A Schema Based Model of the Praying Mantis

Giovanni Pezzulo and Gianguglielmo Calvi

Institute of Cognitive Science and Technology - CNR
Via S. Martino della Battaglia, 44 - 00185 Roma, Italy
giovanni.pezzulo@istc.cnr.it - gianguglielmo.calvi@noze.it

Abstract. We present a schema-based agent architecture which is inspired by an ethological model of the praying mantis. It includes an inner state, perceptual and motor schemas, several routines, a fovea and a motor. We describe the design and implementation of the architecture and we use it for comparing two models: the former uses reactive, stimulus-response schemas; the latter involves also forward models, which are used by the schemas for generating predictions. Our results show an advantage in using anticipatory components inside the schemas[1].

1 Introduction

Schemas [1] are basic functional units, permitting to investigate animal behavior without explicit assumptions about the physiological and neurophysiological realization and localization of the functions[2]. The model we propose is inspired by an ethological model of the praying mantis described in [2] but it is focused on anticipatory capabilities. It includes two kinds of schemas: **perceptual schemas** and **motor schemas**. Some schemas also are closely related (e.g. *detect predator* and *escape*): we call them **coupled perceptual-motor schemas**. In the rest of the paper we will call *schemas* the functional units, and *behaviors* the functions they realize, since many schemas can realize the same behavior.

Schema based design has three advantages: (1) it permits to integrate many competing behaviors in a coherent whole. While the animal has a large repertoire of behaviors (realized by its schemas), only few of them are useful in a given context. For this reason, *the activity level of the schema represents its relevance* [1, 8, 18]. The activity level of a perceptual schema represents a confidence level that a certain entity, encoded in the schema, is or is expected to be present. The activity level of a motor schema represents a confidence level that the behavior encoded in the schema is both applicable and useful in the current situation. (2) it affords distributed control: there is not a central executor, but the behavior of the animal emerges from the competition and cooperation of all the active schemas. (3) it permits to integrate in an unique framework data-driven, bottom-up processes, such as the influence of stimuli on the behavior; and hypothesis-driven, top-down processes, such as forming, maintaining and testing a coherent interpretation of the stimuli.

[1] This work is supported by the EU project **MindRACES**, FP6-511931.

[2] For an hypothesis of implementation of the schemas in the nervous system, see the notion of *command neurons* in neuroethology [17] or [1].

In the rest of the paper we present our schema based agent model and we test it in a simulated environment, with two goals: (1) to evaluate its adaptivity in a dynamic environment, i.e. its capability to select the appropriate schemas for satisfying its drives; (2) to compare anticipatory vs. reactive strategies.

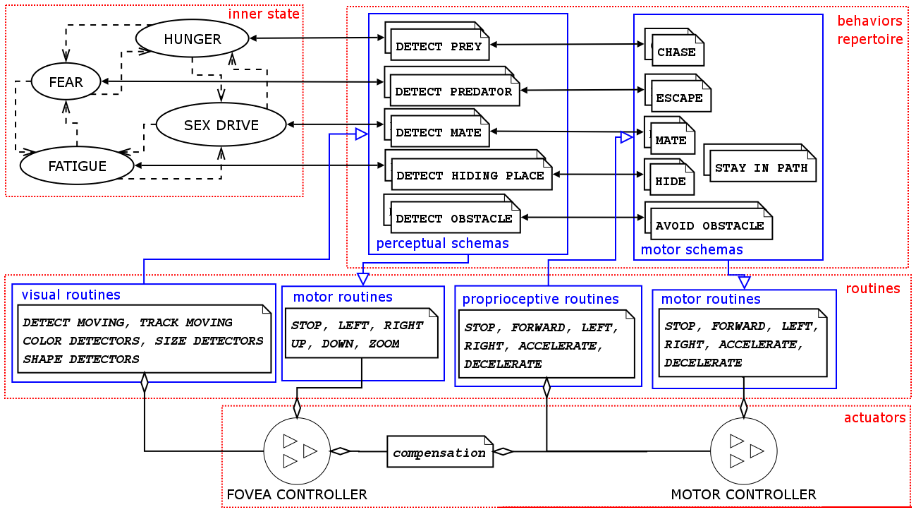

Fig. 1. The Components of the Mantis Architecture

Fig. 1 shows the main components of the model: the **Inner State** (the drives); the **Behavior Repertoire** (Perceptual and Motor Schemas); the **Routines** (Visual, Motor and Proprioceptive Routines); the **Actuators** (the Fovea and Motor Controllers), that we introduce in the next Section.

