CERA-CRANIUM: A Test Bed for Machine Consciousness Research

Raul Arrabales RARRABAL@INF.UC3M.ES
Agapito Ledezma LEDEZMA@INF.UC3M.ES
Araceli Sanchis MASM@INF.UC3M.ES
Carlos III University of Madrid
Avda. Universidad, 30.
28911 Leganés, Madrid, SPAIN

Abstract

This paper describes a novel framework designed as a test bed for machine consciousness cognitive models (MCCM). This MCCM experimentation framework is based on a general-purpose cognitive architecture that can be integrated in different environments and confronted with different problem domains. The definition of a generic cognitive control system for abstract agents is the root of the versatility of the presented framework. The proposed control system, which is inspired in the major cognitive theories of consciousness, provides mechanisms for both sensory data acquisition and motor action execution. Sensory and motor data is represented in the proposed architecture using different level workspaces where percepts and actions are generated thanks to the competition and collaboration of specialized processors. Additionally, this cognitive architecture provides the means to modulate perception and behavior; in other words, it offers an interface for a higher control layer to drive the way percepts and actions are generated and how they interact with each other. This mechanism permits the experimentation with virtually any high level cognitive model of consciousness. An illustrative application scenario, autonomous explorer robots, is also reviewed in this work.

Keywords: Cognitive architectures, cognitive modeling, machine consciousness.

1. Introduction

From the point of view of an Artificial Intelligence engineer, most of the existing theories of consciousness, which typically come from philosophy or psychology, do not provide a fully plausible explanation of what a conscious being is and how consciousness could be produced in a machine. Instead, they offer a more or less metaphorical description of consciousness, but not a model that can be directly implemented in computational terms. Nevertheless, cognitive theories of consciousness, like Global Workspace Theory (Baars, 1997) or Multiple Draft Model (Dennett, 1991), have some aspects in common that can be taken as a functional guideline for the design of at least a partial computational model of consciousness.

Although authors use different names or descriptions, cognitive theories of consciousness share the assumption that the unity of self produced in conscious beings has its roots in non-unitary mechanisms. More precisely, it is argued that conscious contents emerge as a result of

competition and collaboration between specialized processors (Minsky, 1988; Dennett, 1991; Hofstadter, 1995; Baars, 1997; Shanon, 2008). Different theories offer different explanations or metaphors regarding the specific way in which these processes of competition and collaboration take place; however, all theories agree on their highly adaptable and dynamic nature.

Some remarkable examples of machine consciousness implementations inspired in these sorts of theories are Shanahan's cognitive architecture (Shanaham, 2005, 2006) and LIDA (Ramamurthy et al., 2006). Given that the detailed way in which consciousness is produced is not explained by the aforementioned theories, each existing MCCM take a different approach regarding the concrete way in which perceptual and action flows are built and managed. There exists however a common denominator in relation to the underlying mechanism used to perform low-level cognitive processes: the concurrent collaboration and competition of multiple specialized processors in a shared workspace. What differs from one implementation to another is the specific technique applied to orchestrate the collaboration and competition processes. Additionally, each particular implementation is usually oriented towards specific environments and problem domains, making it difficult to compare their relative performance. It is our aim to provide a platform where different high-level cognitive approaches can be tested and compared with each other. In order to design such a test bed we have developed a generic but configurable low-level cognitive architecture based on multiple level workspaces. The proposed framework aims to provide the main functional features of a general-purpose cognitive architecture that can be used as the base of a higher level computational model of consciousness. Taking into account the description of conscious content formation described by the main cognitive theories of consciousness, mechanisms for specialized processors creation, association, combination, and competition have been implemented as well as appropriate means to regulate these processes.

Adopting a purely cognitive perspective as introduced above does not necessarily imply that phenomenal aspects of consciousness are neglected in our work. Although our efforts are specifically focused on functional features of consciousness, phenomenology is expected to be the main subject of study after all key functional aspects are successfully implemented and tested.

The details of the proposed framework are discussed as follows. Section 2 provides an overview of CERA-CRANIUM layered design and an overall description of the perception and action cognitive flows. Section 3 covers the application of the proposed framework to the particular domain of unknown environment exploration using a mobile robot. Finally, conclusions and open research issues are discussed in section 4.

2. CERA-CRANIUM Overview

In order to build an efficient framework for the development and testing of cognitive models of consciousness we have designed and implemented the following main components: CERA, a control architecture structured in layers, and CRANIUM, a tool for the creation and management of high amounts of parallel processes in shared workspaces. As we explain below, CERA uses the services provided by CRANIUM with the aim of generating a highly dynamic and adaptable perception processes orchestrated by a computational model of consciousness.

