ARCHITECTURAL METHODOLOGY BASED ON INTENTIONAL CONFIGURATION OF BEHAVIORS

François Michaud*, Gérard Lachiver and Chon Tam Le Dinh

Department of Electrical and Computer Engineering

Université de Sherbrooke

Sherbrooke (Québec) Canada J1K 2R1

Tel.: (819) 821-7141

Fax: (819) 821-7937

{michaudf, lachiver, ledinh}@gel.usherb.ca

*(819) 821-8000 (2107), michaudf@gel.usherb.ca, corresponding author.

ABSTRACT

Intelligence has been an object of study for a long time. Different architectures try to capture and to reproduce these aspects into artificial systems (or agents), but there is still no agreement on how to integrate them into a general framework. With this objective in mind, we propose an architectural methodology based on the idea of intentional configuration of behaviors. Behavior-producing modules are used as basic control components that are selected and modified dynamically according to the intentions of the agent. These intentions are influenced by the situation perceived, the knowledge about the world and internal variables that monitor the state of the agent. The architectural methodology preserves the emergence of functionality associated with the behavior-based paradigm in the more abstract levels involved in configuring the behaviors. Validation of this architecture is done using a simulated world for mobile robots, in which the agent must deal with various goals like managing its energy, its well being, finding targets and acquiring knowledge about its environment. Fuzzy logic, a topological map learning algorithm and activation variables with a propagation mechanism are used to implement the architecture for this agent.

Key words: Architectural methodology, behavior-based, reasoning, motivation, fuzzy logic, topological representation.

1. INTRODUCTION

Artificial intelligence, psychology, biology, neuroscience, linguistics and many other fields try to get insights on the principles, methodologies and concepts associated with intelligent capabilities. Being able to model the world, plan and predict events, deliberate from possible alternatives, learn, adapt, react, and many other abilities can be associated at different levels with intelligence. Designing an agent that manifests all of these abilities is a complex problem usually decomposed into smaller parts. Until recently, intelligent systems were designed with a strong emphasis on symbolic knowledge representation of the world (using a hierarchy of abstractions) and based on a functional decomposition of the decision process (like perception, modeling the world, executing plans and monitoring events) (Albus 1991). While this methodology allows agents to reason about their environment, designing autonomous agents that have to operate 'intelligently' in unknown, unstructured and unpredictable environments requires a different approach.

The introduction of the behavior-based paradigm (Arkin 1998; Brooks 1986) created a shift from the previous focus in AI on knowledge representation and planning toward the notion of sensing and acting within the environment (Arkin 1998; Kortenkamp *et al.* 1998). Behavior-based systems use behavior-producing modules (also called behaviors) that work in parallel to decide what the agent should do according to its state and the perception of the world. The approach has shown to be especially effective for mobile robots, allowing them to adapt to the dynamics of real-world environments without operating upon abstract representations of reality (Brooks 1991). One important contribution of this approach is that it shows the importance of having a functionality emerge from the interactions between the behavior-producing modules and the environment. However, this does not mean that knowledge representation and planning are of no use in designing intelligent autonomous agents. Extensions to behavior-based systems are necessary to handle more complex tasks requiring planning or for managing multiple and conflicting goals. Different hybrid architectures (Arkin 1998) have been designed and successfully validated (Kortenkamp *et al.* 1998), showing how behavior-based systems can benefit from abstract representation knowledge. A basic principle for making the coupling is to dynamically reconfigure the behaviors according to reasoning done over the representation knowledge available. But deriving an adequate representation depends on what can be modeled and what effect the representation must have on the decision process. This in turn is affected by the agent's own limitations in sensing, in acting and in processing, and by the interaction dynamics with the environment. Therefore, it is important to give to the agent the ability to cope efficiently with these limitations. The agent needs to be able to decide by itself on what it should base the configuration of its behavior-producing modules. In that regard, motivations (McFarland and Bösser 1993; Maes 1991; Parker 1992; Breazeal 1998) and emotions (Simon 1967; Sloman and Croucher 1981; Pfeifer 1988; Ortony *et al.* 1988; Albus 1991; Bates 1994), concepts that are gaining importance in the design of autonomous agents, may be very useful in making an efficient connection between adapting to the contingencies of the world and making the agent accomplish its goals.

To combine all of these aspects, we propose an architectural methodology that is based on the idea of intentional configuration of behaviors, with the intentions being influenced by the situation perceived, the knowledge acquired or innate about the world and internal variables that monitor the state of the agent. The architectural methodology preserves the emergence of functionality associated with the behavior-based paradigm in the two more abstract levels involved in configuring the behaviors. Section 2 presents the architectural methodology and its general characteristics. Validation of the architecture is done using a simulated environment for mobile robots with particular mechanisms designed for these experiments, as described in Section 3. Section 4 presents the results obtained, followed in Section 5 by a discussion about the properties of the architecture demonstrated in the experiments. Section 6 summarizes related works, and Section 7 concludes the paper.

2. ARCHITECTURAL METHODOLOGY BASED ON INTENTIONAL CONFIGURATION OF BEHAVIORS

The architecture, shown in Figure 1, is made of three levels: the *Behavior* level, the *Recommendation* level and the *Motivation* level. The *Behavior* level is composed of the behavior-producing modules, responsible for deriving actions from sensory information. Behaviors are the agent's skills for responding to the situations encountered in the environment. These behaviors all run in parallel and their resulting commands are combined using a fixed or flexible arbitration mechanism (like Subsumption (Brooks 1986), motor schemas (Arkin 1998), fuzzy logic (Saffiotti *et al.*, 1995) or other methods) to generate the control actions of the agent.

FIGURE 1. Architecture for intentional configuration of behaviors

Based on this repertoire, the other modules of the architecture are responsible for changing the selection of behaviors and to configure them according to the agent's goals and the situations experienced in the world. The *Recommendation* level uses three modules that formulate, in parallel, behavior recommendations from different monitoring conditions. The names chosen for these modules try to characterize the conditions they monitor and their effects on the agent. The *External Situation* module evaluates sensory information to affect behavior selection. The *Needs* module selects behaviors based on the motives of the agent, possibly correlating them with perceived events. The third recommendation module, called *Cognition*, exploits or acquires knowledge about the environment to plan or to prepare the use of behaviors. This knowledge may be influenced by perceived conditions in the world, by motives or by the intentions of the agent. Different kinds of knowledge can be incorporated in this module according to what can be useful for the agent. Cognitive recommendations can also be influenced by information coming from behaviors via the *Internal Parameters* link, as they can configure specific behaviors using the same link. This allows including mechanisms that generate virtual inputs to behaviors, adapt the control parameters of a behavior or use behaviors to recognize specific objects in the world (Arkin 1998).

The use of three recommendation modules has been influenced by the hypothesis that human behavior is affected by the environment, the knowledge acquired or innate about it, and the needs of the individual. The importance of each of these influences explains why the global behavior of a person is more reactive, rational, or egotistic. These three modules can be compared to three 'behavior-recommending' modules responsible for making decisions about which behavior-producing modules must be activated or inhibited according to what the agent wants to do and what situations it is experiencing in the world. These decisions take the form of a *desirability* measure and an *undesirability* measure for each behavior, inspired by the hedonic axiom which indicates that the organisms direct their behaviors to minimize aversions and maximize desirable outcomes (Beck 1983). These measures make it possible to prevent possible conflicts when recommending behaviors, which may occur from the parallel and independent evaluation of these three recommendation modules or from different rules in the same recommendation module. The *Final Selection* module determines the activation of the behaviors, i.e., *Behavior Activation*, by processing these measures. Different methodologies can be used to implement this arbitration, but the underlying idea here is to activate behaviors that are more desirable than undesirable in order to minimize aversions and maximize desirable outcomes. The intentions of the agent are derived from behavior recommendations and activation.

The third level of the architecture is the *Motivation* level made of the *Motives* module. The *Motives* module is used to monitor the agent's goals and to coordinate the proper working of the other modules. Methods to manage motivations or even emotions, as in Breazeal (1998) and Maes (1991), can be implemented in this module. The term 'motive' refers to something that prompts an agent to act in a certain way. Because of the many aspects associated with motivations and emotions (Strongman 1987), 'motive' seems to be more appropriate for describing the purpose of this module in the architecture. Motives can help manage the reactive, rational, or egotistic influences of the recommendation modules in two different ways: locally in each one; or globally by affecting their importance in the *Final Selection* module. Motives can be influenced by the environment (from sensed conditions), the intentions of the agent, knowledge about the world (managed by the *Cognition* module) and by observing the effective use of the behaviors. This last influence is related to the link called *Behavior Exploitation*. This is different than *Behavior Activation* in the sense that an **active** behavior is allowed to participate to the control of the agent, and it is said to be **exploited** if it is actually used to control the agent (by reacting to sensed conditions according to the control rules of the behavior). An activated behavior is not exploited when it is not releasing commands that affect the actions of the agent.

3. VALIDATION OF THE ARCHITECTURAL METHODOLOGY USING *BUGWORLD*

The procedures used in the modules of the architecture are designed according to the agent's capabilities and purpose. Using mobile robots, we have used behavior-producing modules with the Subsumption mechanism (Brooks 1986) and the *Cognition* module to learn an *Interaction model* for dynamic and nonstationary environments (Michaud *et al.* 1998), or to interpret of a language based on visual signs and motives (Michaud *et al.*, 1999). The experiments reported in this paper aim at validating the entire architecture for an autonomous agent that has to deal with managing its energy, its well being, finding targets and acquiring knowledge about its environment. To do so, we used *BugWorld* (Almàssy 1993), a simulated world for mobile robots. An agent in *BugWorld* has a circular body equipped with distance sensors (similar to range finders) for detecting obstacles and targets. For our experiments, the agent is placed in a room where it can find targets and a charging station. The agent must be able to efficiently reach the targets and recharge its energy when needed. The agent does not have any *a priori* knowledge about its environment, and has a limited memory to acquire information that can be helpful in its task. For sensing, the agent has at its disposal eight analog straight-line proximity sensors for obstacle detection, each separated by 45° starting from its nose. It also has two target sensors, one on each side. One target in the room is used as a charging station. The agent can also sense its energy level, its speed and its rotation[1]. For actions, the behaviors can affect the speed, the rotation and the color of the agent.

