Integrating Epistemic Action (Active Vision) and Pragmatic Action (Reaching): A Neural Architecture for Camera-Arm Robots

Dimitri Ognibene[1,2], Christian Balkenius[3], and Gianluca Baldassarre[1]

[1] LARAL-ISTC-CNR, Lab. of Autonomous Robotics and Artificial Life, Istituto di Scienze e Tecnologie della Cognizione, Consiglio Nazionale delle Ricerche, Via San Martino della Battaglia 44 - 00185 Roma, Italy
[2] DIST, Dip. di Informatica Sistemistica e Telematica, Universita' di Genova, Via all'Opera Pia 13 - 16145 Genova, Italy
[3] Lund University Cognitive Science, Kungshuset, Lundagård SE - 222 22 Lund, Sweden
dimitri.ognibene@istc.cnr.it, christian.balkenius@lucs.lu.se, gianluca.baldassarre@istc.cnr.it *

Abstract. The active vision and attention-for-action frameworks propose that in organisms attention and perception are closely integrated with action and learning. This work proposes a novel bio-inspired integrated neural-network architecture that on one side uses attention to guide and furnish the parameters to action, and on the other side uses the effects of action to train the task-oriented top-down attention components of the system. The architecture is tested both with a simulated and a real camera-arm robot engaged in a reaching task. The results highlight the computational opportunities and difficulties deriving from a close integration of attention, action and learning.

1 Introduction

Consider a primate exposed to a new environment scattered with bushes carrying red fruits. It might initially look at the bright green foliage and trunks of bushes popping out of the scene, and try to interact with them without any useful result. Then it might look at a fruit and then pick and taste it. Now that it understands that fruits are useful, how can it find more of them? As its gaze often focuses on the bushes' foliage, it should learn to look away from them, and below them, as the fruits of these bushes hang below their leaves. It should also learn to trigger reaching actions on the basis of the fruits' sight and to shape actions on the basis of the gaze direction.

This example shows typical interactions between attention, perception, action and learning processes taking place in an organism acting in a natural context. These interactions have often been overlooked by the *information-processing*

* This research was supported by the EU Projects *ICEA*, contract no. FP6-IST-027819-IP, and *MindRACES*, contract no. FP6-511931-STREP

framework widely used in machine vision, initiated with Marr's theory of vision [1]. This framework views attention and vision as processes directed to construct "objective" detailed general-purpose representations of the environment later used to guide action and learning [2]. Computationally, building representations totally detached from the embodiment and the specific needs of the system tends to produce scene representations containing an overwhelming amount of non-needed information and hence often computationally heavy or intractable.

The *active vision approach* [3] introduced action in visual processes to allow a high-sensitive *fovea* to scan the scene and perform heavy computations only on portions of it relevant for the task in hand, similarly to what happens in human attention [4, 5]. Moreover, it proposed to exploit gaze motion to simplify representations and learning processes, for example by using "deictic representations" encoding information with respect to the current state of sensors, or by applying object or feature recognition processes only to the foveated points [3].

As it emphasizes the importance of action in perception, the active vision perspective has been fully embraced by *evolutionary robotics* [6]. This has proposed systems that fully integrate actions directed to gather information (*epistemic actions*) and actions directed to accomplish the systems' goals in the environment (*pragmatic actions*) [7]. In general, with respect to active vision, evolutionary algorithms have the advantage of co-evolving complementary fovea movements and feature detectors [8, 9]. Moreover, they are not affected by the *perceptual aliasing problem* [10] introduced by the fovea's partial view of scenes as reinforcement learning algorithms are. In this respect, an important feature of the architecture proposed here is that, while it uses reinforcement learning to exploit the advantages of on-line adaptation, it ameliorates the aliasing problem by using a *potential action map (PAM)* that stores information on past percepts in the form of potential actions (*memory* is a typical solution to aliasing).