2.1 CERA

CERA (*Conscious and Emotional Reasoning Architecture*) is a layered cognitive architecture designed to implement a flexible control system for autonomous agents. Current definition of CERA is structured in four layers (see Figure 1): sensory-motor services layer, physical layer,

mission-specific layer, and core layer. As in classical robot subsumption architectures, higher layers are assigned more abstract meaning; however, the definition of layers in CERA is not directly associated to specific behaviors:

CERA sensory-motor services layer comprises a set of interfacing and communication services which implement the required access to both sensor readings and actuator commands. In order to endow an agent with a full CERA controller system, every sensor must have its corresponding sensor service; analogously, every actuator must have its corresponding motor service. These services provide the physical layer with a uniform access interface to agent's physical (or simulated) machinery.

CERA physical layer encloses agent's sensors and actuators low-level representations. Additionally, according to the nature of acquired sensory data, the physical layer performs data preparation and preprocessing. Analogous mechanisms are implemented at this level with actuator commands, making sure for instance that command parameters are within safety limits. Although sensory information binding does not take place at this level, low-level contextualization parameters, like relative positions and timestamps, are calculated and annotated in the physical layer.

CERA mission-specific layer (formerly referred to as instantiation layer) produces and manages elaborated sensory-motor content related to both agent's vital behaviors and particular missions (one mission will typically involve several goals). At this stage is when single contents acquired and preprocessed by the physical layer are combined into more complex pieces of content, which have some specific meaning related to agent's goals. The mission-specific layer can be modified independently of the other CERA layers according to assigned tasks and agent's needs for functional integrity.

CERA core layer, the highest control level in CERA, encloses a set of modules that perform higher cognitive functions. The definition and interaction between these modules can be adjusted in order to implement a particular MCCM. In some of our former works (Arrabales et al., 2006, 2007, 2008), we have identified the following core modules: attention, status assessment, preconscious management, memory management, and self-coordination. Nevertheless, CERA is designed to allow a custom definition of core modules. The objective of the mentioned modules is discussed below as well as the mechanisms they use to modulate the way CERA lower layers work.

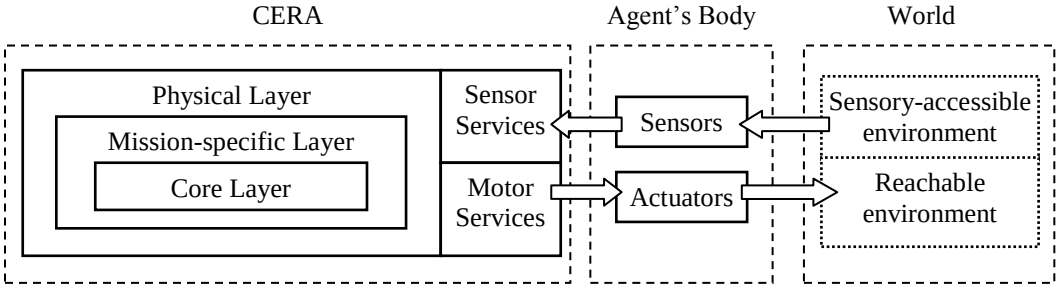

Figure 1. CERA cognitive architecture layered design.

In order to perform an experiment using the proposed test bed, four variables have to be assigned a particular value: cognitive model of consciousness to be tested, physical (or simulated) agent, assigned mission, and environment. This so-called instantiation process involves the definition of CERA core layer modules, implementation of interfaces for the particular agent's sensors and actuators (CERA sensory-motor services layer), and definition of mission-specific routines (CERA mission-specific layer). For the particular case of MCCM comparative study, environment, mission, and agent have to remain constant; therefore, the only changes required in order to test different cognitive models of consciousness have to be made within the CERA core layer.

Physical and mission-specific layers are characterized by the inspiration on cognitive theories of consciousness, where large sets of parallel processes compete and collaborate in a shared workspace in the search of a global solution. Actually, a CERA controlled agent is endowed with two hierarchically arranged workspaces which operate in coordination with the aim to find two global and interconnected solutions: one is related to perception and the other is related to action. In short, CERA has to provide an answer for the following questions continuously:

1. What must be the next content of agent's conscious perception?
2. What must be the next action to execute?

Typical robot control architectures are focused on the second question while neglecting the first one. Here we argue that a proper mechanism to answer the first question is required in order to successfully answer the second question in a human-like fashion. Anyhow, both questions have to be answered taking into account safety operation criteria and the mission assigned to the agent. Consequently, CERA is expected to find optimal answers that will eventually lead to human-like behavior. As explained below, CRANIUM is used for the implementation of the workspaces that fulfill the needs established by the CERA architecture.

