Optimisation of Gaze Movement for Multitasking Using Rewards

Cem Karaoguz, Tobias Rodemann, Britta Wrede

2011

Accepted for IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 25-30 September 2011 in San Francisco, USA

Optimisation of Gaze Movement for Multitasking Using Rewards

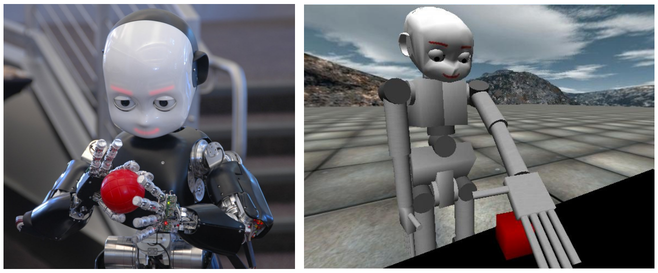
Cem Karaoguz[1,2], Tobias Rodemann[2], Britta Wrede[1]

Abstract— **Domestic tasks such as grasping or navigation for robotic systems can be supported by vision. However, the environment provides a vast amount of visual information and concentrating on the information related to the task being undertaken is an important job. Active vision is an approach that provides such a filtering mechanism by using camera movements to bring relevant information into the focus of attention. However timing of gaze shifts (i.e. when to look where) is crucial for cognitive tasks to proceed simultaneously (multitasking). We developed a framework that learns task dependent management of gaze control. We adopted a systems approach where individual visual processes were formalised as modules such as a colour saliency module or object recognition module. Modules may generate motor commands for gaze shifts to acquire visual information relevant to their operation. The system learns how to use its modules (i.e. when to give motor control access to which module) for a task in a reward-based concept. The framework was used in a reaching-while-interacting scenario using the humanoid iCub in a simulation environment.**

I. INTRODUCTION

The information our environment presents to us is simply too vast to deal with in its entirety. In the visual domain, humans employ eye movements in order to seek and acquire task-relevant information. Yarbus led the study of visual exploration by recording eye movements of observers examining natural scenes showing that eye movement patterns were heavily influenced by the task that is being undertaken [21]. Recent findings were compiled by Land et al. who observed fixation patterns of humans in tasks like reading, tea making, driving, or playing ball games [13]. One of their conclusions was that the human vision system has to resort to time-sharing for multitasking where multiple stimuli have to be monitored concurrently. This is achieved via sequential gaze shifts on the stimuli that are relevant to tasks being undertaken at the moment.

An approach in which a visual system is able to adjust its visual parameters to aid task oriented behaviour is called active vision [1]. Such an approach may also benefit artificial systems that use vision as the primary sensory instrument. However, an application of the active vision approach to modular large scale systems may impose the "when to look where" problem. Usually such systems employ different cognitive processes (i.e. modules) running in parallel. In such systems modules may produce conflicting motor commands for gaze-shifts. If a system has to fulfill multiple tasks at the same time, decisions on when and where to employ gaze-shifts made during the execution of tasks may be crucial. For example, for a robot that has to navigate and localise specific objects at the same time, an obstacle detection module may require visual information from the nearest object while an object search module may intend to fixate on an arbitrary object further away. The conflict can be resolved by prioritising modules, however, it is often difficult to form an optimum hierarchy among modules in real world scenarios since the importance of modules may differ from task to task.


(a)

(b)

Fig. 1. The iCub humanoid robot and the iCub Simulator.

We propose a solution where the utilisation of gaze control among different modules of a large-scale system is optimised with a reward-based learning mechanism. Several examples of such large-scale interacting audio-visual systems were previously presented [2], [5], [8], [9]. These systems combine a number of modules performing different cognitive operations such as object recognition, object tracking or audio-visual saliency-based attention control. Combining these modules to construct a system requires an integration and arbitration system that selects which of these modules is given control of the gaze direction at any point in time.

To demonstrate the utility of the framework a similar system was constructed and two scenarios with different level of complexity were applied to the system using the iCub humanoid in a simulation environment (Fig. 1) [3], [17]. In the first scenario, the task for the robot was to locate, reach and touch an object that is continually moving on a table. In the second scenario an interaction partner was introduced. The main task was still to locate and reach the moving object but the system had to be sociable towards the interaction partner at the same time. We will show that using our framework the system is able to learn optimum strategies for both scenarios.

