# Gaze Allocation Analysis for a Visually Guided Manipulation Task

Jose Nunez-Varela*[1], B. Ravindran[2], and Jeremy L. Wyatt[1]

School of Computer Science, University of Birmingham,
B15 2TT Birmingham, UK
{j.i.nunez,j.l.wyatt}@cs.bham.ac.uk
Department of Computer Science and Engineering, IIT Madras,
600 036 Chennai, India
ravi@cse.iitm.ac.in

**Abstract.** Findings from eye movement research in humans have demonstrated that the task determines where to look. One hypothesis is that the purpose of looking is to reduce uncertainty about properties relevant to the task. Following this hypothesis, we define a model that poses the problem of where to look as one of maximising task performance by reducing task relevant uncertainty. We implement and test our model on a simulated humanoid robot which has to move objects from a table into containers. Our model outperforms and is more robust than two other baseline schemes in terms of task performance whilst varying three environmental conditions, reach/grasp sensitivity, observation noise and the camera's field of view.

**Keywords:** Gaze control, reinforcement learning, decision making

## 1 Introduction

Acting under uncertain and incomplete information is required every time humans engage in some task. This uncertainty can be reduced by looking at relevant parts of the environment. The development of portable eye trackers has made it possible to study the role of human gaze during natural tasks, such as driving, tea and sandwich making, sports activities, etc. [10, 12]. The key findings are that, where we look is determined by the task being performed, the purpose of looking is to reduce task relevant uncertainty, and that gaze patterns reflect extensive learning at several levels [12]. Furthermore, Johansson et al. [9] studied in detail the temporal and spatial relation between manipulation actions and gaze. Subjects had to grasp a bar and move it in a vertical plane to a target point, whilst avoiding an obstacle. The main conclusion was that gaze fixations specify landmark positions to which the manipulation actions are subsequently directed. In addition, empirical evidence from neurophysiological experiments has demonstrated that the human brain controls gaze via reward signals [8].

* We gratefully acknowledge the support given by CONACYT-Mexico (Reg.179604), UKIERI (SA06-0031), CogX (FP7-ICT-215181).

In order to increase our understanding about how gaze can be coordinated, it is useful to create models that can be formally analysed in different conditions. Building on the work of Sprague et al. [11], we have defined a gaze allocation model based on a reinforcement learning (RL) framework [14] that allows for the control of the oculomotor system, multiple parallel motor systems, actions of variable duration, and that reasons about the expected actual reduction in uncertainty due to possible perceptual actions [4]. We implement and test our model using the iCub simulator[1] [2] (Fig. 1.A). Our manipulation task consists of picking up objects from the table top and then placing them inside one of two containers. In this work, the arms cannot interact with each other (e.g. to perform bi-manual grasps), so the task is divided into two sub-tasks, one for each arm. Thus, objects reachable by the right arm are placed inside the right container, with the same happening for the left side. A new object appears on the table every 60 seconds and also every time an object is put inside a container. We reward the robot for task performance. The key idea is to select the gaze at any moment that increases task rewards most by reducing task relevant uncertainty about the location of objects and containers. The only precisely known location to the robot is the centre of the table. In this paper we define our reward based model and characterise how task performance varies in terms of three environmental conditions, namely reach/grasp sensitivity in manipulation actions, the level of observation noise, and the size of the camera's field of view.

Besides Sprague et al. model, another reward based gaze control model was presented in [5] for a multitasking scenario. However, they do not consider multiple motor systems, only two fixation points are considered, and it is not a manipulation task. Other works have approached the problem of where to look based on models of saliency in order to detect regions of interest which may provide possible fixation points [16]. Nevertheless, these systems do not explicitly reason about how to reduce the uncertainty relevant to the task.

## 2 Coordinating Gaze and Actions

In this section we first describe how the task is modelled, then we explain the different components of our system and their interaction as Fig. 1.B illustrates. The robot initially learns the task and then performs the task by deciding what actions to perform and where to look.