2.2 CRANIUM

CRANIUM (*Cognitive Robotics Architecture Neurologically Inspired Underlying Manager*) provides a software library in which CERA can execute thousands of asynchronous but coordinated concurrent processes. In addition to the design guideline based on the main cognitive theories of consciousness, CRANIUM is also inspired by the way brain works from the systems-level point of view, where specialized regions process information coming both from the senses or from other specialized regions. According to the global access hypothesis (Baars, 2002), neural connections between specialized areas make possible the emerging global coordination.

A CRANIUM workspace can be seen as a particular implementation of a *pandemonium*, as described in (Dennett, 1991), where *demons* compete with each other for activation. Each of these *demons* or specialized processors is designed to perform a specific function on certain types of data. At any given time the level of activation of a particular processor is calculated based on a heuristic estimation of how much it can contribute to the global solution currently sought in the workspace. The concrete parameters used for this estimation are established by the CERA core layer as explained below. As a general rule, CRANIUM workspace operation is constantly modulated by commands sent from the CERA core layer.

In the proposed framework we use two separated but connected CRANIUM workspaces integrated within the CERA architecture. The lower level workspace is located in the CERA physical layer, where specialized processors are fed with data coming from CERA sensor

services. The second workspace, located in the CERA mission-specific layer, is populated with higher-level specialized processors that take as input either the information coming from the physical layer or information produced in the workspace itself (see Figure 2). The perceptual information flow is organized in packages called *single percepts*, *complex percepts*, and *mission percepts*. The details about these constructs are explained in the next subsections.

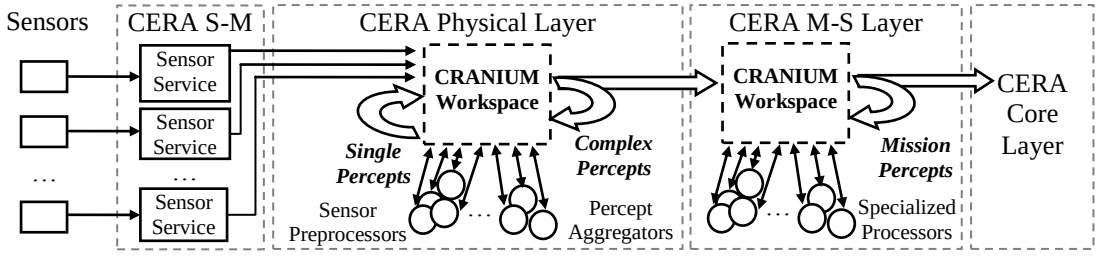

Figure 2. CERA-CRANIUM bottom-up flow: perception.

In addition to the bottom-up flow involving perception processes, a top-down flow takes place simultaneously in the same workspaces in order to generate agent's actions. Physical layer and mission-specific layer workspaces include *single actions*, *simple behaviors*, and *mission behaviors* (see Figure 3). The detailed way in which these motor representations are managed is also covered in the following subsections.

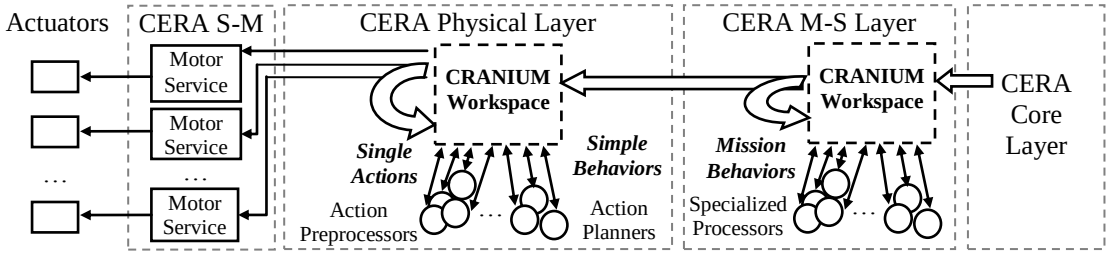

Figure 3. CERA-CRANIUM top-down flow: behavior generation.

One of the key differences between CERA-CRANIUM bottom-up and top-down flows is that while percepts are being iteratively composed in order to obtain more complex and meaningful representations, high level behaviors are iteratively decomposed until a sequence of atomic actions is obtained. As described below, there are different types of specialized processors that can be implemented and associated with CRANIUM workspaces. At first glance, it seems that specialized processors either take percepts or behaviors as input. Nevertheless, some specialized processors are designed to generate behaviors as a function of received percepts. For instance, high-priority reactive responses are rapidly generated in the physical layer, typically without any intervention from the upper layer. If a complex percept indicating a physical threat appears in the physical workspace, the specialized processor in charge of detecting this sort of threat will be activated and it will generate a reactive simple behavior as a response (see Figure 4). The obtained evasive simple behavior will be selected and the corresponding sequence of actions will be executed by the CERA sensory-motor services layer.