1.1 Functional Principles of the Architecture

According to the previous definition, schemas are concurrent processes, each one encapsulating the procedures to realize a behavior. In order to be effective and adaptive, the agent has to adopt the most relevant schemas: in our implementation, this depends on schemas activity level. In our parallel architecture the activity level of each component (including schemas) determines the priority of its thread of execution; activity level thus represents the "power and influence" of a schema even in computational terms. As we will see, a more active perceptual schema can process the visual input more quickly and a more active motor schema can send more commands to the motor controller. Routines have a variable priority, too, reflecting their relevance for the schemas which exploit them: for example, in some situations color-detectors can be very relevant and size-detectors can be less. The activity level of schemas is set according to three parameters: absolute relevance, contextual relevance and predictive success.

The *absolute relevance* represents how much a schema is relevant by default; for example, *detect predator* is more relevant in absolute than *avoid obstacle* for a living creature, even if sometimes the latter is more contextually relevant. If the animal has a repertoire of schemas for the same behavior (e.g. many specialized *detect prey* schemas, e.g. for gray or red, big or small preys), some of them are more relevant in absolute, e.g. because gray preys are more common. Ceteris paribus, more absolutely relevant schemas have higher activation levels.

The *contextual relevance* represents how much the schema is relevant in the current situation. If a prey is detected and the mantis is hungry, *chase* is very relevant; but it is much less relevant if there is no prey or if the mantis is not hungry. The contextual relevance is not centrally calculated, but emerges from the dynamics of the components of the distributed architecture, in two ways. The first way is preconditions matching; for example, *chase* has as a precondition the presence of a prey (or, to be more precise, an high activity level of *detect prey*). Preconditions are not necessary but facilitating conditions: they have fuzzy values, so they can match even partially and provide graded reinforce (the more the match, the more activation is gained by the schema). The second way is exploiting the links between the components, affording spreading activation. Our design methodology includes both pre-designed links (stroke edges in Fig. 1) and evolved ones. Thanks to the pre-designed links, drives spread activation to related perceptual and motor schemas (e.g. *hunger* to *detect prey*), and coupled perceptual-motor schemas (e.g. *detect prey* and *chase*) spread activation to each other. As an emergent result of the dynamics involving both kinds of links, the most relevant schemas become more active in a context-sensitive way.

The *predictive success* also regulates schemas activation. As we will see, the schemas incorporate a predictive component (a forward model) which generates expectations; schemas generating accurate expectations gain activation. The rationale behind this principle is that schemas which predict well are "well attuned" with the current situation; for example, if *detect prey* is activated by error by a big, gray entity (assuming that some preys are gray in the environment), the schema will try to track the prey (by moving the fovea) according to its forward model, i.e. as a moving object. If the object is not a prey but, say, a stone, its tracking activity will fail (because the entity does not move). The *detect prey* schema is not well attuned with the environment, while the *detect obstacle* is: in fact, its forward model predicts a static object. While in the beginning *detect obstacle* could be not very active, as long as its forward model predicts well it becomes more and more active and overwhelms *detect prey*[3].

Schemas activation is also regulated by a general architectural principle. There is a *limited amount of activation* (i.e. computational resources) shared by all the components. All the components thus compete for limited resources and active schemas prevent other ones to gain more activation[4].

[3] In the current implementation expectations are treated as preconditions only in the next schema cycle; if partially matched, they provide it graded activation.

[4] This is similar to having lateral inhibition between the competing components, but the total amount of activation can be manipulated (for example by the drives).

2 The Four Components of the Mantis Model

Here we introduce the four components of the mantis model: the **Inner State**, the **Schemas**, the **Routines** and the **Actuators**.

2.1 The Inner State

The inner state includes four drives: *hunger*, *fear*, *sex drive* and *fatigue*, which have inhibitory links (dashed edges in Fig. 1). All the drives except *fear* are regulated by an endogenous factor, a "biological clock", creating an *habit* system: the mantis routinely needs food and repair and spreads activation to the related schemas for fulfilling these needs (stroke edges in Fig. 1); of course schemas can operate only if there are appropriate environmental conditions. The drives also receive exogenous influences, i.e. the activity level of the related schemas; for example, if *detect predator* is very active, *fear* grows up. There is thus an activation loop between internal drives and schemas. In the current implementation the mantis do not starve and is not really harmed by predators; on the contrary, fatigue has a real effect: it diminishes the overall amount of activation available.