[1] Research Institute for Cognition and Robotics (CoR-Lab), Bielefeld University, 33594 Bielefeld, Germany {ckaraogu, bwrede}@cor-lab.uni-bielefeld.de

[2] Honda Research Institute Europe GmbH, Carl-Legien-Str. 30, 63073 Offenbach, Germany tobias.rodemann@honda-ri.de

II. RELATED WORK

The problem being addressed is an optimisation issue in the most general sense. Allocating access to gaze control among several independent agents in a certain time scale has to be optimised in order to provide the system with maximum benefit. A straightforward approach to solve this problem is pre-programming involving definition of every state-action pair. While this may be applicable to simple systems, it becomes intractable as the system grows more complex. Prioritisation or subsumption is another method where different modules are given different priorities and the one with the highest priority among the active components has access to the motor resources [6]. This approach may work for more complex systems which pre-programming cannot extend to, however, it may not be applicable for dynamic environments and multiple tasks where the priorities are not fixed or conflicting. Moreover, setting priorities for modules implies manual programming as in the pre-programming case.

Wolpert introduced the Probability Collectives (PC) framework that addresses the problem of optimisation of systems composed of multiple independent agents [20]. The agents can modify their policy in order to maximise their utility function (or minimise the cost function). The process reaches an equilibrium when the agents can no longer improve their rewards by changing actions. The core insight of PC theory is to concentrate on how the agents update the probability distributions across their possible actions. A global system utility can be defined for guiding this optimisation process. Several applications of this framework have been presented [19], [4]. We propose to achieve a similar process via a learning based method.

Regarding specific applications in the vision domain, Itti et al. proposed a saliency based gaze control scheme that solves the problem of where to launch gaze-shift by simply selecting the most conspicuous location as gaze target [11]. However, this approach does not address the problem of when to launch a gaze-shift. Additionally, this scheme models only the bottom up attention system. Although later advancements for this framework have been done in order to encompass top down modulation, it is disputable that such models can really represent human like gazing behaviour in complex tasks [13].

A reinforcement learning based method was introduced by Sprague et al. where policies for selecting actions generated by different microbehaviours were learnt [15]. A microbehaviour is defined as a complete sensory-motor routine incorporating information acquisition from the environment and acting on it to achieve specific goals. Learnt policies were used to arbitrate the gaze control among different microbehaviours by estimating the cost of uncertainty over the microbehaviours. While they present an elegant way of solving the problem of gaze arbitration, decisions on which microbehaviours were relevant for which tasks were done based on a pre-programmed scheme. In our framework the selection of task-relevant modules emerge in the course of learning autonomously.

Reward-based methods have previously been used for cue integration applications [18]. Our framework may also be employed to similar applications where temporal characteristics of the cues have an impact on the integration process. For example, in a previous work we investigated different visual depth estimation methods for humanoid applications within interaction range [12]. These methods vary not only in their performance but also in their requested execution time. For an application where different methods have to be combined for better estimations, it may be beneficial to use the reward-based framework to learn when it pays off to execute additional, time consuming methods for distance estimation.

III. CONCEPT

A selection mechanism was implemented to arbitrate the motor commands for gaze-shifts received from the modules. We developed a gaze control scheme where a weight is assigned to each module defining the basic dynamics of the gaze control (Fig. 2). A reward mechanism generates reinforcement signals for the weights with respect to the outcomes of attempts to fulfil the task.

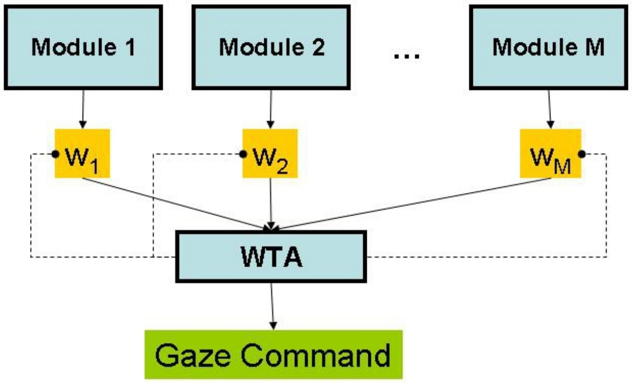


Fig. 2. The proposed gaze control mechanism. Every module is attached to a weight and the module of which motor command is selected for execution inhibits the output of other modules (inhibitory links are shown as dashed lines).

A. Gaze Control

A weight w_j was assigned to each module that determines the strength of the module in a competition of access to the motor control among others. The winner module inhibits the output of the other modules for a specific time determined by a decay rate parameter τ_j. We use 2D Gaussian (called activation maps) to encode the motor command generated by a module. Horizontal and vertical dimensions of an activation map correspond to motor command space relative to current posture of cameras in pan and tilt directions respectively. The peak position in the map is the location of the gaze command. The whole gaze control process can be formalised as:

$$S = \sum_{j=1}^{M} w_j n_j S_j, \qquad (1)$$

where S is the resulting activation map, S_j is the activation map from module j (computations of individual activation maps are explained further in the paper), w_j is the weight assigned to module j and n_j is the inhibition value for module j and computed as:

$$n_j = \begin{cases} 1 & \text{if } j = k, \\ e^{-\tau_k c} & \text{otherwise;} \end{cases} \tag{2}$$

where k is the index of the currently active module, τ_k is the decay rate of module k (decay rates were set to a constant value) and c denotes the current time of an internal clock that measures the total selection period of a module. When a module is first selected to gain the motor control access the clock is set to zero. This is when the inhibition of non-selected modules is at its peak. Elapsed time (i.e. increasing value of c) diminishes the strength of this inhibition (Eq. 2), resulting in an increase of the probability of other modules being selected (Eq. 1). When the strength of the inhibition drops below a threshold e_{inh} a new selection is made (this could be the reselection of the previously selected module) and the clock is set to zero again. The motor command to be executed is sampled randomly from a distribution over the motor space acquired from the total activation map S. The Boltzmann distribution that provides a decent exploration/exploitation balance for action selection was used for the computation of the distribution [16]:

$$p(\theta^P = p, \theta^T = t) = \frac{e^{S(p,t)/Z}}{\sum e^{S(\theta^P,\theta^T)/Z}}, \tag{3}$$

where (θ^P, θ^T) denote the pan and tilt dimensions of the activation map, (p, t) denote the pan and tilt values respectively and Z is a positive parameter called temperature. Low temperatures cause a greater difference in selection probability for actions that differ in their value estimates, while the actions become more equi-probable for higher temperatures.

The proposed gaze control mechanism is similar to the Winner-Take-All approach that is extensively used in neural networks [14]. Also, Eq. 1 resembles the simple saliency based gaze control approach where activations from several bottom-up saliency mechanisms are summed up (explained further in this section). In our approach such a bottom-up saliency mechanism may emerge by setting all module weights to one or a pre-defined model which can be used in periods where the system has no task to fulfil.

B. Learning

The time window between the start and end of a task realisation process is called an epoch. For every successful outcome of an epoch a reward r was generated as:

$$r = 1 + \frac{t_{max} - t}{t_{max}}, \tag{4}$$

where t_{max} is the maximum epoch time, $t < t_{max}$ is the timestep in which the reaching action was concluded. This increases the value of the tasks fulfilled in a shorter time. For unsuccessful outcomes of tasks no reward is generated. The reward is then used to update the weights of modules following an unsupervised learning rule:

$$w^{e+1}_{B(k)} = w^{e}_{B(k)} + \alpha \cdot \gamma^k \cdot r \cdot w^{e}_{B(k)}, \tag{5}$$

where $B(k)$ is a list that contains the indices of the selected modules in the last t_{hist} timesteps (i.e. $k \in 1, 2, ...t_{hist}$) of the epoch e, α is the learning rate and γ is the discount factor. Inspired from the Hebbian theory, the learning rule reinforces the weights of modules, which were selected during the course of the epoch [10]. The buffer size t_{hist} controls how far the rewards were propagated in history. This mechanism ensures that a reward is propagated over the modules which contributed the rewarding outcome from a series of actions. The propagation of rewards is similar to the eligibility traces used in reinforcement learning [16]. The relative values of weights of modules, which contributed a positive outcome, further increased by blunting the weights of the modules, which were not selected in the last t_{hist} timesteps:

$$w^{e+1}_{B(k)'} = w^{e}_{B(k)'} \cdot (1 - \epsilon), \tag{6}$$

where $B(k)'$ denotes the indices of modules other than $B(k)$ and ϵ is the blunting factor. This operation is called decision sharpening in TransSARSA framework and provides rapid learning by honing the value of states, which lead to positive outcomes [7]. Learning rate α and blunting factor ϵ decays exponentially over time to keep the values of the weights stable:

$$\alpha = \alpha_0 e^{-t/\tau_\alpha},$$
$$\epsilon = \epsilon_0 e^{-t/\tau_\epsilon};$$

where α_0 and ϵ_0 are initial values and τ_α and τ_ϵ are decay rates. Weights are initialised to one to achieve equi-probable weight distribution in the beginning and they are kept between 0 and 1 during the learning process.

IV. APPLICATION

The utility of the concept was demonstrated in a *reaching-while-interacting* scenario where the system has to locate, reach and touch a moving object without losing the attention of an interaction partner. The position information about the object is only updated when the object is in the field of view. Additionally, the system has to look at the interaction partner every once in a while otherwise the partner gets bored and the scenario cannot be fulfilled. The system has no prior knowledge about how often it has to attend to the interaction partner. The gazing behaviour has to be optimised in order to fulfil these tasks simultaneously.