Interestingly, within psychology, Allport [11] proposed a new perspective on attention that, in line with the ideas of active vision, claims that attention serves primarily to guide organisms' action, for example by directly setting some of its parameters [12]. Within the modeling literature, Balkenius [13] echoes this view and specifies the *attention-for-action perspective* with four basic principles: (1) inhibition can be used to disengage the focus of attention from the current location; (2) attentive (epistemic) actions can be computationally treated as other (pragmatic) actions; (3) focussing processes can lead to select targets for (pragmatic) action; (4) gaze direction can be used to produce implicit arguments for action. These principles not only emphasize that attention and action are closely coupled, but they also stress that learning principles generally used to acquire pragmatic actions can also be used to learn attentive actions (principle (2)). Indeed, in the past several systems have been proposed that, contrary to processes that detect information on the basis of intrinsic salience of images' features (*bottom-up attention*; e.g. [14]), exploit *reinforcement learning* algorithms to learn to detect task-relevant information (*top-down attention*; e.g. [15]).

This work proposes a novel neural-network architecture where perception, attention (bottom-up and top-down), action, and learning are integrated to an

extent that goes well beyond what is done in existing models. The architecture is tested both with a simulated and with a real camera-arm robot engaged in a reaching task. As we shall see, Sect. 2 on methods and Sect. 3 on results show that the principles of attention-for-action proposed in [13] are fully integrated in the system, either by design or as features emerging from the learning processes. The overall value of this research resides not only in the mechanisms that are proposed to implement the aforementioned integration, but also in the analysis of the system that shows the computational advantages that derive from it.

2 Methods

This section first overviews the system and then explains in detail its components' functioning. The system (Fig. 1b) integrates two previous models: (1) a bottom-up and top-down attention model [16]; (2) an arm control model [17]. These models are based on common computational principles: population codes (here 2D neural maps) to represent sensorimotor information and probability distributions of variables controlling eye/arm behavior [18, 19]; dynamic neural-field networks to integrate information and select actions through biased competition mechanisms [20, 21]; a progressive developmental of skills of the neural components (cf. [17]). These principles were chosen for their biological plausibility.

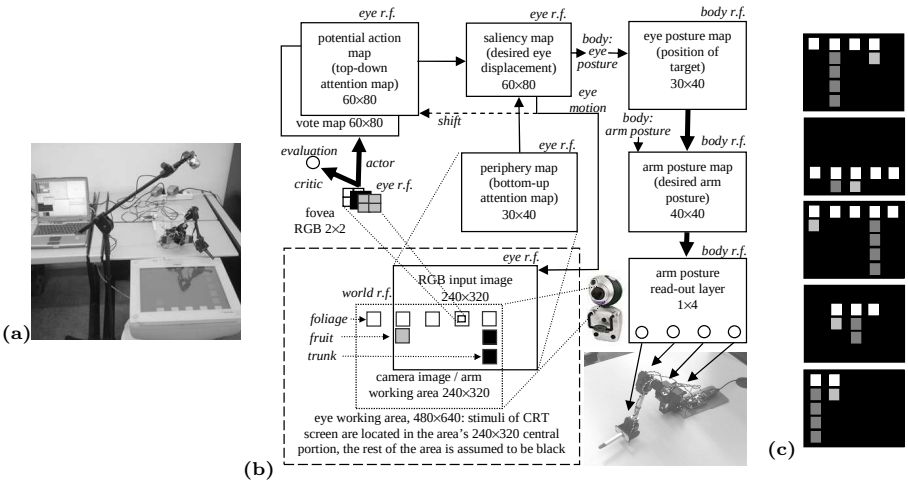

Fig. 1. (a) Robotic setup. (b) Model's architecture. (c) Examples of input images.

The simulated/hardware experimental setup is formed by a down-looking webcam set above a robotic arm (Fig. 1a). The arm's working plane is a CRT monitor. A host computer grabs the camera images, runs the robot's controller, issues motor commands to the arm, and controls the images of the monitor (task). A moving sub-image (*input image*) is extracted from the camera image to simulate eye movements. The input image is used by a *periphery map* that

implements bottom-up attention. The central part of input image (*fovea*) is the input of *reinforcement-learning actor-critic* component that learns to predict the spatial position of the rewarded arm targets with respect to the foveated cues (top-down attention). A *potential action map (PAM)* accumulates this evidence while the fovea explores various cues. A *saliency map* integrates information from the periphery map and the PAM to select the next eye movement using a biased competition. Each fixation point, encoded in a *eye posture map*, suggests a potential arm target to a *arm posture map*: when the eye fixates a location for long, the arm posture map triggers a related action on the basis of a biased competition. If the reached target is a "fruit", the system gets rewarded otherwise it gets slightly punished (energy consumption).