2.2 The Schemas

The schemas (*perceptual* and *motor*) are the main components of the model. As shown in Fig. 1, many schemas realize the same behavior; as an example, there are *detect prey* schemas specialized for gray or red preys, or for big or small ones.

The Perceptual Schemas The model includes five kinds of perceptual schemas: *detect prey*, *detect predator*, *detect mate*, *detect hiding place*, *detect obstacle*. Each perceptual schema has three components: a *detector*, a *controller* and a *forward model*. The main role of the detector is to acquire relevant input (preconditions) from the the fovea. The main role of the controller is to send motor commands to the fovea: in this way the mantis is able to orient its attention. Perceptual schemas are not passive data processing structure, but active ways for "navigating" the visual field [19]. The main role of the forward model is anticipate visual stimuli, i.e. the activation level of appropriate visual routines.

The perceptual schemas become more active if the kind of stimuli they process are indeed present in the environment. In our implementation, the detector has (graded) preconditions which are associated to visual routines; when the relevant visual routines are active, the schema gains activation. For example, in the mantis environment preys are gray; if the *gray-detector* visual routine is very active, the *detect prey* schema gains activation, too. The perceptual schemas also run their forward models: schemas which predict well gain activation.

The perceptual schemas receive activation from the inner states, too; a fearful mantis will search for predators even in absence of real danger. An "hallucinatory" phenomenon is in play: when a mantis is fearful, predators appear closer, moving entities appear to be predators (and get it even more fearful). In the long run hallucinations are ruled out: since the perceptual schemas also feedback on the inner states, the lack of dangerous signs (and the predictive errors of *detect predator*) will make the mantis less fearful.

Active perceptual schemas have two ways to induce top-down pressures. Firstly, they send motor commands to the fovea, orienting it toward relevant entities; more active schemas send commands with higher fire rate. By orienting the fovea, the schemas are able to partially determine their next input (they have an active vision, [19]). In an anticipatory framework, this functionality is mainly used to test the predictions of the forward models: for example, tracking a moving object is a way to acquire new stimuli in order to test the expectations. For this reason, the schemas orient the fovea towards the more informative points, i.e. those able to determine whether or not their predictions are correct. Secondly, they spread activation to the related visual routines. For example, *detect prey* schema activates the *gray-detector* visual routine, even in absence of real stimuli. This induces an "hallucinated" state (like a fake gray entity) which is close to **visual imagery** in [16]). As in the previous case, without real stimuli the hallucinated state lasts shortly.

As an effect of top-down pressures, the same stimulus is interpreted in different ways depending on the active perceptual schemas. For example, if a prey and an obstacle have the same color (say gray) and the gray-detector is very active, both *detect prey* and *detect obstacle* detect it. However, the more active schema detects it faster and takes controls of the fovea: if *detect prey* is more active, it is likely that the fovea will try to track it as a moving object (the *detect obstacle* schema, on the contrary, would have monitored it as a static object). Of course, more active perceptual schemas activate more their related motor schema, too.

The Motor Schemas The model includes six motor schemas: *stay in path* (the default behavior), *chase*, *escape*, *mate*, *hide*, *avoid obstacle*. They have three components: a *detector*, which sets the value of the preconditions by monitoring the state of the perceptual schemas (e.g. *detect prey* is *very_active*); a *controller* (an inverse model), which send commands to the motor (e.g. *move left*); and a *forward model*. The motor schemas receive activation from the related perceptual schemas in the form of matched preconditions: a very active *detect prey* activates *chase* (which learns to interpret it as: "there is a prey"). The motor schemas receive also activation from the inner states: a fearful mantis activates its motor routines for escaping even in absence of real danger; as in the case of perceptual routines, they can only remain active if the right stimuli are in place. The main role of the controller is to send commands to the motor. The main role of the forward model is to produce expectations about perceptual stimuli (to be matched with sensed stimuli, including vision and proprioception).

