 6 gives two examples of the three important steps towards a pointing gesture. Gestures in "A1" and "B1" are generated by the MRAN network directly; gestures in "A2" and "B2" are produced by the Q-learning algorithm, where both positions are each at the closest position to the object. Gestures in "A3" and "B3" are made by random movements. Note that these pictures are taken before the human demonstrator starts to interact; they are only subsequently recognised by the demonstrator as pointing gestures.

Fig. 7 demonstrates the MRAN neuron increasing curve in learning both the reaching and the pointing abilities. The horizontal axis is the number of training iterations; and the vertical axis is the number of the MRAN's neurons. Although this paper does not particularly address the learning of the reaching movement, we implement both learning phases to better reflect the incremental features of the MRAN network. Twenty-seven neurons in total are generated during the training phase. Twenty neurons are used to handle the reaching ability. The number stops increasing at about the 190th training iteration.

After the network has been trained 600 iterations, we add the pointing data to the network training patterns. The neuron number therefore, starts to increase again at the 600th training, and seven new neurons have been produced. This incremental feature fits our developmental robot very well. Note that in this work, the shrinking capability of the MRAN network is not exploited. This is because when the robot begins to learn the pointing gesture, no more reaching data is sent to the MRAN. If however, this feature were used, certain MRAN hidden neurons that handle precise reaching movements might be incorrectly deleted.

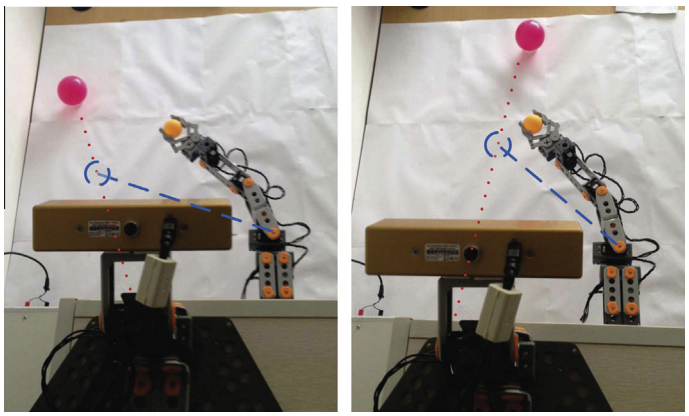

Fig. 4. Unsuccessful pointing gestures.

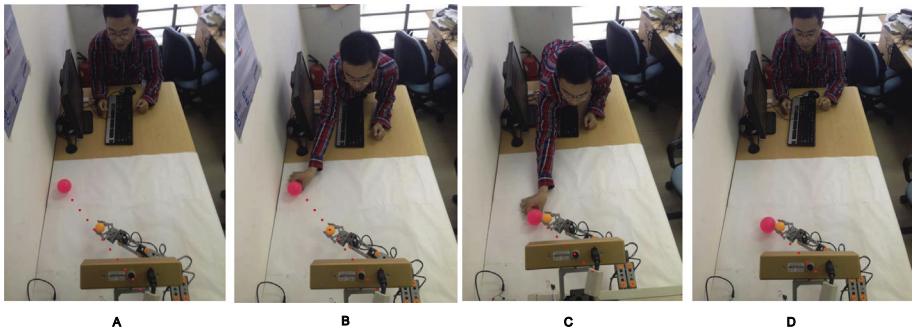

Fig. 5. Human robot interactions.

