A Swarm Cognition Realization of Attention, Action Selection and Spatial Memory

Pedro Santana
LabMAg, University of Lisbon
Pedro.Santana@di.fc.ul.pt

Luís Correia
LabMAg, University of Lisbon

Last revision: January 4, 2012

Abstract

This paper reports a study on modelling covert visual attention as a parallel process that unfolds in synergy with the embodied agent’s action selection process. The parallel nature of the proposed model is according to the multiple covert attention hypothesis and thus suitable for the concurrent search for multiple objects in the embodied agent’s visual field. In line with the active vision approach, the interaction with the action selection process is exploited by the model to deploy visual attention in a by-need way. Additionally, the multiple focuses of attention considered in the proposed model interact in a way that their collective behaviour robustly self-organises for a proper handling of the speed-accuracy trade-off inherent to visual search tasks. Besides the self-organisation of a global spatio-temporal visual attention policy, the model also produces parallel, sparse, and active spatial working memories, that is, local maps of the environment. The underlying mechanisms of the model are based on the well known formalism that describes the self-organisation of collective foraging strategies in social insects. This metaphor is particularly interesting because of its similarities with the problem tackled in this paper, that is, the one of parallel visual attention. This claim is validated by experimental results on a simulated robot performing local navigation, where the ability of the model to generate accurate visual attention and spatial memories in a parsimonious and robust way is shown.
1 Introduction

More than an information process, perception is a sensorimotor process on its own right (B Ballard et al., 1997; Pfeifer and Scheier, 1999; O’Regan and Noe, 2001). This active nature of perception is essential to enable robustness and efficiency (Bajcsy, 1988; Aloimonos et al., 1988; Ballard, 1991). This is in line with the idea that modules within a cognitive system are dynamical systems mutually entrained, as well as they are entrained with the external world through the agent’s body (Beer, 1995; Thelen and Smith, 1996). To focus perception in a by-need way, this article addresses the particular entrainment observed between action selection and visual attention.

Visual attention ultimately results in the motion of sense organs towards the relevant stimulus source. This is called overt attention. A faster process is the one of mentally focussing particular aspects of the sensory stimuli. This is called covert attention. Studies on human subjects support the hypothesis that multiple covert attention processes co-exist in the brain (Pylyshyn and Storm, 1988; Doran et al., 2009). Based on this, the model herein proposed considers covert visual attention as a process composed of a set of parallel focuses of attention, which cohabit and interact among themselves and with the action selection process for a better and by-need coverage of the embodied agent’s visual field.

One of the most powerful formalisms capable of providing these properties to systems composed of many interacting elements is the one that describes the self-organisation of collective intelligence displayed by social insects (Franks, 1989). In this line, the proposed model uses the ant foraging metaphor to control the collective behaviour of a set simple virtual agents that inhabit the embodied agent’s sensorimotor and decision spaces. The behaviour of each of these virtual agents implements a locally coherent covert attention policy. The self-organisation of these virtual agents leads in turn to a global spatio-temporal collective pattern of coherent, robust, and efficient covert attention policy. Parallel, sparse, and active spatial working memories are also maintained by the system, which besides helping the action selection process to reach better decisions, also improve the visual attention process by endowing it with some predictive power.

The experimental apparatus is composed of a simulated robot that must focus its perceptual resources to enable the detection and retention of memories of the obstacles that are essential to ensure safe local navigation. As argued by Slocum, Downey, and Beer (Slocum et al., 2000), “if we hope to . . . analyse in detail model agents exhibiting genuinely cognitive behaviour, it is essential to focus on the simplest possible agent-environment systems that exhibit the cognitive behaviour of interest”. It not being the goal of the model to encompass the complexity of object identification and/or recognition, means that the only cognitive behaviour of interest is the one of directing multiple covert visual attention processes in a by-need basis. Hence, the simplest agent-environment system in this case is the one where the perceptual process needs to search a sensory space, whatever its dimensions, and the identification/recognition aspects are reduced to a well defined binary decision process. Bearing this in mind, the simulated robot is provided with a body-centred two-dimensional image of the local environment in which obstacles can be detected by simply checking which image pixels are occupied.
This article is organised as follows. Section 2 surveys related work in both natural and applied sciences. Then, in Section 3, an overview of the proposed model is presented, as well as a focused inspection on the parallel perceptual system. The experimental results, obtained with a simulated mobile robot, are reported in Section 4. A discussion about key aspects of the model is given in Section 5. Finally, a set of conclusions is drawn, and pointers to future work are suggested in Section 6.

2 Related Work

The ability to shift the focus of attention, ubiquitous in animals (Land, 1999), highlights the active nature of perception (Bajcsy, 1988; Aloimonos et al., 1988; Ballard, 1991). From this perspective, body, nervous system, and environment must be seen in an holistic way (Gibson, 1979; Ashby, 1952; Beer, 1995; Telen and Smith, 1996). An advantage of active vision is the possibility of the embodied agent to act in order to shape its sensory information flow (Sporns and Lungarella, 2006), and consequently make its processing tractable. Sensorimotor coordination (Dewey, 1896) thus plays a key role for adaptive behaviour (Brooks, 1991; Ballard et al., 1997; Pfeifer and Scheier, 1999; O’Regan and Noe, 2001; Pfeifer and Bongard, 2006; Mossio and Taraborelli, 2008).

Models of visual attention, a component of active vision, typically assume the existence of a sensory-driven bottom-up pre-attentive component (Treisman and Gelade, 1980; Koch and Ullman, 1985; Itti et al., 1998; Palmer, 1999; Corbetta and Shulman, 2002; Hou and Zhang, 2007), which is modulated by top-down context aware pathways (Yarbus, 1967; Wolfe, 1994; Tsotsos et al., 1995; Corbetta and Shulman, 2002; Torralba et al., 2003; Frintrop et al., 2005; Navalpakkam and Itti, 2005; Walther and Koch, 2006; Neider and Zelinsky, 2006; Rothkopf et al., 2007; Hwang et al., 2009), as it has been shown by recent neurophysiological studies (Egner et al., 2008).

The outcome of the interplay between bottom-up and top-down processes is typically assumed to be some sort of moving “spotlight” from the most to the least relevant regions of the image. However, it is unlikely that a massively parallel structure, as the human brain is, only runs a single sequential processing stream. In fact, studies with human subjects (Pylyshyn and Storm, 1988; Doran et al., 2009) revealed exactly the opposite. Moreover, in vision in general, interacting perceptual pathways dedicated to different purposes are more likely to exist (Milner and Goodale, 1995; Goodale, 2008), than a global and isomorphic representation of the environment (Marr, 1982).