Coupled Perceptual-Motor Schemas Fig. 2 shows the pseudo-closed loop between controllers and forward models in a coupled perceptual-motor schema. The controllers send a control signal to the actuators, which integrate them and act accordingly; on the same time, an efference copy of the (final) command signal is sent to the forward models of all the schemas, which compute the next expected input. The dashed lines indicate that a feedback signal is received (via visual or proprioceptive routines); the dashed circles indicate that there is a comparison between the actual input stimulus and the expected stimulus. The degree of (mis)match between actual and expected stimulus has two functions: (1) *Adjustment of Control*: the predicted signal can compensate time delays, filter or replace missing or unreliable stimuli; see [7] for a comparison with Kalman filtering; (2) *Schema Selection*: schemas which predict well gain activation.

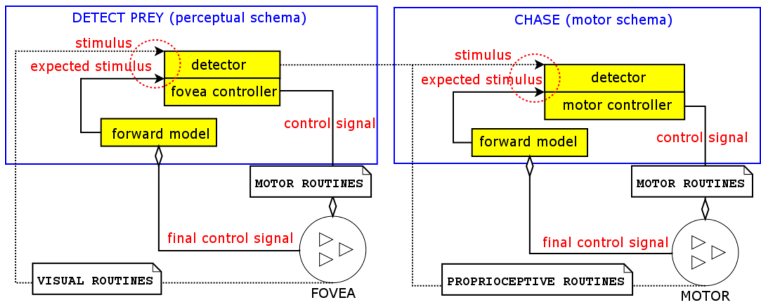

Fig. 2. A coupled perceptual-motor schema: *Detect Prey* and *Chase*

Patterns of Actions Some schemas include many concatenated actions (in a way similar to [4]); e.g. *detect prey* and *chase* have the following structure:

```
DETECT PREY:  IF red AND moving THEN find_prey
(loop)        ELSE IF prey_found THEN maintain_prey
              ELSE IF prey_lost THEN re_find_prey
              ELSE IF prey_maintained THEN maintain_prey

CHASE:        IF prey_found THEN approach_prey
(loop)        ELSE IF prey_close THEN grab_prey
              ELSE IF prey_in_contact THEN eat_prey
```

Schemas are always-looping procedures; for each cycle, depending on preconditions, an action is selected. This means that actions can be executed in different sequences, in parallel and also skipped: for example, *re_find_prey* is only needed when a prey is lost. Coupled perceptual-motor schema can realize complex strategies by coordinating their patterns of actions. Fig. 3 illustrates the example of a chasing behavior involving two schemas, *detect prey* and *chase*.

In the beginning, only *detect prey* is active, because some of its preconditions are true (e.g. the visual routines *red-detector* and *movement-detector* are highly active). *Detect prey* both activates its first action (*find prey*, which sends commands to the fovea) and spreads activation to *chase*. The first applicable action

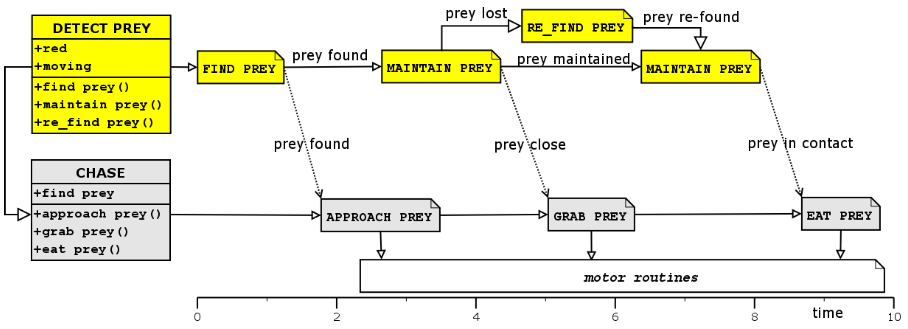

Fig. 3. Complex patterns of actions within a coupled perceptual-motor schema

of *chase* (*approach prey*) can only start when *find prey* succeeds; subsequently, the actions of the two schemas continue in a coordinated way: as long as the perceptual schema succeeds in finding and maintaining the prey, the motor schema tries to reach, grab and eat it.