To achieve its goals, the agent operates according to the following scenario. First, the agent starts to acquire topological knowledge about the environment by following boundaries, reaching a target or a charging station deliberately or not. When the agent is able to recognize its location in the environment, it can start exploring other regions. Eventually, when the agent judges that it knows enough about the environment, it can use this knowledge for reaching memorized targets. The following subsections describe the mechanisms designed to implement this scenario using our architecture. Fuzzy logic is used at the *Behavior* level. Fuzzy logic is also used at the *Recommendation* level for recommending behaviors in the *External Situation* module and the *Needs* module, and for combining these recommendations in the *Final Selection* module. To construct an internal representation of the environment, the *Cognition* module uses a topological graph algorithm. The *Motives* module uses internal variables and a propagation mechanism to manage the goals of the agent. Note that influences between the *Motives* module and the *Final Selection* module or the *External Situation* module are not required in the experiments reported in this paper.

3.1. Fuzzy *Behavior* Module

A fuzzy behavior uses rules and linguistic variables to establish the relation between sensations and actions. The processing steps are similar to the ones for fuzzy systems (Lee 1990), which are fuzzification to convert input data into linguistic values, rule inference, and defuzzification. The only difference here is that rule firing strength is affected by μ_{act}, the activation of the

[1] The rotation is used only to indicate if the agent is moving or not.

behavior given by the *Final Selection* module. The processing steps for the fuzzy behaviors are summarized below:

- **Fuzzification (1).** This operation converts input data (i.e., sensations for the simulated robot) into linguistic values (A_i) characterized by a label and a membership value. Figure 2 gives an example of membership functions used for fuzzification of the front sensor of the agent (i.e. its nose). In this case, the fuzzy representation of a sensory input of 15 is given by two linguistic values: a membership of 0.67 for the *Danger-in-front* fuzzy set, and a membership of 0.33 for the *Near-front* fuzzy set.

$$Sensations \rightarrow \mu_{A_i}(Sensation) \tag{1}$$

Figure 2. Example of membership functions

- **Fuzzy implication of rule *r* for behavior *j* (2).** The operator ⊗ is the minimum and is used for the fuzzy conjunction of the *n* antecedents of rule *r*. The result is called the firing strength of the rule, which is a measure of the contribution of the rule to the fuzzy control action. The firing strength of the rule is associated with its fuzzy consequence, which is a linguistic variable (*B*) for a fuzzy control action. This processing step is repeated for all the rules of a behavior, and for all the activated behaviors. When NOT is used in front of an antecedent, the complement of its membership value (i.e., 1 - μ_A) is used. Examples of rules with their membership functions for the EMERGENCY behavior and for the TURN180 behavior are presented in Figure 3 and in Figure 4. For the rule *Slow-down-danger* of Figure 3, if *Danger-in-front* has a membership value of 0.67 and *Speed-null* has a membership value of 0 (which indicates that the agent is moving), then the membership value of *Slow-down-fast* is 0.67.

$$\mu_{B_{rj}}(Action) = \otimes\left[\mu_{A_n}(Sensation)\right] \tag{2}$$

- **Adjustment of the firing strength of the rules using $\mu_{act}(j)$, the activation of behavior *j* (3).** The linguistic variable *C* has the same label as *B* but a different membership value. The minimum is also the fuzzy conjunction operator used here. For example, if the activation of the EMERGENCY behavior is 0.5 and the firing strength of rule *Slow-down-danger* is 0.67, then the adjusted membership value of the fuzzy consequence *Slow-down-fast* for this rule is 0.5.

$$\mu_{C_{rj}}(Action) = \otimes\left[\mu_{B_{rj}}(Action), \mu_{act}(j)\right] \tag{3}$$

- **Union of the fuzzy consequences (4).** The fuzzy disjunction operator ⊕ maximum is used to combine membership values of the same fuzzy consequence, for all the linguistic values (C_o) associated with the actions. For instance, the rule *Slow-down-danger* of the EMERGENCY behavior and the rule *Immobilization* of the TURN180 behavior share the same fuzzy consequence, *Slow-down-fast*. If the adjusted membership values for *Slow-down-fast* are 0.5 and 0.9 respectively, then their union gives a membership value of 0.9.

$$\mu_{C_o}(Action) = \oplus\left[\mu_{C_{rj}}(Action)\right] \tag{4}$$

- **Defuzzification using the center of area method (5).** This step converts the fuzzy consequences into a 'crisp' (numerically precise) output, for each action possible (in our case speed, rotation and color). The idea is to take the fuzzy control actions that have been derived using the rules and the fuzzification of the sensations, and determine the actual action value that should be used to control the simulated robot. The parameter w_{Cx} is the support value at which the membership function for C_x reaches the maximum value μ_{Cx}[2], and x represents the linguistic variables for a common control action. This step allows smooth blending of the fuzzy commands given by the activated and exploited behaviors. For example in Figure 4, the *Slow-down-fast* fuzzy consequence is a linguistic variable with $w_{Slow\text{-}down\text{-}fast} = -3.25$ if $\mu_{Slow\text{-}down\text{-}fast} = 0.5$. If fuzzy rules also generate the fuzzy consequence $\mu_{Accelerate} = 0.1$ with $w_{Accelerate} = 4$, then the resulting control action *Acceleration* (associated with speed control) is -2.04.

---

[2] The average is used when there is two support values with the same membership strength.

$$Action = \frac{\sum_x \mu_{C_x}(Action) \cdot w_{C_x}}{\sum_x \mu_{C_x}(Action)} \tag{5}$$

Overall, twelve behaviors were designed to control the agent. The first is EMERGENCY, responsible for handling obstacles that are too close in front of the agent. The rules are presented in Figure 3 while Figure 4 shows the membership functions. Using this behavior, the agent slows down if it is in front and very close to an obstacle; it turns away from an obstacle at its side (the variable *x* is for *left* or *right*, and *y* denotes the opposite direction); and it makes a wide turn left if the obstacle is right in front.

Figure 3. Rules for the EMERGENCY and TURN180 behaviors

Other behaviors are AVOID to move away from obstacles, SPEED to maintain a constant cruising velocity, ALIGN to follow boundaries, TARGET to search for a target, RECHARGE to search for a charging station and to energize the agent, BACKING to move back, SPIN to make the agent turn around on itself, TURN90 to move away from a boundary, TURN180 to make a U-turn, ALARM to express some internal state of the agent by changing its color to red, and a behavior for identification of topological states (see Section 3.5). The membership functions and rules for the TURN180 behavior are also presented in Figures 3 and 4. The behavior starts by slowing down the agent. It then makes it turn away from a boundary until the agent perceives the other side. The *Cognition* module via the *Internal Parameters* link modifies the underlined antecedents and consequences of these rules, depending on the side of the boundary.

FIGURE 4. Membership functions for the EMERGENCY and TURN180 behaviors

3.2. *Motives* Module

Motives are responsible for coordinating and supervising the goals of the agent according to the situations experienced and the design scenario. These goals are to find a charging station and recharge itself, to reach targets, to detect improper use of its behaviors, to explore its environment and acquire knowledge from it; and to use this knowledge whenever possible.

To do so, the agent uses ten motives. Each one is implemented as an internal variable ranging from 0 to 1 that gets incremented or decremented by influences coming from sensations, behavior activation or exploitation, recommendation modules or from other motives. These motives can be grouped into four categories:

- **Physiological motives**. Motives like HUNGRY and EAT are used to monitor the energy level of the agent and to control the use of the RECHARGE behavior via the *Needs* recommendation module. The energy level directly influences the motive HUNGRY, which influences the motive EAT. Using two distinct physiological motives for energy supervision, the agent is able to show opportunism by wanting to recharge when it reaches a charging station, even if it is not hungry. The HUNGRY motive is also designed to fully recharge the agent before allowing it to leave the charging station.
- **Good Operation motives**. These motives are particularly influenced by the *Behavior Exploitation* link to detect improper use of behaviors. Because behaviors are fuzzy, *Behavior Exploitation* μ_{exp} is a fuzzy measure defined in equation (6), approximating the contribution of the behavior j to the fuzzy control actions formulated before defuzzification. *Behavior Exploitation* combines the activation μ_{act} of a behavior with its reactivity to the environment (based on the activation of the rules of the behavior, as presented by the second term of the right side of the equation).

$$\mu_{exp}(j) = \mu_{act}(j) \otimes \left(\oplus \left[\mu_{B_{rj}}(Action) \right] \right) \tag{6}$$

Two motives are designed to ensure good operation of the agent. First, the motive DISTRESS is used to monitor the proper working of behaviors like EMERGENCY, AVOID and SPEED. These first two behaviors must normally be exploited very briefly to move the agent away from trouble areas. However, if their μ_{exp} remains approximately constant for a long period of time, this may be a sign of conflict between the behaviors used. Also, another sign of trouble is when these two behaviors are fully activated but not exploited. For the SPEED behavior, a full exploitation for a long period of time indicates that the agent is not able to reach its desired velocity. The SPEED behavior is composed of only two rules indicating when to increase or to decrease the speed of the agent. These rules use two fuzzy linguistic variables overlapping by 50% at the desired velocity, so a constant behavior exploitation of 0.5 indicates normal working condition. Second, the motive DECEPTION examines behaviors responsible for reaching a goal location. The motive increases if the agent is moving away from a target or a charging station, detected by a decrease in the exploitation of the TARGET or RECHARGE behaviors respectively. This motive influences the use of the TURN180 behavior via the *Cognition* module to change the direction of the agent.

- **Accomplishment motives**. Only one motive, called FULFILLMENT, is used to make the agent find targets at periodic intervals. This motive decreases at each cycle unless a target is reached (which causes an increase of 0.3) or that the SPIN behavior is used.

FIGURE 5. Cognition motives

- **Cognition motives**. These motives, illustrated in Figure 5, supervise the acquisition and the use of the knowledge of the *Cognition* module. A solid arrow indicates a positive influence from another motive, as opposed to a shaded one. Motives CONFIDENCE and CERTAINTY are responsible for monitoring conditions from the topological graph mechanism implemented in the *Cognition* module, while motives EXPLORE, EXPLOIT[3] and BORED are used to manage the “cognitive” recommendation of behaviors. The motive CONFIDENCE is associated with the agent's ability to locate itself in previously memorized topological sites.

[3] Not to be confused with *Behavior Exploitation.* The motive EXPLOIT is an internal variable that monitors specific conditions, while *Behavior Exploitation* is a measure derived from the use of activated behaviors.