Overt attention being a sensorimotor process, and supported by the evidence that both overt and covert share the same neural mechanisms (de Haan et al., 2008), we speculate that covert attention can also be modelled as a sensorimotor process. Under this assumption, it is reasonable to consider the focus of attention as a dynamical entity inhabiting the sensory space of the embodied agent. This entity can then proactively move, that is, guide the focus of the embodied agent, towards the most relevant regions of the sensory space in a sensorimotor coordinated way. This entity, which can be modelled as a virtual agent, thus behaves as a locally sequential
covert attention process. The feasibility of such agent-based modelling has considerable support from the fields of embodied cognition and active vision, where simple virtual agents performing sensorimotor coordination have been successfully synthesised (Beer, 1996; Scheier et al., 1998; Slocum et al., 2000; Nolfi and Marocco, 2002; Beer, 2003; Balkenius et al., 2004; Floreano et al., 2004; Nolfi, 2005; Suzuki and Floreano, 2006; Pfeifer and Bongard, 2006; Kim and Moeller, 2006; Sporns and Lungarella, 2006; de Croon and Postma, 2007; Choe et al., 2008).

Brain computational modelling with multiple virtual agents is not a new idea (Minsky, 1988; Chialvo and Millonas, 1995). Although the first realisations of the idea to solve computer vision related problems are also not new (Poli and Valli, 1993; Liu et al., 1997), only more recently it has received considerable attention (Ramos and Almeida, 2000; Owechko and Medasani, 2005; Antón-Canalís et al., 2006; Mobah et al., 2006; Broggi and Cattani, 2006; Mazouzi et al., 2007; Zhang et al., 2008). These parallel computational models are mostly stand-alone engineered parallel perceptual systems, lacking the interaction with action selection and mapping processes. This deficit undermines their explanation power regarding the mechanisms actually building up adaptive behaviour. Conversely, sensor planning, which is a relatively stable field in computer vision and robotics communities, is actually trying to bridge the gap between body motions and information gathering through the sensors (Dickmanns et al., 1990; Nabbe and Hebert, 2003; Kwok and Fox, 2004; Patel et al., 2005; Hernandez et al., 2007; Sprague et al., 2007). However, none of these models considers parallel covert attention operating in an intricate way with the action selection process.

A remarkable metaphor from the natural world encompassing the characteristics of parallel deployment of attention is the foraging behaviour of army ants. These ants are able to leave their nests and cover large areas to search for food items in a purely parallel and robust way (Deneubourg et al., 1989), exhibiting a sort of collective intelligence (Franks, 1989). This mapping between insect-based swarms and neuron-based brains of vertebate has been also suggested by parallel and independent work (Passino et al., 2008; Couzin, 2009; Marshall et al., 2009; Marshall and Franks, 2009). Our work, instead, approaches the problem of studying cognition through social insects behaviour by building it, following the synthetic approach to embodied cognition (Pfeifer and Scheier, 1999) and artificial life (Bedau, 2003). The advantages of using artificial life models for this purpose, though without the support of any practical realisation, have also been dissected in a parallel study (Trianni and Tuci, 2010). All these accounts can be framed in the emerging multidisciplinary field of swarm cognition (Trianni and Tuci, 2009), which attempts to uncover the basic principles of cognition, that is, adaptive behaviour, recurring to self-organising principles, mainly those exhibited by social insects. Furthermore, conceiving cognition as a self-organising process is essential to understanding how open-ended learning can develop, and consequently, how it can be synthesised.
3 Proposed Model

This section starts by providing an introduction to the biological inspiration of foraging strategies in social insects as a model of parallel covert visual attention. This introduction is followed by the description of the algorithm specifying the way perception, which includes visual attention, and action selection interact in the proposed model. Details on both perceptual and action selection processes are provided. The model is described under the context of local navigation. That is, the action selection process aims at deciding which motor action should be engaged by the embodied agent to allow its safe progression in the environment. This decision is influenced by a desired heading of motion and constrained by the obstacles present in the environment, whose detection is responsibility of the perceptual process.

3.1 Biological Inspiration

Covert visual attention is mostly related to the parallel search of objects in the embodied agent’s visual field. A remarkable metaphor of this process is the one of army ants engaging on foraging behaviour. Assuming that the environment that these ants inhabit corresponds to the embodied agent’s visual field, each ant can be seen as an individual covert visual attention process. Their collective behaviour can then be seen as a parallel covert visual attention process.

Following this metaphor from the natural world, the visual process in the proposed model is composed of a swarm of simple homogeneous virtual agents (hereafter p-ants) that inhabit the visual field of the embodied agent (hereafter simply agent). These p-ants are probabilistically created (recruited) to search (forage) the agent’s visual field along those regions where detected obstacles are likely to affect the action selection process more strongly. Hence, p-ants operate on a by-need basis being affected by the action selection process. As in natural ants, p-ants do this search in a stochastic way. P-ants interact through stigmergy for better coverage and tracking of detected objects. It allows the coexistence of positive and negative feedback loops that lead to robust collective behaviour. In conclusion, random fluctuations and both positive and negative feedback, which are necessary ingredients for self-organisation to occur (Bonabeau et al., 1999), are integral part of the model.

3.2 Overall Process

Fig. 1 illustrates the connectivity between perceptual and action selection processes, whose control logic is outlined in Algorithm 1. Basically, after receiving a new frame, $I$, both perceptual and action selection processes interact (thicker arrows) for several iterations before a final motor action decision is reached and eventually engaged. These interactions occurring between both processes allow them to progressively unfold in parallel, and consequently, to enable accurate deployment of visual attention, which is an integral part of the perceptual process. The following describes these interactions for a given frame.

At each iteration, the action selection process sends a message to the perceptual one with an action utility vector, $u = (u^1, u^2, \ldots, u^k)$, where $u^j \in [0, 1]$ is the utility of performing action
Algorithm 1: Overall system’s execution pseudo-code.

Input: desired heading, $h$, desired speed, $s$

Data: $i_{\text{max}}, i_s$ are empirically defined constants.

1. create an empty set of p-ants, $A \leftarrow \emptyset$
2. while true do
3.     reset iterations counter, $i \leftarrow 0$
4.     reset stable iterations counter, $c \leftarrow 0$
5.     clear the set of obstacles, $O \leftarrow \emptyset$
6.     clear free-space connectivity information, $C \leftarrow \emptyset$
7.     reset perceptual shared mediums, $R \leftarrow -1, P \leftarrow 0$
8.     reset best action, $v \leftarrow 0, v'' \leftarrow 0$
9.     get new frame, $I$
10.    estimate agent motion, $(\Delta x, \Delta y, \Delta \theta)$
11.    // iterate until best action stabilises for $i_s$ iterations or a maximum nr. of iterations, $i_{\text{max}},$ is reached
12.    while ($c < i_s \land i < i_{\text{max}}$) do
13.        $i \leftarrow i + 1$
14.        // iterate the action selection process
15.        compute the action utility vector, $u = (u^1, u^2, \ldots, u^k)$, based on $C, h,$ and $v''$
16.        save previous best action, $v' \leftarrow v$
17.        obtain the highest utility action in $u$, $v \leftarrow \arg\max_j u^j$
18.        // iterate the perceptual process
19.        iterate obst. search,
20.        $(R, P, A, O') \leftarrow \text{iterateDetection}(I, (\Delta x, \Delta y, \Delta \theta), R, P, A, u, s, i, k)$ // see Algorithm 2
21.        update the set of obstacles, $O \leftarrow O \cup O'$
22.        compute free-space connectivity, $C$, based on $O$
23.        // update counter of stable iterations
24.        if $v = v'$ then
25.            $c \leftarrow c + 1$
26.        else
27.            $c \leftarrow 0$
28.        end
29.    end
30.    generate motor action for $v$ and desired speed $s$
31.    save motor action sent to the actuators, $v'' \leftarrow v$
32. end
Figure 1: Building blocks of the proposed model.

