17

Using Probabilistic Cellular Automaton for Adaptive Modules Selection in the Human State Problem

Martin Lukac[1], Michitaka Kameyama[2] and Marek Perkowski[3]
[1,2]*Tohoku University*
[3]*Portland State University*
[1,2]*Japan*
[3]*USA*

1. Introduction

The field of the HRI is an important area of research in robotics with wide range of applications and open problems. One of the main reasons for being an attractive research field, is the fact that robots designed to act in environment side by side with humans require highly complex and adaptive interaction mechanisms. This requirement for complex processing is due to the fact that one of the main components of the HRI is the emotional expression. However, emotions are not only present in human communication but also have been found to take part in memory mediation, focus, imagination or complexity reduction (4–7). Because both the human emotional expression and emotional mechanisms are still an open area of research it is not possible to formulate a precise function that would truly model these mechanisms. Thus robots designed for human interaction such as service bots, social robots or simple social agents require a lot of adaptive mechanisms allowing them to at least partially account for the complexity of human communication and adapt quickly to the unpredicted situations arising from hidden emotional human states.

Robots evolving in the social environment cannot always expect explicit feedback. The feedback from a user with respect to a robot can be either explicit (pressing a button, voice command) or indirect; an automated robotic system that extracts the change of user expression, body posture or facial expression can use such information to adapt its behavior. The successful usage of such indirect feedback can be very useful in some robotic applications. However, despite being less invasive, the indirect feedback is more difficult to use because a robotic application might not be always able to determine the proper cause of the human emotional state (the robot might not be able to determine if the patient is unhappy because of wrong service or because of bad internal (personal) state). A good example is a hospital service robot: some patients might not always be able to properly communicate their state explicitly. Thus estimating their internal state or intentions is very important.

To analyze the above introduced problems we propose a so-called *Human State Problem*(HSP). The HSP consist of a robotic application and of a user with some unknown expectations about the robot performance. The robot's behavior is generated with the goal to keep (or bring) the user to a desired target emotional state. The robotic application has only access to the

indirect human feedback (emotional state estimation, posture estimation, etc.) to evaluate its performance. The indirect human feedback is unlike classical human feedback - passively captured by robot sensors and without the user's explicit and symbolic feedback. In this manner the robot behavior is seamlessly adapted to user expectations and the user obtains a unique and pleasant experience. Such feedback is considered to be noisy, difficult to analyze and not always represent only the user expectations about the robot's performance. The robot thus cannot analyze fully the environment as well as completely map user preferences (intentions) and must use active emotional-state-mining approach to extract a maximum information from a particular user state to plan its next actions.

In this chapter we first look at the Human State Problem from a probabilistic point of view within the context of the Live-Feeling Communication. As one possible approach to solving the HSP we introduce the Adaptive Functional Module Selection (AFMS) framework of robotics aimed to act in a social context. The AFMS robotic model uses a set of black boxes (computational processes) to map in real-time an input-to-output by quickly selecting the most appropriate computational resources from a pool of available intelligent processing modules. The selection mechanism is based on the indirect user feedback, and in this chapter we provide the initial problem formulation and description of the overall system. We describe a Bayesian Network (BN) example as a method of arbitrating the AFMS mechanism. The BN approach is described as example of real life situations problem solver however, it might not be practically the most efficient approach because the real-life environment does not always posses a hierarchy that could be efficiently exploited in the BN methodology. As an alternative we present a Cellular Automaton (CA) based approach for AFMS mechanism and demonstrate its potential use in such situations where models such as BN cannot be efficiently used.

This chapter is divided as follows. Section 2 introduces the details of the studied problems; the Section 2.1 introduces the details of the HSP and Section 3 introduces the probabilistic formulation of the HSP. Section 4 describes the Adaptive Functional-Modules Selection (AFMS) mechanism (10; 11) and Section 5 presents the AFMS based robotic platform. Section 6 describes the use of Cellular Automaton for behavior arbitration and Section 7 provides a machine learning approach to the HSP using Bayesian Network. Finally Section 8 concludes the chapter and provides discussion on the results and future work.

2. Predicting human intentions from the observations of its emotional expression

2.1 The human state problem framework

The problem framework is shown in Figure 1. A robotic application with a set of sensors and actuators is designed with the goal to keep the user in a *happy* state as long as possible. The mapping from sensors to actuators is done by assigning resources from the *Intelligent Processing Resources* pool by the controller that receives the feedback from the user. The user, observing the output of the robot, changes its state such as from happy to unhappy, or happy to happy, and the change of state is provided as the feedback to the controller.

Let s_{uh} be a user state from a set of all possible observable human emotional states $U = \{u_0, \ldots, u_k, \tilde{u}\}$, with $\tilde{u}$ being the target user (set of) state(s), and let c_j be the robot state from the set of all possible robot states $C = \{c_0, \ldots, c_p\}$. Each state c is represented by an output that corresponds to the full process of mapping a given input to the output in the state c_j. In the rest of this chapter we use c for an arbitrary robotic state and u for an arbitrary user state. A configuration is given by the state of the user $u \in U$ and the state of the robotic application $C \in \{c_0, \ldots, c_j\}$: $\gamma(s_u, s_r)$.

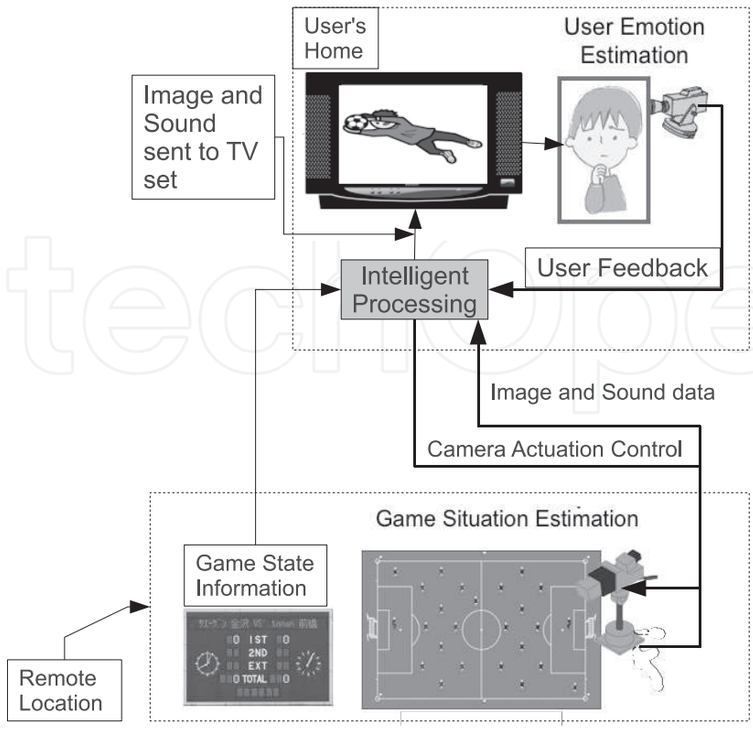

Fig. 1. The Live-feeling communication application. The user sitting in front of the TV has its emotional state passively extracted from a set of sensors on the TV. The emotional state is used to adaptively reconfigure the robotic behavior that controls the actuators of the cameras on the remote location. The decision making is done inside of the box *Intelligent processing* .