The motive EXPLORE directly uses the level of CONFIDENCE to influence the cognitive recommendation of the TURN90 behavior, making the agent explore the center of the room. Moreover, the motive EXPLOIT increases gradually if the motive CONFIDENCE is greater than zero, indicating the increasing ability of the agent to know where it is in the environment. EXPLOIT is also increased if no more nodes in the topological graph are available. If EXPLOIT reaches the value 1, it inhibits the motive EXPLORE to stop the acquisition of knowledge and to use the topological graph constructed for going toward memorized targets. Behavior recommendations then occur according to the path planned using the topological graph and the information stored in nodes. The ability to plan a path toward a goal is reflected by the CERTAINTY motive. Having EXPLOIT at value 1 also makes the motive BORED increase at each cycle, except when an unvisited target location is reached, causing BORED to be decreased by 0.3. The purpose of BORED is to make the agent exploit its topological graph as long as possible. Eventually, because all targets have been visited or the agent is not able to plan efficient paths and to reproduce them, the motive BORED reaches the value 1 and reinitializes the EXPLOIT motive, and exploration of the environment is then resumed.

3.3. Fuzzy *External Situation* Module

The *External Situation* module recommends behaviors according to sensed conditions in the environment. As in equations (1) and (2), this module uses rules to formulate fuzzy recommendations indicating the use (*desirability*) or the inhibition (*undesirability*, identified by NOT) of behaviors. These recommendations take the form of fuzzy desirability measure μ_{des} and fuzzy undesirability measure μ_{und} for behavior j. Figure 6 shows the rules used by the *External Situation* module. Rule <*Danger*> recommends the use of the EMERGENCY behavior if the agent comes too close to an obstacle. Rule <*Obstacle*> recommends using AVOID and inhibiting TARGET in front of an obstacle, while SPEED and ALIGN are recommended by rule <*Normal*> otherwise. If the agent is moving, rule <*Topological states*> recommends the use of the behavior for identification of topological states. Finally, rule

<*Charging*> facilitates the positioning of the agent by inhibiting ALIGN as the agent comes near a charging station.

FIGURE 6. Rules for the *External Situation* module

3.4. Fuzzy *Needs* Module

This module also uses fuzzy logic, with the difference of using conditions based on motives. This means that motives are fuzzified into linguistic variables, and these variables are used by the fuzzy rules presented in Figure 7 to recommend behaviors. Rules <*Want-to-recharge*>, <*Charging-station-near-x*> and <*Charging-station-nearer-x*> use the linguistic variable *Want-recharge* based on the motive EAT to find a charging station by recommending the use of the RECHARGE behavior, and to improve the influence of this behavior as the agent comes near a charging station. Rule <*Difficulties*> recommends using BACKING and inhibiting other behaviors if DISTRESS is sufficiently activated. Rule <*Accomplishment*> recommends the use of TARGET if the motive FULFILLMENT is small. When FULFILLMENT is high enough, rule <*Happiness*> recommends the use of the SPIN behavior and the inhibition of SPEED and ALIGN to make the agent turn on itself without trying to move forward or to follow boundaries.

FIGURE 7. Rules for the *Needs* module

3.5. Topological *Cognition* Module

The algorithm implemented in the *Cognition* module tries to answer the following question: how the agent can build a representation of the environment only by using proximity sensors. We chose to design a topological graph algorithm to construct a representation of the world. The *Cognition* module memorizes various information and recommends the use or the inhibition of behaviors using this representation. This module is by far the most complex of our implementation and its components, explained in the following paragraphs, are depicted in Figure 8.

FIGURE 8. Components of the *Cognition* module

Construction of the topological graph. At its lowest level, the graph is constructed from topological states identified by the TOPOLOGICAL STATE IDENTIFICATION behavior. This procedure is similar to the work of Mataric (1992), but with a different identification behavior. If activated, this behavior examines sensations coming from the front, the back and the two sides of the agent. The presence or the absence of obstacles at a certain distance in these four directions is used to infer one of 16 possible topological states: *Right side*, *Left side*, *Dead-end right*, *Dead-end left*, *Dead-end at the back*, *Dead-end in front*, *Right corner turned*, *Left corner turned*, *Right corner to turn*, *Left corner to turn*, *Corridor*, *Nothing*, *Stuck*, *Against a wall*, *Obstacle in front* and *Perpendicular to a corridor*. The topological state identified by the behavior is sent to the *Cognition* module using the *Internal Parameters* link of Figure 1. Figure 9 gives an example of topological states identified as the agent turns a corner and passes near an obstacle while following boundaries.

FIGURE 9. Example of identification of topological states

Our topological graph consists of an array of two types of nodes. The type is determined according to the characteristics of a sequence of topological states. A stable landmark node is used if the same topological state like *Right side*, *Left side*, *Corridor* or *Nothing*, is perceived. Otherwise, a transition node is created by analyzing the topological states identified between two stable landmark nodes, mainly to characterize and approximate the rotation made by the agent. Unlike Mataric, our topological representation does not use position estimate or a compass. Instead, our algorithm uses twenty-six regular expressions, with symmetrical conditions for the left and right sides of the agent, for a lexical analysis (Aho *et al.* 1988) of the topological states identified during this sequence. Figure 10 presents examples of right side regular expressions used for each of the topological type possible for a transition node. On the right side of the expressions, the operators indicate the number of the same consecutive topological state that must be in the sequence: '+' indicates more than one; '-' indicates only one; '?' is for zero or one. These expressions are evaluated in parallel after the consecutive identification of the same topological state. If a topological state does not correspond with the active state in a regular expression, then the regular expression is dismissed. Priority is assigned according to the order of definition of the regular topological expressions[4]. If an expression is completely validated, the result at the left side of the expression is memorized. If there is no result obtained for the sequence of topological states identified during a transition, the best guess according to the expressions evaluated is used. These regular expressions can be seen as symbolic behaviors reacting to topological state sequences to characterize the transition made by the agent.

FIGURE 10. Example of regular topological expressions

As the agent moves in the environment, nodes are created by memorizing the landmark type[5], its length (i.e., the number of topological states for the node), the orientation of the agent[6] and the identification number of the branch in construction in the graph. Other information, like the number of visits to the node, the presence of a goal like a target or a charging station, the occurrence of a motive like DISTRESS or DECEPTION, the use of particular behaviors like TARGET and SPIN, cognitive recommendations for TURN90 and TURN180, and path planning variables are other fields in a node. Note that the information coming from motives and behaviors are memorized to characterize the situation in which the nodes were created, and may be very useful in planning paths that avoid situations difficult to reproduce.

Nodes are connected together with bi-directional links. These links store information about the anticipation of the state (direct or opposite) of the connected node. This indicates the

---

[4]The first ones have priority on the others.

[5]The topological state for a stable landmark, or the result obtained from a transition landmark analysis.

[6]For stable landmark only, cumulated from previous nodes according to the rotation approximated in transition nodes.

direction in which the agent is progressing in the graph, information especially important in a junction created when the graph is followed in reverse direction. An uncertainty measure about the length of the node is also memorized with a link. The uncertainty is evaluated based on the topological states observed during the junction of a stable landmark node and a transition node. For a link between a stable landmark node and a transition node, the number of identical topological states at the beginning of the transition is used as the uncertainty measure. For a link between a transition and stable nodes, the number of identical states at the end of the transition is used. In other cases, the default value used is 1. Figure 11 presents the graph constructed based on the topological states identified in Figure 9. The uncertainty measures are shown with the links.

FIGURE 11. Topological graph for Figure 9

Positioning. Because position estimates are not used in the graph, a node may not be distinct from other nodes. To find out if the agent is located on a previously visited landmark, a search in the nodes must be initiated. During this search, the algorithm tries to find a sequence of three nodes similar to the three most recent nodes constructed. Similarities are evaluated according to the landmark type[7], the length of the nodes, and sometimes the orientation of stable landmarks[8]. Orientation is not always used as a similarity criterion because of approximation errors using topological regular expressions, and because of the necessity of resynchronization if the agent starts from an unknown location in the graph. Furthermore, orientation is used to indicate the possibility of searching for similar nodes belonging to the same branch if the trajectory of the agent covers 360° for the branch (to avoid errors in case of a symmetrical sequence of

[7] According to the anticipation of the state derived from the link followed to establish the sequence.

[8] An error tolerance is used to take into consideration approximation errors made during the lexical analysis with the regular topological expressions. The range of this tolerance varies between 45° to 75° according to the number of transition nodes in the branch.

landmarks). When only one similar sequence is detected, the recent nodes can be eliminated and a loop is established in the topological graph. The agent is then situated in its memorized topological representation and can compare its state according to the following nodes in the graph, updating the number of visits in reused nodes. The motive CONFIDENCE increases by 0.3 each time a similar node is eliminated and by 0.025 if the current topological state corresponds to the anticipated node type in the graph. If a divergence is observed, a new branch is initiated and the search process is reactivated. In addition, if no more nodes are available to construct new paths or when the agent wants to exploit the topological graph, three buffer nodes are used to locate the agent in its topological graph.

Planning. Similarly to Mataric (1992), path planning with the topological graph is done by activation spreading from the current location toward a node referring to a particular goal, as illustrated in Figure 12. Activation is spread following the links to the nodes, preferring the paths in the same direction of the agent (identified with positive activation) with the fewest nodes, the smallest length, and avoiding special conditions that occurred during the creation of the node like DISTRESS, DECEPTION, SPIN and U-turns[9]. The motive CERTAINTY is fully activated if a path is accepted, and is affected by the ability of the agent to follow the planned path. Planning is done when the agent wants to reach a charging station or a target[10] and that no planned path have been derived, or that the one found is not correctly followed.

FIGURE 12. Path planning by activation spreading

Optimization. Optimization is an essential process for the topological graph algorithm because of possible duplications of nodes for the same landmark and because of the limited

---

[9] It would not do the agent any good to follow a path where it has to come back from where it came from.

[10] For targets, the EXPLORE motive must be inhibited.

number of nodes[11]. Optimization must be done without interfering with the agent's decision processes and when sufficient knowledge has been acquired about the environment. Optimizing when the agent is recharging and with motive EXPLOIT sufficiently activated is then an appropriate choice. Only the useful nodes are then kept in the graph. A node is considered to be useful if it has been visited more than once, if it is at the start of an exploration branch[12] or if it is part of the optimal path from a node referring to a charging station toward a node referring to a target. After optimization, the number of visits for each node is reset to one, to eventually forget a node not visited since the last optimization.

Cognitive influences. These influences take the forms of modifying conditions in behaviors like TURN180 and TURN90 (as explained in Section 3.1), influencing the cognition motives, and recommending behaviors. Using binary[13] conditions, the cognitive recommendations are:

- if DECEPTION has reached the value 1 or that the planned path requires the agent to change direction, TURN180 is recommended while SPEED and ALIGN are inhibited so that they do not interfere;
- if EXPLORE has reached the value 1 and that the agent is at the middle of a node like *Right-side* or *Left-side*, TURN90 is recommended while SPEED and ALIGN are inhibited;
- when a planned path is followed, cognitive recommendations memorized in the nodes for TURN90 and TURN180 are repeated to reproduce the path;
- based on a planned path, if the agent evaluates that it will not have enough energy to reach a charging station, the ALARM behavior is recommended.