\( j \), and \( k \) is the cardinality of the agent’s action repertoire. This vector is computed according to a desired heading of motion, \( h \), and constrained by information about free-space connectivity of the local environment, \( C \), which has been sent by the perceptual process in the previous iteration. In this study, actions \( j \) are defined as linear trajectories centred on the agent and directed outwards in a radial pattern.

In turn, the perceptual process uses the just received vector \( u \) to iterate its search for obstacles. The set of obstacles detected in the current iteration, \( O' \), is appended to the set \( O \), which accumulates obstacles previously detected. This set is then used to compute \( C \), which encompasses range information regarding those radial sectors of the local environment that contain free-space for agent motion. Finally, \( C \) is sent back as a message to the action selection process so as to affect its next iteration. Since the perceptual process is affected by the incoming vector \( u \), which states the utility of performing each possible action and consequently which regions of the environment should be more carefully analysed, it is considered to operate on a by-need fashion.

The mutual influence between action selection and perceptual processes is carried per frame: (1) for a maximum number of iterations, \( i_{\text{max}} \); or (2) until the action with highest utility is the same over \( i_s \) iterations. Finally, the highest utility action at the time interactions cease is passed to the low-level motion controller through a low-pass filter. In this way, sudden changes at the system’s output are smoothed to avoid jitter at the actuators level.

Details regarding both perceptual and action selection processes are provided in the following sections.

### 3.3 Perceptual Process

As mentioned, the iteration of the perceptual process is three fold (see Algorithm 1). First, a set of obstacles, \( O' \), is detected in the current iteration. Second, \( O' \) is appended to the cumulative set \( O \). Third, \( O \) is used to re-compute \( C \), that is, the local environment’s free-space connectivity.

The set of obstacles \( O \) is used to determine the maximum distance, \( d^j \), the agent is able to travel along each possible linear trajectory, \( j \in [1,k] \), without hitting any obstacle. Refer to (Santana and Correia, 2006) for further details. The set \( C \) is then the aggregate of all free-space,
Figure 2: Snapshot of the visual process activity, in a situation where the action utility vector states that moving to the right is preferred to moving to the left, and both are preferred to moving forward. (a) P-ants (small grey circles) in search (over non-obstacle regions) and tracking (over obstacles, in black) behaviour. Note that the input image is a body-centric two-dimensional Euclidean image of the local environment. The point where p-ants are created is at the bottom-centre of the input image, indicated by the arrow. (b) Shared medium $R$ representing the pixels classified as obstacle (white), as non-obstacles (black), and that have not been tested so far (grey). (c) Shared medium $P$ (in grey level) updated by p-ants as obstacles are detected.

defined in terms of pairs $(j, d^j)$. With this information the action selection process is able to assess the quality of each possible motor action.

At each iteration, the set $O'$ is created as follows (see Algorithm 2). The perceptual process starts by extending the set of p-ants, $A$, with a new one, $A'$. The rate at which p-ants are created is a function of the agent’s speed, $s \in [0, 1]$. This allocates more perceptual resources when the agent is moving faster and consequently when an action must be selected sooner. The creation rate is also a function of the action utility vector. The higher the utility of a linear trajectory, $j$, the higher the chances of creating a p-ant, $a_j$, with the following initial position:

$$p_{a_j} \leftarrow [\zeta_j, 0]^T,$$

which is defined in polar coordinates in the image plane and where $\zeta_j$ is the direction of linear trajectory $j$. This position corresponds to the bottom-centre of the visual field, so that p-ants can start their search for obstacles in the close vicinity of the agent (see Fig. 2(a)).

A newly created p-ant, $a_j$, is endowed with an initial energy level, $o_{a_j} \leftarrow \rho$, which is reduced at each iteration and restored when an obstacle is detected (see below). P-ants with null energy are removed from the system to maintain memory and computation within boundaries. A p-ant initiates its operation with the activation of the search behaviour (see below), $b_{a_j} = \text{SEARCH}$. With this behaviour, the p-ant will move on the agent’s visual field with the purpose of detecting an obstacle, which is then tracked by the activation of other behaviours.

If the current iteration is the first after obtaining the current frame, $i = 1$, p-ants are not created and the agent’s motion since the last frame, $(\Delta x, \Delta y, \Delta \theta)$, is estimated recurring to wheel odometry. This estimate is then used to compensate the position of every p-ant $a \in A$, for the agent motion:
Algorithm 2: iterateDetection

Input: $I, (\Delta x, \Delta y, \Delta \theta), R, P, A, u, s, i, k$
Output: $R, P, A, O'$
Data: $(p_a, z_a, o_a, b_a, r_a)$ is the set of properties of a given p-ant $a \in A$ (explained in the text).
Data: $\eta, \zeta, v, p, r_{max}$, and $l$ are empirically defined constants and $FOV$ is the set of pixels composing the agent’s visual field (explained in the text).

1 create an empty set of obstacles, $O' \leftarrow \emptyset$
2 sample a number from an uniform distribution, $y \in [0, 1]$
3 if $y < s \land i > 1$ then // the faster the agent the higher the chances of creating p-ants
4 // create a new set of p-ants, $A'$, to be added to $A$
5 $A' \leftarrow \emptyset$
6 foreach $j \in [0, 1, \ldots, k]$ do // go over all possible actions
7 sample a number from an uniform distribution, $x_j \in [0, 1]$
8 if $x_j < u'$ then // the higher the utility of the action the higher the chances of creating a p-ant
9 create new p-ant $a_j$ with $(p_{a_j}, z_{a_j}, o_{a_j}, b_{a_j}, r_{a_j}) \leftarrow ([z_j, 0]^T, [z_j, 0]^T, 0, p, SEARCH, r_{max})$
10 add the new p-ant $a_j$ to set $A'$, $A' \leftarrow A' \cup \{a_j\}$
11 end
12 end
13 add the new set of p-ants, $A \leftarrow A \cup A'$
14 end
15 // iterate all p-ants included in $A$
16 foreach $a \in A$ do
17 reduce p-ant’s energy available, $o_a \leftarrow o_a - 1$
18 if $i = 1$ then // new frame received
19 use $(\Delta x, \Delta y, \Delta \theta)$ in Equation 2 to compensate p-ant’s position, $p_a$, for agent motion
20 end
21 if $o_a \leq 0$ then // p-ant without energy
22 remove p-ant, $A \leftarrow A \setminus \{a\}$
23 end
24 // iterate search behaviour
25 if $b_a = SEARCH$ then
26 iterate search behaviour, $(R, P, A, O') \leftarrow iterateSearch(I, R, P, A, O', FOV, a)$ // see Algorithm 3
27 end
28 // iterate track behaviour
29 if $b_a = TRACK$ then
30 iterate track behaviour, $(R, P, A, O') \leftarrow iterateTrack(I, R, P, A, O', FOV, a, \zeta, i)$ // see Algorithm 4
31 end
32 // iterate local search behaviour
33 if $b_a = LOCAL\_SEARCH$ then
34 iterate local search behaviour,
35 $(R, P, A, O') \leftarrow iterateLocalSearch(I, (\Delta x, \Delta y, \Delta \theta), R, P, A, O', FOV, a, \eta, \nu, l, i)$ // see Algorithm 5
36 end
37 return $R, P, A, O'$
where $p_a^*$ corresponds to $p_a$ in Cartesian coordinates, $R(-\Delta \theta)$ is a two-dimensional rotation matrix in order to compensate for the angle $\Delta \theta$. This procedure helps p-ants maintaining their positions with respect to the objects in the environment independently of agent motion. The motion compensated position of each p-ant can also be seen as an interesting point upon which covert attention should be deployed when a new frame arrives. This introduces a prediction component to the covert attention process. During the motion compensation process, some p-ants will eventually leave the agent’s visual field and so will ultimately implement a sparse local map of the environment. Since the motion compensation is performed by all p-ants, these already out of the visual field will also be affected by the process. This is essential to maintain the local map updated and also to allow some of those p-ants to re-enter the agent’s visual field when an environment is revisited.