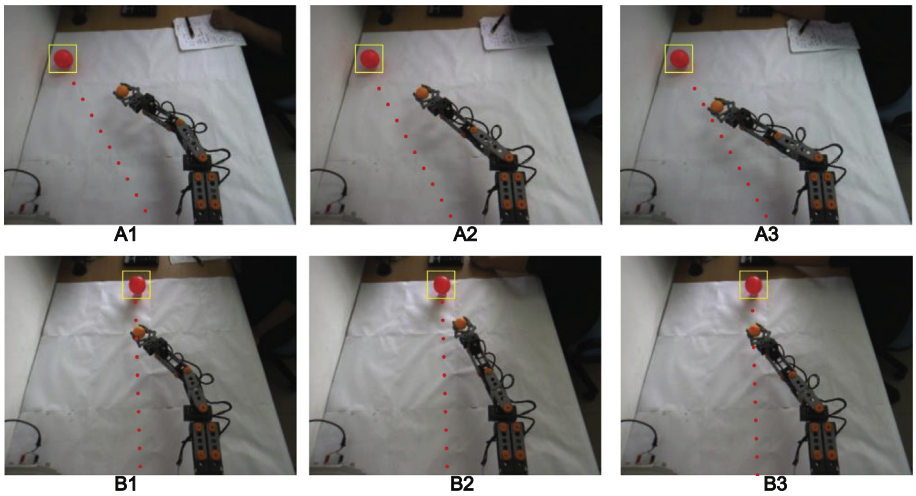

Fig. 6. Three steps to a pointing gesture.

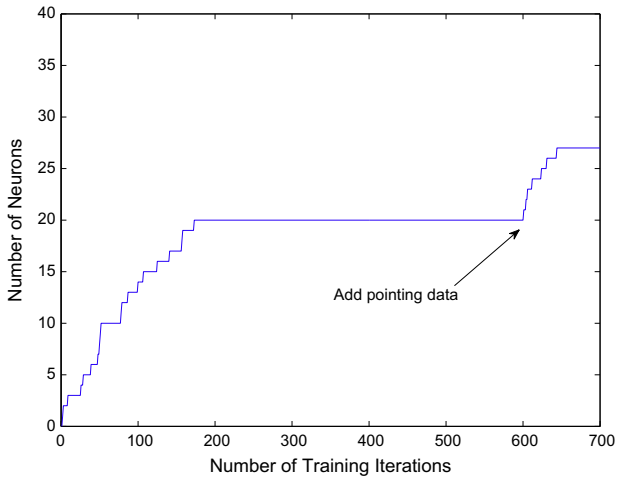

Fig. 7. Increase in number of MRAN neurons.

Fig. 8 presents the MRAN network's convergence curve. This figure also combines the reaching and the pointing training results. Here, the root mean square of the output errors (ERMS) is used to depict the network performance. The horizontal axis is the number of training iterations, and the vertical axis is the value of ERMS. The error drops rapidly at the beginning,

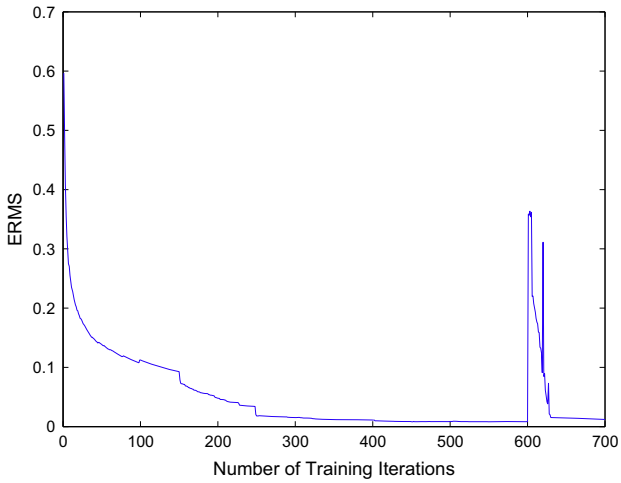

Fig. 8. Output error of MRAN network.

while remaining at a very low level after approximately 250 training iterations. When we start to train the pointing patterns, the error becomes large again and then, falls very rapidly because, at this point, more neurons are responsible for the new learning data.

Figs. 7 and 8 jointly indicate that our learning system is incremental and continuous. The system learns when there is a need for learning, which can occur at any time, even when the robot is performing different tasks. This feature is considered important in developmental robotics [2,42].