---

[11] The limit was set to 150 for the experiments reported in this paper.

[12] To remember where exploration has been done.

[13] Binary conditions are special cases of fuzzy membership strengths of 0 or 1. Fuzzy logic was not necessary for the cognitive recommendations.

3.6. *Final Selection* Module

With the *Final Selection* module, the *Behavior Activation* is evaluated based on a hedonic continuum established from the fuzzy desirability μ_{des} and undesirability μ_{und} measures. First, these measures from the recommendation modules are respectively and equally combined for each behavior using the fuzzy disjunction operator maximum. Then, the *undesirability* is subtracted from the *desirability* measure and the behavior is activated if the result is greater than zero. So to be activated, the *desirability* of the behavior must be higher than its *undesirability*. Equation (7) shows these operations where *m* represents the recommendation modules.

$$\mu_{act}(j) = \max\left(0, \oplus\left[\mu_{des_m}(j)\right] - \oplus\left[\mu_{und_m}(j)\right]\right) \tag{7}$$

4. EXPERIMENTAL RESULTS

The results presented here are parts of longer trajectories followed by the agent, but they clearly show the use of the mechanisms described in Section 3 and illustrate the general behavior of the agent. Also note the following conditions used for the experiments: the energy of the agent decreases at each cycle; when it reaches a charging station, the agent detects that it is recharging by sensing an increase of its energy level; and a target reached is inhibited for 200 cycles.

Using the same room configuration that comes with *BugWorld*, Figure 13 illustrates the initial trajectory followed by the agent starting from the upper left corner, along with the activation level of some of its motives. The agent starts by reaching the target in the upper left corner and continues its path by following boundaries. It stops at the lower left corner where the charging station is located. It then continues to follow boundaries, reaching the lower right target and the upper right target successively. As indicated in Section 3.2, the FULFILLMENT motive is increased by 0.3 when a target is reached. After that, the agent reaches again the upper left target and the charging station. The EAT motive is fully activated when the agent is recharging, confirming that the behavior RECHARGE is used. At this point, the agent is able to detect similar sequences in its topological graph and to create a loop with the nodes. The agent has a higher level of CONFIDENCE and wants to EXPLORE. It starts exploring the center of the room just after leaving the charging station by using the TURN90 behavior in the middle of a *Right side* node.

FIGURE 13. Trace and motives when the agent starts exploring its environment

To illustrate the influences of the recommendation modules, recommendations for the TARGET behavior are presented in Figure 14. The *External Situation* module inhibits the use of TARGET if an obstacle is detected in front of the agent (see rule <*Obstacle*> of Figure 6). The *Needs* module is in favor of using TARGET if the motive FULFILLMENT is small, but not if the agent is recharging (see rules <*Want-to-recharge*> and <*Accomplishment*> of Figure 7). The activation of TARGET is the result of equation (7).

FIGURE 14. Trace and recommendations of the TARGET behavior

As indicated in Section 2, an activated behavior may not be exploited. Figure 15 shows the exploitation of the behaviors activated for the trace presented in Figure 13. The activation of the EMERGENCY and AVOID behaviors are very similar to their exploitations because their activating conditions are used in the rules of the behaviors. The SPEED and ALIGN behaviors are activated by the rule <*Normal*> of Figure 6, but their exploitations differ because the conditions from which they issue commands are different. Finally, the RECHARGE behavior is activated and exploited right away when the agent passes through the charging station, showing opportunism for recharging itself.

FIGURE 15. Behavior activation and exploitation

Finally, Figure 16 presents the topological graph constructed for the same trace. Each node is plotted at the middle of the region it characterizes, and is identified by its index number in the topological graph. The search process starts at node 4, and a similar sequence is accepted at node 6[14]. Nodes are eliminated to create the link between nodes 21 and 2, and exploration starts at node 9 by the new branch composed of nodes 30 to 32. Information about the nodes constructed is presented in Table 1.

FIGURE 16. Topological graph

TABLE 1: Node information

The agent continues to explore its environment this way, only using the TURN90 behavior at landmarks where it has never been used. Figure 17 shows the following explorations carried out. Results show that the agent does not start exploring in previously explored regions because of the knowledge memorized in its graph.

FIGURE 17. Next explorations made by the agent

After a while, the agent feels confident for long periods of time because it is able to locate itself with its topological graph and it cannot find new sites to explore. As illustrated in Figure 18, the motive EXPLOIT increases until it reaches a preset value of 0.9. It then inhibits the EXPLORE motive, and the buffer nodes are used to locate the agent in its graph. The EXPLOIT motive is set to 1 when the agent arrives at a charging station to optimize its topological graph. Then, the agent tries to use its graph to plan a path toward a target. In this example, the agent is able to position itself in the graph up to a certain point (as we can see by the zero CONFIDENCE level before and after the 2600 step). The agent is also unable to plan paths or to reach targets during that time, and the motive BORED increases until it resets the EXPLOIT motive. This can

[14]After rejecting three consecutive sequences because of the orientation. This is one case where the orientation is not considered to evaluate similarities.

be interpreted as if the agent decides that it cannot efficiently use its knowledge anymore, and needs to explore further its environment.

FIGURE 18. Motives when the topological graph is exploited

Figure 19 shows the topological graph before and after optimization. Because the similarities between nodes are evaluated based on a sequence of similar nodes and not by factors uniquely identifying each possible landmark in the environment, the graph can be composed of parallel branches or multiple nodes for the same site. After optimization, the graph is reduced by approximately 50%. The paths following the boundaries of the environment are kept without being explicitly specified in the optimization procedure. This way, the useful paths emerge from the experiences of the agent and its abilities to use the topological representation.

FIGURE 19. Topological graph before and after optimization

In Figure 20, the agent starts from another location in the environment. Because of a conflict between EMERGENCY, AVOID and ALIGN behaviors, the agent gets stuck in the lower right corner. The simultaneous constant exploitations of EMERGENCY and of AVOID excite the motive DISTRESS from which the BACKING behavior is recommended by the *Needs* module. Observing the exploitation of these behaviors revealed useful here to avoid conflicting commands to control the agent. After having backed away from the corner, the agent starts moving toward the charging station but observes a decrease in the exploitation of the TARGET behavior. This indicates that the agent is moving away from a target, which in this case is the upper right target. The motive DECEPTION is then increased and the agent makes a U-turn by using the TURN180 behavior. The agent continues its path by following boundaries until it reaches the charging station. This demonstrates how *Behavior* Exploitation can serve as a useful indication of the progress of the agent in satisfying its goals. A SOS is emitted before arriving at this point because the agent has only three cycles of energy left.

FIGURE 20. Trace and conditions for the motives DISTRESS and DECEPTION

Another room configuration, bigger than the first, has a target at the center that cannot be detected from the boundaries, and discontinuities in its upper left side. This environment allows to evaluate the possibility of reproducing a memorized topological path and the efficiency of the topological graph algorithm when a discontinuous region is present in the world. First, Figure 21 shows two parts of a trace made by the agent. The first trace is made when the agent wants to explore its environment. The agent is then able to reach the target at the center of the room. Later, the agent is able to plan a path toward this target and to reproduce the path using the buffer nodes and its optimized topological graph presented in this same figure. As it can also be seen from this graph, the agent is able to locate itself in its topological graph, even in the presence of disturbances in the upper left side of the room.

FIGURE 21. Path reproduction

Figure 22 shows a special condition that occurred when a moving obstacle was placed in the same room with the agent. At one point, the obstacle is moving toward the agent. The agent tries to move away from the obstacle but could not do so because its back side collides with the moving obstacle. The agent does not have any behaviors to avoid obstacles from its back, but by observing that the SPEED behavior is fully exploited for a long period of time[15], the motive DISTRESS is excited so that the BACKING behavior can be used. *Behavior Exploitation* can then be useful to compensate for limited perception of the agent.

FIGURE 22. DISTRESS motive when a mobile obstacle came toward the agent

[15]Because the agent wants to move forward but simply cannot get some speed.

5. DISCUSSION

These results show the ability of the agent to efficiently interact with its environment by being able to adapt to the situations experienced, the knowledge acquired and the goals managed. However, a lot of expertise must be transmitted from the designer to the agent by programming the different modules. The behavior repertoire of the agent, the recommendation conditions and the motives used have profound effects on the way the agent behaves. But these choices are not that hard to do if a good design methodology is followed. First, EMERGENCE, AVOID, SPEED and ALIGN behaviors, activated by the *External Situation* module, were designed to make the agent move in the environment by following boundaries. Then, the behavior TARGET with the motive FULFILLMENT, along with the rule in the *Needs* module, were added to make the agent find targets. Other design strategies for distress, recharging, exploring, exploiting the topological graph, making a U-turn, were also used. Starting from simple behavior selection strategies to more complex ones simplified the overall interactions between the modules of our architecture and the identification of possible conflicts. These conflicts can be managed during the design of behaviors (by membership function positioning and the choice of rules), by using the *undesirability* measure (allowing a possible conflicting behavior to be inhibited by any recommendation module), and by the influences on motives. Motives, especially from *Behavior Exploitation* influences, are important in that regard since they serve as a memorization mechanism of the history of past events relevant in affecting behavior selection.

This design strategy also helps preserve the emergence of functionality at all three levels of the architecture. Emergent computing occurs at the *Behavior* level by having behaviors that issue actions simultaneously on the same actuator but based on different sensed conditions. The interactions between the behaviors and the environment generate an emergent functionality that is not a property of either the agent or the environment in isolation but rather a result of the interplay between them (Arkin 1998). Similarly, emergent computing occurs at the *Recommendation* level by having modules issuing recommendations simultaneously on the same behavior but based on different conditions, and also by the mechanism used in these modules to formulate these recommendations and to activate behaviors. The topological graph also emerges from the interactions of the agent with its environment, based on the mechanisms[16] contained in the *Cognition* module. Finally, observing the exploitation of behaviors over time revealed to be a very important source of information on the emerging functionality that comes from the behavior-producing modules and the recommendation modules. This can be explained by the fact that *Behavior Exploitation* combines both a representation of the environment (i.e., the sensory information used by a behavior) and of the control policy (since the exploitation of a behavior depends on its activation, the behavior arbitration mechanism and the conditions associated with its activation). *Behavior Exploitation* provides important feedback about the interactions the agent is having in its operating environment (Michaud *et al.* 1996; Michaud 1997; Michaud *et al.* 1998), giving it a sense of awareness. These interactions emerge from the computation done in the three abstraction levels of the architecture, and involve the actual situation (short-term), the intentions of the agent (medium-term) and the accomplishment of its goals (long-term).