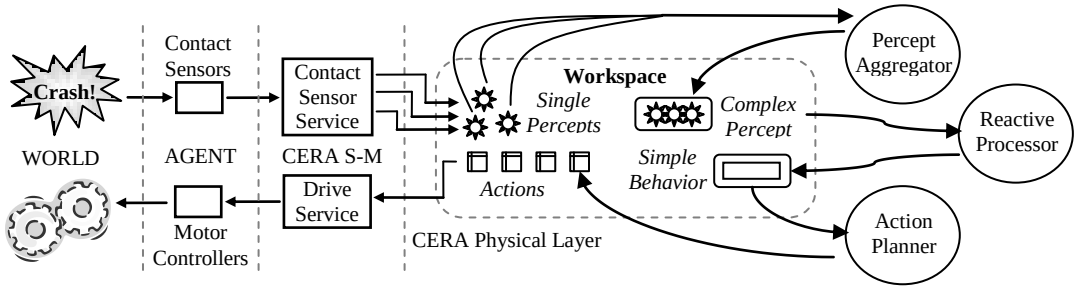

Figure 4. Simplified scheme of the reflex mechanism in a mobile robot controlled by CERA-CRANIUM.

A CRANIUM workspace provides a shared and global access working memory where data can be read or written by any associated specialized processor. In the particular case depicted in Figure 4, which has been simplified for the sake of clarity, three single percepts are generated and added to the physical workspace as a consequence of an impact detected almost simultaneously in three contiguous bump panels of a mobile robot (this particular example is studied in detail in the next section). The percept aggregator processor reads these single percepts and builds a new complex percept out of their data. The concrete set of single percepts selected in order to form the new complex percept is determined thanks to the application of multiple context criteria. Once the new complex percept is generated and published in the workspace, a reactive processor can read it and detect a condition in which a quick reactive response is required. If such a condition is met, the reactive processor is able to build a simple behavior designed to diminish or prevent the consequences of the detected undesired situation. The presence of the reactive simple behavior in the workspace triggers the activation of the action planner specialized processor, which in turn will produce the corresponding sequence of single actions.

As discussed in the former example, there are various types of specialized processors. All these processors work as asynchronous independent programs that are able to subscribe to a workspace, read certain data types from it, perform a particular processing, and then submit new elaborated data back to the workspace, where it becomes available for the rest of specialized processors. Basically, a CRANIUM workspace interacting with its associate processors constitutes a blackboard system (Nii, 1986), where CERA plays the role of the blackboard control shell. Most important CRANIUM processor types are briefly described as follows (an exhaustive list of all processor types is out of the scope of the present work):

Sensor preprocessors collect raw sensory data that appears in the physical workspace and build single percepts by combining sensor readings with contextual information that can be associated to them. The generated single percepts are automatically sent to the same workspace. In order to perform this task, sensor preprocessors also retrieve system information available in the workspace, like limbs relative positions and timing.

Action preprocessors prepare atomic actions generated by action planners (another type of processor) to enter the execution cycle. Basically, action preprocessors build the so-called single action constructs which include contextual data about actions. For instance, every single action encloses the exact timestamp corresponding to the moment it was created (planned) and also the actual timestamp assigned for execution by the CERA action dispatcher. This information is used to abort the execution of actions that have been queued for too long. Proprioceptive sensory data is also included in order to adapt actions to the current position of the actuators.

Percept aggregators are the processors in charge of building complex percepts out of interrelated single percepts. While single percepts represent atomic sensory information, complex percepts are more elaborated and meaningful combinations of the former. Multiple context criteria can be considered in the formation of complex percepts. Consequently, different types of percept aggregators will focus on different parameters for the selection of the single percepts they combine. As soon as complex percepts are generated, percept aggregators send them to the workspace, where other processors could use them for further aggregation processes.

Reactive processors are typically located in the physical layer in order to provide a quick response to stimuli that are considered harmful or highly undesired for the agent. These processors monitor the generated single and complex percepts looking for particular unsafe conditions. If such conditions are met, the processors build simple behaviors intended to mitigate the detected risk.

Action planners are processors able to take a behavior as input and generate the corresponding sequence of atomic actions that will lead to behavior completion. Thanks to action planners all active behaviors in a workspace are processed and the corresponding action sequences are submitted to be eventually executed.