Fig. 9 illustrates the saturation situation of the robotic system. The global excitation value in the vertical axis, which is calculated by Eq. (10), is used to indicate whether the learning system is stable. The curve starts with an object inside the robot arm's range. The horizontal axis is the number of the stimulus. The global excitation falls to about 0.19 at the 16*th* stimulus. Then, the robot's arm goes to the rest area, and the object is placed outside the arm's range. Failing to reach the object is a novel stimulus to the robot; therefore, the global excitation rises to a high value. However, the robot still cannot reach the object; thus, the global excitation falls again. After that, the temperature is changed. Therefore, the robot performs random movements until the human demonstrator moves the object. These situations are caused by the changing of the constraints that we set in the robotic system. As such, the entire robotic system's developmental procedure is driven by the constraints.

Fig. 10 shows example trajectories of the process of learning to point. It presents only the starting and the end position of each movement, and uses arrows to indicate each movement's direction. There are three labeled movements in the figure, corresponding to those cases as given in Fig. 6. Label 1 designates the first movement (the largest movement in Fig. 10) generated by the MRAN network. The movements between Label 1 and Label 2 are produced by the Q-learning algorithm. These movements make the robot's arm move to the boundary of its second joint. The remaining movements (from Label 2 to Label

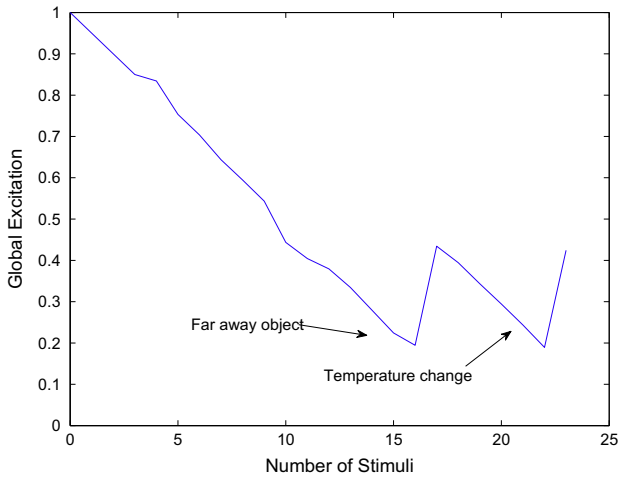

Fig. 9. Global excitation during training.

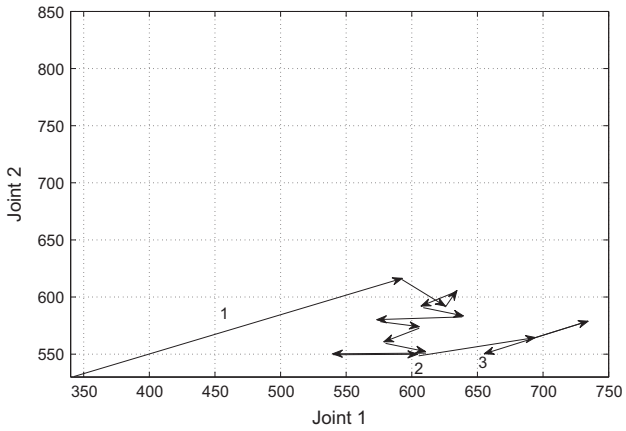

Fig. 10. Trajectories of learning to point.

3) are random ones that stop at a boundary position of Joint 2 (Label 3), indicating that a pointing gesture requires the elbow to be straight.

Fig. 11 shows the comparison against the human pointing gestures. The top two belong to the human demonstrator, and the bottom two to the robot. A dash line is added in each picture to help assess the gestures. In each of these four pictures, no matter where the object is placed, both the human and the robot's arms are straight. In addition, the demonstrator's head and hand and also, the object are in a straight line. Correspondingly, the robot's head and end-effector and also, the object