An overview to the system is shown in Fig. 3 which is composed of two major parts: a limb control mechanism that controls the arm of the robot to manipulate the environment and a vision architecture that supports the limb control mechanism with information about the environment. The vision architecture is composed of several modules that are

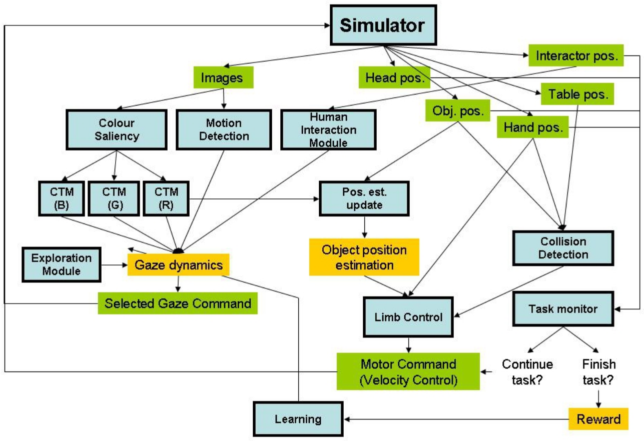


Fig. 3. An overview to the system on which the proposed framework was applied. Six independent modules are responsible for various visual processing operations and they may generate motor commands on demand. Gaze dynamics are shown in Fig. 2 in detail. Limb control and collision detection operations are responsible for reaching movement. A task monitor generates rewards depending on the state of the system and the learning component uses rewards for weight updates.

responsible for various visual processes and may generate motor commands for gaze-shifts to various areas of the visual field. The following subsections explain the basic building blocks of the system in detail.

A. *Vision Architecture*

The architecture used in the experiments employed the following modules: the *Colour Tracking Modules*, *Visual Exploration Module*, *Motion Detection Module* and *Human Interaction Module*. Since the modules are independent, internal parameters of modules were selected manually in the way that each module shows an optimum performance.

Every module $j \in 1, 2, ...M$ (M being the number of modules) generates an activation map highlighting desired motor commands represented as a 2D Gaussian:

$$S_j(\theta^P, \theta^T) = \mu_j \cdot \exp\Big(-\frac{1}{2}\Big[\frac{(\theta_j^P - \theta^P)^2}{\sigma_{j,P}^2} + \frac{(\theta_j^T - \theta^T)^2}{\sigma_{j,T}^2}\Big]\Big), \tag{7}$$

where (θ^P, θ^T) denote the pan and tilt dimensions of the activation map respectively, (θ_j^P, θ_j^T) is the motor command selected by module j, μ_j is peak value and $\sigma_{j,P}$ and $\sigma_{j,T}$ are spreads in pan and tilt dimensions respectively. μ was set to the maximum of the conspicuity map for Colour Tracking Modules and 1 for the rest of the modules. Spread parameters were fixed for all modules.

1) Colour Tracking Module (CTM): For detection of coloured objects (red, green and blue) a colour saliency computation was done. The conspicuous maps C_i were computed via the colour opponency method [11]:

$$C_i(x, y) = \sum_{i'} I_i(x, y) - (\eta I_{i'}(x, y) + \vartheta), \tag{8}$$

where i is the colour index the conspicuous of which is being computed, i' denotes the colours other than i and $I_n(x, y)$ indicates the pixel value at position (x, y) on colour channel n. Threshold parameters were set to $\eta = 1.2$, $\vartheta = 10$. Conspicuity maps were used by three CTMs each of which was assigned a colour from red, green and blue. These modules generate motor commands for gaze-shifts towards the most conspicuous position if the peak value is above a certain threshold which was set to 0.8.

2) Visual Exploration Module (VEM): Generating motor commands for gaze-shifts towards positions which were not visited for long time was done by the VEM. An exploration map P_{VEM} covering the whole head motor space in two dimensions (i.e. pan and tilt) was used for this purpose. At every timestep, values of the map were updated in the following way[1]:

$$P_{VEM}(\theta^P, \theta^T) = P_{inh}(\theta^P, \theta^T) \cdot (P_{VEM}(\theta^P, \theta^T) + \xi). \tag{9}$$

This involves an increment of all values by an amount given by parameter ξ (set to 0.01) and an inhibition of the values around the current gaze position $(\overline{\theta^P}, \overline{\theta^T})$ done by:

$$P_{inh}(\theta^P, \theta^T) = 1 - \exp\Big(-\frac{1}{2}\Big[\frac{(\overline{\theta^P} - \theta^P)^2}{\sigma_P^2} + \frac{(\overline{\theta^T} - \theta^T)^2}{\sigma_T^2}\Big]\Big), \tag{10}$$

where σ_P and σ_T determine the spread of the 2D Gaussian in the pan and tilt dimensions respectively and both were set to 2. Subsequently the probability distribution map was normalised as it sums to one and a single motor command is sampled randomly using this distribution. The VEM generates motor commands at every timestep.