### 2.1 Modelling the task

As described in Section 1, the task is to pick up objects from the table top and then place them inside one of two containers. We consider two motor systems: the right and left arm/hand. We divide the task into two sub-tasks, one for each motor system. Furthermore, each sub-task is modelled with two levels. The

[1] The main advantage of using the iCub simulator is that it works with the same controllers used by the real robot. Thus, the transition between the simulation and the real robot is relatively straightforward.

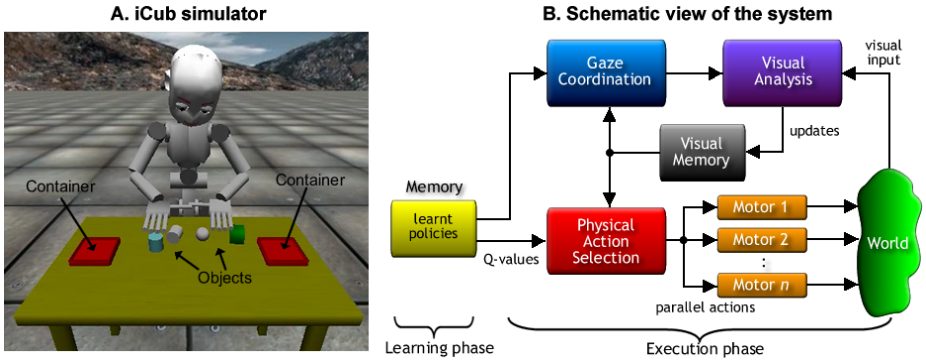


**Fig. 1.** A. Snapshot of the iCub simulator, where the task is to pick up and place objects from the table top to the containers. B. Schematic view of the system.

high-level models the sub-task qualitatively as a semi-Markov decision process (SMDP)[2] [13], with a discrete state representation. Whereas the low-level models each high-level action with a continuous state space representation.

In the high-level SMDP, each motor system $ms \in MS$ (where $MS$ is the set of motor systems), is modelled as a tuple $\langle \mathcal{S}_{ms}, \mathcal{O}_{ms}, \mathcal{T}_{ms}, \mathcal{R} \rangle$, where $\mathcal{S}_{ms}$ is the set of discrete states, $\mathcal{O}_{ms}$ is the set of temporally extended high-level actions (called *options* as in [7]), $\mathcal{T}_{ms} : \mathcal{S}_{ms} \times \mathcal{O}_{ms} \times \mathcal{S}_{ms} \times \mathbb{N} \rightarrow [0, 1]$ is the transition probability distribution, where $\mathbb{N}$ is the set of natural numbers representing the execution time of each option, and $\mathcal{R} : \mathcal{S}_{ms} \times \mathcal{O}_{ms} \rightarrow \mathbb{R}$ is the reward function. For this domain, the reward function is identical for all motor systems.

In our case, an option $\mathcal{O}^j_{ms} \in \mathcal{O}_{ms}$ is modelled with continuous states ($\mathcal{S}^j$) and actions ($\mathcal{A}^j$), and with a non-stochastic transition function ($\mathcal{T}^j$). Options are defined as commands provided by the motor controllers available to the iCub [3]. These controllers receive 3D positions in robot coordinates and they calculate and execute trajectories in the joint space of the robot.

The set of motor systems for the high-level SMDPs is defined as $MS = \{right\_arm, left\_arm\}$. A factorised discrete state space is used for both arms ($\mathcal{S}_{right\_arm}$ and $\mathcal{S}_{left\_arm}$), with the state variables: *armPosition* = {*onObject, onTable, onContainer, outsideTable*}, *handStatus* = {*grasping, empty*}, and *tableStatus* = {*objectsOnTable, empty*}. The set of options available for both arms ($\mathcal{O}_{right\_arm}$ and $\mathcal{O}_{left\_arm}$) are: *moveToObject*(2.65sec), *moveToTable*(2.95sec), *moveToContainer*(3.25sec), *graspObject*(2.87sec), and *releaseObject*(1.0sec). Next to each option is the average completion time in seconds, which was obtained by executing all options in sequence for 60 minutes. It is important to point out that for option *graspObject* we use a special command, defined in the iCub simulator, that makes the hand act like a magnet. Even though we simplify the problem of grasping, we can control its sensitivity by checking the offset between