Sensory Predictors monitor a particular source of sensory information incessantly, interpreting it as a continuous signal that can be predicted in the short term. When the sensory input under analysis differs significantly from the prediction, these processors build a mismatch complex percept that is placed in the corresponding workspace. Mismatch complex percepts are used to concentrate attention on unusual perceptions.

Having a shared workspace, where sensory and motor flows converge, facilitates the implementation of the multiple feedback loops required for adapted and effective behavior. The winning simple behavior is continuously confronted to new options generated in the physical layer, thus providing a mechanism for interrupting behaviors in progress as soon as they are no longer considered the best option. In general terms, the activation or inhibition of perception and behavior generation processes is modulated by CERA according to the implemented cognitive model of consciousness. Figure 5 shows a schematic representation of typical feedback loops produced in the CERA architecture. These loops are closed when the consequences of actions are perceived by the agent, triggering adaptive responses at different levels.

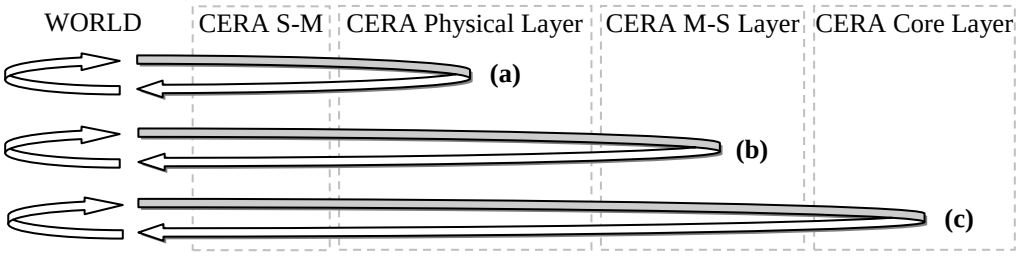

Figure 5. Different feedback loops produced in the CERA-CRANIUM.

Curve (a) in Figure 5 represents the feedback loop produced when an instinctive reflex is triggered as in the example depicted in Figure 4. Figure 5 curve (b) corresponds to a situation in which a mission-specific behavior is being performed unconsciously. Finally, curve (c)

symbolizes the higher level control loop, in which a task is being performed consciously. These three types of control loops are not mutually exclusive; in fact, same percepts will typically contribute to simultaneous loops taking place at different levels.

As explained above, the implementation of CERA-CRANIUM computational model has strong requirements in terms of concurrency and asynchronous input/output. The software architecture designed to address this issue is described in the next subsection.

2.3 Software Architecture

The concepts about CERA-CRANIUM described above refer to the cognitive-level architecture. However, building such a system also requires an underlying well designed software architecture. A good software engineering strategy will allow us to have a robust and extensible artifact that can be easily modified, enhanced and reused. Additionally, performance and scalability are factors that cannot be ignored due to the large computational demands correlated with a high number of specialized processors to be executed concurrently. Bearing these considerations in mind, as well as the requirement of a powerful physics simulator, the software development platform selected for the implementation of CERA-CRANIUM is Robotics Developer Studio 2008 (Microsoft, Corp., 2008).

The Robotics Developer Studio runtime is based on two key components: the CCR or Concurrency and Coordination Runtime (Richter, 2006), which we use for asynchronous programming and specialized processors concurrency management; and the DSS or Decentralized Software Services (Nielsen and Chrysanthakopoulos, 2006), which provide us with a framework for implementing a light-weight distributed service-oriented architecture. Applying the service orientation paradigm (Singh and Huhns, 2005), each CERA layer has been defined as an independent service that, if needed, can be executed in a separate machine. Consequently, the communication between layers is implemented using the DSS protocol (DSSP). Each CRANIUM workspace allocates at least one managed high-performance thread pool that dispatches specialized processors tasks across all available CPUs. CRANIUM thread dispatching mechanism and asynchronous I/O coordination patterns are adjusted by CERA core layer commands. Figure 6 depicts a simplified view of main software architecture components and their communication scheme (circles between modules indicate asynchronous communication implemented using CCR task ports).