3) Motion Detection Module (MDM): In order to detect motion in the visual field the temporal difference of images was computed via the MDM. Whenever a motion is detected, a motor command is generated towards this position. Since this basic method cannot deal with ego-motion, the module does not perform any computation during gaze-shifts.

4) Human Interaction Module (HIM): The capability of localising humans in the environment enhances the social aspects of robotic systems. Various cues not only in vision domain, but also in audio domain can be used for this purpose. Face detection and sound localisation are two operations that can be used for human localisation using such cues. The HIM simulates a similar module that would produce gaze shifts towards an interaction partner present in the scene. Due to difficulties in generating both auditory (e.g. sound) and visual (e.g. face) elements using the simulator, the assumed position of the interaction partner (in front of the robot, behind the table) is given to the system manually.

B. *Limb Control and Reaching*

The task, reaching a red coloured object moving on the table, has to be fulfilled in a specific time determined by a parameter t_{max}, otherwise it was considered as failure. The reaching process is fulfilled by minimising the distance between the position of the left hand and perceived object

[1] Products indicate element-wise multiplication.

position. The position of the hand is acquired directly from the simulator (i.e. proprioception). For object position an internal memory was created. The memory is updated whenever the object resides in the field of view (i.e. having its projection on the camera images). When the object was not visible, the memory was not updated but the object was still moving. An uncertainty factor was also applied to the memory that would introduce a noise proportional to the time the object was not visible. Two degrees of freedom from the arm group of joints and one degree of freedom from the waist group of joints were used to move the hand in three dimensions for reaching. The motion of the hand was realised by velocity control. Velocity commands were generated/updated in a ballistic fashion (i.e. no feedback control was used) using the errors between the hand position and the perceived object position. A reaching action was fulfilled if the euclidean distance between hand and object was below a threshold e_{reach}.

Apart from the reaching process, a collision detection mechanism was implemented to monitor the position of the hand, and other items in the environment and detect any contact of the hand with other items in the environment (e.g. the table). In case of a collision the reaching process resumes after the hand is moved back to a safer location in the vicinity.

C. Reward Scheme

Every successful completion of the task was rewarded as explained in Eq. 4. This involves completing the reaching action without losing the social interaction with the partner. If reaching time exceeds a pre-defined threshold value ($t > t_{max}$) the task was considered unsuccessful. Additionally, the interaction partner encoded a boredom variable that increased linearly during the time the robot did not look at her and was set to zero otherwise. The rise of the boredom was determined by a boredom rise rate parameter τ_b. If the boredom reached one the interaction partner was considered as bored and the scenario failed. For unsuccessful outcomes no reward was generated.

V. RESULTS

A. Simulation Layout

A snapshot from the simulator and the major components of the system running on the simulator is shown in Fig. 4. The iCub Simulator was used as the simulation engine [17]. Fig. 5 shows the flowchart of simulations. Two sets of experiments were conducted: the first set was designed for proof of concept. In these experiments the interaction partner was excluded and the task was just to locate and reach an object. The second set of experiments included the interaction partner in order to form a multitasking scenario. All experiments included three objects on the table in red, green and blue colours. The red object was the target for reaching and continually moving on the table. Green and blue objects were distractors. The parameters were set empirically. For modules, the parameters were chosen to maximise individual performance of each module. For the gaze control and limb control systems the parameters did not heavily affect the outcome of the scenario. The learning parameters were set using a more simple simulation framework developed for rapid prototyping. All parameters are shown in Tab. I.

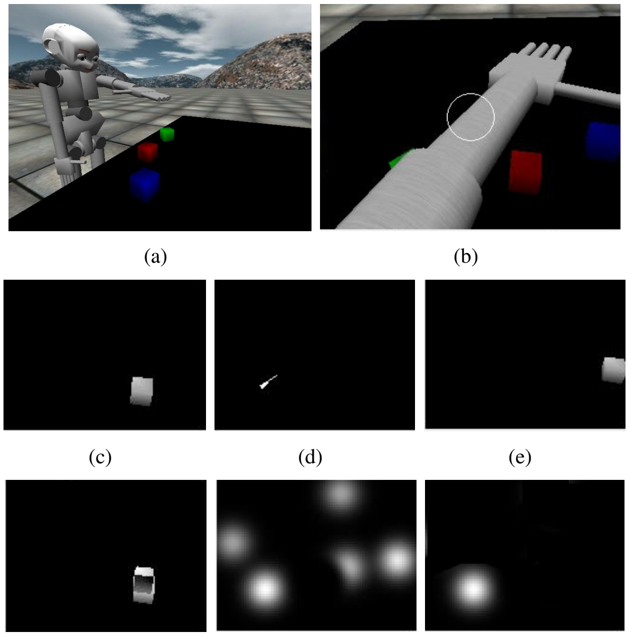

(a) (b) (c) (d) (e) (f) (g) (h)

Fig. 4. A snapshot of the system during a reaching action. (a) An external view of the robot and the environment from the simulator, (b) view from the left camera with white circle indicating the centre of the image, (c)-(e) colour conspicuous for red, green and blue colours, (f) result of motion detection, (g) combined activation map computed as in Eq. 7, (h) selected motor command for gaze-shift (relative to current posture).