[2] In contrast with the Markov decision processes (MDPs), SMDPs allow us to model actions with variable duration.

the centre of the hand and the centre of the object. Except for *releaseObject*, all options can fail if the offset between the centre of the hand and the desired final position is greater than some threshold (e.g. 1 cm). The iCub controllers have a limited accuracy and the minimum value of that threshold is 0.5 cm.

### 2.2 Visual Memory

The central component in the system is what we call the *visual memory* (Fig. 1.B), which captures the continuous state information about object pose needed for low-level control and supplies the discrete objects id's needed to create the high-level discrete state. We define the visual memory as the set of ordered pairs $\mathcal{VM} = \langle (e_1, bel(e_1)), (e_2, bel(e_2)), \ldots, (e_n, bel(e_n)) \rangle$, where $e_i$ is the $i^{th}$ entity of interest on the table, where an entity can be an object or a container. Every time a new entity is seen it is added to the visual memory. $bel(e_i)$ is a probability density (or belief state), associated with the location of the entity $e_i$. This location is used by the low-level options. Each object has a diameter of 4 cm, and its location refers to its centre projected in the X-Y plane in robot coordinates. Containers are 10x10x3 cm in width, length and height, and the location refers to its centre as well. The belief state for each entity $e_i$ is approximated by a particle filter [15]. Each particle filter contains a set of particles $\mathcal{G}_i$, where each particle $g \in \mathcal{G}_i$ represents a possible location for entity $e_i$.

### 2.3 Learning Phase

As described above, each high-level SMDP models a given sub-task, and learning this sub-task is achieved via reinforcement learning (RL) using *SMDP Q-learning* [6]. Each motor system $ms$ learns a policy $\pi_{ms} : \mathcal{S}_{ms} \rightarrow \mathcal{O}_{ms}$, that defines a mapping from states to options. A policy contains Q-values, i.e. the cumulative expected reward for each state-action pair. During learning we limit the number of objects appearing on the table to 10. We follow a minimal time to goal strategy, so for any option taken the robot receives -1 unit of reward. When the task is completed (i.e. no more objects appear on the table) it receives 0 units of reward. The robot initially learns how to perform the task under an assumption of complete observability, i.e. the visual memory contains the complete list of entities ($e_i$), and their corresponding belief ($bel(e_i)$) indicates the true location of that entity.

### 2.4 Execution Phase

During execution the robot is in charge of maintaining the visual memory, which does not contain any entity at the beginning. The robot has to look at entities in order to add its pair $(e_i, bel(e_i))$ into visual memory. Every time an entity $e_i$ is observed, its belief $bel(e_i)$ is updated with new information.

**Physical Action Selection:** The robot selects options for each of its motor systems by following the *Q-MDP algorithm* [1] in terms of particle filters:

$$o_{ms} = \arg\max_{o \in \mathcal{O}_{ms}} \frac{1}{|\mathcal{G}_i|} \sum_{g \in \mathcal{G}_i} weight(g) cost(g, o), \tag{1}$$

where $\mathcal{G}_i$ is the set of particles representing the belief state of entity $e_i$, and $g$ refers to an individual particle. $weight(g)$ defines the weight given to the particle $g$. Each belief $bel(e_i)$ determines the likelihood of success or failure of options, which in turn determines changes in the discrete space described in Section 2.1. For instance, the discrete state variable *armPosition* takes the value *onObject* if and only if the offset between the centre of the hand and the centre of the object is less or equal than some threshold (e.g. 1 cm). The $cost(g, o)$ takes the value of $Q_{ms}(s, o)$ (i.e. the Q-value taken from the policy $\pi_{ms}$, where $s$ is the current discrete state) if the offset between the centre of the hand and the particle $g$ is less than or equal to some threshold, otherwise it takes the value of $\min_{o \in \mathcal{O}_{ms}} Q_{ms}(s, o)$, indicating that the option failed.