After these preparatory procedures, the currently active behaviour of each p-ant in $A$ is iterated. That is, each p-ant will perform a motion step according to a given stochastic perception-action rule. The following sections describe the iteration of each of the three possible behaviours, namely search, track, and local search.

### 3.3.1 Search Behaviour

The iteration of the search behaviour is a simple stochastic motion step on input frame $I$ along the preferred search direction defined at p-ant creation time (see Equation 1). In the case of a p-ant $a \in A$ with current position $p_a$ the motion step is defined as follows:

$$p_a \leftarrow p_a + \begin{bmatrix} N(0,1) \\ 0 \end{bmatrix} \begin{bmatrix} \lambda_1 \\ \lambda_2 \end{bmatrix},$$

where $N(0,1)$ samples a number from a Gaussian distribution with mean 0 and variance 1 (see Fig. 2(a)), $\lambda_1$ and $\lambda_2$ are empirically defined scalars. High $\lambda_1$ and $\lambda_2$ values facilitate fast detection of large obstacles, at the cost of missing smaller ones. Small values result on slower, though finer, detection.

After performing a motion step, p-ant $a$ invokes the function $(R,d) \leftarrow \text{detect}(I,R,p_a)$, whose result is $d = 1$ if an obstacle is found at position $p_a$, and $d = 0$ otherwise. Parameter $R$ is a shared medium that the function uses to check whether any other p-ant has already tested the position in question (i.e., if $R(p_a) \neq -1$). If that is the case, then $R(p_a)$ is used as the result. Conversely, if the position is tested for the first time, the function applies the obstacle detector and updates the shared medium accordingly, that is, it does $R(p_a) \leftarrow 1$ if the result is positive and $R(p_a) \leftarrow 0$ if otherwise. This helps saving computation when the cost of applying the obstacle detector is high. In sum, the function $\text{detect}$ abstracts the detection process as well as the management of the shared medium $R$. With the exception of this function call, the rest of the perceptual process is about visual attention.
After detecting an obstacle, a set of steps is executed by the p-ant $a$. First, its position is appended to the set of obstacles $O'$, $O' \leftarrow O' \cup \{p_a\}$, which is also updated by other p-ants and eventually used to feed the action selection process. Furthermore, the p-ant’s level of energy is restored, $o_a \leftarrow \rho$, and its behaviour changed to track behaviour, $b_a \leftarrow TRACK$, so that the detected obstacle can be followed across frames. Then, a shared medium $P$ is updated to reflect the localisation of the detected obstacle. This is done by adding to $P$ a top-view pyramidal shape of top magnitude 20 and linear decay (0.9) outwards, centred on $p_a$. This information will then be used by other p-ants to have an idea of the density of p-ants on the region (see below).

Finally, a p-ant in search behaviour that leaves the agent’s visual field is removed from the system, $A \leftarrow A \setminus \{a\}$. See Algorithm 3 for further details.

### Algorithm 3: iterateSearch

**Input:** $I, R, P, A, O', FOV, a$

**Output:** $R, P, A, O'$

1. use Equation 3 to update the p-ant’s position, $p_a$, along a preferred search direction
2. if $p_a \not\in FOV$ then // p-ant out of the visual field
3. remove p-ant, $A \leftarrow A \setminus \{a\}$
4. else // p-ant in the visual field
5. apply detector at p-ant’s position, $(R, d) \leftarrow detect(I, R, p_a)$
6. if $(d = 1)$ then // an obstacle has been detected in input image $I$
7. report the presence of obstacle at p-ant’s position, $O' \leftarrow O' \cup \{p_a\}$
8. raise the obstacles density shared signal at p-ant’s position, $P(p_a)$
9. restore p-ant’s level of energy, $o_a \leftarrow \rho$
10. change p-ant to track behaviour, $b_a \leftarrow TRACK$
11. end
12. end
13. return $R, P, A, O'$

#### 3.3.2 Track Behaviour

A p-ant engages in track behaviour when it finds an obstacle on its current position. Whenever a new frame is acquired, $i = 1$, that p-ant has its position compensated for the agent motion (see above), which ultimately results in the construction of a local map of the environment. However, if the motion estimation is noisy or the obstacle is dynamic, the updated position of the p-ant may no longer be the same as the one of the obstacle. Therefore, immediately after obtaining a new frame, every p-ant in track behaviour invokes the detect function. In the presence of a positive result, the p-ant’s position is appended to the set of obstacles, $O'$, and its energy restored. See Fig. 3 for an illustration of the spatial working memory operation. As in the search behaviour, the shared medium $P$ is updated to reflect the localisation of obstacles detected in the agent’s visual field.
Figure 3: Graphical output generated by the perceptual process to illustrate the mapping capabilities in two sequential situations. (a) The agent faces a set of obstacles, which are detected and registered in the action selection process. Consequently, the decision to turn to the right is taken. In turn, the decision modulates the visual process, compelling p-ants to flow more on that direction. (b) After a displacement with rotation to the right, some obstacles move out of the agent’s visual field (to the left). However, those obstacles remain represented by means of their associated p-ants, and thus still influencing the action selection process. Refer to Fig. 5 for the meaning of the lines associated to some p-ants. The grey-level colour coding is as in Fig. 2.

With the purpose of rapidly covering detected obstacles, p-ants in track behaviour that are in the agent’s visual field are allowed to locally replicate themselves. At each track behaviour iteration, a clone \( a' \) of a given p-ant \( a \) may be created and positioned in a randomly chosen location \( q \), \( p_{a'} = q \), provided that this position is not densely populated with other p-ants, \( P(q) < \eta \), and it contains an obstacle. \( q \) is selected by searching for a pixel at a distance \( (\lambda + \beta \cdot y_a) \) of \( p_a \), where \( y_a \in [0, 1] \) is a number sampled from an uniform distribution. After appending the clone \( a' \) to the set of p-ants, \( A \leftarrow A \cup \{a'\} \), its position is reported as an obstacle to \( O' \) and \( P \) is updated accordingly.