For comparison, two additional gaze control mechanisms were implemented. One of these is *saliency*, a simple application of a saliency-based bottom-up attention mechanism presented by Itti et al. [11]. This was carried out by generating a cumulative saliency map via the superposition of activation maps received from all modules:

$$S_{sal} = \sum_{j=1}^{M} S_j. \tag{11}$$

The most conspicuous location in the resulting saliency map was selected as the motor command for a gaze-shift if its value exceeds a previously defined threshold. An inhibition of return procedure was applied to the saliency map as explained in Sec. IV-A.2 to prevent previously visited locations to reappear on the saliency map too soon.

A second method, *pre-programming*, prescribes setting up heuristics for modules to follow in certain conditions. Since the main task of the system is to locate and reach a specific coloured object, intuitively a good strategy is to use the VEM when the object being reached to is not in the visual field and use the CTM tuned to the colour of the object when the object is found. It will be shown that this method was competent to solve the reaching task alone (i.e. scenario without an interaction partner). When it comes to

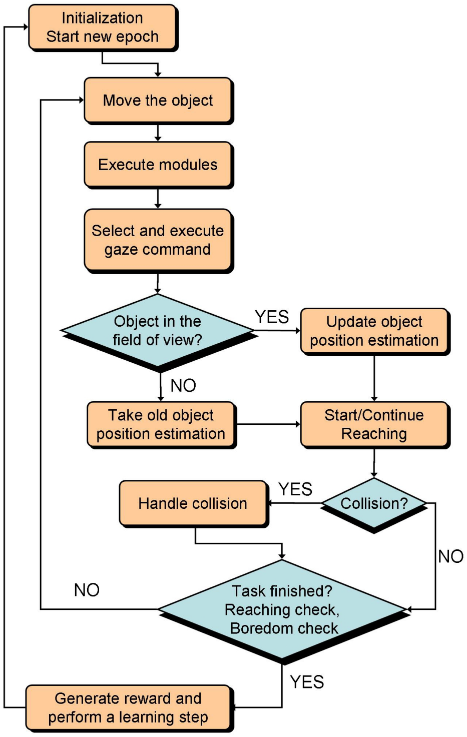


Fig. 5. The simulation flowchart. Before each epoch the robot was moved to its initial posture, which prescribes a straight arm position over the table, and the objects were randomly distributed on the table. At every timestep in an epoch the object is moved randomly in horizontal and depth axes (± 5 cm.)

reaching-while-interacting scenario the method is expected to fail since it does not have any knowledge about the second task. Of course a second pre-programmed method can be implemented to deal with the new situation. However, in the real world it is often not very easy to infer such heuristics. In such cases the system has to live with its current capabilities. We want to show here that our proposed solution can cope with both situations.

TABLE I
PARAMETER VALUES USED IN THE EXPERIMENTS.

Simulator		**Limb Control**	
Image size	160 × 120	t_{max}	25
τ_b	0.2	e_{reach}	0.12
Modules		**Learning**	
μ_j, $(j \neq CTM)$	1	α_0	0.05
$\sigma_{j,P}$	5	γ	0.9
$\sigma_{j,T}$	5	ϵ_0	0.001
Gaze Control		τ_α	250
Z	0.25	τ_ϵ	250
τ_j	0.95	t_{hist}	10
e_{inh}	0.1		

B. Experiment 1: Simple scenario

The first set of experiments were conducted excluding the role of the interaction partner. The task in this more simple scenario is just to locate and reach a specific object. The *pre-programming* method for gaze control was specifically designed to solve this task. Comparing results from the *pre-programming* and the proposed framework we can see if our concept can learn a good strategy to fulfil this basic task.

Fig. 6 shows the performance of three methods as the amount of acquired rewards with time (averaged over last 100 epochs). It is clear that our framework can learn a strategy as good as a pre-programming method would achieve. The performance of methods were very close to each other: the task has been achieved 94%, 84% and 92% of the time with *pre-programming*, *saliency* and the proposed method respectively.