**Gaze Coordination:** The selection of perceptual actions works as follows:

1. Each entity $e_i$ listed in visual memory represents a fixation point, i.e. a perceptual action $p \in \mathcal{P}$, where $\mathcal{P}$ is the set of perceptual actions at any given time. The number of perceptual actions varies depending on the number of entities in visual memory.
2. We define $ms_f$ as the motor system that will benefit the most if it is given access to perception:

$$ms_f = \arg\max_{ms \in MS} \left[ \max_{p \in \mathcal{P}} \{V^p_{ms}\} - \max_{o \in O_{ms}} \{V^o_{ms}\} \right], \tag{2}$$

where $MS$ is the set of motor systems, and the expression inside the square brackets represents the expected gain that would result if gaze is allocated to motor system $ms$. $V^p_{ms}$ is the expected value for motor system $ms$ assuming perceptual action $p$ is taken. $V^o_{ms}$ is the expected value with the current uncertainty. In fact, $\max_{o \in O_{ms}} \{V^o_{ms}\}$ has been calculated during the option selection using (1), so we can cache this value. The difference between the maximum of these two values tells us how much we gain if gaze is allocated to this motor system. To calculate $V^p_{ms}$ we follow:

$$V^p_{ms} = \frac{1}{|\Omega|} \sum_{\omega \in \Omega} \max_{o \in O_{ms}} \left( \frac{1}{|\mathcal{G}_i|} \sum_{g \in \mathcal{G}_i} weight(g, \omega) cost(g, o) \right), \tag{3}$$

where $\Omega$ is the set of observations, $\omega$ is a particular observation, $\mathcal{G}_i$ is the set of particles, and $g$ is a single particle. $weight(g, \omega)$ represents the weight of particle $g$ after having observed $\omega$, and $cost(g, o)$ takes the value of $Q_{ms}(s, o)$ if the offset between the centre of the hand and the particle $g$ is less than or equal to some threshold, otherwise it takes the value of $\min_{o \in \mathcal{O}_{ms}} Q_{ms}(s, o)$.

This equation is trying to predict the value of moving the gaze to a specific fixation point by using "imaginary" observations $\omega$ to update the current belief, and then checking the effect this possible new belief would have in the selection of options.

3. After a motor system is selected in (2), we define $p_f$ as the perceptual action that will be executed:

$$p_f = \arg\max_{p \in \mathcal{P}} \left\{ V^p_{ms_f} \right\}, \tag{4}$$

which is straightforward if we cache the results of $\max_{p \in \mathcal{P}} \{V^p_{ms}\}$ when calculating (2).

The observations $\omega$ are sampled assuming that the robot would fixate on the mean of the cloud of particles ($mean(\mathcal{G}_i)$). First, a number of particles $g_j$ are uniformly selected. Then a bivariate Gaussian distribution is chosen for each $g_j$ by indexing an observation model learnt off-line (described below), according to the location of particle $g_j$ and the imaginary fixation point ($mean(\mathcal{G}_i)$) with respect to the also imaginary oculomotor position. From the chosen bivariate Gaussian distributions the observations $\Omega$ are sampled.

**Visual Analysis:** During a fixation, visual input is read from the right camera. The entities inside the camera's field of view (FoV) are detected[3], and their 3D locations (in robot coordinates) are calculated using the plane of the table to obtain an estimate of the depth. Since only one camera is used the estimated 3D locations are noisy[4]. In order to handle this noise during execution, an observation model was learnt off-line. This model is used for updating the particle filters and for sampling imaginary observations. The observation model was created by systematically moving the robot's gaze and an object in the robot space, and storing the coordinates depicted in Fig. 2.A. With these data points 405 bivariate Gaussian distributions were fitted, each representing a particular relationship between the position of the camera, the location of the fixation point and the object. Fig. 2 shows three of these distributions. Distributions $B$ and $D$ represent the noise when objects are found at the left and the right of the fixation point respectively, their location is less accurate. Distribution $C$ represents the noise when objects are found in line with the fixation point, which gives a more accurate location. Finally, if an entity in visual memory is not seen for more than 1.5 seconds, Gaussian noise with zero mean and 1.0 cm of standard deviation is added to its current estimate.