6. RELATED WORK

Over the years, a lot of theories and architectures (Arkin 1998) have been proposed to describe some of the organizational principles associated with intelligence. But still, there are many things unknown about these principles, principally how to combine them into a unified framework (Albus 1991; Corfield *et al.* 1991). Motivated by this objective, our research aimed at studying the principles, methodologies and concepts related to intelligence from the fields of artificial intelligence, psychology and ethology. Trying to synthesize and to combine their

---

[16] From the lexical analysis of the topological states, the search for positioning in the graph, path planning by activation spreading and removing nodes unfrequently visited during optimization.

strengths resulted in the architectural methodology presented in this paper. Too many approaches and works influenced ours to be listed in this paper, and a summary of the principal influences is presented in Michaud (1996). Its principal novelty is that it integrates concepts from reactivity to reasoning and to emotion (Michaud 1996; Michaud *et al.* 1996). The work of Albus (1991) is the one closest to ours because it also considers these concepts, but differs in its dependence on a symbolic and central world model and its hierarchical decomposition with different planning scopes (in time and in space). Instead of focusing on knowledge representation, we tried to preserve the principle of emergence in all three levels of our architecture to still be able to make decisions in situations that cannot be modeled.

The mechanisms implemented to validate the architectural methodology with *BugWorld* are inspired by other approaches. The use of fuzzy logic for the behavior-producing modules is inspired from the work of Saffiotti *et al.* (1993, 1995). The *Local Perceptual Space* module that they use is not reproduced in our experiments but could be integrated in the *Cognition* module of our architecture. Also note that their use of the term *desirability* corresponds to our variable μ_C, and differs significantly from the *desirability* measures used for behavior recommendation. Tunstel *et al.* (1996) also use fuzzy behaviors and coordination mechanism. But the approach is based on behavior hierarchy with no influences from motives or acquired knowledge about the world. Goodridge *et al.* (1994) use fuzzy behaviors along with a motivational state but do not consider the other aspects that can influence behavior coordination. Our work is able to combine the properties of all of these approaches.

As indicated in Section 3.5, the topological graph algorithm shares some similarities but also has clear distinctions with the work of Mataric (1992). Finally, the work of Maes (1991) inspired us for the *Motives* module, but instead of using the propagation mechanism to affect behavior arbitration, it is used to affect internal variables that manage the goals of the agent. Using mechanisms from three different approaches also show that the architecture methodology allows to combine very distinct concepts into a general framework, leading to the design of agents more intelligent.

7. CONCLUSION

This paper describes an architectural methodology that is based on behaviors, and the idea is to dynamically configure them according to the intentions of the agent. Taking the form of behavior recommendations, these intentions are influenced by the situation perceived, the knowledge acquired or innate about the world, and internal variables called *Motives* that monitor the internal states of the agent. The results presented show that the architecture and the mechanisms implemented gave the agent the ability to interact intelligently with its environment, adapting to the situations experienced, the topological knowledge acquired about the world and the goals it needed to achieve. They also demonstrate how emergence can be incorporated into processing levels more abstract than behaviors, and how planning and deliberation can be used to influence how the agent reacts and behaves in its environment. Reasoning about the purpose of behaviors using *Behavior Exploitation* revealed to be a powerful mechanism to monitor effectively and to use appropriately the emerging phenomena. By following the organizational principles of this architectural framework, we hope to gain a better understanding on various aspects related to intelligence like reactivity, planning, modeling, learning, communication, group and social behavior, motivation, emotion and so on, to design increasingly intelligent artificial systems.

ACKNOWLEDGMENTS

Support from the Natural Sciences and Engineering Research Council of Canada (NSERC) was greatly appreciated. We would also like to thank Nikolaus Almàssy for letting us use *BugWorld*.

REFERENCES

ARKIN, R.C. 1998. Behavior-Based Robotics. The MIT Press.

AHO, A.V., SETHI, R., and ULLMAN, J.D. 1988. Compilers, Principles, Techniques, and Tools. Addison Wesley.

ALBUS, J.S. 1991. Outline for a theory of intelligence. IEEE Trans. on Systems, Man, and Cybernetics, **21**(3):473-509.

ALMÀSSY, N. 1993. BugWorld: A distributed environment for the development of control architectures in multi-agent worlds. Tech. Report 93.32 (ftp anonymous, almassy@ifi.unizh.ch), Department of Computer Science, University Zurich-Irchel.

BATES, J. 1994. The role of emotion in believable agents. Communications of the ACM, **37**(7):122-125.

BECK, R.C. 1983. Motivation. Theories and Principles. Prentice Hall, Second Edition.

BREAZEAL, C. 1998. A motivational system for regulating human-robot interaction. *In* Proc. AAAI.

BROOKS, R.A. 1986. A robust layered control system for a mobile robot. IEEE Journal of Robotics and Automation, **RA-2**(1):14-23.

BROOKS, R.A. 1991. Intelligence without reason. A.I. Memo 1293, MIT Artificial Intelligence Laboratory.

CORFIELD, S.J., FRASER, R.J.C., and HARRIS, C.J. 1991. Architecture for real-time control of autonomous vehicles. Comput. Control. Eng. J., **2**(6):254-262.

GOODRIDGE, S.G., LUO, R.C. and KAY, M.G. 1994. Multi-layered fuzzy behavior fusion for real-time control of systems with many sensors. *In* Proc. IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems, pp. 272-279.

KORTENKAMP, D., BONASSO, R.P. and MURPHY, R. 1998. Artificial Intelligence and Mobile Robots. The MIT Press.

LEE, C.C. 1990. Fuzzy logic in control systems: fuzzy logic controller. IEEE Trans. on Systems, Man, and Cybernetics, **20**(2):404-435.

MAES, P. 1991. A bottom-up mechanism for behavior selection in an artificial creature. *In* From Animals to Animats. Proc. First Int'l Conf. on Simulation of Adaptive Behavior. The MIT Press, pp. 238-246.

MATARIC, M.J. 1992. Integration of representation into goal-driven behavior-based robots. IEEE Trans. on Robotics and Automation, **8**(3):304-312.

MCFARLAND, D. and BÖSSER, T. 1993. Intelligent Behavior in Animals and Robots. Bradford Book, The MIT Press.

MICHAUD, F. 1996. Nouvelle architecture unifiée de contrôle intelligent par sélection intentionnelle de comportements, Ph.D. Thesis (in French), Université de Sherbrooke, Department of Electrical and Computer Engineering.

MICHAUD, F. 1997. Adaptability by behavior selection and observation for mobile robots. *In* Soft Computing in Engineering Design and Manufacturing. Springer-Verlag, pp. 363-370.

MICHAUD, F., LACHIVER, G., and LE DINH, C.T. 1996. A new control architecture combining reactivity, deliberation and motivation for situated autonomous agent. *In* Proc. Int'l Conf. Simulation of Adaptive Behavior (SAB). The MIT Press, pp. 245-254.

MICHAUD, F. and MATARIC, M.J. 1998. Learning from history for behavior-based mobile robots in non-stationary environments. Machine Learning, **31**(1/2/3):141-167.

MICHAUD, F. and VU, M.T. 1999. Managing robot autonomy and interactivity using motives and visual communication. *Accepted for presentation at the* Autonomous Agents Conference.

ORTONY, A., CLORE, G.L., and COLLINS, A. 1988. The Cognitive Structure of Emotions. Cambridge University Press.

PARKER, L.E. 1992. Adaptive action selection for cooperative agent teams. *In* From Animals to Animats 2. Proc. Second Int'l Conf. on Simulation of Adaptive Behavior. The MIT Press, pp. 442-450.

PFEIFER, R. 1988. Artificial intelligence models of emotion. *In* Cognitive Perspectives on Emotion and Motivation. Kluwer Academic Publishers, pp. 287-320.

SAFFIOTTI, A., KONOLIGE, K., and RUSPINI, E. 1995. A multivalued logic approach to integrating planning and control. Artificial Intelligence, **76**:481-526.

SAFFIOTTI, A., RUSPINI, E., and KONOLIGE, K. 1993. A fuzzy controller for Flakey, an autonomous mobile robot. Technical Note 529, SRI International.

SIMON, H.A. 1967. Motivational and emotional controls of cognition. Psychological Review, **74**:29-39.

SLOMAN, A. and CROUCHER, M. 1981. Why robots will have emotions. *In* Proc. Seventh Int'l Conf. on Artificial Intelligence. pp. 197-202.

STRONGMAN, K.T. 1987. The Psychology of Emotion. John Wiley & Sons, Third Edition.

TUNSTEL, E., LIPPINCOTT, T. and JAMSHIDI, M. 1997. Behavior hierarchy for autonomous mobile robots: Fuzzy-behavior modulation and evolution. International Journal of Intelligent Automation and Soft Computing, **3**(1):37-50.

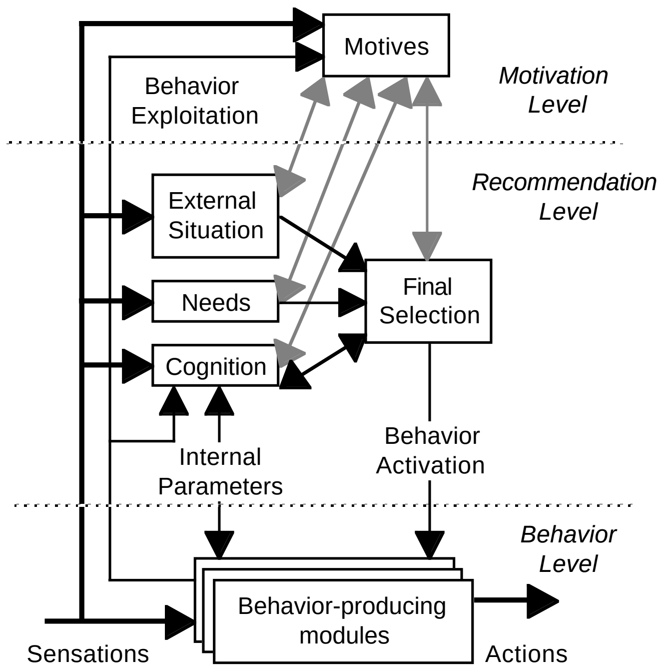
LIST OF SYMBOLS

A	Linguistic variable for a sensation
B	Linguistic variable for a consequence before being affected by behavior activation
C	Linguistic variable for a consequence after being affected by behavior activation
μ	Membership level of a linguistic variable
μ_{act}	Activation level of a behavior
μ_{des}	Desirability level of a behavior
μ_{exp}	Exploitation level of a behavior
μ_{und}	Undesirability level of a behavior
w_C	Support value at which the membership function C reaches the maximum value
$\otimes$	Fuzzy conjunction operator
$\oplus$	Fuzzy disjunction operator