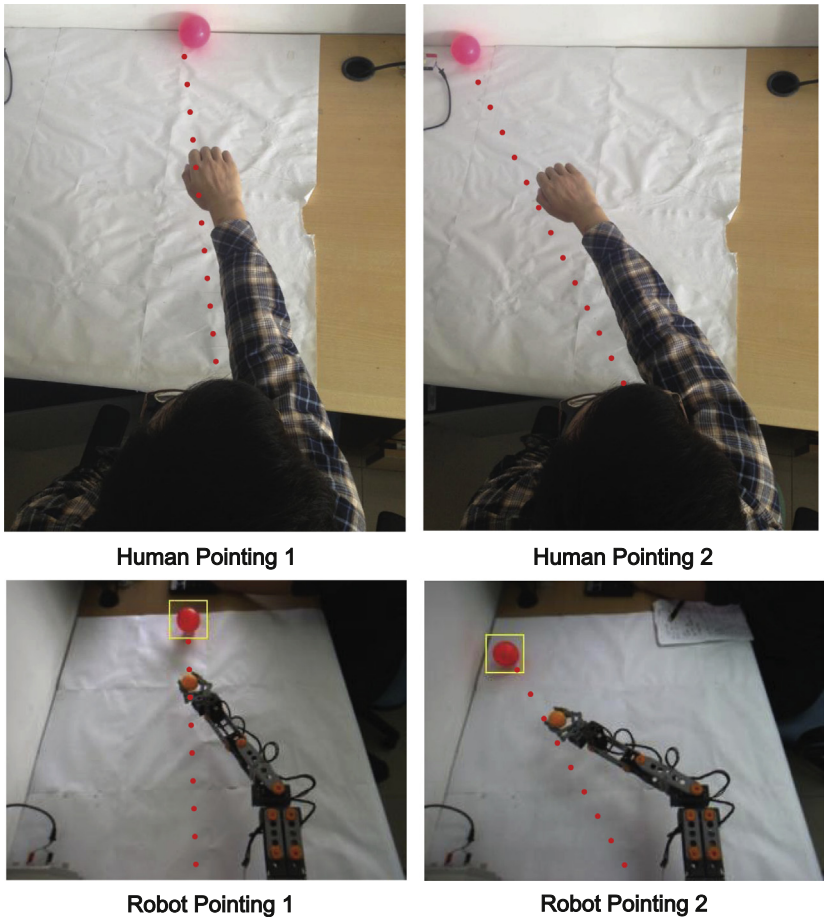

Fig. 11. Comparison with human pointing gesture.

Table 1
Comparison with existing approaches.

Option	Existing approaches	Our approach
Control system implementation	Static artificial neural networks, and Jacobian matrix	Constructive neural networks
Infant features	Few features exploited	The procedure of "trying to reach objects – failing to reach objects – interacting with an adult human-knowing how to point" followed
Developmental stages	No work supported	Staged behavioral patterns in cooperated
Incremental learning	Little work supported	Phased learning entailed
Human–robot interaction	No human–robot interaction involved	Human–robot interactions form the key

are in a straight line too. These results conform to the assumption taken in this research (as given in Section 2). The robot's pointing gestures are indeed very similar to natural pointing.

4.2. Discussion

Based on the above experimental observations, we believe that the proposed developmental approach is successful in generating pointing behaviours. This work significantly differs from existing work in that our robot uses attempt movements to express the robot's desire. The human demonstrator's involvement is also important for the robot. Without human interactions, the robot is unable to establish a common understanding of pointing with humans. The LCAS algorithm leads the robot to develop from reaching to pointing. The switching between the robot's different behavioural patterns is caused by developmental constraints. By using the MRAN network, the robot's learning is incremental. As such, the robotic system has a developmental procedure similar to that of human infants.

The above experimental results demonstrate how the robot learns to point at the salient object via the human demonstrator's interactions. They also show how the reinforcement learning algorithm and the developmental learning mechanism cooperate to enable the robot's learning. To further reflect the strengths of this research, a comparison with typical robotic pointing approaches is summarised in Table 1. In particular, the comparison is focused on the following five important features: (1) control system's implementation, (2) infant characteristics, (3) developmental stages, (4) incremental learning, and (5) human–robot interactions.

Many existing approaches prefer to apply static neural networks to implement the hand-eye nonlinear transformation for robotic pointing. However, our approach is to implement this transformation using a constructive neural network – MRAN. The topology and weights of the MRAN network change simultaneously during the training. This property fits well with a number of important psychological theories on cognition [42].