2.1 Robotic Setup and Task

In the following, with the exception of some weight matrices, the bold symbols of mathematical notations represent column vectors.

Camera. A low cost Spacecam 150 Live webcam (by Trust; see Fig. 1) was used to acquire visual information from the environment. The webcam grabs ten RGB images per second with a 240×320 pixel definition (24 bit/pixel). The webcam is set above the robotic arm with its view field covering exactly the monitor's screen forming the arm's working plane. The webcam is connected to the host computer via a USB port and is interfaced with software built with Java Multimedia Framework libraries (by SUN). In the tests running in simulation, the camera is simulated by directly using the task's monitor images.

Robotic Arm. The robotic arm was built using low cost components (e.g. by Lynxmotion). The arm (Fig. 1) is composed of a base and 3 segments (upper arm $15.9cm$, forearm $17.5cm$, and "hand" $9.5cm$). The arm has four degrees of freedom: two at the shoulder (planar rotation, $15° - 165°$, and vertical rotation, $20° - 140°$), one at the elbow ($35° - 145°$), and one at the wrist ($110° - 220°$). Each joint is powered by one digital servo (by Hitech) with the exception of the shoulder vertical-rotation joint having two servos. The servos are controlled by a servo controller SSC32 (by Lynxmotion) connected to the host computer via a serial port. The simulated version of the arm is a simplified kinematic plant with segments' size and degrees of freedom like those of the real arm.

Environment and Task. The horizontal working plane of the arm is a CRT monitor screen ($37 \times 28cm$) connected to the host computer. The monitor generates the images of the task used to test the system. These images are formed by red, green and blue squares set on the vertexes of a 5×5 grid covering the whole screen. In particular, the images form stylized "trees" (see Fig. 1) with 2-5 green blocks representing the foliage (100% luminosity), 1-4 blue blocks representing the trunk (80% luminosity), and 1 red block representing a fruit (80% luminosity). After each reaching action, the system gets a reward of 1 if it touches a fruit and a punishment of -0.05 otherwise. Saccades are not directly rewarded or punished. A new tree randomly structured and positioned in the image is generated after the execution of each reaching action.

2.2 Attention Control Components

Preprocessing Filters. The 240×320 pixel RGB image of the webcam is noisy and contains reflections, so it is first subtracted by an image grabbed by the camera with the screen switched off (black image), and then it is filtered into main-color components. The resulting image in copied into the centre of a bigger 480×640 pixel black image, and a 240×320 pixel image is extracted from this to simulate the system's *input image* grabbed by a moving eye.

Periphery Map (Bottom-Up Attention). The 30×40 periphery map **pm** is activated with a grayscale image: first the input image is divided into 30×40 blocks of 8×8 pixels each, then the RGB color values of the pixels of each block are averaged to obtain a gray value. A more sophisticated bottom-up saliency (e.g. as in [14]) is not needed as this research focuses on top-down attention.

Actor-Critic Component (Top-Down Attention). The fovea is simulated with an image $\mathbf{f}$ of 2×2 RGB pixels taken from the input image centre. This image is fed into two feedforward neural networks forming a reinforcement-learning actor-critic architecture [22]. The *critic* is a network with a linear output unit v_t which learns to evaluate the current state on the basis of the expected future discounted rewards. The system gets a reward r_t after the execution of a reaching action, and this, together with v_t, is used to compute the *surprise signal* s_t [22] used to update both the critic's weights $\mathbf{w}^c$ and the actor's weights $\mathbf{W}^a$. The *actor* is a network whose output layer is a *vote map* **vm** of 60×80 sigmoid neurons which signal to the PAM the possible positions of rewarded targets with respect to the currently foveated visual cue ($\gamma = 0.9$; T is the transpose operator):

$$v_t = \mathbf{w}^{cT}\mathbf{f} \quad s_t = (r_t + \gamma\, v_t) - v_{t-1} \quad \mathbf{vm} = g\,[\mathbf{W}^a\mathbf{f}] \quad g[x] = 1/(1+e^{-x}) \quad (1)$$