Interaction-Oriented Representations It is worth noting that inside the forward models schemas process information in an interaction-oriented format, and the mantis only has deictic representations. For example, *detect prey* is only able to represent "the prey I am looking at". However, even without storing extra information, the architecture implements a certain kind of object permanence. Since schemas activation decay gracefully, it is highly probable that schemas which where very active remain quite active even if the stimulus disappears for a while. This effect is magnified by the presence of drives, which have a stabilizing effect on behavior: since they continue to fuel schemas for a given span of time, drives can be seen as **task-specific memories**, introducing commitment without central control. Moreover, since schemas act according to their predictive models, they remain attuned with relevant entities by actively searching them. For example, during a successful chase *detect prey* and *chase* gain activation thanks to the success of their predictions. On the contrary, failure in finding, maintaining or reaching a prey weakens the schemas and eventually the chase ends, if another behavior becomes more active. This example shows how, in stable enough environments, deictic representations and agent-environment engagement based on predictions can realize (at least a limited version of) complex functionalities such as maintaining objects permanence.

2.3 The Routines

The perceptual schemas do not receive raw input from the fovea: a number of preprocessing units, the *visual routines*, filter fovea information (although with different priority). In the current implementation there are several routines of each kind, such as color-detectors specialized for detecting different colors, as well as for colors, sizes, shapes and for detecting and tracking moving entities. The activation level of the visual routine directly encodes the presence of absence of associate entities; for example, an active red-detector encodes directly the presence of red entities as provided by the 3D engine, without learning. In a similar way there are *motor routines*, commanding the fovea and the motors, and *proprioceptive routines*, providing feedback information from the motors.

2.4 The Actuators: Motor and Fovea Controllers

The actuators receive commands from the motor routines and perform command fusion. Differently from many systems in literature (see [5] for a review), in which many schemas can be partially active at once but only one is selected for commanding the actuators, in this model each active schema sends its motor command. Since we adopted a parallel architecture in which schemas can have different priority, commands are sent asynchronously and with different fire rates. *Fire rate encodes relevance*: more active (and thus more relevant) schemas are able to send more commands to the actuators and to influence it more.

Command Fusion As already discussed, selection is needed both for adopting the most appropriate schema(s) for realizing the same behavior, and for adopting the most appropriate behavior. As an example of the first case, consider that there can be many *detect prey* schemas which are specialized e.g. for small and quick ones or for very big and red ones; in order to realize prey detection, often many *detect prey* schemas are needed, as in the “mixture of experts” model [13]. The case is similar for motor schemas. As an example of the second case, consider that the agent has a repertoire of behaviors and has to arbitrate between them (e.g. *chase* vs. *escape*), as long as it can not fulfill all them together.

In both cases, the fuzzy based command fusion mechanism we adopted [15] produces the course of actions accounting for more drives and stimuli. Strictly speaking, there is no actual “selection” since all the active schemas send their commands to the actuators, although with different fire rate; the course of action results from the graded contribute of all the active schemas. For example, a detect prey behavior is often realized not by a single *detect prey* schema, but integrating the graded contribute of many ones. The rationale is that exemplars of preys do not fall into clear cut categories (which prototypes are encoded in the schemas), such as “quick” or “slow” so it is often necessary to fuse the commands of the two schemas specialized for quick and slow preys. As an example, consider that a moving prey can match the preconditions of both schemas, although with different degrees of matching; thus, the degree influence they have on the fovea depends on the degree of membership of the prey to their prototype (expressed in fuzzy terms in the current implementation).

Mixed courses of actions can also emerge from the contribute of schemas realizing different behaviors. As an example, Fig. 4 shows the activation levels of two schemas, *stay in path* (black boxes) and *avoid obstacle* (white boxes), during obstacle avoidance. Both schemas are involved, with different priorities over time, as long as they can both be satisfied together. Note that the trajectory and the turning points are not preplanned but dynamically emerge depending on the size of the obstacle and the initial direction of the agent.

Exploitation also happens when the results of a schema are exploited by another behavior. For example, a mantis which is *escaping* can activate an *hide* schema as a part of the escaping strategy; the latter schema is not selected per se, but activated and exploited by the former behavior.

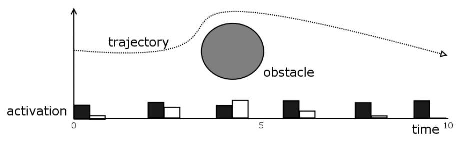

Fig. 4. Evolution over time of the activation of the schemas during obstacle avoidance.