In order to make the problem at least partially tractable and allow further investigation, we simplify further the overall framework. We assume the following conditions:

1. The state of the machine c changes the state of the human u immediately
2. If a machine state c does not change the human state u immediately, it is assumed that it will have no effect on later changes on the human state u while the machine remains in the state c
3. For now, the human state as well as the human state change is considered to be only the application related and it is assumed that the desired state is reachable using the robotic application.

The robot changes its state from c_j to c_k using a state change relation $f : (u_l, c_j) \rightarrow (u_m, c_k)$. It is assumed that for each robotic state c there is a corresponding change in human state given by $u_j = (u_i, c)$. But following condition 1, the human state change caused by robot state change is not necessarily observable by the robot.
Figure 2 shows the high level of this framework. On the left is a non-terminal user state: $\bar{u} = U - \tilde{u}$ and on the right side it the user in the target state $\tilde{u}$. The change from an non-terminal state to the desired target state can be triggered by various sequences of robotic states (actions). Formally, from the point of view of the robot architecture the goal is specified by:

$$\{\forall c \in C,\ f|(u,c) \xrightarrow{min_G} (\tilde{u},c)\} \tag{1}$$

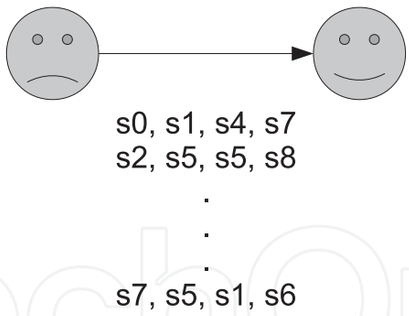

Fig. 2. The problem statement: what is the minimal set of states the robotic controller has to go through to make the user be happy?

This means that we are looking for such a relation f or a set of functions $\hat{f} = \{f_0, \ldots, f_o\}$ that will in a finite number of steps bring the user from an arbitrary state u to the target desired state $\tilde{u}$.

2.2 The human state problem

Definition 1 (The Human State Problem)**.** The problem of finding such a mechanism that would in a particular context of a robotic application successfully estimate human intentions related to the application only from indirect emotional feedback.

Figure 3 shows the concepts and the main components of the HSP setup. The G^t, G^{t+1} represents two consecutive states of the application context - the environment. Similarly, C^t, C^{t+1} represents two consecutive states of the robot and U^t, U^{t+1} represents two consecutive states of the user. The full arrows represent state transitions and the dotted arrows represent inputs to state transition functions. The environment is considered to be

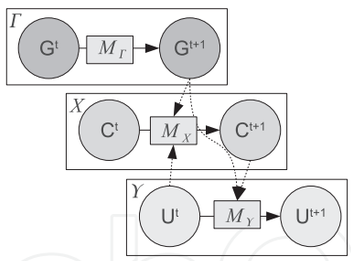

Fig. 3. The HSP context

an autonomous machine - neither the robot neither the user can change how the environment change its states[1]. The change of the environment state triggers a change in the robot state that then triggers the change of the user state. Observe that while the environment is an autonomous machine the robot and the user are both interconnected.

The robotic application used to formulate this model is shown in Figure 1. The robot is represented by the box labeled *Robotic Agent* and its function is to provide the user with a camera configuration most suitable for the user preferences and given a particular state of the environment (live sports event). The user provides feedback in the form of emotional expressions that are used to adaptively change the camera work.

[1] This condition is imposed only in the present application context.

The environment machine Γ represents the live event, The machine Ξ is the robotic agent and the machine Y is the user. For the sake of formalization it is assumed that the user state changes only as the result of the game state and of the machine state.

3. The probabilistic human state problem

The HSP is more realistic when defined using probabilistic approach. This is because beside the states C of the machine both the environment and the user states are incompletely defined and highly noisy. Thus with respect to Figure 3 let $G^i = \{g_0, \ldots, g_k\}$ be the set of the states of the environment (for one particular live event i), $C = \{c_o, \ldots, c_j\}$ the set of all the states of the robot and $U^h = \{u_0, \ldots, u_l\}$ the set of observable user emotional states/expressions (for a particular user and its preferences towards the game i, expressed with exponent h). Let Γ be the autonomous state machine of the environment performing the mapping $M_\Gamma : G^i \to G^i$ on G. Both the user and the robot are interconnected so that we represent the user by a machine Y performing a mapping $M_Y : G^i \times C \times U^h \to U^h$ on U and the robot by a machine Ξ performing a mapping $M_\Xi : G^i \times C \times U^h \to C$ on C.
For any two arbitrary states g_q, g_r of Γ the probability of reaching the state g_r from g_q is given by:

$$p(g_r|g_q) \tag{2}$$

Equation (2) represents the probabilistic autonomous state machine of the environment.
Similarly for any two arbitrary state c_m, c_n of Ξ the probability of reaching c_n from c_m is given by $p(c_n|c_m, u_i, q_r)$ and for u_i, u_j of Y the probability is given by $p(u_j|u_i, c_n, q_r)$.
The two conditional probabilities introduced above can be broken into simpler conditions given by:

$$p(c_n|c_m, u_i, g_r) = p(c_n|c_m) \times p(c_n|u_i) \times p(c_n|g_r) \tag{3}$$

and by

$$p(u_j|u_i, c_n, g_r) = p(u_j|u_i) \times p(u_j|c_n) \times p(u_j|g_r) \tag{4}$$

Equations 2 to 4 represent the formulation of the HSP. The component probabilities $p(c_n|u_i)$, $p(c_n|g_r)$, $p(u_j|c_n)$ and $p(u_j|g_r)$ represent the user preferences while the probability $p(c_n|c_m)$ represent the desired mapping constrained by $p(u_j|u_i)$. In other words, the desired robotic mapping is such that

$$p_{max}(c_n|c_m) \leftrightarrow p_{max}(u_j|u_i), \text{ for some desired } u_j \tag{5}$$

The goal of the robot is thus provide such a mapping that will satisfy the user. This can be written as shown in (6).

$$p(c_n, v|c_m, u_i, g_r) = p(u_j|u_i, c_n, g_r) \approx p(c_n|c_m, u_i, g_r) \tag{6}$$

3.1 Human emotional state data mining

In the previous section we were assuming that condition 3 from Section 2.1 was valid. However in real world applications the user providing feedback to the robotic application might not always express its emotional state only as related to the robot task performance. For instance, the application framework illustrated in Figure 1 can be used for an automated Live-performance TV-set. In such case, the actuators control the camera and the microphone and the sensors accept data from the camera and microphone. The output of the camera and the microphone is transmitted to the user via the TV-set. Thus in a particular sport event, the

user can be angry at the team in question (game quality) or can be unhappy about the viewing angle of the camera (robot performance quality). In both cases the expression can be the same and the robot is required to identify the proper cause of user state.
Our approach to deal with this problem is based on Human-Emotional-State data-mining (HESdm). A user in a given state will show different sensitivity to changed robot's actions depending whether yes or no he/she was displeased with the robot previous behaviors. For instance, assume that the user is watching a sport game, and is dissatisfied; i.e. either the performance of the players is not satisfactory either the viewing angle is not adequate. By following the conditions 1 and 2 from Section 2.1 change in user behavior can be described by two cases:

1. The user will change its state proportionally to the change in robot performance (state). This is the case described by all three conditions 1, 2 and 3 in Section 2.1.
2. The user will not change its state proportionally to the change of robotic behavior, because its emotional state is mainly based on the content of the robot's output rather than on the robot's behavior it self.