The critic is trained on the basis of s_t, used as error signal, and the input signal $\mathbf{f}$ [22]. The actor is trained with a Hebb rule involving the activation of the saliency map $\mathbf{sm}_t$ (encoding the last eye displacement, see below) and the input signal $\mathbf{f}$ so as to increase or decrease the probability of doing the same saccadic movement again on the basis of the surprise signal s_t [16] ($\eta^c = 10^{-7}$, $\eta^a = 10^{-5}$):

$$\mathbf{w}^c_{t+1} = \mathbf{w}^c_t + \eta^c\, s_t\, \mathbf{f}_t \quad \mathbf{W}^a_{t+1} = \mathbf{W}^a_t + \eta^a\, s_t\, \mathbf{sm}_t \bullet (\mathbf{vm}_t \bullet (1 - \mathbf{vm}_t))\, \mathbf{f}_t^T \quad (2)$$

where $\bullet$ is the entrywise product operator.

Potential Action Map (Top-Down Attention Memory). The PAM **pam** is formed by 60×80 leaky neurons and accumulates evidence, furnished by the vote map **vm** via topological connections, on the possible positions of rewarded targets. During each saccade the map's activation is shifted in the direction opposite to the eye's motion to maintain eye-centred representations (as it might happen in real organisms [23]). The PAM is reset each time the tree image from the camera changes (also this might happen in real organisms [24]).

Saliency Map. The 60×80 saliency map **sm** selects saccade movements on the basis of the sum of the topological input signals **pm** and **pam**. The saccade movement is selected by first identifying the unit with the maximum activation and then by activating the map with a Gaussian population code centred on it

($\sigma = 1$). The eye movement is the winning neurons' preferred eye displacement $(\Delta x, \Delta y)$. This selection mechanism is a computationally fast approximation of a biased dynamic competition process as the one reported in [21] (cf. Eq. 3).

2.3 Arm Control Components

Eye Posture Map. This 30×40 neuron map encodes the current eye posture as a Gaussian population code **emp** ($\sigma = 0.3$).

Arm Posture Map. This 40×40 map *apm* is the output layer of a neural network pre-trained with a Kohonen algorithm (see below and [17]) and encodes arm postures in the 2D map space. During the tests reported in Sect. 3, a neural biased competition [21] takes place in the map (similarly to what happens in real organisms [20]) in order to select a target for reaching actions when any neuron achieves a threshold th ($th = 0.3$; $\delta = 0.1$):

$$\mathbf{apm}_{t+1} = max\left[(1-\delta)\,\mathbf{apm}_t + \mathbf{W}^{apm\,l}\mathbf{apm}_t + \mathbf{W}^{apm}\mathbf{epm}_t, 0\right] \qquad (3)$$

where $\mathbf{W}^{apm\,l}$ are the weights of *lateral* close-excitatory far-inhibitory connections having a Gaussian distribution dependent on the distance between neurons (see [17] for details), and $\mathbf{W}^{apm}$ are the weights from the eye posture map.

Arm Posture Readout Layer. This is a layer of four sigmoid neurons **aprl** that encode the desired arm joint angles issued to the arm real/simulated servos. The map is activated by the arm posture map through the weights $\mathbf{W}^{aprl}$.

Training. The weights $\mathbf{W}^{aprl}$ and $\mathbf{W}^{apm}$ are trained using the simulated arm to avoid stressing the hardware robot. Training is composed of three succeeding learning phases based on random movements of the arm (*motor babbling*). To avoid redundancy problems during training, the hand segment is kept parallel to the working plane at a fixed distance from it (see Fig. 1a; see [25] for a version of the model addressing redundancy issues). In these phases the system (see [17] for details): (a) performs a vector quantization of postures, within the arm posture map, on the basis of a Kohonen algorithm; (b) learns the inverse kinematic mapping ($\mathbf{W}^{apm}$) between the gaze directions corresponding to the seen hand (**epm**) and the corresponding arm posture (**apm**) with supervised learning; (c) trains the arm posture readout map ($\mathbf{W}^{aprl}$) with supervised learning.