Regarding the human infant aspect, conventional approaches tend to focusing on building the pointing ability directly, few methods exploit the inspirations from the infant development [40]. In contrast, our work follows the behaviour of human infants developing their capability in the procedure of "trying to reach objects – failing to reach objects – interacting with an adult human-knowing how to point". As a result, our approach allows the robot to exhibit more similar behaviours to an infant.

Another point to note is that existing methods do not show how a resulting robot may gradually develop its competence. However, our work enables the robot to develop from originally simple tasks to the eventual much more complex tasks, with its functions becoming more and more proficient throughout its learning phase. This is confirmed by the experimental results. Figs. 9 and 10 illustrate such a situation where staged behavioural changes take place. Such changes appear to be very similar to the human infant developmental process. Also, the learning mechanism developed in this research allows the robot to learn different tasks incrementally. In particular, it can learn the reaching ability first, and then the pointing ability. Such a property is not possessed by the existing approaches. Besides, the conventional methods do not involve the human–robot interaction in training the robot's pointing ability. The key innovation of our approach is to use the human–robot interactions to establish the understanding of the pointing gestures, in order to develop the robot's pointing ability.

5. Conclusion

This paper has presented an approach for generating pointing behaviour in developmental robotics. The method is achieved by: (1) modelling human infant development procedure; (2) applying human–robot interactions; and (3) using the LCAS algorithm to drive the robot to learn pointing ability. The observations from the experiments demonstrate that the robot system exhibits an increasing progression behaviour from initially performing failed reaching movements to getting closer to the target and, then, generating correct pointing movements. These observations illustrate three advantages of this research over existing work:

- The proposed robotic learning progression is very similar to that of a human infant's development.
- The robot performs incremental and cumulative features in its learning.
- The robotic system possess more autonomous and psychological characteristics.

In the present work however, we only use static RGB cameras to implement the robotic vision. In future, we propose to enhance the robot's vision ability, thereby increasing our robot's potential to operate within a more complicated environment. Also, as proof of concept, our research is currently tested against a laboratory setup. Application of this work to a carefully identified real world problem remains active research. In addition, this work utilises just object and the temperature parameter in the application of Q-learning as the system's developmental constraints. In our further work, we will focus on other types of constraint such as human speech or gesture recognition, which are also important for developmental robotics.

Acknowledgements

This work was supported by the National Natural Science Foundation of China (Nos. 61203336, 61273338 and 61003014) and the Major State Basic Research Development Program of China (973 Program) (No. 2013CB329502). The authors are very grateful to the anonymous reviewers for their constructive comments which have helped significantly in revising this work.