A p-ant is only allowed to clone for a limited number of times, \( r_{\text{max}} \). A clone inherits the number of replications of its ancestor so as to control the diffusion process. This means that the number of descendants of a single p-ant can amount up to \( r_{\text{max}}! \). In practice the number is much smaller as the cloning process is constrained by the obstacle boundaries and by the density of p-ants in the location in question. Therefore, a single p-ant is generally not capable of cloning up to the point of fully covering a large obstacle. If the object is in a region of the environment that is really important to the action selection process, its uncovered region will be seen as free space and consequently attract new p-ants towards itself. These new p-ants will eventually reach the obstacle, a new diffusion process will be started, and the obstacle will be fully covered. See Fig. 4 for an illustrative example of the diffusion process.

If a given p-ant \( a \) in track behaviour happens to be in a region with a high density of p-ants, \( P(p_a) > \zeta \), or detects that it is no longer on an obstacle, then it leaves the track behaviour to initiate a local search behaviour, \( b_a \leftarrow \text{LOCAL\_SEARCH} \). By making \( \eta < \zeta \), a sort of hysteresis is implemented, and with that, massive fluctuations of neighbouring p-ants entering and leaving local search behaviour are avoided.
Finally, the energy of p-ants in track behaviour that are out of the agent’s visual field receive an extra decrement. As a consequence, p-ants not refreshed with immediate sensory information are maintained in the system for a smaller period of time. See Algorithm 4 for further details.

3.3.3 Local Search Behaviour

Local search is modelled as a random walk around an anchor point, whose initial position is defined as the p-ant’s position at the time local search was initiated, \( z_a \leftarrow p_a \). Hence, with the goal of re-detecting a lost obstacle or of finding a less cluttered region of a tracked obstacle, the local search behaviour randomly changes the position of a p-ant \( a \), around an anchor point, \( z_a \):

\[
\begin{align*}
p^*_a & \leftarrow z^*_a + \sigma_a [N(0, 1), N(0, 1)]^T, \\
\end{align*}
\]

where \( \sigma_a \leftarrow \min(\sigma_a + 1, l) \) and \( N(0, 1) \) samples a number from a Gaussian distribution with mean 0 and variance 1. By increasing \( \sigma_a \) at each iteration, the local search spreads up to the upper-bound \( l \), which constrains the search to avoid migration of p-ants between obstacles.

The anchor point changes to the p-ant’s current position, \( z_a \leftarrow p_a \), whenever the density of p-ants there is higher than the one at the current anchor’s location, yet not too high, \( \eta > P(p_a) > P(z_a) \). This directs the local search towards regions where other p-ants, but not too many, reported the existence of an obstacle. Conversely, if any of these conditions is not met, then the anchor position is not changed and the level of energy of the p-ant in question is dramatically reduced by an amount of \( \eta \). This compels p-ants in local search behaviour to remain in the system only for a reduced number of frames. Nevertheless, since p-ants may be in local search behaviour across frames, the anchor points must be compensated for the agent’s motion the same way p-ants positions are, at \( i = 1 \) (see above).

If a p-ant in local search happens to detect a not too cluttered obstacle, it is appended to \( O' \), the shared medium \( P \) is updated accordingly, the p-ant’s energy is restored, and it changes to track behaviour. Finally, a p-ant \( a \) in local search that leaves the agent’s visual field is removed.
Algorithm 4: iterateTrack

**Input:** $I, R, P, A, O', FOV, a, \zeta, i$

**Output:** $R, P, A, O'$

1. apply detector at p-ant’s position, $(R, d_p) \leftarrow \text{detect}(I, R, p_a)$
2. if $i = 1 \land (p_a \notin FOV \lor (p_a \in FOV \land d_p = 1))$ then // update obstacles only at frame onset
   3. report the presence of obstacle at p-ant’s position, $O' \leftarrow O' \cup \{p_a\}$
   4. raise the obstacles density shared signal at $P(p_a)$
3. if $p_a \in FOV \land d_p = 1$ then
   4. restore p-ant’s level of energy, $o_a \leftarrow \rho$
   5. if $r_a > 0$ then // limit the number of replications
   6. randomly generate a neighbour position $q \in FOV$ for p-ant replication
   7. apply detector at neighbour position, $(R, d_q) \leftarrow \text{detect}(I, R, q)$
   8. if $P(q) < \eta \land d_q = 1$ then // new position at not too crowded obstacle region
   9. decrease the number of p-ant’s available replications, $r_a \leftarrow r_a - 1$
   10. replicate p-ant, $a' \leftarrow a$, do $p_{a'} \leftarrow q$, and add new p-ant to the p-ants
   11. set, $A \leftarrow A \cup \{a'\}$
   12. report the presence of obstacle at new p-ant’s position, $O' \leftarrow O' \cup \{q\}$
   13. raise the obstacles density shared signal at $P(q)$
   end
4. end
5. if $p_a \notin FOV$ then // p-ant out of visual field
   6. extra reduction in p-ant’s energy available, $o_a \leftarrow o_a - 1$
6. else // p-ant in the visual field
   7. if $P(p_a) > \zeta \lor d_p = 0$ then // p-ant at too crowded or on non-obstacle region
   8. change p-ant to local search behaviour, $b_a \leftarrow \text{LOCAL\_SEARCH}$
   9. set an anchor point for the p-ant’s local search, $z_a \leftarrow p_a$
   10. set the initial spread of the local search, $\sigma_a \leftarrow 0$
   11. lower the obstacles density shared signal at $P(p_a)$
   end
7. end
8. return $R, P, A, O'$
from the system, $A \leftarrow A\setminus\{a\}$. See Algorithm 5 for further details and Fig. 5 for an illustrative example of the local search behaviour.

**Algorithm 5: iterateLocalSearch**

**Input:** $I, (\Delta\mathbf{x}, \Delta\mathbf{y}, \Delta\mathbf{\theta}), R, P, A, O', \text{FOV}, a, \eta, \nu, l, i$

**Output:** $R, P, A, O'$

1. **if** $i = 1$ **then** // new frame received
2.  
3. **end**

4. use Equation 4 to update p-ant’s position around anchor point, $z_a$

5. **if** $p_a \not\in \text{FOV}$ **then** // p-ant out of visual field
6.  
7. **else**
8. apply detector at p-ant’s position, $(R, d) \leftarrow \text{detect}(I, R, p_a)$
9. **if** $d = 1$ **and** $P(p_a) < \eta$ **then** // p-ant on non-crowded region of obstacle
10. report the presence of obstacle at p-ant’s position, $O' \leftarrow O' \cup \{p_a\}$
11. raise the obstacles density shared signal, $P(p_a)$
12. restore p-ant’s level of energy, $o_a \leftarrow \rho$
13. change p-ant to track behaviour, $b_a \leftarrow \text{TRACK}$
14. **else** // local search must carry on
15. **if** $P(z_a) < P(p_a) < \eta$ **then** // anchor point less crowded than p-ant’s position
16.  
17. **else**
18. reduce dramatically p-ant’s energy available, $o_a \leftarrow o_a - \nu$
19. **end**
20. **end**
21. increment the spread of the local search, $\sigma_a \leftarrow \min(\sigma_a + 1, l)$
22. **end**
23. **return** $R, P, A, O'$

### 3.3.4 Location-Specific P-Ants

The described mechanism for p-ants creation is basically driven by the action selection process output. With it, the perceptual process is capable of adapting to the dynamic nature of the surroundings. However, there are some specific spots in the agent’s workspace that demand for invariant attention. To take into account these particular cases, new p-ants are deterministically deployed in each frame at points of interest. In this work we do it for the region right in front of the agent, where obstacles must be detected as soon as possible. These p-ants are setup to immediately engage in local search behaviour.
Figure 5: Graphical output generated by the perceptual process with erroneous motion estimate to illustrate the local search behaviour. Although the agent does not move, the motion estimate, which is noisy, reports a straight ahead motion. By motion compensation, p-ants move away from the obstacle. As the time unfolds (from $t_1$ to $t_4$) they manage to find the way back to the obstacle. Lines connect anchor’s and p-ant’s current positions. The grey-level colour coding is as in Fig. 2.