3 Results

This section analyses the behavior of the system tested in the simulated and real robot and the functioning of its neural components emerged with learning. Fig. 2.a shows the reward received by the robot for every reaching action during learning. After the first reward, the performance of both the simulated and real robot increases rapidly and soon reaches a near-optimal steady state. Fig. 2.b shows the average distance between reaching actions' targets and the τ^{th} saccade's target executed before such actions during learning. For $\tau = 1$ the distance initially increases from $2cm$ to $4cm$ and then goes back to $2cm$, for $\tau = 2$, 3, 4 the distance goes from $4 - 8cm$ to $2cm$, for $\tau = 5$ the distance decreases from

$6 - 8cm$ to $3cm$ and then goes to $3 - 4cm$. This data indicate that initially only the last saccade ($\tau = 1$) is related to the reaching target but with learning the attention-action coordination increases until the last four/five saccades are focussed on the target. Fig. 2.c shows how the average number of saccades per reaching action evolves during learning. Initially this number is about 20 but then increases to 50 in correspondence to the maximum learning progress in reaching (Fig. 2.a), and finally stabilises at about 10. This dynamics is due to the fact that the system initially tends to trigger reaching actions directed to bushes' foliage (which has a high salience) or trunk, then it learns to inhibit these actions so that eye exploration increases, and finally it learns to anticipate the position of fruits so that saccades become very efficient in localising fruits and in triggering correct reaching actions.

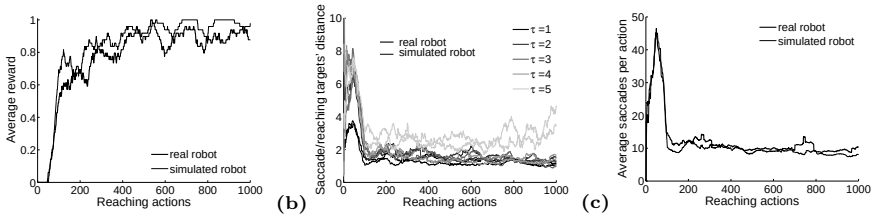

Fig. 2. Learning dynamics during 1000 reaching actions of the simulated and real robot. (a) Moving average of reward. (b) Moving average of the distance between the targets of the last five saccades (τ) and the reaching targets. (c) Moving average of saccades per reaching action. All moving averages have a 50-step window.

This interpretation is corroborated by data reported in Fig. 3 that shows the 20 most frequent sequences of objects foveated by the trained simulated and real robots in 1000 reaching actions. In both cases, the two most frequent sequences start with a saccade on foliage followed by 4-5 saccades on fruit that trigger action: the system has learned to suitably inhibit the high-saliency foliage cues and to stay on the fruit once found. Other sequences focus only on fruit: these are the "lucky" cases where the eye is already on the fruit in the new tree image. Finally, other sequences are those that start with a foliage saccade followed by a trunk saccade and then a fruit: they indicate that the PAM retains information on the first "ambiguous" saccade target and integrates it with the information from the second saccade target, so in part solving the partial observability problem caused by the limited view of the fovea (see Sect. 1). Note that the most frequent sequences are quite similar for the simulated and real robot. However, the number of total sequences in general is higher for the real robot (53) than for the simulated one (391) due to a higher noise which in the latter case tends to lead saccades to the background.

Fig. 4 shows the activation of the vote map, the PAM and the saliency map in a sequence of three saccades targeted respectively to the foliage, trunk and fruit. While foveating the foliage (or the trunk), the vote map activates as follows (Fig. 4a-c): (a) a cluster of neurons activates below 0.5 in correspondence to the whole row of foliage elements (or the column of trunk elements): this biases

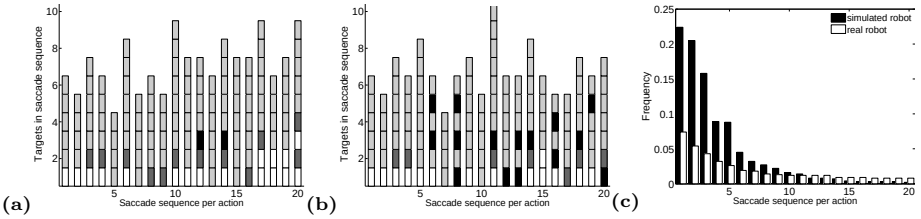

Fig. 3. Most frequent sequences of targets foveated in 1000 reaching actions by the simulated (a) and real robot (b), and corresponding frequencies (c).