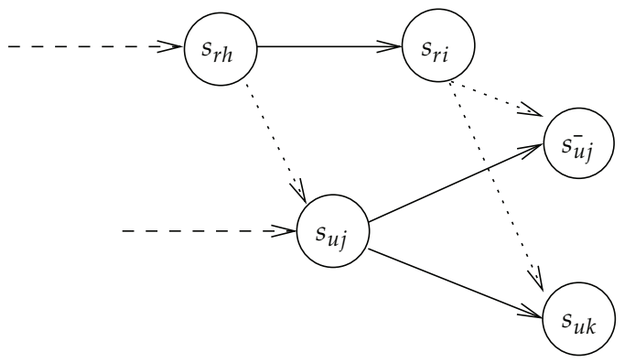

Fig. 4. The human emotional state data mining. The two possible human state changes; the robot changes to a state that either triggers a state of human emotional state or is mostly ignored. This is the trail-and-error approach to determine the relation of user state to the robot's performance.

These two cases are illustrated in Figure 4. The robot is in the state c_h and the human is in the state u_j. It can be seen that we assume that the user-state u_j was generated as the reaction to the robotic state c_h (more or less depending on the content as well). At this point, and depending on the content of the robotic output, the user state is not the desired user state $\tilde{u_j}$. The robot consequently changes the state to c_i to determine the reason of the user state. If the user state is only content related, the resulting user state will be $\bar{u_j}$. The new user state being similar to the previous user state indicates that the user is unhappy because of the content rather than because of the robot performance. In such case the robot can keep the current state until further event either change the user state or particular case of the game will trigger a different robot state. In the opposite case, when the resulting state is u_k, the robot can conclude that the user dissatisfaction is related directly to the robot performance and thus a change in robot actions is required.

4. Adaptive Functional-Modules selection

The Adaptive Functional-Modules Selection (AFMS) (10) was introduced as a mean to provide a mechanism to generate robotic behaviors in real time by recombining component functions.

The AFMS principle is different from the well know behavioral approach to robotics (2) in that in the behavioral approach one deals with monolithic blocks representing behaviors. Instead in the AFMS the monolitic blocks are on the level of functions and each combination of these blocks create a different behavior. The goal of the AFMS mechanism is to provide a lrger repertoire of behaviors while preserving the same amount of computational ressources. Each combination of a set of functions provides a robotic behavior. Using the classical hierarchical robotic paradigm let $P = \{p_\alpha, \ldots, p_\omega\}$ corresponds to a set of functions processing input, $E = \{e_\alpha, \ldots, e_\omega\}$ be a set of functions corresponding to estimation and $A = \{a_\alpha, \ldots, a_\omega\}$ be a group of function controlling the actuation. Also let $B_\Xi = \{b_0, \ldots, b_n\}$ be a set of all possible robot behaviors given by $C \times G$. Let every robot state c_l correspond to a selection of functional modules given by:

$$c_l = p_\alpha \times e_\beta \times a_\delta \tag{7}$$

and the output of the robot to the user is given by a composition of selected functional modules from each group with the current state G:

$$b_j = a_\delta \circ e_\beta \circ p_\alpha(G) \tag{8}$$

resulting in the observable behavior b_j of the robot. Equations 3 and 4 now can be rewritten to a more concise form:

$$p(c_n|c_m, u_i, g_r) = p(c_n|u_i, b_j) = p(c_n|u_i) \times p(c_n|b_j) \tag{9}$$

and

$$p(u_j|u_i, c_n, g_r) = p(u_j|u_i, b_j) = p(u_j|u_i) \times p(u_j|b_j) \tag{10}$$

Note, that unlike the distinct states from C, the behaviors B are not distinct and most of them can be considered equivalent from the point of view of the user. Thus the desired mapping we wish to obtain is given by $p_{max}(c_n|b_j) \leftrightarrow p_{max}(u_j|u_i)$. Because the game states are not completely known and cannot be completely predicted, it is possible to restrict $p(c_n|b_j)$ to $p(b_k|b_j)$ to such set of robotic behaviors that would correspond to unique $C \times G$. This means that for $b_k = b_j$ with $b_k = c_n \times g_q$ and $b_j = c_m \times g_r$, $c_n = c_m$ and $g_q \approx g_r$. In the case when $b_k \neq b_j$, then both $c_n \neq c_m$ and $g_q \neq g_r$. This means that when the robot will estimate that the game states g_r and g_q are similar it will always perform the same camera work. However, from a practical point of view, such restriction could entail that the user will not be able to obtain a maximum satisfaction from the robot performance because the robot will always perform the same camera work for a given group of game states and the user will not be satisfied.
When such generalization cannot be done, in practical application it is the most appropriate to use some sort of machine learning approach because both the user states and the game states are only partially known and cannot be used without a prior learning for making generalizations and predictions.
Before showing the machine learning approach we illustrate the mechanism by which the module selection allows to change robot behavior and how it allows the user to be more or less satisfied with the audio-video data sent to the TV-set.

Example 4.1 (Adaptive Functional-Modules Selection)**.** *The robotic setup shown in Figure 5 shows that the robot has the following functional modules:* player detect (pd), ball detect (bd), position estimator (pe), speed estimator (se), camera movement (cm) *and* camera zooming (cz). *A robotic state C corresponds to a selection of exactly three functional modules; one modules is selected from each column taking input from the previous module (or sensory inputs) and sending its output to the next module or directly to the actuators. In this manner each selection correspond to the assignment of the values to the control variables D,E,A (Figure 5).*

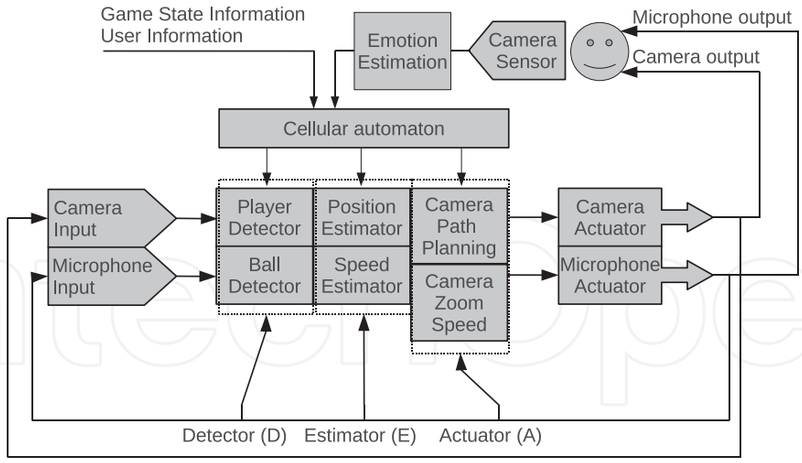

Fig. 5. The Robotic Application schema using the Adaptive Functional-Module Selection Mechanism

Because the generated behaviors are analogous to the reactive behaviors in the model of Arkin (1) or Brooks (2) the output of the robot is content dependent and thus cannot be tested without a proper environmental settings. However the schematic representation of the AFMS architecture shown in Figure 5 is only a simplest example: each real-time behavior controls only one degree of freedom at a time and is represented by a cascade connection of functional modules. It is expected that using AFMS more general methods of computing outputs from each column are available allowing a higher degree of freedom in the selection process.