[3] We have a noiseless object detection by using the simulator, this let us focus just on a single source of uncertainty, namely the estimated 3D locations.

[4] Using both cameras for triangulation results in a perfect estimate, as the cameras are perfectly aligned in the simulator.

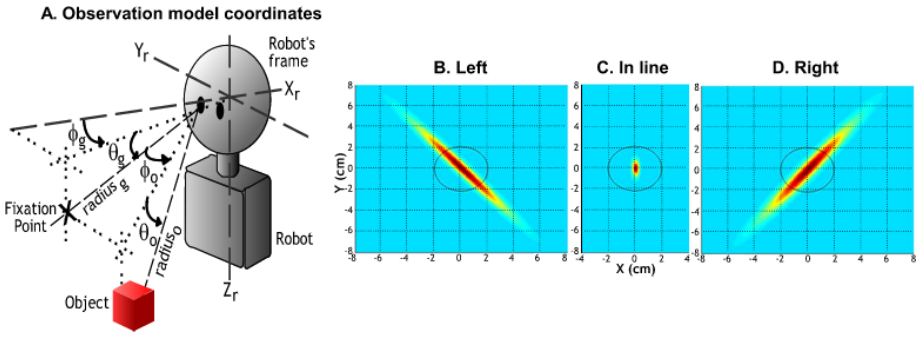


**Fig. 2.** Observation model. A. The coordinates specifying fixation points and object locations. B. Example of a distribution representing objects appearing at the left of the fixation point. C. Objects in line with the fixation point. D. Objects appearing at the right of the fixation point. The circle in the graph represents an object

## 3 Experiments

We characterise our model by varying three environmental conditions: reach/grasp sensitivity in the manipulation actions, the level of observation noise and the camera's field of view. In order to compare our reward based model, we also implemented a *random*, and a *round robin* gaze allocation strategies.

**Reach/Grasp Sensitivity Analysis:** As described in Section 2.4, the success or failure of options is determined by a threshold. This threshold controls the sensitivity whilst reaching and grasping. We have defined six threshold values: 0.5, 1.0, 1.5, 2.0, 2.5 and 3.0 cm. Fig. 3.A shows the results of the average number of objects correctly placed in the containers for all gaze strategies. A total of 15 trials of 5 minutes each were performed for each strategy. The observation noise is 1.0, meaning that we use an unmodified version of the observation model, and the field of view (FoV) is 60°horizontally and 40°vertically (the default in the simulator), these angles refer to the complete FoV. The error bars represent the 95% confidence intervals. The results show how as the sensitivity increases (i.e. when it moves towards 0.5 cm), the performance of all three strategies decreases, since the task requires more accuracy. Notice that the ratio between the uninformed strategies (i.e. random and round robin), and the reward based strategy increases as the sensitivity increases. This means that if we require more accuracy, we also require an efficient way to allocate gaze. The main reason is that as the sensitivity increases the robot needs to fixate several times on an object before reaching and grasping can succeed, something that the reward based strategy is capable of doing but not the uninformed strategies.

**Observation Noise Analysis:** For these experiments we added noise to the observation model to test for robustness. As explained above, the observation

model is a set of bivariate Gaussian distributions. Noise was added to the distributions by multiplying the two standard deviations that define each distribution by some factor (1.0, 1.5, 2.0 and 2.5, in this case). Fig. 3.B presents the results of the average number of objects correctly placed in the containers for all gaze strategies. A total of 15 trials of 5 minutes each were performed for each strategy. For this set of experiments the reach/grasp sensitivity is 1.0 cm and the FoV is 60˚x40˚. As the level of observation noise increases the performance of all strategies decreases. But notice how the ratio between the uninformed strategies and the reward based strategy grows as the observation noise increases, showing that the reward based strategy is much more robust to noise.