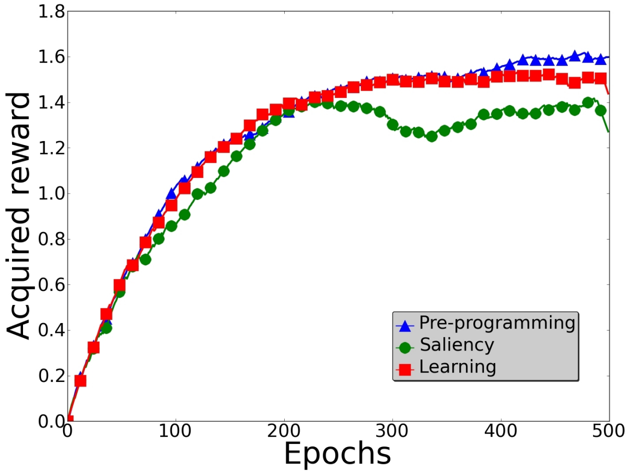


Fig. 6. Mean of last 100 acquired rewards in a simple reaching scenario.

C. Experiment 2: Reaching-while-interacting

The second set of experiments incorporated an interaction partner in the scenario. The remaining elements were the same as in the first set of experiments. The main task was still to locate and reach the moving red coloured object but this time the system had to attend to the interaction partner simultaneously by gazing at her from time to time.

Fig. 7 shows the acquired rewards of the three methods for the multitasking scenario. Clearly, the *pre-programming* method specifically designed for the reaching task was not sufficient for the multitasking scenario. The scenario was also too complex for the *saliency* method. The proposed framework was able to learn an optimum distribution of gaze control among various modules to fulfil the desired scenario. Success rates of the task were 38%, 51% and 63% with *pre-programming*, *saliency* and the proposed method respectively. Please note that these statistics include the training time for the proposed method as well.

The learning time was around 200 epochs. Average runtime of an epoch was 10.6 timesteps in learning experiments and every timestep takes approximately one second. This is equal to 35 minutes of learning time which is a considerably short period compared to common reinforcement learning methods. The system performed 8.7 gaze-shifts on average during the learning time. This means approximately 1800 gaze-shifts were enough to learn the presented multitasking behaviour.

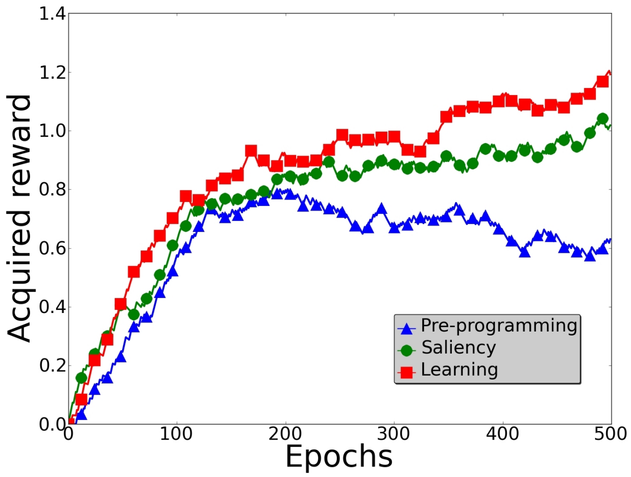


Fig. 7. Mean of last 100 acquired rewards in a *reaching-while-interacting* scenario.

VI. CONCLUSION

For vision systems, when multiple tasks have to be achieved at the same time a scheduling mechanism is necessary in order to distribute motor resources of the system among different cognitive processes dealing with these tasks. An adaptive framework was introduced where the utilisation of gaze control among different tasks is learnt through a reward mechanism. The framework was demonstrated in a reaching scenario where an effective resource allocation policy was necessary to gain maximum reward. The system eventually learnt such a policy. The results showed that it is possible to apply the methodology to large-scale systems that have to schedule different tasks and this is more beneficial than strategies using fixed policies.

One of the major benefits of the proposed framework is that it can reveal task-relevant modules in a system and use them. This may be applied to resource constrained systems, in which parallel execution of modules is restrained. It was also shown that learning was accomplished in a short time. On the other hand, since the concept is based on a reward mechanism, the credit assignment is crucial. For instance, even though the gaze control mechanism followed a good strategy, reaching process may fail due to some other reason. Gaze control mechanism should not be penalised in such cases. In this work we established a stable environment by using the same limb control strategy and having high number of trials in all experiments. However, in a real world application it is important to make accurate credit assignments.

Future work entails migration of the system to the real robot. Shifting from a controlled simulation environment to the real world brings technical challenges in its wake. Our adaptive framework may assist addressing difficulties related to operations of visual processing modules by finding the best sets of modules for specific tasks. Additionally, decay rates for modules were set to a constant value in this work. Adaptation framework will be extended to these parameters as well that will allow learning temporal sequencing of modules.