The AFMS principle is based on biologically-inspired approach to human dealing with infinite complexity using finite amount of resources. In such context the AFMS mechanism provide a mechanism for decision making by switching behaviors not only in the traditional manner - based on the input but also based on feedback and off line learning. As introduced above the presented model is the simplest one. It is not difficult to imagine such AFMS that selects multiple functional modules in each functional column and then propagates the fused result to next column.

Figure 5 shows a feature-based robotic system, that chooses which features should be used to generate the current behavior. Such selection can be altered by allowing the AFMS mechanism representing different algorithms for each of the features. In such case we set of features remains constant in each computation of output from input but the algorithms used are different. Finally, the AFMS can be seen as adaptive mechanism to minimize the internal allocation of resources. For instance, assume that a face is detected on an image. In order to analyze the face in general a set of pattern-templates are used; the selection of the templates is in this case performed by the AFMS in order to minimize the number of patterns that have to be tried on the face.

Thus AFMS can be seen as both, a mechanism for adaptively choosing resources for intelligent processing and as a mechanism of building internal model. The next section illustrates how using a Bayesian Network a module allocation algorithm based on probabilistic reasoning is obtained.

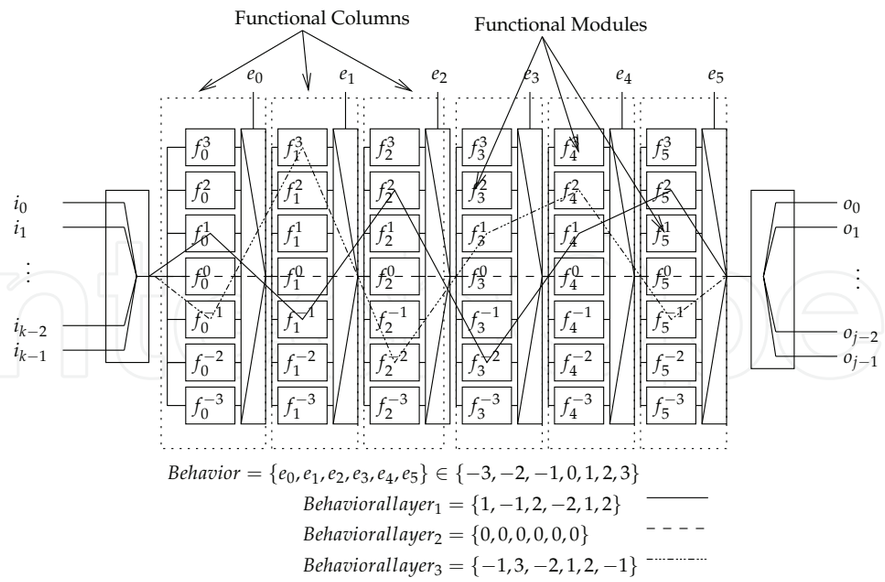

Fig. 6. Example of the behavioral matrix robotic processor architecture

5. The proposed model

To address the above introduced problems we designed a hardware architecture that can in real time generate behavioral layers by adaptively combining functional modules. This hardware architecture designed as the platform for the AFMS mechanism is called the behavioral matrix.

5.1 Behavioral matrix

The principles that this platform is based upon are the following:

- Each behavioral layer can be build in real time by interconnecting a set of pre-defined functional blocks (functions).
- Each behavioral layer built in real time is composed from input-activated functions and output-filtered functions.
- Both valid behaviors and invalid ones can be created
- The output from each function in a behavioral layer can be based on current or past input values

The proposed hardware architecture is shown in Figure 6. It is built from functional columns (hardware blocks); each column is built from a set of functions (functional modules), from an output buffer, from an activation module and from an arbitration module (Figure 7). The activation module allows to decide where the data input goes - i.e. which function will generate output. The arbitration module allows to select which functional output will be selected as the output of this column. The set of functional modules can be any type of function: from neural network processor to a Fourier transform or a face detection algorithm. The behavioral arbitration module is represented by a MUX letting only one functional module to propagate its output to the output buffer. The activation module is either a DEMUX (propagating the input only to a single functional module) or is completely ignored and thus allows the input data to be propagated to all functional modules (Figure 6).

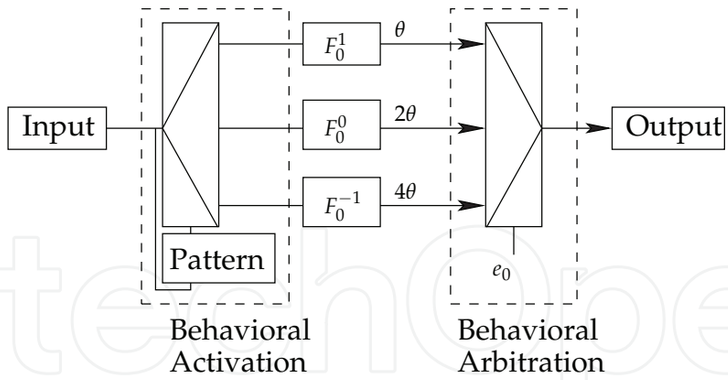

Fig. 7. Example of a functional column with the function activator and the function arbitrator

The output buffer in each functional column allows to be filled with the currently selected functional module output value and thus to propagate sensory input i^t to the output $o^t = f_j(i^t)$. This can be seen in the case when input data reach a functional column and both the activation and arbitration modules are pointing to the same function. The output buffer can also propagate a past input to the output $o^{t-k} = f_j(i^{t-k})$. This is the case when either the arbitration or the activation modules in a functional column point to a different function; the output obtained on the output will be the value last stored in the output buffer.
Observe that using our robotic platform the functionality of each behavioral layer is created automatically in real time by a serial combination of functions. Each function from a column is selected using a control variable $\{e_0, \ldots, e_5\}$ in Figure 6. The output of the last functional column is either propagated to the actuators or can be withheld to be used as a part of the next input. In this manner each functional column can be used as a combinational circuit or as a sequential circuit.