**Field of View Analysis:** For these experiments we vary the horizontal and vertical angles of the field of view with the values: (60˚x40˚), (50˚x35˚), (40˚x30˚), (35˚x25˚), (25˚x20˚). Fig. 3.C presents the results of the average number of objects correctly placed in the containers for all gaze strategies. A total of 10 trials of 5 minutes each were performed for each strategy. For this set of experiments the reach/grasp sensitivity is 1.0 cm and the observation noise is 1.0. As the field of view is reduced the performance of all strategies decreases, since less objects appear in the FoV. Although the reward based strategy outperforms the other two in all cases, it is interesting to notice that the performance of the uninformed strategies do not decrease so much. This is because as the FoV decreases *visual search* becomes a critical issue for the good performance of the task. The uninformed strategies are not too affected by this problem because they move the gaze more around the table, finding new objects indirectly. So, for instance, if the reward based strategy does not see objects on the left side of the table, the left arm remains idle. At the moment we have implemented a heuristic where every time the visual memory is empty the robot performs a systematic search for objects across the table. However, these results show that a better visual search strategy is required that can work together with the gaze allocation model in order to maintain a good performance.

There are two factors that should be taken into account for the analysis of all the previous results. First, the *peripheral information* has a major impact in the performance, especially when the FoV is large. We update all entities inside the FoV, this means that there might be objects that are never directly fixated but the robot still succeeds in grasping them, particularly if the sensitivity and/or observation noise decrease. This is why the performance of the uninformed strategies is so good in some cases. The effect of the peripheral information is more evident when the reach/grasp sensitivity threshold is between 2.0 to 3.0 cm, where round robin is even slightly better than our model. Second, the *decision time* is also an important factor. The reward based strategy takes in average 0.65 sec to decide where to look, compared to the almost immediate response of random and round robin. Unfortunately, it is not possible to directly compare our model to Sprague et al. model [11]. They model tasks with multiple concurrent goals within a single motor system, whereas we model tasks with multiple motor systems, with a single goal for each motor system.

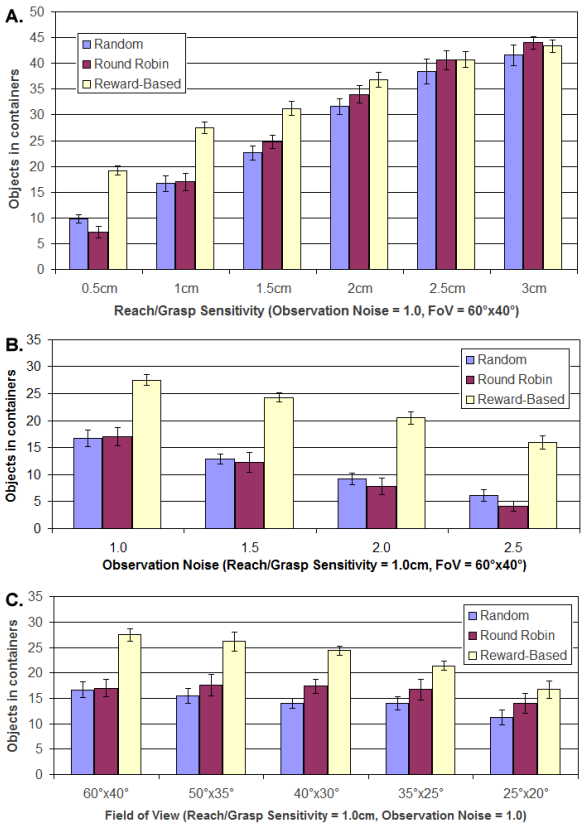


**Fig. 3.** A. Results for reach/grasp sensitivity analysis, with observation noise = 1.0 and field of view = 60°x40°. B. Results for observation noise analysis, with reach/grasp sensitivity = 1.0 cm and field of view = 60°x40°. C. Results for field of view analysis (horizontal x vertical angles), with reach/grasp sensitivity = 1.0 cm and observation noise = 1.0. The error bars represent the 95% confidence intervals.