Constructive Perception As discussed in the introduction, schemas permit to model both bottom-up and top-down phenomena, for example in perception. A classic experiment [25] shows that human attention varies with the nature of the task. When there is not an explicit task specification, bottom-up processes are mainly responsible for determining the salience of objects in the scene [12]. On the contrary, if there is an explicit task specification, top-down and volitional processes frame the scene and drive attention to the task-specific relevant objects in the scene [9]. We call this process *constructive perception*.

In this framework perception is distinct from sensing: *the active perceptual schemas represent multiple concurrent perceptual hypotheses which compete for being accepted*; they are prioritized according to the accuracy of their preconditions and predictions, i.e. how much their requirements are compatible with the actual perception. Schemas also actively drive perceptual exploration of the environment by orienting the fovea. The constructive process does not only influence stimuli categorization (such as prey vs. obstacle), but also behavior selection. An example of goal oriented constructive perception can clarify the point: if the mantis is not hungry and is escaping, it can approach a prey as an obstacle and activate *avoid obstacle*. Constructive perception is thus the abductive process of producing and testing hypotheses: the most active schemas drive subsequent actions (i.e. chase a prey or avoid an obstacle), active perception (where to orient the fovea) and visual imagery (to which visual routines give priority). Indeed, information is selected and it serves to confirm or disconfirm the running hypotheses, not to mirror the environment.

As discussed above, the behavioral and perceptual spaces of the mantis are also shaped by its internal drives. Drives provide activation to the behaviors, which can thus perform more epistemic actions, predict more often and influence more the fovea. An hungry mantis is much more likely to interpret ambiguous evidences as food; more precisely, an hungry mantis puts much more resources in classifying an evidence either as food or not food (instead of e.g. shelter or not shelter) even if the affordances of the object are the same.

3 Implementation and Testing

We implemented the mantis model by using the cognitive modeling framework AKIRA [11, 21], and the 3-D engine Irrlicht [10], having realistic physics. Our aim is to investigate if our architecture fulfills adaptively its drives in a dynamic environment. We followed the above described architectural design, inspired by the ethological model reported in [2]; we also set up two learning phases. In the former the components of each schema, controllers (inverse models) and forward models, and the parameters such as the schemas absolute relevance or the weight of the edges, were first learned individually in a simple environment having a limited number of features (e.g. only preys or predators). In the latter all the schemas were integrated in an unique architecture. Since the framework puts schemas in competition, adding new behaviors did not waste the performance of old ones; the challenge is now to coordinate them in a complex environment. In this phase the inverse and forward models did not learn any more, but schemas which were active in the same span of time evolved energetic links (in addition to those shown in Fig. 1) with hebbian learning [15]. Schemas which were not active in a given context learned to spread energy to more successful ones, too, with a mechanism described in [20]. The rationale is that the energy has to be conveyed in a context-sensitive way from less to more relevant schemas[5].

Each schema is implemented by using a single thread which activation is set according to the principles explained above: absolute relevance (learned in the first phase), spreading activation (via edges learned in both phases) and degree of match of its preconditions and expectations. All the representational elements (activation, preconditions and expectations) have fuzzy values: in this way it is possible to compare all them and obtain graded results. For example, a *very_active* internal drive (hunger) provides high match with the *quite_hungry* precondition of *detect prey*. Or a poorly active *detect_movement* routine provides low match for the expectation *prey_moving* produced by the forward model of *detect prey*. Drives, inverse and forward models were implemented by using both Fuzzy Cognitive Maps and Neural Networks [15], with minor differences. Drives values vary according to their links, their "biological clock" and the input they receive from the active schemas. In turn, the values of the drives become input for the schemas, as described above. Even the motor commands have the form of fuzzy statements such as *turn_left* and a fuzzy controller is responsible for command fusion, as in [20]. A motor routine (*compensation*) compensates the

[5] The main reason of having two phases is that it is very complex to learn many behaviors together. For example, the prediction error of the forward model can be interpreted either as scarce relevance of the schema or as poorly accurate forward model. Learning each forward model individually permits to disambiguate this signal.

movements of the agent, permitting to maintain the right orientation of the fovea during movement.