6. Cellular automaton for AFMS

Using the behavioral matrix, the arbitration of behaviors become a a selection of control variables with respect to a particular set of inputs and a feedback.
Because the problem introduced cannot be analyzed from a global point of view, we propose to use an arbitration mechanism based on local rules and feedback. Using local rule based arbitration mechanism we can on one hand build in real time a global representation of the environment based on local rules and on the other hand use computational models requiring less resources. Using a Cellular Automaton (CA) to assign in real time computational resources in the behavioral matrix, saves computational resources (only used functional modules are turned on), uses only local state transition function and as required allows to build a global environment representation. Also and more important, the use of CA for behavior arbitration allow for cyclic sampling of the environment and action generation. The idea behind this approach is the fact that the execution of a task can be done by a sampling through the environment with both various sensors and various actuator actions. This means that a robot with a set of possible actions, can be instructed to solve a particular task by cyclically selecting from a sub-set of these actions and arriving at its goal. The main advantage compared to behavioral robotics for instance is the resource saving; while in standard robotics all resources are computing in parallel at all times, the architecture proposed here allows to selectively activate particular set of computational modules and thus save energy.

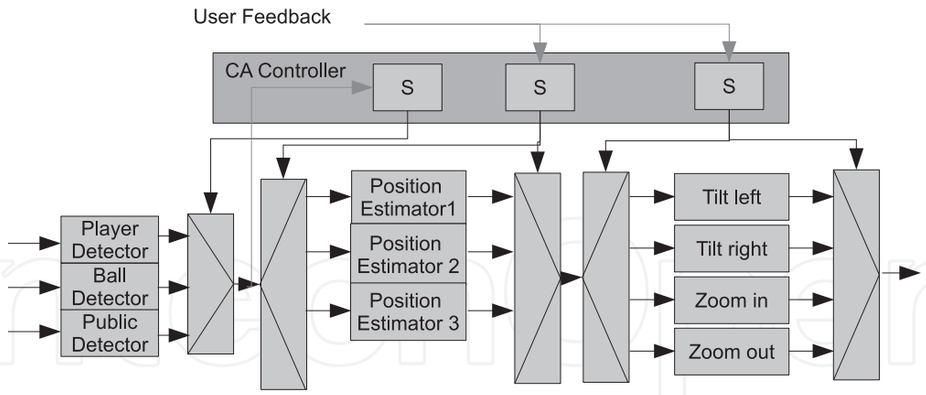

Fig. 8. Example of a behavioral controller based application. The first column represents the detector allowing to detect a player, a ball and the audience, the second column shows three different black-box estimators and the last columns are actuator commands. Observe there is one internal feedback from the first column to the controller as well as the external user feedback allowing to change the output of the robotic application.

Figure 8 shows an example of robotic application using the behavioral matrix with three columns controlled by a three-cell CA. The application that this processor was designed for is the intelligent TV-set. The purposed is to allow the user to change behaviors of cameras and thus allowing the user to fully enjoy for instance a live sport event (Figure 1).
Each functional column has either three or four input functions and thus each of the control cells is considered to use three/four valued logic. The robotic processor has two types of feedback. First, an internal feedback is used to lock the first column; if a given feature is detected the CA will not change the detection module. A change in the detected feature is triggered only by the global robot state change. Second feedback is coming from the user and allows to force the CA to change its state. Thus if both feedbacks are positive the robot will continue to perform the current operation unchanged. In any other case the CA will evolve according to a local rule until both feedbacks are satisfied.
To illustrate the reason of usage of the CA for the real-time behavior arbitration the following example shows how the trail-and error or behavioral arbitration results in minimization of the number of states required to bring user from a non-terminal state to terminal (desired) state.

Example 6.1 (Minimizing path to desired user state)**.** *Let M be a robotic application with a CA controller. The state transition function is specified by the reachability graph shown in Figure 9 and is generated using a three-cell neighborhood CA state transition function f. The two dark squares shows the state transitions of interests. In particular the larger square shows an example situation where the user state and the machine state are respectively given by (s_{uk}, s_{rj}) Given this initial configuration the machine proceed in changing its state from* 023 *to* 013 *and so on until the state* 013 *generates the user state $\tilde{s_u}$ which is the target user state. Because the goal of the robot is to satisfy the user as quickly as possible or even to keep the user constantly happy, the robot uses local rule optimization to reduce the number of required steps between the (s_{uk}, s_{rj}) and the $(\tilde{s_u}, s_{rj+4})$. This is shown in Figure 10 where the modified state-transition function partially restructured the reachability graph of the machine M.*
[End of example]

The advantages of the CA based robot arbitration concept introduced in this chapter seem can be summarized by three powerful concepts: the multi-modal-perception (3; 8; 9) decision making, the non-abellian decision making and the sensory noise resistance.

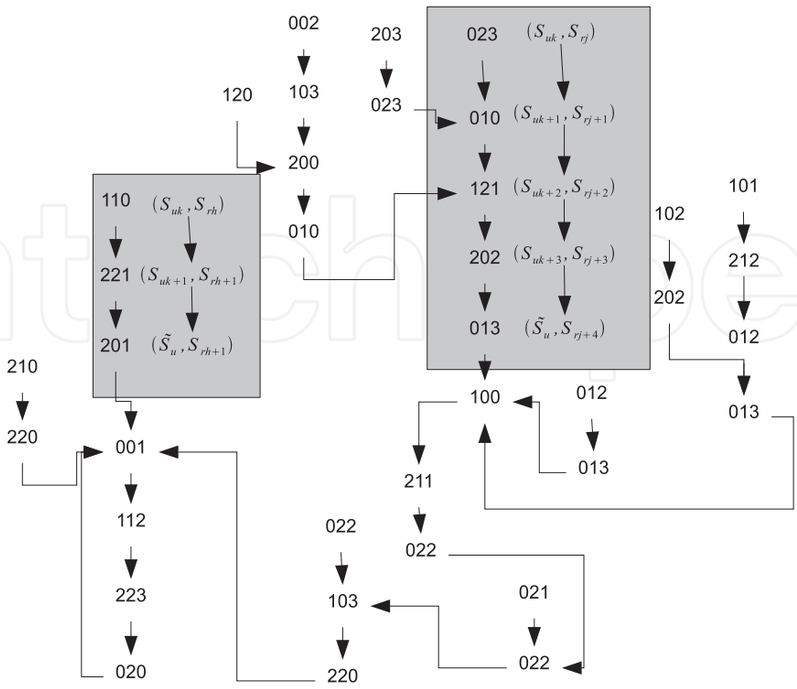

Fig. 9. State reachability graph of machine M.