References

[1] A. Alvarez-Alvarez, J.M. Alonso, G. Trivino, Human activity recognition in indoor environments by means of fusing information extracted from intensity of wifi signal and accelerations, Inform. Sci. 233 (2013) 162–182.
[2] M. Asada, K. Hosoda, Y. Kuniyoshi, H. Ishiguro, T. Inui, Y. Yoshikawa, M. Ogino, C. Yoshida, Cognitive developmental robotics: a survey, IEEE Trans. Auton. Mental Dev. 1 (2009) 12–34.
[3] G. Butterworth, Pointing is the royal road to language for babies, in: S. Kita (Ed.), Pointing: Where Language, Culture, and Cognition Meet, Psychology Press, 2003, pp. 9–33.
[4] C.S. Chan, H. Liu, Fuzzy qualitative human motion analysis, IEEE Trans. Fuzzy Syst. 17 (2009) 851–862.
[5] F. Chao, M.H. Lee, An autonomous developmental learning approach for robotic eye-hand coordination, in: Proceeding of Artificial Intelligence and Applications – 2009, Innsbruck, Austria, 2009, pp. 639–013–1–6.
[6] F. Chao, M.H. Lee, J.J. Lee, A developmental algorithm for ocular-motor coordination, Robot. Auton. Syst. 58 (2010) 239–248.
[7] F. Chao, P. Wang, M. Shi, M. Jiang, Pointing learning in developmental robotics based on constructive neural network and reinforcement learning algorithm, in: Proceedings of the 3th International Conference on Intelligent Computing and Intelligent Systems, Guangzhou, China, 2011, pp. 614–619.
[8] F. Chao, X. Zhang, H. Lin, C. Zhou, M. Jiang, Learning robotic hand-eye coordination through a developmental constraint driven approach, Int. J. Automat. Comput. 10 (2013) 414–424.
[9] J. Chen, W. Bian, D. Tao, Locally regularized sliced inverse regression based 3D hand gesture recognition on a dance robot, Inform. Sci. 221 (2013) 274–283.
[10] Y.J. Chen, K.S. Hwang, W.C. Jiang, Policy sharing between multiple mobile robots using decision trees, Inform. Sci. 234 (2013) 112–120.
[11] M.W. Doniec, G. Sun, B. Scassellati, Active learning of joint attention, in: Proceedings of the 6th IEEE/RAS International Conference on Humanoid Robots, IEEE/RAS, 2006, pp. 34–39.
[12] B. Fernandez-Gauna, I. Marques, M. Grana, Undesired state-action prediction in multi-agent reinforcement learning for linked multi-component robotic system control, Inform. Sci. 232 (2013) 309–324.
[13] T. Fong, I. Nourbakhsh, K. Dautenhahn, A survey of socially interactive robots, Robot. Auton. Syst. 42 (2003) 143–166.
[14] V.V. Hafner, F. Kaplan, in: Lecture Notes in Computer Science, Learning to Interpret Pointing Gestures: Experiments with Four-Legged Autonomous Robots, vol. 35 75, Springer Verlag, 2005, pp. 225–234.
[15] K.S. Hwang, Y.J. Chen, W.C. Jiang, T.W. Yang, Induced states in a decision tree constructed by Q-learning, Inform. Sci. 213 (2012) 39–49.
[16] K.S. Hwang, H.Y. Lin, Y.P. Hsu, H.H. Yu, Self-organizing state aggregation for architecture design of Q-learning, Inform. Sci. 181 (2011) 2813–2822.
[17] J.M. Iverson, S. Goldin-Meadow, Gesture paves the way for language development, Psychol. Sci. 16 (2005) 367–371.
[18] L. Jamone, L. Natale, F. Nori, G. Metta, G. Sandini, Autonomous online learning of reaching behavior in a humanoid robot, Int. J. Human. Robot. 9 (2012). 1250017-1–1250017-26.
[19] L. Jamone, L. Natale, G. Sandini, A. Takanishi, Interactive online learning of the kinematic workspace of a humanoid robot, in: Proceedings of International Conference on Intelligent Robots and Systems, IEEE/RSJ, Vilamoura, Algarve, Portugal, 2012, pp. 2606–2612.
[20] J. Law, M.H. Lee, M. Huelse, Infant development sequences for shaping learning in humanoid robots, in: Proceedings of the Tenth International Conference on Epigenetic Robotics, Lund University Cognitive Studies, 2010, pp. 65–72.
[21] J. Law, M.H. Lee, M. Huelse, The infant development timeline and its application to robot shaping, Adapt. Behav. 19 (2011) 335–358.
[22] J. Law, P. Shaw, M. Lee, A biologically constrained architecture for developmental learning of eyechead gaze control on a humanoid robot, Auton. Robots 35 (2013) 77–92.
[23] M.H. Lee, Q. Meng, Psychologically inspired sensory-motor development in early robot learning, Int. J. Adv. Robot. Syst. 2 (2005) 325–334.
[24] M.H. Lee, Q. Meng, F. Chao, Developmental learning for autonomous robots, Robot. Auton. Syst. 55 (2007) 750–759.
[25] M.H. Lee, Q. Meng, F. Chao, Staged competence learning in developmental robotics, Adapt. Behav. 15 (2007) 241–255.
[26] A. Lemme, A. Freire, G. Barreto, J. Steil, Kinesthetic teaching of visuomotor coordination for pointing by the humanoid robot iCub, Neurocomputing 112 (2013) 179–188.
[27] U. Liszkowski, M. Carpenter, T. Striano, M. Tomasello, 12-and 18-month-olds point to provide information for others, J. Cognit. Dev. 7 (2006) 173–187.
[28] H. Liu, A fuzzy qualitative framework for connecting robot qualitative and quantitative representations, IEEE Trans. Fuzzy Syst. 16 (2008) 1522–1530.
[29] H. Liu, D.J. Brown, G.M. Coghill, Fuzzy qualitative robot kinematics, IEEE Trans. Fuzzy Syst. 16 (2008) 808–822.
[30] Y. Lu, S. Narashiman, P. Saratchandran, Performance evaluation of a sequential minimal radial basis function (RBF) neural network learning algorithm, IEEE Trans. Neural Netw. 9 (1998) 308–318.
[31] M. Lungarella, G. Metta, R. Pfeifer, G. Sandini, Developmental robotics: a survey, Connect. Sci. 15 (2003) 151–190.
[32] M.J. Marjanovic, B. Scassellati, M.M. Williamson, Self-taught visually-guided pointing for a humanoid robot, in: Proceedings of the 4th Int. Conf. on Simulation of Adaptive Behavior, 1996, pp. 35–44.
[33] Q. Meng, M.H. Lee, Novelty and habituation: the driving forces in early stage learning for developmental robotics, Neural Learning for Intelligent Robotics, LNCS, 2005, pp. 315–332.