Related Literature Similar models in literature are MOSAIC [24] and HAMMER [6], implementing coupled inverse and forward models for motor control and basing schema selection on predictive success; schema architectures [1, 2] and the "mixture of experts" model [13]. However, we use a parallel architecture in which computational resources (such as speed) encode "responsibility"; command fusion is asynchronous and based on fuzzy logic. Our model also includes hebbian learning and spreading activation between the schemas.

Anticipatory vs. Reactive Systems We compared the performance of two variants of the mantis model (MANTIS and MANTIS-R) in a complex environment including obstacles, preys, predators and hiding places. The first model (MANTIS) is the one we have described throughout the paper. The second model (MANTIS-R) lacks the forward models. Obviously, in this case *predictive success* can not be used for allocating activation; only absolute and contextual relevance are used. Some anticipatory capabilities are however *implicitly* present in MANTIS-R, too: for example, a perceptual schema spreading activation to a visual routine implicitly assumes that its results will be useful; and this prediction is grounded on the history of their past interactions. However, endowing schemas with a predictive component (a forward model) which produces *explicit* expectations (as in MANTIS) permits also to use predictive error for action control and schema selection.

The two agents dwell separately in the same environment with the same inner states. **Drives satisfaction** was used as success metric: each agent had to satisfy its drives, i.e. to keep their values close to zero. Moving increases *fatigue*, while resting in hiding places lowers it; eating preys lowers *hunger*; the presence of close predators increases *fear*, while their absence lowers it; mating lowers *sex drive*. Four analysis of variance (ANOVA) with mean *fatigue*, *fear*, *hunger* and *sex drive* satisfaction (calculated as $1 - mean\_drive\_value$, in 100 real-time, 3-minutes simulations) as dependent variables were carried out.

DRIVE	**MANTIS**	**MANTIS-R**
Fatigue	0,845	0,657
Fear	0,812	0,665
Hunger	0,758	0,703
Sex Drive	0,891	0,793
Average	0,826	0,704

Table 1. MANTIS vs. MANTIS-R (mean satisfaction)

Tab. 1 shows the results. The main effect is significant for all drives ($F(1,99) = 130,53; p <,00001$ for *fatigue*; $F(1,99) = 68,01; p <,00001$ for *fear*; $F(1,99) =$

$24,82; p <,00001$ for *hunger*; $F(1,99) = 50,65; p <,00001$ for *sex drive*): in all cases, MANTIS satisfies its drives better than MANTIS-R. Our results indicate that in a scenario involving multiple entities and drives it is advantageous to exploit anticipatory representations, and this advantage overwhelms the cost to maintain and to run them in real time. However, our results can be considered preliminary: further investigation is needed for understanding at which level of environmental complexity anticipatory strategies become advantageous.

4 Conclusions and Further Work

We have presented a schema based agent architecture; illustrated its components and its action selection strategy; tested its behavior in a complex environment, also comparing it with a simpler model only having *implicit* anticipatory capabilities in an adaptive drives satisfaction task. Our results show that there is a significant advantages in using *explicit* expectations, produced online by forward models, for action control and schemas selection. Recently many authors [3, 7, 24] have provided evidences for a crucial role of anticipations and forward models in these and other cognitive functions, which we are now investigating.

In our architecture the top-down influences introduced by drives follow the **ideomotor principle** [14], arguing that action planning takes place in terms of anticipated features of the intended goal. For example, the drive *hunger* endogenously activates related perceptual and motor schemas (*detect prey* and *chase*), which in turn pre-activate visual and motor routines related to preys and pray-chasing. In [20] we have also show that this mechanism can also realize more complex goal states such as *search the red prey.* We are now investigating how to extend this principle to realize decoupled, off-line processing, such as planning by off-line producing, evaluating and comparing hypothetical, alternative goal states and courses of events, even if new and never experimented before.

In our architecture an high activation level of a perceptual schema represents the (actual or expected) presence of related entities, and motor schemas can exploit this information; in a similar way, active visual routines are exploited as preconditions by the perceptual schemas. We are now investigating how to extend this principle to schemas organized hierarchically; [20] describes the preliminary implementation of a layered architecture including feature-specific and increasingly complex concept-specific schemas, in which the activity of schemas in the lower layers *is interpreted as an information* by schemas in the higher layers, and in which *complex schemas exploit simpler ones* which realize some of their preconditions or expectations. Schemas in the higher layers are specialized for satisfying the drives of the agent; on the contrary, [22], also based on [13], shows that if hierarchical systems are evolved for a single task they do not specialize in a feature- or concept-specific way.