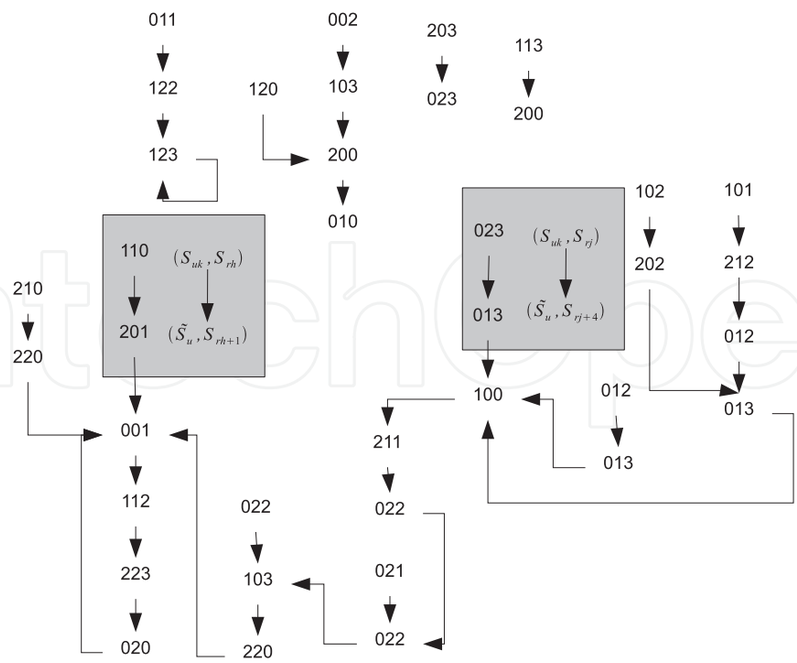

Fig. 10. Two path minimized by changing single rules in the state transition function.

To explain these notions let first assume that the robot has a video camera and a set of image processing algorithms: a set of low level transforms such as Sobel, Hadamard, Haar, HOG (Histogram of Oriented Gradients) as well as a set of high level feature extracting tools such as face detection, face tracking, object detection, shape extraction, color tracking and so on.
The *multi-modal perception* is a well know area in robotics and in general it means that a given decision is made on multiple perception sources and levels. This is illustrated with the approach shown in Figure 8 where various sensor sources can be combined as given by the CA cycle of state change. In this case, using the *Output-per-final-configuration output generation* the output can be generated using the history based approach combining multiple sources of information.
The *non-abellian decision making* is a generalized concept and it means that a sequence of applied behaviors to a given sequence of inputs is not equivalent to the application of another sequence of the same behaviors to the same input sequence. This can be observed from a simple example. Assume a robot is supposed to solve a puzzle consisting in moving blocks around by pushing them. Any action resulting in a block being adjacent directly to the wall does not allow the robot to move that block away from the wall. Such situation can result either from bad planning but also from bad sensory input-sequence evaluation. Because each behavioral layer pursues its own goal and sometimes such goals can lead to mutually exclusive situations, the order of applying of the behavioral layers outputs bears extreme importance. Such choices can allow to explore novel regions or improvise a solution found by mistake.

7. Learning adaptive behavior from training samples

In the previous section we introduced a mechanism of selecting functional modules using Cellular Automaton. However, the design of such automaton might turn out to be difficult. In this section we present a mechanism for selecting functional modules using a statistical method. Notably we show how to use a Bayesian Network for module selection by using symbolic information as inputs to the BN and obtaining outputs a set of user's preferences. The BN is designed so as to generate additional information for the computer which when combined with information from standard image and sound processing will allow a higher degree of control over the automated process. Figure 11 shows the high level schema of the system. Three parallel processes are extracting the ball, the players and the referee(s) by color filtering the raw input image. Each of the players' and the referees' movements are recorded into a sequential memory, where it is used to extract information about their performance in the game (Figure 11: box labeled *Human Behavior Recognition*). At the same time players facial expressions and body gestures are used to determine their emotional state (Figure 11: box labeled *Human Expression Recognition*). This pre-processed information, with the coordinates of all the players, referees and the ball as well as information about user preferences and game information are the inputs to the module M.
Figure 12 shows in more details the inputs and outputs to the module M. In particular it can be seen that the module M contains a Bayesian network and additional computational resources that will together be used to generate the final output. The combined information is shown in Figure 13 on an example containing a schematic of a baseball field with some players on it. Observe that each object (player, referee, ball) has a set of coordinates and a percentage value. The percentage value represents the degree of user's interest in a given object.
To understand this concept assume that the default behavior of the robot is to track the ball with the camera. Such behavior will capture - with appropriate zoom - most of the action

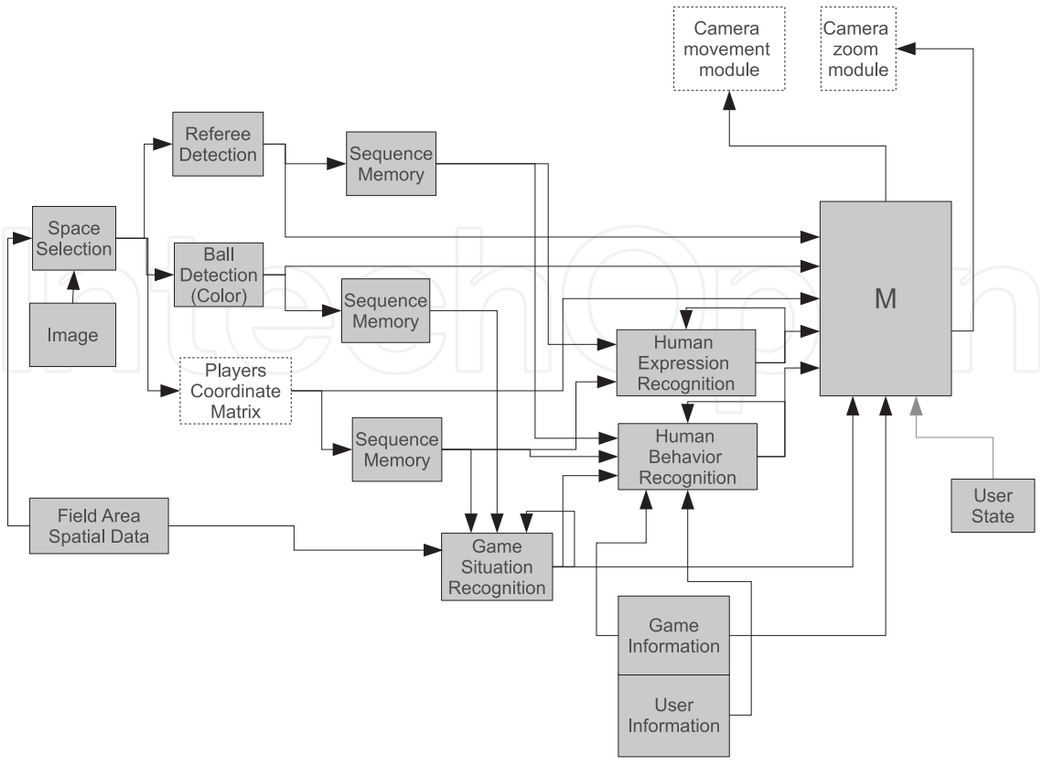

Fig. 11. High level architecture for a Live-Feeling system. Module M represents the processing that is used to provide automated camera and microphone behavior.