the eye to move away from them and constitutes an *emergent* form of self-tuned *inhibition of return* related to visual cues (as in organisms [24]) and not to spatial locations, as in the *hardwired* implementations of it of previous models (e.g. [14, 16]); (b) a cluster of neurons activates above 0.5 in correspondence to the row of elements below the foliage (or left and right to the trunk): this biases the eye to move there and captures the spatial relations existing between the foliage (or trunk) and the fruit. While foveating the fruit, the vote map exhibits a high-contrast Mexican-hat-shaped activation formed by a cluster of neurons activated above 0.5, surrounded by neurons activated below 0.5, in correspondence to the centre: this bias the eye to stay on the target. The activations of the PAM (Fig. 4d-f) show how this memory plays an important function in integrating information in time (the system's performance decreases of 50% without this memory, see [17]). In particular, Fig. 4e shows that if the system first foveates the foliage and then, by chance, the trunk it maintains a strong inhibition in correspondence to the foliage and sums up the bias to go on the fruit (below the foliage and laterally to the trunk) coming from both the foliage and trunk cues. Last, the saliency map (Fig. 4g-i) shows how top-down information is suitably integrated with bottom-up information in order to select the most promising locations. For example, notice the strong activation in correspondence to the fruit, laterally to the trunk, in Fig. 4h compared to Fig. 4e.

4 Conclusions and Future Work

This paper presented an architecture for controlling a camera-arm robot that integrates attention, perception, action and learning well beyond existing models. The integrated nature of the system allows it to instantiate the four principles of attention-for-action [13], and this gives the system several interesting properties and strengths: (1) it leads the system to learn self-tuned object-related inhibitions that allow it to disengage attention from scanned or non-relevant visual cues: this can be considered as an emergent inhibition-of-return mechanism commonly hardwired in attentional systems (e.g. [14, 16]); (2) it allows using similar neural structures and algorithms, such as reinforcement learning, to train both epistemic and pragmatic actions [13]; (3) it allows selecting the targets of pragmatic action, and triggering the latter, on the basis of attention processes: this lead to a strong integration of the *decision* and *parametrisation* of actions, as observed in real organisms' brains [20]; (4) it allows using the direction of gaze to furnish an implicit parameter to reaching actions: this simplifies

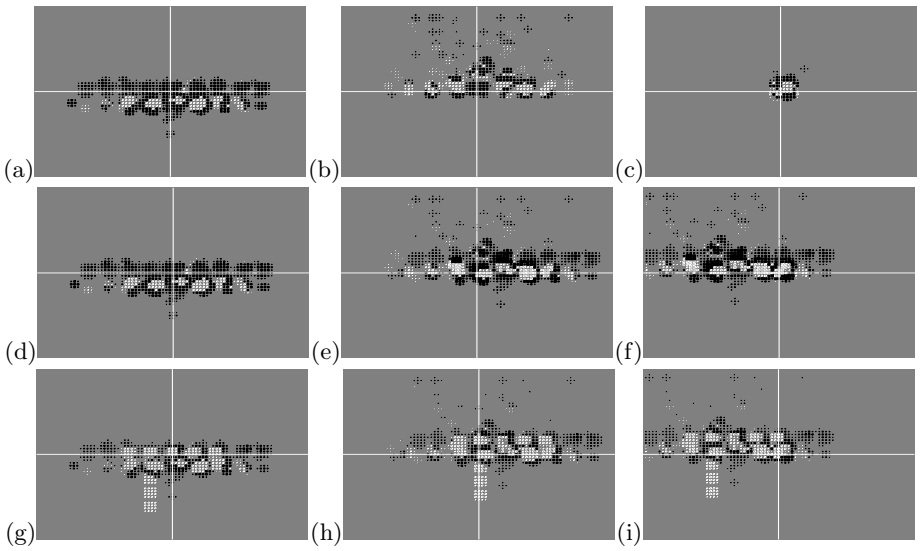

Fig. 4. Activation of the vote map (a-c), PAM (d-f), and saliency map (g-i) in a sequence of three saccades targeted to the foliage (a, d, g), trunk (b, e, h), and fruit (c, f, i) of the tree of the top graph of Fig. 1.c. White and black dots indicate neurons activated respectively above and below 0.5 (a-c) or above and below 0 (d-i).

computations as it allows extracting a simple and clean information for guiding action (the "where" of targets) from complex visual scenes [11, 12]. A further advantage produced by the integration is that the architecture does not need to be furnished the representation of "target objects" to which associate a reward signal, as it usually happens (cf. [8]) in other top-down attention learning systems (e.g., [15]). In fact. the reward produced by behaviour allows the system to autonomously build representations of objects that should trigger actions.