References

1. Arbib, M.A. (1992) Schema Theory. in the *Encyclopedia of Artificial Intelligence*, 2nd Edition, Editor Stuart Shapiro, 2:1427-1443, Wiley.

2. Arkin, R.C., Ali, K., Weitzenfeld, A., Cervantes-Prez, F. (2000) Behavioral models of the praying mantis as a basis for robotic behavior. *Robotics and Autonomous Systems*, Vol. 32, No. 1, pp. 39-60.
3. Barsalou, L. W. (1999) Perceptual symbol systems. *Behavioral and Brain Sciences*, 22, 577-600.
4. Bryson, J. (2000) *The Study of Sequential and Hierarchical Organisation of Behaviour via Artificial Mechanisms of Action Selection.* M.Phil Thesis. University of Edinburgh.
5. Crabbe, F.L. (2004) Optimal and non-optimal compromise strategies in action selection. In *Proceedings of SAB 2004.*
6. Demiris, Y., Khadhouri, B. (2005). Hierarchical Attentive Multiple Models for Execution and Recognition (HAMMER). *Robotics and Autonomous Systems Journal.*
7. Grush, R. (2004) The emulation theory of representation: Motor control, imagery, and perception. *Behavioral and Brain Sciences*, 27 (3): 377-396.
8. Gurney, K.N., Prescott, T.J., Redgrave P. (2001) A computational model of action selection in the basal ganglia, I. A new functional anatomy, *Biological Cybernetics* 84, 401-410.
9. Hopfinger, J. B., Buonocore, M. H., Mangun, G. R. (2000) The neural mechanisms of top-down attentional control. *Nature Neuroscience*, 3(3):284291.
10. http://irrlicht.sourceforge.net/
11. http://www.akira-project.org/
12. Itti, L., Koch, C. (2001) Computational modeling of visual attention. *Nature Reviews Neuroscience*, 2(3):194-203
13. Jacobs, R.A., Jordan, M.I., Nowlan, S.J., Hinton, G.E. (1991). Adaptive mixtures of local experts. *Neural Computation*, 3:79–87.
14. James, W. (1890) *The Principles of Psychology.* Dover Publications, New York.
15. Kosko, B. (1992) *Neural Networks and Fuzzy Systems*, Prentice Hall International Inc, Singapore.
16. Kosslyn, S.M., & Sussman, A.L. (1994). Roles of imagery in perception: Or, there is no such thing as immaculate perception. In M. Gazzaniga (Ed.), The cognitive neurosciences (pp. 1035-1042). Cambridge, MA: MIT Press.
17. Kupfermann, I., Weiss, K. (1978) The command neuron concept. *Behavioral and Brain Sciences* 1:3-39.
18. Maes, P. (1990) A Bottom-Up Mechanism for Behavior Selection in an Artificial Creature. In Jean- Arcady Meyer and Stewart W. Wilson, editors, *Proceedings of SAB 1990*, pages 238-246, Cambridge, Mass. The MIT Press.
19. O'Regan J.K., Noe, A. (2001) A sensorimotor account of vision and visual consciousness. *Behavioral and Brain Sciences*, 24(5):883–917
20. Pezzulo, G., Calvi, G., Lalia D., Ognibene D. (2005) Fuzzy-based Schema Mechanisms in AKIRA. *Proceedings of CIMCA'2005*, M. Mohammadian (ed.). Vienna.
21. Pezzulo, G., Calvi, G. (2005). Dynamic Computation and Context Effects in the Hybrid Architecture AKIRA . In Anind Dey, Boicho Kokinov, David Leake, Roy Turner, *Modeling and Using Context: 5th International and Interdisciplinary Conference CONTEXT 2005.* Springer LNAI 3554.
22. Tani, J., Nolfi, S. (1999) Learning to perceive the world as articulated: An approach for hierarchical learning in sensory-motor systems. Neural Networks 12:1131–1141.
23. Tyrrell, T. (1993) *Computational Mechanisms for Action Selection.* PhD thesis, University of Edinburgh.
24. Wolpert, D. M., Kawato, M. (1998) Multiple paired forward and inverse models for motor control. *Neural Networks* 11(7-8):1317-1329
25. Yarbus, A. (1967) *Eye Movements and Vision.* Plenum Press, New York.