LIST OF FIGURES

(Page # in the text – Page # of the figure)

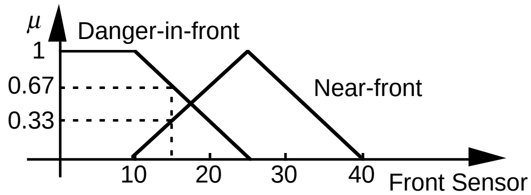
LIST OF TABLES



François Michaud, Gérard Lachiver, Chon Tam Le Dinh

Architectural methodology based on intentional configuration of behaviors

FIGURE 1. Architecture for intentional configuration of behaviors



François Michaud, Gérard Lachiver, Chon Tam Le Dinh

Architectural methodology based on intentional configuration of behaviors

Figure 2. Example of membership functions

EMERGENCY

<Slow-down-danger>
IF Danger-in-front
AND NOT (Speed-Null)
THEN Slow-down-fast

<Danger-*x*>
IF Danger-front-*x*
AND NOT (Danger-front-*y*)
THEN Turn-*y*

<Danger-in-front>
IF Speed-Null
AND Danger-in-front
AND Danger-front-right
AND Danger-front-left
THEN Turn-left-big

TURN180

<Immobilization>
IF NOT (Speed-almost-null)
THEN Slow-down-fast

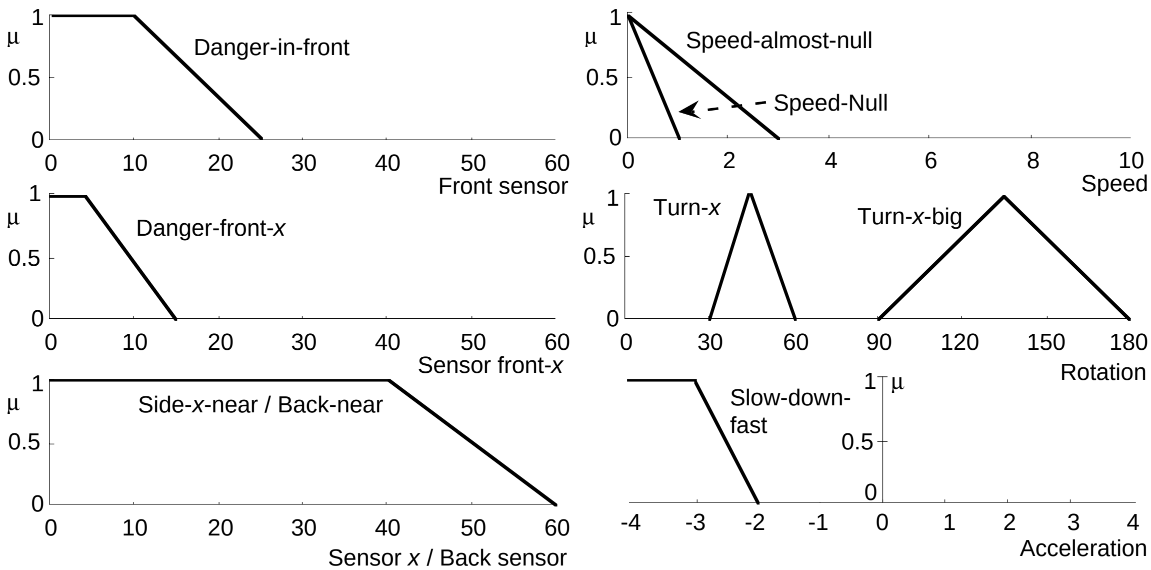
<Turn-180-left>
IF NOT (Side-left-near)
AND Speed-almost-null
THEN Turn-left

<Turn-180-left-end>
IF Side-left-near
AND Back-near
AND Speed-almost-null
THEN Turn-left

François Michaud, Gérard Lachiver, Chon Tam Le Dinh

Architectural methodology based on intentional configuration of behaviors

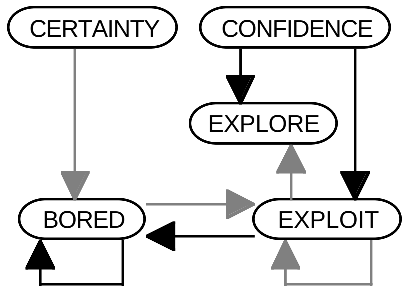
Figure 3. Rules for the EMERGENCY and TURN180 behaviors



François Michaud, Gérard Lachiver, Chon Tam Le Dinh

Architectural methodology based on intentional configuration of behaviors

FIGURE 4. Membership functions for the EMERGENCY and TURN180 behaviors



François Michaud, Gérard Lachiver, Chon Tam Le Dinh

Architectural methodology based on intentional configuration of behaviors

FIGURE 5. Cognition motives

<Danger>
IF Danger-in-front
OR Danger-front-right
OR Danger-front-left
THEN EMERGENCY
<Obstacle>
IF Obstacle-in-front
THEN AVOID AND NOT(TARGET)
<Normal>
IF NOT(Obstacle-in-front)
THEN SPEED AND ALIGN
<Topological states>
IF NOT(Speed-null)
AND NOT(Rotation-null)
THEN TOPOLOGICAL STATE IDENT.
<Charging>
IF Speed-almost-null
AND Charging-station-visible-left
AND Charging-station-visible-right
THEN NOT(ALIGN)

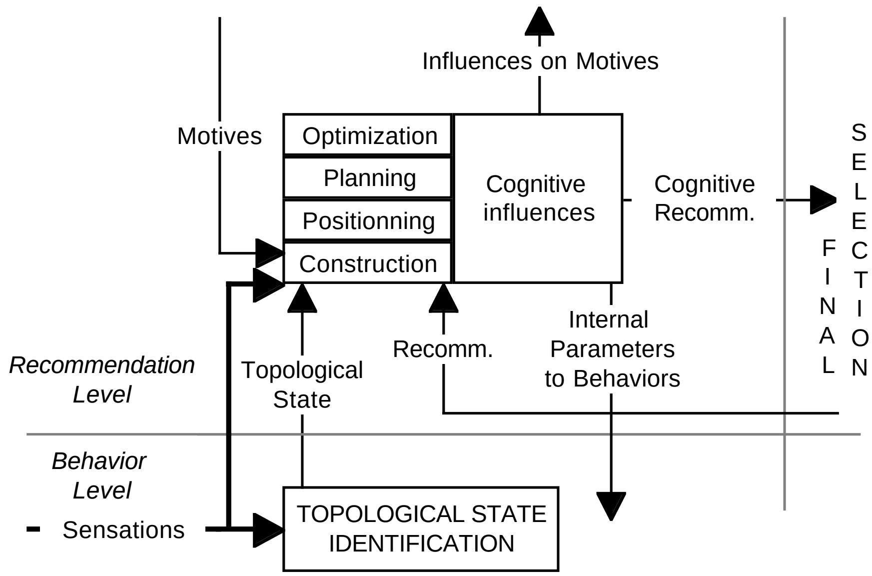
François Michaud, Gérard Lachiver, Chon Tam Le Dinh

Architectural methodology based on intentional configuration of behaviors

FIGURE 6. Rules for the *External Situation* module

```
<Want-to-recharge>
    IF      Want-Recharge
    THEN    RECHARGE, NOT(SPIN)
            NOT(TURN90) AND NOT(TARGET)
<Charging-station-near-x>
    IF      Want-Recharge
    AND     Charging-station-visible-x
    THEN    NOT(SPEED)
<Charging-station-nearer-x>
    IF      Want-Recharge
    AND     Charging-station-nearer-x
    THEN    NOT(ALIGN)
<Difficulties>
    IF      Distress-exists
    THEN    BACKING, ALARM  AND NOT(ALIGN)
<Accomplishment>
    IF      Fulfillment-small
    THEN    TARGET
<Happiness>
    IF      Fulfillment-big
    THEN    SPIN, NOT(SPEED) AND NOT(ALIGN)
```

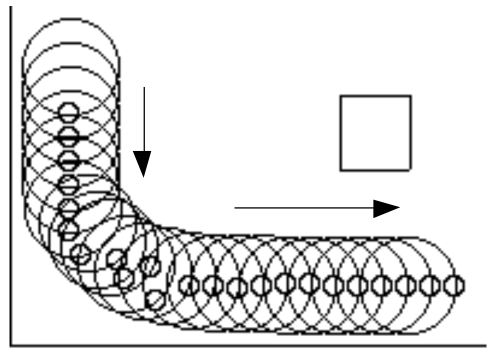

FIGURE 7. Rules for the *Needs* module



François Michaud, Gérard Lachiver, Chon Tam Le Dinh

Architectural methodology based on intentional configuration of behaviors

FIGURE 8. Components of the *Cognition* module



Right side - Right side - Right side - Right side - Right side -
Right corner to turn - Right corner to turn -
Dead-end right - Dead-end right - Right corner turned -
Right corner to turn - Right corner turned -Right corner turned -
Right side - Right side - Right side - Right side -
Right side - Right side - Right side -
Corridor - Corridor -Corridor

François Michaud, Gérard Lachiver, Chon Tam Le Dinh

Architectural methodology based on intentional configuration of behaviors

FIGURE 9. Example of identification of topological states

Internal *Right* Corner - 110° =
Right corner to turn+ Dead-end *right*+
Right corner turned- *Right* corner turned-
Right corner turned-
Internal *Right* Corner - 90° =
Right corner to turn+ Dead-end *right*+
Right corner turned- *Right* corner turned?