3.4 Action Selection Process

The action selection process produces at each iteration the multi-valued output $\mathbf{u}$. As mentioned, actions in this article are instantiated as linear trajectories centred on the agent and directed outwards in a radial pattern. For this study, $j = 0$ and $j = k$ correspond to $+90^\circ$ and $-90^\circ$ linear trajectories, respectively, both perpendicular to the straightforward motion. Hence, the utility of each possible action is defined as the value of moving the agent along the corresponding linear trajectory. The closer an obstacle is from the agent, the lower the utility of the linear trajectory affected by the obstacle.
The specific fast obstacle avoidance method employed in this study (Santana and Correia, 2006) determines which linear trajectory the agent should take to produce faster progress along the direction of the goal heading, $h$, without hitting any obstacle reported by the visual process. Its geometric nature ensures smooth navigation in cluttered environments. Accordingly, the non-normalised utility of a given linear trajectory, $j \in [0, k]$, represented by the pair $(j, d^j) \in C$, is

$$v^j = \beta_1 \cdot \left( \frac{(d^j/r) \cdot \cos(|\angle_j - h'|) + 1}{2} \right) +$$

$$\beta_2 \cdot (1 - |(d^j/r) \cdot \sin(\angle_j - h')|) +$$

$$\beta_3 \cdot \left( 1 - \left| \frac{j - v''}{k + 1} \right| \right),$$

where $v''$ is the action sent to the actuators in the previous frame, $\angle_j$ is the angle between linear trajectory $j$ and the agent’s main axis of motion, $h'$ is desired heading $h$ transformed to the agent’s frame of reference, and $r$ is the maximum obstacle detection range. The utility is higher in directions whose range without obstacles, $d^j$, projected on the desired direction of motion is larger (first term), whose lateral displacement from that direction is smaller (second term) and that are closer to the last actually sent to the actuators (third term) in order to introduce some inertia. $\beta_1$, $\beta_2$, and $\beta_3$ are empirically defined importance weights set up to avoid oscillatory behaviour.

Then, those trajectories whose utility equals the highest one have their utilities amplified three times. This empirically defined scaling factor allows the best linear trajectory to become even more interesting. The utility normalisation finally occurs by doing $u^j = \max \left( \frac{v^j}{\arg \max_j, v^j}, \varepsilon \right)$, where $\varepsilon$ is an empirically defined minimum residual utility every linear trajectory must contain. In this way all linear trajectories have a non-zero probability of being analysed by p-ants.

4 Experimental Results

4.1 Experimental Setup

All experiments were carried out on a Centrino Dual Core 2GHz running Linux/Ubuntu. A set of representative videos with the proposed model’s output is available in the authors’ website. In this study, a wheeled robot simulated in Player/Stage (Gerkey et al., 2003) with a width of $w = 0.3$m is used as test-bed. The robot’s sensory input takes the form of $300 \times 150$ body-centric Euclidean two-dimensional images of the robot’s $16\text{m} \times 8\text{m}$ frontal environment, constrained by a $90^\circ$ field-of-view. Obstacles are represented by black pixels, whereas non-obstacle regions are represented by white pixels. To emulate occlusions, only the closest obstacle along each radial direction outwards the robot is imaged.

1Videos with results overlaid at http://www.uninova.pt/~pfs/index/SwarmVideos_2.html
The following describes the model’s parameterisation. The number of iterations, $i$, and $i_{\text{max}}$, described in Section 3.2, have been set to 10 and 20, respectively. A higher number of iterations provided no significant added value to the stability of the system. The stochastic motion control parameters from Equation 3, $\lambda_1$ and $\lambda_2$, have been empirically defined as 0.3 and 1.0, respectively. These values produce an adequate speed-accuracy trade-off, for the environments where the simulated robot has been tested. The hysteresis control parameters used to determine when an obstacle is too cluttered with p-ants, $\eta$ and $\zeta$ (Section 3.3.2), have been set to 100 and 150, respectively. With these values, p-ants are allowed to densely populate obstacles without exhibiting too much overlap. The diffusion process control parameters (see Section 3.3.2), $\lambda$ and $\beta$, have been set to 3 and 7, respectively. These values have been chosen in order to generate a fast diffusion process, and also a relatively dense coverage of the obstacles. The number of times a p-ant is allowed to clone itself is $c = 10$. P-ants top energy is $\rho = 1000$ (see Section 3.3), which corresponds roughly to a 5s lifespan. This is sufficient to maintain a useful short-term memory for local navigation. The control parameter to avoid migration of p-ants between obstacles when performing a local search (see Section 3.3.3), $l$, has been set to 10. Location-specific p-ants (Section 3.3.4) are deployed with polar coordinates $[0, 20]^T$ (see Equation 1). At this position, the p-ant will guarantee that obstacles in the close vicinity of the robot are surely found, independently of the remaining configuration of the environment. The cardinality of the robot’s action repertoire, $k$, is 80. This action granularity is sufficient to ensure navigation in cluttered environments. The importance weights for the action selection process (see Section 3.4), $\beta_1$, $\beta_2$, and $\beta_3$, have been set to 0.35, 0.55, and 0.1, respectively. The residual action utility, $\epsilon$, has been set to 0.005.

Two 16m $\times$ 16m environments have been devised to carry out the experiments. The first (see Fig. 6), denominated lines environment, contains large (easily detectable) obstacles in the far-field. A proper behaviour in this case requires good look-ahead capabilities. Conversely, the second environment (see Fig. 7), denominated dots, is cluttered with small (hard to detect) obstacles. In this case the robot should be able to perform thorough near-field obstacle detection. In a given sense, the lines environment affords speed whereas the dots environment demands accuracy. Using the same parameterisation for both environments, that is, without explicit context awareness, we show the system self-organising in order to handle the different speed-accuracy trade-offs.

The ability of p-ants located outside the robot’s field-of-view to report their associated obstacles will be activated only in the last experiment, when explicitly mentioned. In this way, in the remaining experiments, we avoid any influence this feature may have in the results.