Notwithstanding its strengths, the model has various limits: a simplified feature extraction component, based on simple colour-detection, a simplified bottom-up attention component, based only on luminosity (see the components used in [14]), and a hardwired reset of the PAM memory when the scene changes (cf. [24]). However these limits, which will be tackled in future work, concern the specific system's components used here and not its overall architecture.

References

1. Marr, D.: Vision: A Computational Investigation into the Human Representation and Processing of Visual Information. W. H. Freeman, New York (1982)
2. Fermuller, C., Aloimonos, Y.: Vision and action. Image Vision Comput **13**(10) (1995) 725–744
3. Ballard, D.: Animate vision. Artif Intell **48** (1991) 57–86
4. Posner, M.I.: Orienting of attention. Q J Exp Psychol **32**(1) (Feb 1980) 3–25

5. Treisman, A.M., Gelade, G.: A feature-integration theory of attention. Cognit Psychol **12**(1) (Jan 1980) 97–136
6. Nolfi, S., Floreano, D.: Evolutionary Robotics: The Biology, Intelligence, and Technology. MIT Press, Cambridge, MA, USA (2000)
7. Floreano, D., Kato, T., Marocco, D., Sauser, E.: Coevolution of active vision and feature selection. Biol Cybern **90**(3) (Mar 2004) 218–228
8. Cliff, D., Noble, J.: Knowledge-based vision and simple visual machines. Philos T Roy Soc B **352**(1358) (Aug 1997) 1165–1175
9. de Croon, G., Postma, E.: Sensory-motor coordination in object detection. In: IEEE Symp ALIFE07. (2007) 147–154
10. Whitehead, S.D., Ballard, D.H.: Learning to perceive and act by trial and error. Mach Learn **7**(1) (July 1991) 45–83
11. Allport, D.: Selection for action: Some behavioral and neurophysiological considerations of attention and action. In: Perspectives on perception and action. Volume 15. Erlbaum, Hillsdale, NJ (1987) 395–419
12. Neumann, O.: Direct parameter specification and the concept of perception. Psychol Res **52**(2-3) (1990) 207–215
13. Balkenius, C.: Attention, habituation and conditioning: Toward a computational model. Cogn Sci Quart **1**(2) (2000) 171–204
14. Itti, L., Koch, C.: Computational modelling of visual attention. Nat. Rev. Neurosci. **2**(3) (Mar 2001) 194–203
15. Schmidhuber, J., Huber, R.: Learning to generate artificial fovea trajectories for target detection. Int J Neural Syst **2**(1-2) (1991) 135–141
16. Ognibene, D., Balkenius, C., Baldassarre, G.: A reinforcement-learning model of top-down attention based on a potential-action map. In: The Anticipatory Approach. Springer-Verlag, Berlin (2008)
17. Ognibene, D., Rega, A., Baldassarre, G.: A model of reaching that integrates reinforcement learning and population encoding of postures. In: 9th Int Conf Simul Adapt Behav, Springer (september 2006) 381–393
18. Pouget, A., Ducom, J.C., Torri, J., Bavelier, D.: Multisensory spatial representations in eye-centered coordinates for reaching. Cognition **83**(1) (Feb 2002) B1–11
19. Pouget, A., Zhang, K., Deneve, S., Latham, P.E.: Statistically efficient estimation using population coding. Neural Comput **10**(2) (Feb 1998) 373–401
20. Cisek, P.: Integrated neural processes for defining potential actions and deciding between them: a computational model. J. Neurosci. **26**(38) (Sep 2006) 9761–9770
21. Erlhagen, W., Schöner, G.: Dynamic field theory of movement preparation. Psychol Rev **109**(3) (2002) 545–572
22. Sutton, R., Barto, A.: Reinforcement Learning. MIT Press, Cambridge, MA (1998)
23. Dominey, P.F., Arbib, M.A.: A cortico-subcortical model for generation of spatially accurate sequential saccades. Cereb Cortex **2**(2) (1992) 153–175
24. Klein: Inhibition of return. Trends Cogn Sci **4**(4) (Apr 2000) 138–147
25. Herbort, O., Ognibene, D., Butz, M.V., Baldassarre, G.: Learning to select targets within targets in reaching tasks. In: IEEE 6th Intern Conf Development Learning. (Jul 2007) 7–12