## 4 Conclusions and Future Work

In this paper we have defined a gaze allocation model based on rewards by posing the problem of where to look as one of maximising task performance by reducing task relevant uncertainty. In particular, the robot selects the gaze at any moment that increases task rewards most by reducing task relevant uncertainty about object location. Our experiments show the behaviour of this reward based strategy compared with a random and round robin strategies, while varying three environmental conditions: reach/grasp sensitivity, observation noise, and the field of view. As the sensitivity and/or noise increases, the performance of all strategies decreases. If the FoV is reduced, the performance is expected to decrease, although we have concluded that a visual search mechanism is needed in order to maintain good performance. The results show that our reward based

strategy outperforms the random and round robin strategies in all cases, except when reach/grasp sensitivity = 3.0 cm. This case helped us to identify two factors that should be taken into account in the analysis of the results, namely the peripheral information and the decision time. We are currently implementing a formal method for visual search instead of the current heuristic. Also, we are developing two more gaze strategies derived from our model, so that we can compare with other informed strategies, not just uninformed ones. We are also extending the system to take into account concurrent motor systems, so that the arms can interact with each other. Finally, another interesting extension would be to execute physical actions to help the perception process. For instance, if the object of interest is occluded by another object, it would be better to remove that object to get a better view.

## References

1. Cassandra, A. R.: Exact and approximate algorithms for partially observable Markov decision processes. Ph.D. thesis, Brown University (1998)
2. Metta, G., et al.: The iCub humanoid robot: An open platform for research in embodied cognition. In: Proc. ACM Perf. Metrics for Int. Sys. pp. 50–56. ACM New York (2008)
3. Pattacini, U. et al.: An experimental evaluation of a novel minimum-jerk cartesian controller for humanoid robots. In: IEEE IROS. pp. 1668–1674 IEEE Press (2010)
4. Nunez-Varela, J. et al.: Where do I look now? Gaze allocation during visually guided manipulation. In: IEEE ICRA. pp. 4444–4449 IEEE Press (2012)
5. Karaoguz, C. et al.: Optimisation of gaze movements for multitasking using rewards. In: IEEE/RSJ IROS. pp. 1187–1193 IEEE Press (2011)
6. Bradtke, S. and Duff, M.: Reinforcement learning methods for continuous-time Markov decision problems. Adv. in Neural Inf. Proc. Sys. 8 393–400 (1995)
7. Sutton, R., et al.: Between MDPs and semi-MDPs: A framework for temporal abstraction in reinforcement learning. AI Journal. 112(1) 181–211 (1999)
8. Schultz, W.: Multiple reward signals in the brain. Nat. Rev. Neurosci. 1(3) 199–207 (2000)
9. Johansson, R., et al.: Eye-hand coordination in object manipulation. Journal of Neuroscience. 21(17) 6917–6932 (2001)
10. Land, M.: Eye movements and the control of actions in everyday life. Progress in Retinal and Eye Research. 25(3) 296–324 (2006).
11. Sprague, N., et al.: Modeling embodied visual behaviors. ACM Trans. Appl. Percept. 4(2) (2007)
12. Hayhoe, M. and Rothkopf, C.: Vision in the natural world. Wiley Inter. Reviews: Cognitive Science. 2(2) 158–166 (2010)
13. Puterman, M.: Markov Decision Processes: Discrete Stochastic Dynamic Programming. Wiley-Interscience New York (1994)
14. Sutton, R. & Barto, A.: Introduction to Reinforcement Learning. MIT Press (1998)
15. Thrun, S., et al.: Probabilistic Robotics. MIT Press Cambridge (2008)
16. Frintrop, S.: VOCUS: A Visual Attention System for Object Detection and Goal-Directed Search. Springer-Verlag New York (2006)