### 4.2 Results

During five runs per environment, the robot is asked to travel at a speed of $1.0 \text{ms}^{-1}$, along the heading coincident with the one of the robot when in its start position. Position (in meters) and heading (in radians) estimates are polluted with additive Gaussian noise, $(\hat{\Delta}x + 0.1 \cdot N(0, 1), \hat{\Delta}y + 0.1 \cdot N(0, 1), \hat{\Delta}\theta + 0.001 \cdot N(0, 1))$, where $N(0, 1)$ represents samples from a Gaus-
Figure 6: Behavioural results in the *lines* environment. Since all runs with active perception produced very similar trajectories, only one is plotted. The robot starts from the bottom-left aligned with the desired heading in the world frame. Filled trails left by the robot correspond to the ground-truth behaviour. Unfilled trails correspond to runs using the proposed active perception model. They are barely visible in this figure as a consequence of being covered by the filled trails. This means that the trajectory executed with the proposed model is near optimal, in the sense that it follows closely the ground-truth.

Figure 7: Behavioural results in the *dots* environment. The variability exhibited across the five runs (labelled with corresponding numbers) is a result of the environment’s complexity. Small variations in the decision at key points cause considerable differences in the path taken. Nevertheless, in qualitative terms, the behaviour of the robot with the active perception set-up follows closely the ground-truth. Same colour code and process of obtaining ground-truth as in Fig. 6.
sian distribution with mean 0 and variance 1. With this error profile, the re-detection of tracked obstacles is required nearly in almost every frame. This is a reasonable situation for robots performing, for instance, in outdoor (unstructured) off-road environments. For comparison purposes, a ground-truth behaviour is generated by providing the action selection process with all sensory information contained in the robot’s field-of-view, that is, with a fully-informed passive perception set-up.

The proposed model generates smooth robot paths that follow closely the ground-truth (see Fig. 6 and Fig. 7), which is itself near optimal. This is further confirmed by Fig. 8(a), in which the steering angle error per frame is plotted. This plot was obtained by frame-wise averaging the error over the five independent runs, in the *lines* environment. The error metric is computed as the absolute difference between the steering angle cast by the action selection process to the actuators and the one associated with the ground-truth behaviour. The small error magnitude confirms the behavioural analysis, revealing the ability of the proposed model to accurately reproduce the activity of a fully-informed (near-optimal) system.

Following the ground-truth is not a big achievement if the percentage of analysed sensor data is high, thus approaching the fully-informed situation. Fig. 8(b) illustrates exactly the opposite situation for the proposed model on the five runs in the *lines* environment. Namely, the mean percentage of pixels analysed per frame, that is, in which the obstacle detector is actually applied, is only 5% of the robot’s field-of-view. This shows that the method maintains an adequate speed-accuracy trade-off with parsimonious allocation of resources.

Because of the chaotic nature of the *dots* environment, independent runs on it lead inevitably to different paths. Being considerably different, their correlation along time is poor. The quantisation of the performance metrics followed by their frequency analyses, i.e., through histograms, is much more informative in this case. These histograms are built by bin-wise averaging the individual histograms obtained for each of the five independent runs. This frequency analysis highlights the ability of the system to match ground-truth (see Fig. 9(a)) and accurately focus attention (see Fig. 9(b)) in the *dots* environment. These results are comparable to those obtained in the *lines* environment. The frequency analysis for the *lines* environment are coherent with the time-based analysis (cf. Fig. 8).

Fig. 8(c) and Fig. 9(c) demonstrate the ability of the system to maintain a low computational cost, per frame, in what refers both action selection and visual processes. The mean per frame of 2.9 ms for the *lines* environment, and similarly to the *dots* environment, is residual when compared to typical cost of vision-based obstacle detectors, $\approx 100$ ms (Santana et al., 2008). Thus, the overhead caused by the active vision infrastructure will not be noticed when applied to real-life obstacle detectors. Note that the additional cost of image acquisition and motor actuation, which together take $\approx 100$ ms in the Player/Stage simulator, has not been considered. The fast computation obtained is in part due to the small convergence time of the action selection process. This can be observed by the small number of iterations required by the action selection process to stabilise its output (see Fig. 10).

As seen above, the system performs near-optimally and parsimoniously in both environments, with the same parameterisation. Although this shows the model’s robustness, it is interesting to dissect the underlying causes of such adaptability. Fig. 11(a) shows a shift to higher
Figure 8: A set of performance metrics per frame. Values are the mean computed over five independent runs in the lines environment. Standard deviation represented by the error bars.
Figure 9: Histograms of a set of performance metrics. Values are the mean computed over five independent runs. Histograms normalised by the total number of frames. Standard deviation represented by the error bars.
values on the number of agents in local search behavioural mode for the dots environment, when compared to the ones in the lines environment. Fig. 11(b) shows that the same trend is observed for the number of agents in search mode. Refer to Fig. 12 for a detailed view on the number of p-ants in each behaviour during the five runs engaged in the lines environment. The complexity of the dots environment presents a larger number of potential directions of motion, which in turn results in a higher number of p-ants being created. This explains the higher number of p-ants searching in the dots environment. The more p-ants are created the higher their concentration around obstacles, thus explaining the higher number of p-ants engaging in local search due to saturation of shared medium $P$. While trying to find a less saturated spot, p-ants local search increasingly spreads over wider areas around the objects, thus resulting in a qualitatively different behaviour from the observed in the lines environment. Additionally, in cluttered environments p-ants tend to be locked on obstacles closer to the robot, raising the density of p-ants where it is most needed. In sum, the qualitatively different behaviour raises the chances of detecting relevant obstacles. Hence, the more complex the environment the more p-ants are actively exploring it, and the more focused they are on the vicinities of the robot. This can be observed also in Fig. 9(b), where a higher number of analysed pixels in the dots environment than in the lines one is reported.

As previously argued, the use of parallelism is essential to promote load balancing by enabling some predictive power. To verify this argument the system was tested with and “without” parallelism. To avoid interferences of memory in this test, p-ants were not allowed to survive across cycles. Absence of parallelism occurs when: (1) noise in the stochastic search is not employed, that is, $(\lambda_1, \lambda_2) = (0.0, 0.0)$; (2) p-ants replication is inhibited; and (3) p-ants are deployed only in the direction corresponding to the action with highest utility, that is, $v$. In these conditions p-ants operate mostly on a greedy manner without massively engaging in self-organising collective behaviour, and consequently without fully exploiting the advantages of parallelism. As expected, Fig. 13(a) and Fig. 13(b) confirm that in these conditions both
Figure 11: Histograms of the number of p-ants in search behaviours. Values are the mean computed over five independent runs. Histograms normalised by the total number of frames. Standard deviation represented by the error bars.
ground-truth matching error and time to action selection convergence are higher.

Memory is important because it is what enables p-ants to move across frames and consequently to propagate evidence along time. This is confirmed by the reduced number of iterations required for action selection to stabilise when memory is activated, as depicted in Fig. 13(c). In a last experiment, the ability of p-ants located outside the robot’s field-of-view to report their associated obstacles and consequently to affect the action selection process, was activated. Fig. 14 shows that this capability is essential for the robot to be able to avoid dead-ends.