External *Right* Corner - 110° =
Nothing+ *Right* side+ Nothing+
External *Right* Corner - 90° =
Nothing+ *Right* side - Nothing -

Front to *Right* - 90° =
Obstacle in front+ *Right* side+

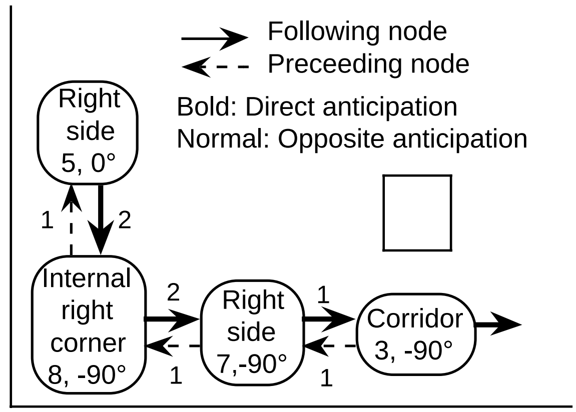
U Turn *Right* - 180° =
Right corner turned- Against a wall-
Left corner turned-

Turn *Right* - 90° =
Right corner turned- Against a wall+

François Michaud, Gérard Lachiver, Chon Tam Le Dinh

Architectural methodology based on intentional configuration of behaviors

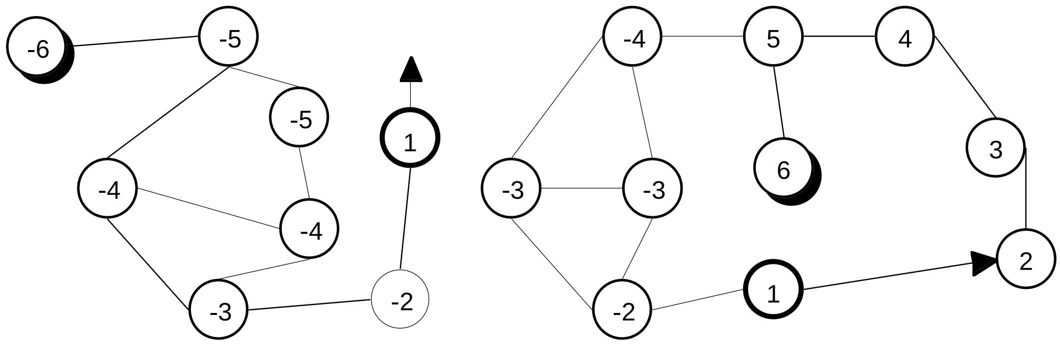
FIGURE 10. Example of regular topological expressions



François Michaud, Gérard Lachiver, Chon Tam Le Dinh

Architectural methodology based on intentional configuration of behaviors

FIGURE 11. Topological graph for Figure 9

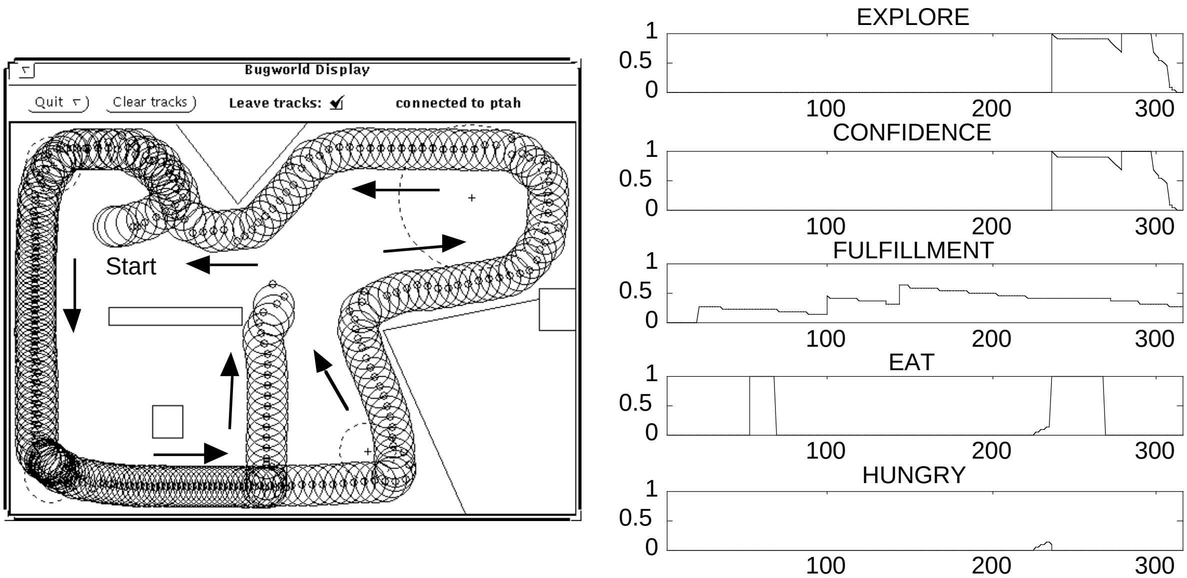


a) Branch in the Topological Graph b) Loop in the Topological Graph

François Michaud, Gérard Lachiver, Chon Tam Le Dinh

Architectural methodology based on intentional configuration of behaviors

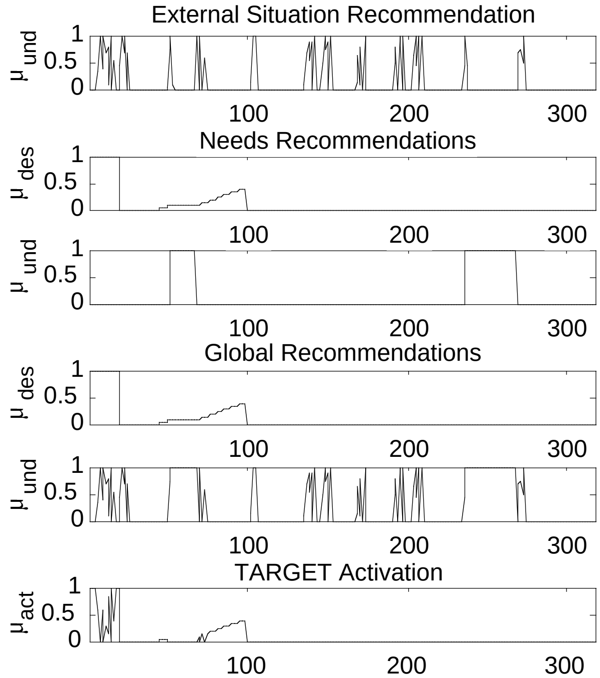
FIGURE 12. Path planning by activation spreading



François Michaud, Gérard Lachiver, Chon Tam Le Dinh

Architectural methodology based on intentional configuration of behaviors

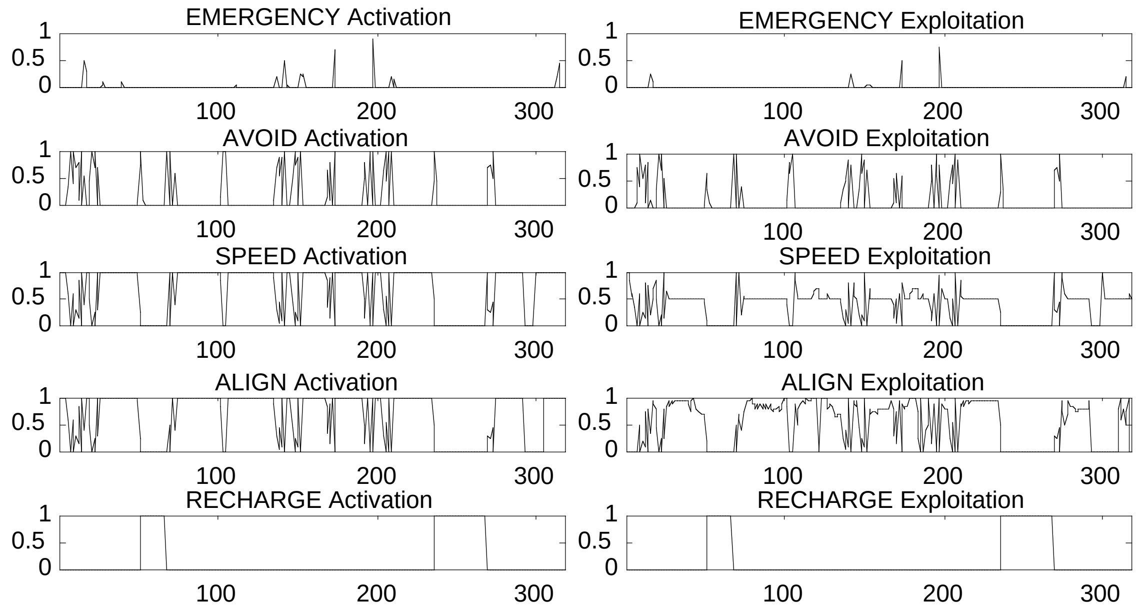
FIGURE 13. Trace and motives when the agent starts exploring its environment



François Michaud, Gérard Lachiver, Chon Tam Le Dinh

Architectural methodology based on intentional configuration of behaviors

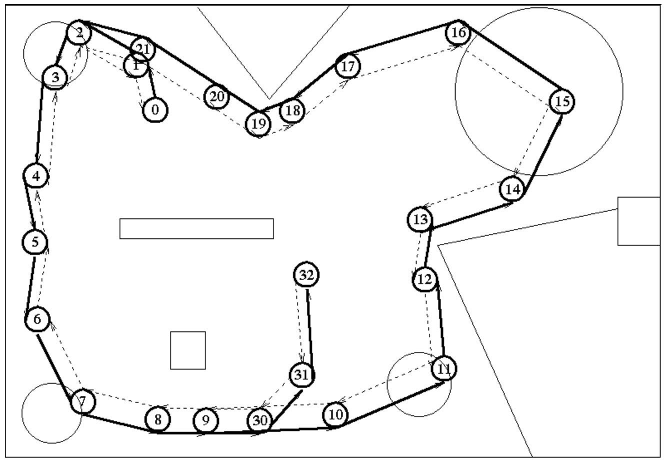
FIGURE 14. Trace and recommendations of the TARGET behavior



François Michaud, Gérard Lachiver, Chon Tam Le Dinh

Architectural methodology based on intentional configuration of behaviors

FIGURE 15. Behavior activation and exploitation



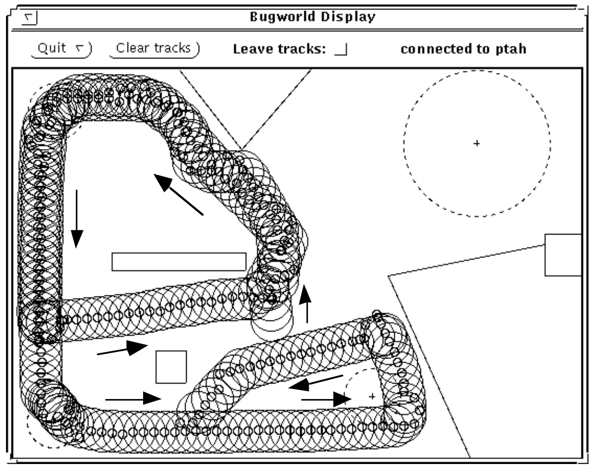
François Michaud, Gérard Lachiver, Chon Tam Le Dinh