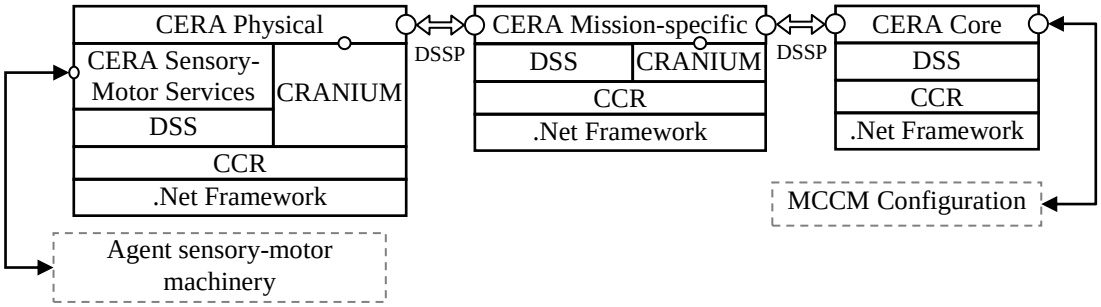

Figure 6. CERA-CRANIUM software architecture.

2.4 Knowledge Representation

Knowledge representation is one of the key problems in artificial general intelligence, and so is in the particular field of machine consciousness. Any machine consciousness model should provide a satisfactory account for world knowledge representation and symbol grounding (Haikonen, 2007). In CERA-CRANIUM, sensory and motor data is iteratively processed inside the workspaces and across layers in order to build higher level meaningful knowledge about the world. Raw sensory data coming directly from the sensors is initially processed by specific sensor preprocessors in the physical layer workspace. These preprocessors build single sensor data representations called *single percepts*. Single percepts are integrations of mono-modal sensor data packages and their associated contextualization parameters. Basically, contextualization parameters characterize the perceived stimulus in terms of relative position and time of the sensing event (see Figure 7). CERA physical layer encloses a number of modules designed to keep track of physical variables. For instance, the timer module implements a precision clock that represents the age of the agent down to a resolution of 1 millisecond. In addition, the proprioception module calculates the position of exteroceptive sensors. These parameters are used by the sensor preprocessors to calculate the relative location of the percept being acquired by the corresponding exteroceptive sensor. As shown elsewhere (Arrabales et al., 2009a), additional contextualization parameters can be established in CERA in order to obtain more accurate and selective perceptions.

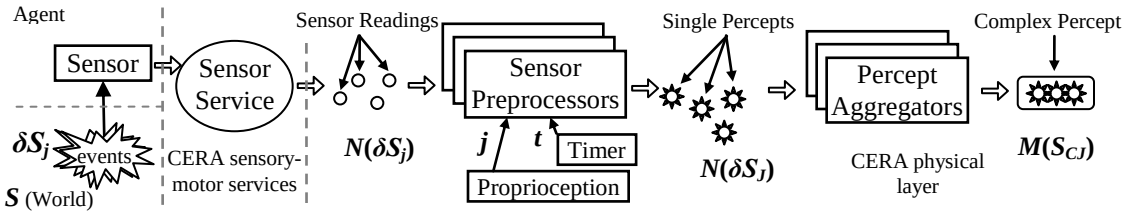

Figure 7. Single percepts generation in CERA physical layer.

Following Aleksander's notation for axioms of neuroconsciousness (Aleksander and Dunmall, 2003), where S is the sensory accessible world and δS_j represents a minimal percept. A necessary requirement for perception is that such a minimal percept must have a correlated agent's internal physical state: $N(\delta S_j)$. According to this notation, j represents the relative location, that is, the encoding of the location where the percept has been originated from the point of view of the observer organism. Consequently, $N(S)$ is the entire internal representation of the world built by the agent. In CERA physical layer, single percepts produced by sensor preprocessors include relative contextual information that we call J (J also include representations to encode relative locations called j referent vectors). In other words, single percepts can be referred to as $N(\delta S_J)$, where J is a set of contextual parameters including j (relative position) and t (timestamp). Therefore, $N(S)$ is the union of the $N(\delta S_J)$ representations produced by CERA (see Equation 1). In CERA, the representation of relative contextual parameters, J, is not encoded using neural networks, but explicit representations in the form of geometrical vectors or integer variables (as explained in the example discussed in next section).

$$N(S) = \cup_J N(\delta S_J) \tag{1}$$

All single percepts are produced within the physical layer CRANIUM workspace, where they become accessible immediately to all physical layer specialized processors. Percept aggregator processors are able to combine two or more single percepts available in the workspace and build a so-called complex percept. Complex percepts can be mono-modal or multi-modal representations depending on the modality of the single percepts they come from. Basically, they constitute more elaborated representations of the world that have been assembled as a result of the application of certain contextualization rules.