5 Discussion

5.1 Role of Parallelism

Focussing all perceptual resources on the analysis of the environment’s region associated to highest utility action is inefficient. Objects detected in other regions may also affect the utility of the most relevant region. The need for perceiving more than one region at a time demands a visual attention policy capable of handling the speed-accuracy trade-off. A detailed accurate analysis of the highest utility region may render impossible the analysis of the other regions. A too coarse analysis will hamper the detection of small obstacles. Defining an optimal visual search policy with this trade-off can only be done for well defined sets of agent-environment configurations. Instead, a self-organising visual parallel search policy is capable of covering a wider envelope at the cost of doing it sub-optimally. Robustness being more relevant than optimality for embodied agents, self-organisation is paramount.

When either the environment or the task change, the importance of an obstacle for the action selection process may also change. Had the sub-optimal parallel search not been performed while the context was static, the system would be overloaded while trying to stabilise both perceptual and action selection processes to the new context. Hence, parallelism is important to
Figure 13: Results on the influence of parallelism and memory. The histograms were built for the *lines* environment by the same method as in Fig. 9. The benefit of using memory and parallelism is reflected in the consistent shift of the corresponding histograms towards the left, in respect to the ones corresponding to the system without these features. (a) Histograms of the ground-truth matching error with and without parallelism. (b) Histograms of the number of iterations required for action selection to stabilise, with and without parallelism. (c) Histograms of the number of iterations required for action selection to stabilise, with and without memory.
Figure 14: Short-term memory results. The robot starts from the bottom-left aligned with the desired heading. The grey robot’s path corresponds to a typical situation where only information contained in the sensor’s field-of-view (90°) is used. In this situation the action selection is only provided with perceptual information concerning a small region of the dead-end. As a result the action selection process, performing in a greedy way, repetitively switches its output between hard-left and hard-right turns, leading ultimately to a deadlock. With an extended field-of-view (180°), enabled by the short-term memory, the action selection process is already able to better assess the utility of each action (lighter robot’s path).

5.2 Implicit Context Awareness

In the proposed model there is no explicit control of the maximum number of p-ants that can be deployed. It is fully defined by the utility vector, and consequently it is an emergent property resulting from the immersion of the system in a specific environment. Formally, the average number of p-ants being created per iteration is \( \sum_{j \in [0,k]} (u_j s_j) \). Practically, this means that for a speed of 0.5 ms\(^{-1}\), in an environment where the \( k = 81 \) possible motor actions exhibit an average utility 0.5, an average of 405 p-ants is created per frame, assuming an execution of \( i_{\max} = 20 \) iterations. In a situation where only three possible motor actions are possible, and with highest utility, the average number of created p-ants drops to 30. That is, the number of p-ants being created varies with the environment. Hence, the system adapts to the context without explicit awareness.

5.3 Comparison With Other Population-Based Methods

The fact that p-ants inspect the input frame according to the agent’s action selection process, in contrast to previous swarm-based related work for image analysis (Ramos and Almeida, 2000; Owechko and Medasani, 2005; Antón-Canalís et al., 2006; Mobahi et al., 2006; Broggi and
is a significant difference from an embodied cognition standpoint. The same applies to the ability of p-ants to implement spatial working memories and to be affected by the agent’s proprioception for motion compensation.

Another possible alternative to p-ants for obstacle search and tracking are Particle Filters. Although supported by a Bayesian framework, these filters exhibit two undesirable properties for them to be included in a self-organising embodied cognition framework. First, particles are sampled by a centralised process. This limits their explanation power with respect to the emergence of cognition in animals and their implementation onto parallel hardware, which is essential to enable robot miniaturisation and energetic sustainability. Second, particles do not communicate with each other, thus neglecting the recognised role of lateral connectivity for sustainable self-organisation in neural structures. In fact, the recently reported out-performance of swarm-based methods over particle filters in visual tracking tasks (Zhang et al., 2008; John et al., 2010) is most often attributed to these two aspects.

Particle filters are recognisably powerful in providing autonomous agents with Simultaneous Localisation and Mapping (SLAM) capabilities (Thrun et al., 2005). The link between particles and p-ants in this case is not straightforward. In general, a particle in SLAM is a sample of the joint distribution robot/map, where a map is composed by a set of Extended Kalman Filters (EKF), each associated to a given landmark of the environment. Conversely, a p-ant is associated to a single landmark of the environment. Hence, a p-ant is more related to the role played by each EKF than with the particles themselves. In contrast to EKFs in SLAM, p-ants perform visual search in addition to tracking, and do it guided by the action selection process. Moreover, while p-ants interact to improve their individual performance, EKFs in particles are independent.

P-ants do not contribute to the estimation of the localisation of the agent, as particles do. Local navigation being the focus of the proposed model, the localisation problem is handled solely with dead-reckoning. While SLAM methods could deliver both localisation and mapping, their high computational and memory cost makes them unsuited for parsimonious local navigation. To reduce computation, usually particle filters are considerably simplified and consequently lose their most important property, that is, their theoretical completeness. From this and the previous paragraph one can conclude that the problem of local mapping in self-organising embodied cognition is more convincingly solved by the proposed model than by SLAM models.

6 Conclusions

A parallel model of covert visual attention was proposed and validated in a simulated robot performing local navigation. The covert visual attention process has been modelled with swarms of simple virtual agents, named p-ants, based on the social insects foraging metaphor, motivated by the multiple covert attention hypothesis (Pylyshyn and Storm, 1988; Doran et al., 2009) and in line with the new born swarm cognition framework (Passino et al., 2008). Basically, p-ants perform local covert visual attention loops, whereas the self-organised collective behaviour maintains global spatio-temporal coherence.
The main goal of the model was to focus attention and consequently to reduce the processed image area. Experimental results confirm that this goal was attained, which suggests that the proposed model is able to run in real-time in vision-based robots with modest computational resources. The underlying rationale of the proposal is that the swarm-based fragmentation of the whole behaviour into simple local rules provides a robust system. Results also confirmed this.

Although the use of multiple virtual agents in image analysis is not new, this is the first report on its interaction with action selection. That is, p-ants opportunistically report their results to the action selection process, being at the same time guided by it. In other words, both action and perceptual processes are loosely coupled and unfold in parallel through cross-modulatory signals. This is fundamentally different from previous work in visual attention. For instance, in connectionist models, as typically considered in embodied cognition, both perceptual and action components share, to a large extent, the same computational units, that is, neurons. Being so tightly coupled, those approaches are difficult to modularise and consequently to scale. Conversely, another typical approach is to decouple both processes to the point they only share information in a master-slave way, such as in typical sensor planning strategies. This forces visual attention to operate at longer time-scales, which has little utility for the covert case.

The instantiation of the model onto a vision-based physical robot, to be pursued in future work, will be straightforward. It requires mostly: (1) the conversion between three-dimensional world and two-dimensional image space; and (2) changing the simple black/white discriminator that operates as obstacle detector by an appearance-based template matcher. In the future, we will also extend the model to enable the emergence of hierarchies among p-ants. This would allow hierarchical decomposition of visual attention and sparse spatial working memory. The parallel nature of the model can be further exploited, due to the emergence of new parallel computational platforms.

7 Acknowledgements

This work was supported by FCT/MCTES (grant number SFRH/BD/27305/2006). We would also like to thank the fruitful comments provided by Magno Guedes, Nelson Alves and anonymous reviewers.

References