[34] Q. Meng, M.H. Lee, Automated cross-modal mapping in robotic eye/hand systems using plastic radial basis function networks, Connect. Sci. 19 (2007) 25–52.
[35] E.L. Newport, Constraints on learning and their role in language acquisition: studies of the acquisition of american sign language, Lang. Sci. 10 (1988) 147–172.
[36] J. Platt, A resource-allocating network for function interpolation, Neural Comput. 3 (1991) 213–255.
[37] S. Qu, R. Grupen, From manipulation to communicative gesture, in: Proceedings of the 5th ACM/IEEE International Conference on Human–Robot Interaction, IEEE Press Piscataway, NJ, USA, 2010, pp. 325–332.
[38] M. Ros, M. Cuellar, M. Delgado, A. Vila, Online recognition of human activities and adaptation to habit changes by means of learning automata and fuzzy temporal windows, Inform. Sci. 220 (2013) 86–101.
[39] A. Shademan, A.M. Farahmand, M. Jagersand, Towards learning robotic reaching and pointing: an uncalibrated visual servoing approach, in: Proceedings of the Canadian Conference on Computer and Robot Vision, 2009, pp. 229–236.
[40] M. Sheldon, M. Lee, A developmental approach to the emergence of communication in socially situated embodied agents, in: Proceedings of the IEEE 9th International Conference on Development and Learning, 2010, pp. 204–210.
[41] V. Southgate, C. van Maanen, G. Csibra, Infant pointing: communication to cooperate or communication to learn, Child Dev. 78 (2007) 735–740.
[42] A. Stoytchev, Some basic principles of developmental robotics, IEEE Trans. Auton. Mental Dev. 1 (2009) 122–130.
[43] R.S. Sutton, A.G. Barto, Reinforcement Learning: An Introduction, MIT Press, 1998.
[44] M. Tomassello, M. Carpenter, U. Liszkowski, A new look at infant pointing, Child Dev. 78 (2007) 705–722.
[45] N.A. Vien, H. Yu, T.C. Chung, Hessian matrix distribution for bayesian policy gradient reinforcement learning, Inform. Sci. 181 (2011) 1671–1685.
[46] L.S. Vygotsky, Mind in Society: The Development of Higher Psychological Processes, 14 ed., Harvard University Press, 1978.
[47] X.S. Wang, Y.H. Cheng, J.Q. Yi, A fuzzy Actor–Critic reinforcement learning network, Inform. Sci. 177 (2007) 3764–3781.
[48] A.L. Woodward, J.J. Guajardo, Infants' understanding of the point gesture as an object-directed action, Cognitive Dev. 17 (2002) 1061–1084.
[49] X. Xu, L. Zuo, Z. Huang, Reinforcement learning algorithms with function approximation: recent advances and applications, Inform. Sci. 261 (2014) 1–31.
[50] A. Yorita, N. Kubota, Cognitive development in partner robots for information support to elderly people, IEEE Trans. Auton. Mental Dev. 3 (2011) 64–73.