VII. ACKNOWLEDGEMENTS

The work described was supported by the Honda Research Institute Europe.

REFERENCES

[1] J. Alomoinos, I. Weiss, and A. Bandopadhay. Active vision. *International journal on Computer Vision*, 1988.

[2] A. Andreopoulos, S. Hasler, H. Wersing, H. Janssen, J. Tsotsos, and E. Koerner. Active 3D object localization using a humanoid robot. In *IEEE Transactions on Robotics*. IEEE, 2011.

[3] R. Beira, M. Lopes, M. Praga, J. Santos-Victor, A. Bernardino, G. Metta, F. Becchi, and R. Saltaren. Design of the robot-cub (iCub) head. In *Robotics and Automation, 2006. ICRA 2006. Proceedings 2006 IEEE International Conference on*, pages 94–100, June 2006.

[4] Stefan R. Bieniawski, Ilan M. Kroo, and David H. Wolpert. Flight control with distributed effectors. In *Proceedings of the AIAA Guidance, Navigation, and Control Conference*, 2005.

[5] B. Bolder, H. Brandl, M. Heracles, H. Janssen, I. Mikhailova, J. Schmuedderich, and C. Goerick. Expectation-driven autonomous learning and interaction system. In *IEEE-RAS International Conference on Humanoid Robots*, 2008.

[6] R. Brooks. A robust layered control system for a mobile robot. *Robotics and Automation, IEEE Journal of*, 2(1):14 – 23, March 1986.

[7] Benjamin Dittes and Christian Goerick. Unsupervised self-development in a multi-reward environment. *Proceedings of the 10th International Workshop on Epigenetic Robotics*, 2010.

[8] C. Goerick, B. Bolder, H. Janssen, M. Gienger, H. Sugiura, M. Dunn, I. Mikhailova, T. Rodemann, H. Wersing, and S. Kirstein. Towards incremental hierarchical behavior generation for humanoids. In *IEEE-RAS International Conference on Humanoids 2007*. IEEE, 2007.

[9] C. Goerick, J. Schmuedderich, B. Bolder, H. Janssen, M. Gienger, A. Bendig, M. Heckmann, T. Rodemann, H. Brandl, X. Domont, and I. Mikhailova. Interactive online multimodal association for internal concept building in humanoids. In *IEEE-RAS International Conference on Humanoids 2009*. IEEE, 2009.

[10] D. O. Hebb. *The Organization of Behavior: A Neuropsychological Theory*. Psychology Press, new edition edition, June 2002.

[11] L. Itti, C. Koch, and E. Niebur. A model of saliency-based visual attention for rapid scene analysis. *IEEE Trans. Pattern Anal. Mach. Intell.*, 20(11):1254–1259, Nov 1998.

[12] C. Karaoguz, A. Dankers, T. Rodemann, and M. Dunn. An analysis of depth estimation within interaction range. In *Proceedings of Int. Conf. on Intelligent Robots and Systems*, 2010.

[13] M. F. Land and B. W. Tatler. *Looking and Acting*. Oxford University Press, 2009.

[14] Matthias Oster, Rodney Douglas, and Shih-Chii C. Liu. Computation with spikes in a winner-take-all network. *Neural computation*, 21(9):2437–2465, September 2009.

[15] N. Sprague, D. Ballard, and A. Robinson. Modeling embodied visual behaviors. *ACM Trans. Appl. Percept.*, 4(2):11+, 2007.

[16] R. S. Sutton and A. G. Barto. *Reinforcement Learning: An Introduction (Adaptive Computation and Machine Learning)*. The MIT Press, March 1998.

[17] V. Tikhanoff, P. Fitzpatrick, G. Metta, L. Natale, F. Nori, and A. Cangelosi. An open source simulator for cognitive robotics research: The prototype of the iCub humanoid robot simulator. In *Proceedings of the Workshop on Performance Metrics for Intelligent Systems, National Institute of Standards and Technology, Washington DC, USA*, August 2008.

[18] T. Weisswange, C. Rothkopf, T. Rodemann, and J. Triesch. Can reinforcement learning explain the development of causal inference in multisensory integration? In *Proceedings of the IEEE 8th International Conference on Development and Learning (ICDL)*. IEEE, 2009.

[19] David Wolpert and Nilesh Kulkarni. Game-theoretic management of interacting adaptive systems. In *Proceedings of NASA/ESA Conference on Adaptive Hardware and Systems*, 2008.

[20] David H. Wolpert. *Information Theory - The Bridge Connecting Bounded Rational Game Theory and Statistical Physics*. Perseus books, February 2004.

[21] A. F. Yarbus. *Eye Movements and Vision*. New York, Plenum P., 1967.