Architectural methodology based on intentional configuration of behaviors

FIGURE 16. Topological graph

TABLE 1: Node information

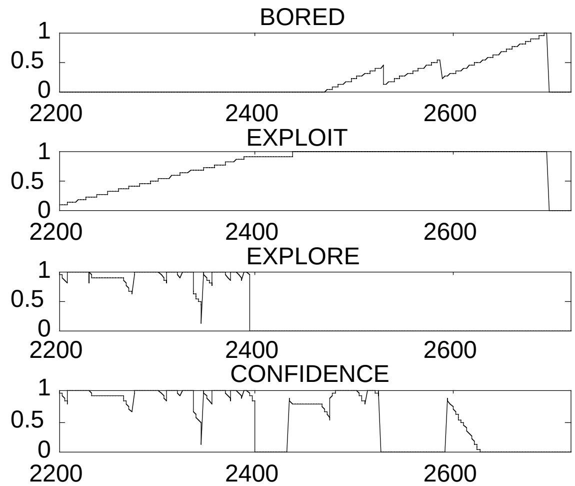
#	Landmark Type	Length	Orientation
0	Nothing	4	0
1	Front to right	1	-90
2	Right side	3	270
3	Internal right corner	7	-90
4	Right side	11	180
5	Corridor	2	180
6	Right side	11	180
7	Internal right corner	11	-80
8	Right side	7	100
9	Corridor	3	100
10	Right side	19	100
11	Internal right corner	6	-110
12	Right side	11	350
13	External right corner	1	90
14	Right side	16	80
15	Internal right corner	18	-90
16	Right side	14	350
17	Internal right corner	7	-50
18	Right side	6	300
19	External right corner	1	90
20	Right side	9	30
21	Internal right corner	8	-50
30	Right side	10	90
31	Turn right 90	7	-90
32	Nothing	11	0



François Michaud, Gérard Lachiver, Chon Tam Le Dinh

Architectural methodology based on intentional configuration of behaviors

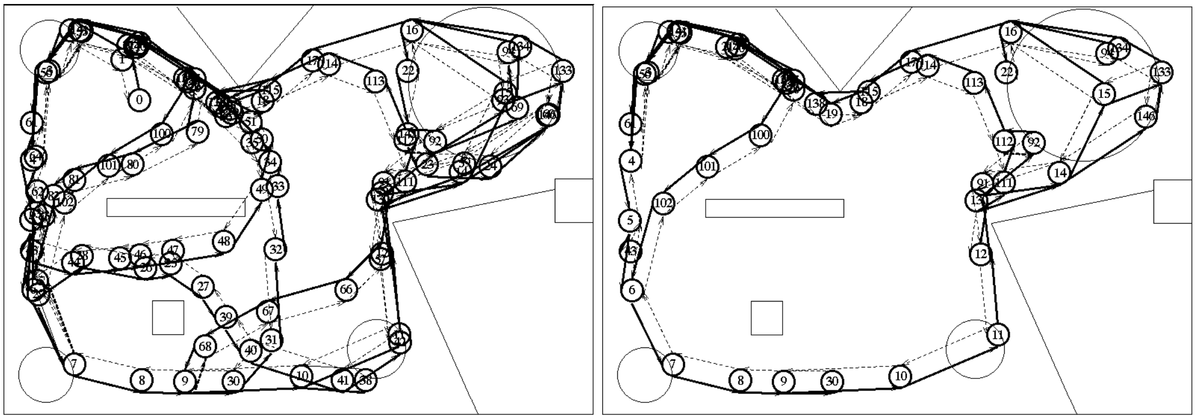
FIGURE 17. Next explorations made by the agent



François Michaud, Gérard Lachiver, Chon Tam Le Dinh

Architectural methodology based on intentional configuration of behaviors

FIGURE 18. Motives when the topological graph is exploited



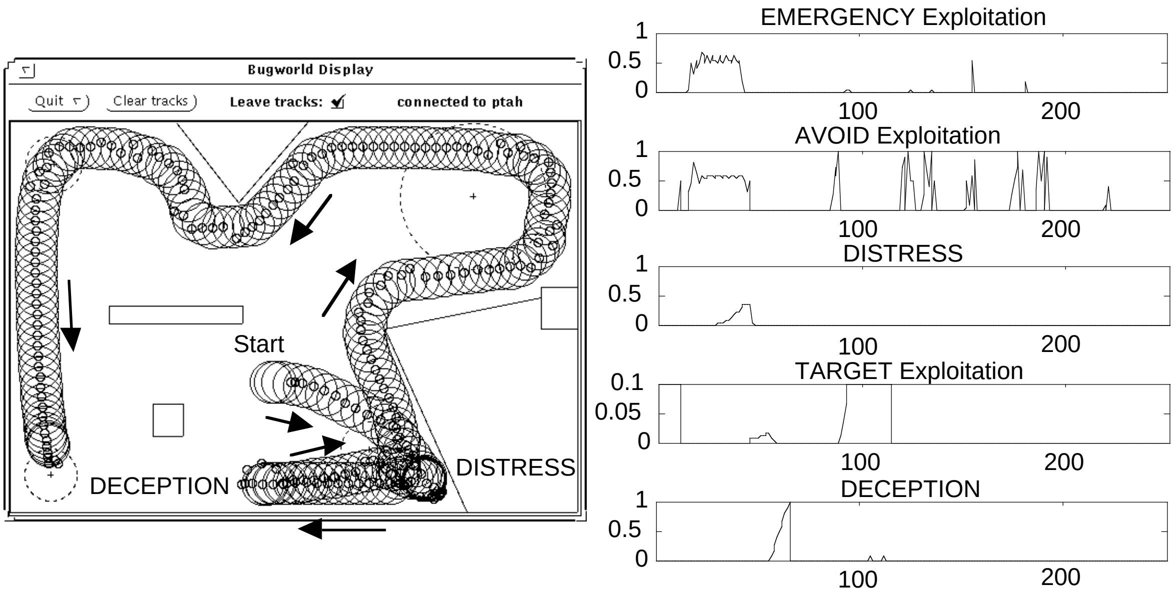
a) Before optimization

b) After optimization

François Michaud, Gérard Lachiver, Chon Tam Le Dinh

Architectural methodology based on intentional configuration of behaviors

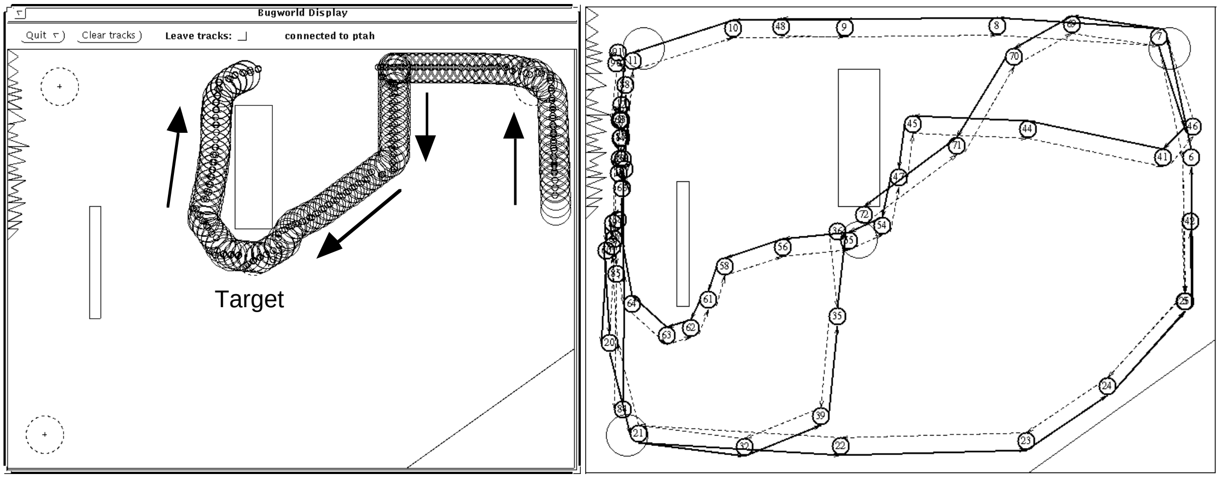
FIGURE 19. Topological graph before and after optimization



François Michaud, Gérard Lachiver, Chon Tam Le Dinh

Architectural methodology based on intentional configuration of behaviors

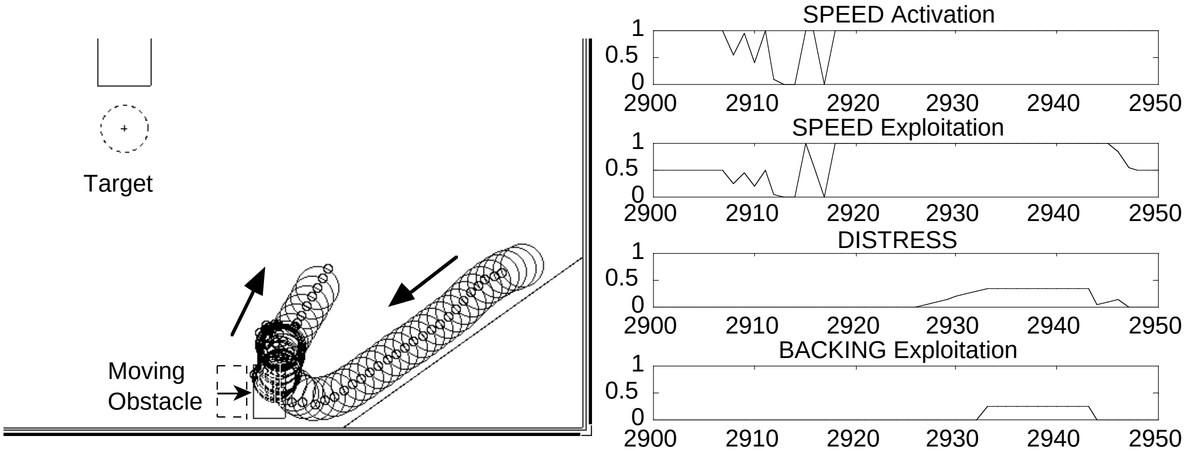
FIGURE 20. Trace and conditions for the motives DISTRESS and DECEPTION



François Michaud, Gérard Lachiver, Chon Tam Le Dinh

Architectural methodology based on intentional configuration of behaviors

FIGURE 21. Path reproduction



François Michaud, Gérard Lachiver, Chon Tam Le Dinh

Architectural methodology based on intentional configuration of behaviors

FIGURE 22. DISTRESS motive when a mobile obstacle came toward the agent