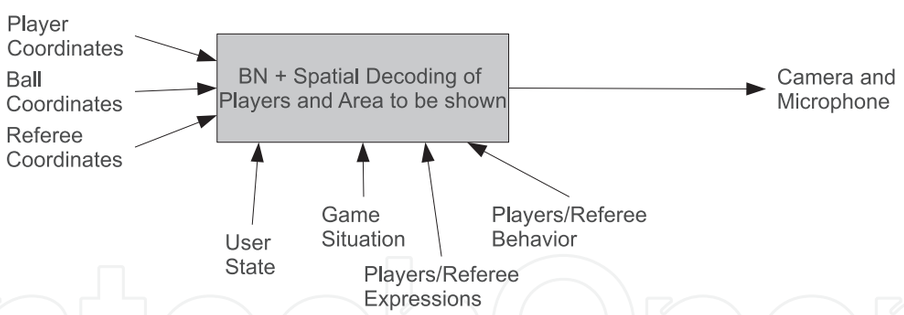

Fig. 12. The inputs and outputs of the module M.

in the game that the user is interested in. However, player's behavior, referee behavior or unusual player's performance can sometimes be more interesting that the center of the game. In such cases it is important to capture such events because an automated system that can detect such highlights of a live event and autonomously show them to the user will provide the user not only with the game elements but also with such scenes that will increase user's happiness.

In order to allow an autonomous system perform such actions, the decision about the content what will be shown to the user as well as about the manner this content is shown must include a prediction about the level of user's interest. In order to do that the following criteria are taken into account:

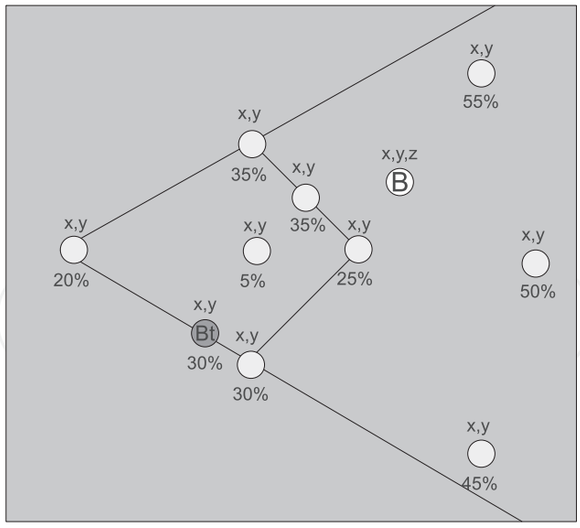

Fig. 13. The complete information generated in the module M.

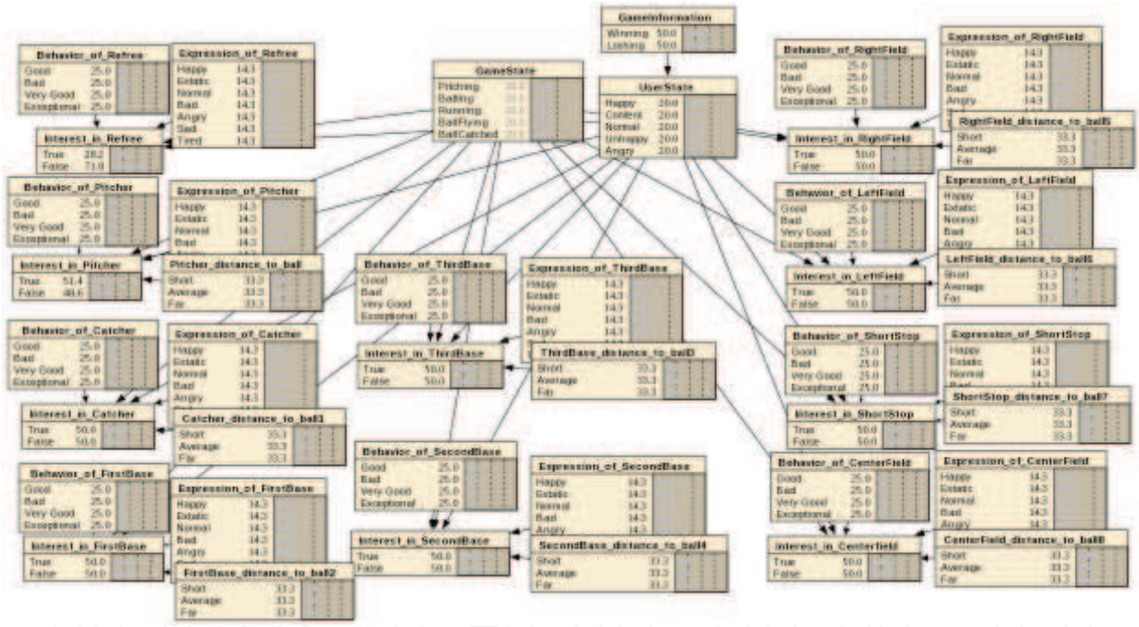

Fig. 14. The complete information generated in the module M.

- Player's emotional behavior: if a player does something unexpected and mostly unrelated to the game performance it increases its chance to be included into the camera image
- Player's professional behavior: if a player performs a directly game related action in an unexpected manner it increases its chance that the camera will focus on him/her.
- Referee's emotional behavior: if the referee performs an action unrelated to the game but with high emotional arousal it increases the chance that the camera will focus on him

An example of system that perform such operation is shown in Figure 14. The BN's inputs are a from preprocessor that has been previously introduced. In particular the BN'sinputs are the players' and referees' emotional expressions, game behavior, the distance between

the players' and the ball, the game state and the user's feedback. The goal of the network is to asses the probability of a given player as a modifier of the default robot's behavior. In most of the cases this is only a matter of proper zooming because as already introduced the ball-following behavior will be in most of cases able to capture the most important features of the game. This can be seen in the Figure 13, where every detected object on the game field is given a percentage representing the degree of user interest in that object.
In order to make this evaluation more realistic, one can assume multiple cameras because one camera pointing at a given location with a certain zoom can in general only see a subsets of players. Because the BN and the preprocessing is performed on the current image in most of the cases the interest degree will be the highest near the ball. This is a precondition in our model that represents the overall game dynamics. However in a realistic situation the changes of zoom or of camera orientation that are often interesting are not directly centered centered on the game but rather on different point of views of the given situation or from different location. In such case, images from different cameras can be processed in parallel and evaluated in order to find the best point of interest.
Finally, with respect to the AFMS method described in Section 4, the usage of the BN resides in the resolution of the network outputs. Without the usage of the AFMS mechanism the spatial coordinates and the degrees of interests calculated by the BN can be directly mapped into camera commands using weighted sum or similar linear methods. When used with the AFMS the output of the BN is similarly mapped into the allocation of the functional modules so as the output behaviors of the robot corresponds to the displaying of the objects with the highest degree of interest. This means that according to the output modules selection will vary.
Naturally one can ask: if the BN allows such direct mapping into the camera movements why AFMS is even considered? This can be answered by looking at the example of the BN shown in Figure 14. The architecture of the network is unfortunately a star network which means that there is no advantage of using BN over any other methods, because each nodes will require a large look-up table. This is contrary to the ideal use of the BN network where hierarchy should be used to minimize the size of the look-up table in each node.
The reason for this architecture is the fact that to evaluate the user's interest is based on a multi-level analysis of individual player's behavior a as well on the evaluation of the global properties of the game. The BN from Figure 14 is just a mere example but from it structure it can be deduced that if one would introduce user's interest of group of players in the network, the structure will essentially be not changed. The network will certainly get more complicated but hierarchy will be still lacking.
The machine learning required to allow a proper AFMS mechanism is well known in the case of the Bayesian Network. From a set of samples it is now a standard procedure to generate probabilities and then use these conditionals in the appropriate nodes. In this approach to machine learning the most important is the quality of the data as the learning by itself is done by probabilistic inference. The final result of the computation of the module M is shown in Figure 13 allows by selecting player tracking or ball tracking set of functional modules in order to move the camera to the desired location.