Similarly to single percepts, newly generated complex percepts are immediately published in the CERA physical workspace, and they are also sent to the mission-specific layer workspace. This means they become available to the specialized processors of both layers. Although current implementation does not include processors that combine several complex percepts into one larger complex percept, this feature could be added just by defining the corresponding specialized processors. Nevertheless, a composed *J*-index (*CJ*) is always calculated for each new complex percept. Percept aggregators combine the *J*-indexes coming from the original single percepts and build an integrated *CJ*-index that represents the contextualization parameters of the whole complex percept being formed. As single percepts are *J*-indexed, they can be grouped in terms of context criteria; for instance, a particular specialized processor might select all single percepts sensed 10 seconds ago in the left hand side of the agent and build the corresponding complex percept. According to this definition, complex percepts can be referred to as $M(S_{CJ})$, a subset of agent's internal world representation (see Equation 2).

$$M(S_{CJ}) \subset N(S) \tag{2}$$

Problems can be encountered when a percept aggregator is building a new complex percept out of contradictory single percepts. This situation can be caused by sensor noise or malfunctioning hardware. Single percepts corresponding to different sensors but associated by a common context could provide contradictory data. In that case, some strategies can be applied in order to build a meaningful complex percept that successfully integrates all the data from the original single percepts. One option is to assign levels of confidence both to the sensory data and contextual parameters obtained by sensors. Other complementary option is to generate mismatch complex percepts that will raise core layer attention towards the unexpected situation.

CERA mission-specific layer hosts a workspace in which complex percepts received from the lower layer become the input of mission-specific processors. Once more, mission-specific layer percepts are sent both to the same layer workspace and to the upper layer (CERA core layer in this case). Mission percepts could have been generated directly using one single CRANIUM workspace where single and complex percepts could be also included. However, using separated workspaces allow us to decouple physical agent specific processors and mission-specific processors. Some examples of mission-specific processors are briefly discussed in the next section.

Motor data representation in CERA-CRANIUM is analogous to the sensory data representation explained above. Atomic actions are defined as δB_I (Arrabales et al., 2007), being *I* the referent that indicates the parameters of movement (like direction and speed). $M(B_{CI})$ represents a generic behavior, which is defined as a sequence of atomic actions. The *CI* notation refers to the integrated *CI*-index, which is the final context expected to be reached when the behavior is completed. For instance, if $M(B_{CI})$ refers to a U-turn movement, *CI*-index will indicate the final position of the agent once the U-turn is completed. If the U-turn behavior is decomposed into atomic actions, a sequence of *I*-indexes corresponding to the sequence of steps required to

perform the U-turn will represent the intermediate positions and speeds. Applying the same notation as used for percepts, $N(\delta B_I)$ refers to the representation hold in CERA physical layer for agent's atomic actions. Analogously to the bottom-up processing carried out by sensor preprocessors, action preprocessors perform a top-down processing in order to build single actions out of atomic actions by including contextual information about time and relative position (see Figure 8).

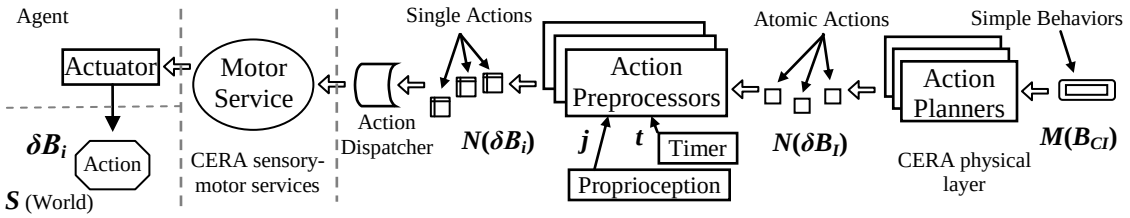

Figure 8. Actions generation in CERA physical layer.

CERA action dispatcher manages an execution queue in which active single action sequences wait for execution. Single actions constructs include some parameters that can be used by the dispatcher to decide how to manage the queue. For instance, actions derived from highest priority simple behaviors are executed in the first place.

Mission-specific behaviors are generated in the corresponding CERA layer, and then submitted to the physical layer where they are decomposed into a sequence of simple behaviors. The activation of mission-specific behaviors is driven by the application of mission goals and commands sent from the core layer. In general, the operation of CERA physical and mission-specific layers is modulated by workspace commands sent from the core layer.

2.5 Workspace Modulation

CRANIUM workspaces are not passive short-term memory mechanisms. Instead, their operation is affected by a number of workspace parameters that influence the way the pandemonium works. These parameters are set by commands sent to physical and mission-specific layers from the CERA core layer. In other words, while CRANIUM provides the neural-like mechanism for specialized functions to be combined and thus generate meaningful representations, CERA establishes a hierarchical structure and modulates the competition and collaboration processes according to the model of consciousness specified in the core layer. This mechanism closes the feedback loop between the core layer and the rest of the architecture: core layer input (perception) is shaped by its own output (workspace modulation), which in turn determines what is perceived.