8. Conclusion

We presented a problem of HRI called the *Human State Problem* and showed how it can be partially analyzed and solved using deterministic hardware based approach using a Cellular Automaton as well as probabilistic Bayesian Network. For this we illustrated our robotic

processor using a CA for adaptive resources selection and showed by example how it can be used for machine learning of robot behavior by modifying the local state-transition function of the CA in real time. The advantages of this approach are: (1) it is designed to be realized in VLSI leading to low cost and low power consumption, (2) it allows a fast switching of behaviors, (3) it uses multi-valued logic and thus leads to fast realizations of behaviors in hardware. The formalization of the HSP problem allows to at least partially to study the possible solutions and propose some solutions.

The purpose of designing such robotic processor is to allow the robot to deal with noise in real environment as well as with extremely complex situations requiring real-time processing. Examples of such environments are for instance human emotional perception, indirect human feedback or general human-robot interaction. The main problem is that in these environments a single sensory input might not be sufficient and more complex sensor processing and general purpose processing might be required. However the processing must be highly adaptive and should allow variations in both inputs and outputs. Thus the CA controlling the robotic processor by larger or smaller cycles can be used to explore/exploit more or less of the given situation for the purpose of achieving its task. For instance analyzing the input only for object detection and face detection will generate a different output than if the same input is analyzed for face detection, motion tracking and emotional expression. Obtaining different set of features from the input depends on the states of the CA and on how many different states the CA goes through. Thus it is directly related to the size of the cycles and to the states it goes through within these cycles of states.

A common question in this approach is why we propose hardware rather than software implementation? The answer is that it is in hardware where the multi-valued logic can bring power saving and thus is ideal to build small low power behavior based robotic processors. In particular it is interesting for robots that can benefit of having the processing on board as opposed receiving all control commands from a computer through a wireless link. As in general the main power consumption is from actuators, eliminating an active wireless link can save a lot of power. Also, on board implementation eliminates additional devices that a remote computer needs such as hard drive, memory and so on. Thus from a global point of view an on-board implementation is beneficial for both the power saving as well as real time processing. Moreover, the design is meant to be learn-able - both the control logic as well as the functional modules are to be either learned in real time of programmed before the task. Thus such controller can be reused by simple reprogramming for various low power robots.

Finally the future of this work consists in developing the hardware platform and implementing a learning mechanism so that the most desirable robotic behavior can be obtained for various applications. The proposed multi-valued basis for design algorithm is to be extended in more details so that ideally behaviors specified by simple constraints can be designed using standard Logic Synthesis methods. Moreover, to solve some of the problems related to complexity and noisy environment we plan to investigate a different method of machine learning based on probabilistic automaton and probabilistic cellular automaton. This approach, could possibly solve some of the shortcomings of the BN in this application and thus either combining different probabilistic computational models for the module allocation would then provide a feasible result.

9. References

[1] R. C. Arkin. *Behavior-Based Robotics*. The MIT Press, 1998.

[2] R. A. Brooks. Intelligence without reason. In *Proceedings of the 12th IJCAI*, pages 569–595, 1991.
[3] B Chandrasekaran. Multimodal cognitive architectures: Making perception more central to intelligent behavior. In *AAAI national Conference on Artificial Intelligence*, pages 1508–1512, 2006.
[4] A. R. Damasio. *Descarte's Error: Emotion, Reason and the Human Brain*. Picador, 1994.
[5] A. R. Damasio. *Looking for Spinoza*. Picador, 2004.
[6] P. Ekman. Facial expression and emotion. *American Psychologist*, 48(4):384–392, 1993.
[7] P. Ekman. *Basic emotions*, chapter 3. Wiley, New York, 1999.
[8] M. P. Hollier, A. N. Rimell, D. S. Hands, and R. M. Voelcker. Multi-modal perception. *BT Technology Journal*, 17(1):35–46, 1999.
[9] M.H. Luerssen, T.W. Lewis, R.E. Leibbrandt, and D.M. Powers. Adaptive multimodal perception for a virtual museum guide. In *3rd Workshop on Artificial Intelligence Techniques for Ambient Intelligence*, 2008.
[10] M. Lukac, M. Kameyama, and M. Perkowski. Adaptive selection of intelligent processing modules and its applications,. In *The 2010 International Conference on Artificial Intelligence, WORLDCOMP'10*, 2010.
[11] M. Lukac, M. Kameyama, and M. Perkowski. Emotion-aware probabilistic robotics. In *Second International symposium on Aware Robotics, ISAC2010*, 2010.

Cellular Automata - Innovative Modelling for Science and Engineering

Edited by Dr. Alejandro Salcido

ISBN 978-953-307-172-5
Hard cover, 426 pages
Publisher InTech
Published online 11, April, 2011
Published in print edition April, 2011

Modelling and simulation are disciplines of major importance for science and engineering. There is no science without models, and simulation has nowadays become a very useful tool, sometimes unavoidable, for development of both science and engineering. The main attractive feature of cellular automata is that, in spite of their conceptual simplicity which allows an easiness of implementation for computer simulation, as a detailed and complete mathematical analysis in principle, they are able to exhibit a wide variety of amazingly complex behaviour. This feature of cellular automata has attracted the researchers' attention from a wide variety of divergent fields of the exact disciplines of science and engineering, but also of the social sciences, and sometimes beyond. The collective complex behaviour of numerous systems, which emerge from the interaction of a multitude of simple individuals, is being conveniently modelled and simulated with cellular automata for very different purposes. In this book, a number of innovative applications of cellular automata models in the fields of Quantum Computing, Materials Science, Cryptography and Coding, and Robotics and Image Processing are presented.

InTech Europe
University Campus STeP Ri
Slavka Krautzeka 83/A
51000 Rijeka, Croatia
Phone: +385 (51) 770 447
Fax: +385 (51) 686 166
www.intechopen.com

InTech China
Unit 405, Office Block, Hotel Equatorial Shanghai
No.65, Yan An Road (West), Shanghai, 200040, China
中国上海市延安西路65号上海国际贵都大饭店办公楼405单元
Phone: +86-21-62489820
Fax: +86-21-62489821

IntechOpen