All theories of consciousness differentiate between implicit and explicit processing (Atkinson et al., 2000). In CERA-CRANIUM all sensory-motor contents being processed in the workspaces are by default implicit or unconscious contents. The selection of a reduced subset of contents which will become available for explicit reasoning is carried out by the competition of both specialized processors and percepts. CERA-CRANIUM provides a mechanism to modulate these competition processes by means of commands sent from the core layer, where the cognitive model of consciousness to be tested is implemented. Specialized processors, behaviors (simple behaviors and mission-specific behaviors) and percepts (single percepts, complex percepts, and mission specific percepts) are assigned an activation level by the workspace. Activations levels are highly dynamic variables that are being constantly updated by the workspace. The role of these activation levels is twofold: on the one hand, percepts with a very low activation level are

not processed any further, thus saving the limited computational resources; on the other hand, percepts with the highest activation levels are iteratively processed until a winner is selected as conscious content. The operation of specialized processors is also affected by their activation levels. Processors with lower activation levels are less likely to be executed as they are assigned less priority in the workspace thread dispatching.

Activation levels for specialized processors and percepts are calculated as a function of multiple parameters, being contextualization criteria the most important ones. Context commands are messages sent to the workspaces specifying a *J*-index. This contextual *J*-index establishes the context criteria (like time and relative location as discussed above) that have to be activated in the workspaces. For instance, a *J*-index might refer to the particular segment of agent's visual field of view that fall between 8º and 14º. A context command sent with such a *J*-index would cause the increase of the activation level of those percepts with *J*-indexes indicating they have been sensed close to that particular position. The assigned activation level is inversely proportional to the distance between context command *J*-index and percept *CJ*-index. When a CRANIUM workspace receives a context command, activation levels of all percepts and processors inside the workspace are automatically recalculated. The distance between the specified contextual *J*-index and existing percepts *J*-indexes is calculated, and percept activations are assigned accordingly. The level of activation of processors is assigned in terms of the input they can process (activation of their potential input).

As discussed in the example presented in the next section, active contexts are typically established in the core layer taking into account agent's goals and feedback obtained from lower layers. Percept activation is also based on the match/mismatch/novelty mechanism proposed by Haikonen (2007). For instance, a mismatch percept will be initially assigned a high activation level because it might represent part of an unexpected situation that may require conscious attention. Once the mismatch signal reach the core layer, the MCCM can induce a contextual bias in the lower CERA levels by sending a context command specifying a *J*-index directed towards the unexpected percept.

The contextual bias induced by the core layer determines the percepts that are formed. Consequently, the perception process is a highly active mechanism rather than a passive data retrieving system. As behaviors are also assigned an activation level, behavior generation is also affected to a great extent by active contexts. At any given time, a number of possible behaviors are generated in the workspaces; however, only those with the highest activation levels are likely to be selected and finally executed. The calculation of activation levels for behaviors is also based on the distance between the contextual *J*-Index and behaviors *CI*-Indexes. In other words, those behaviors which are directed to the same location as the active contexts will be selected. In fact, the contextual *J*-index is not only used for selecting existing behaviors, but also to generate new behaviors directed to current focus of attention. The application of these mechanisms is illustrated in the next section with a domain-specific example. The implementation of a very basic core layer is also discussed.

3. Application in Autonomous Robot Exploration

As discussed above, our proposed framework has been designed to be used as a test bed in multiple scenarios. Typical application environments for machine consciousness research include autonomous robots and virtual agents; see for instance (Holland, 2007) and (Goertzel, 2008). In

the following we briefly introduce an instantiation of the CERA-CRANIUM framework in the domain of unknown environment exploration and mapping using autonomous mobile robots.

3.1 Experimental Setting

Preliminary experiments have been developed using real and simulated Pioneer 3DX robots equipped with front and rear bumper arrays and a ring of eight forward ultrasonic transducer sensors (see Figure 9).

As the mapping task is simplified to a two-dimensional grid, *J*-indexes are composed of *j*-referent vectors that represent relative positions using just two coordinates, being $(x,y) = (0,0)$ the subjective point of view of the robot. In this particular case of unknown environment exploration, spatial contexts have been defined in order to estimate optimal headings during robot navigation. A specific version of CERA mission-specific layer has been coded with the aim of representing the particular complex percepts that are required for the mapping mission. Concretely, sonar single percepts that represent obstacles are combined firstly into complex percepts, and secondly into mission-specific percepts that represent walls and corridors.

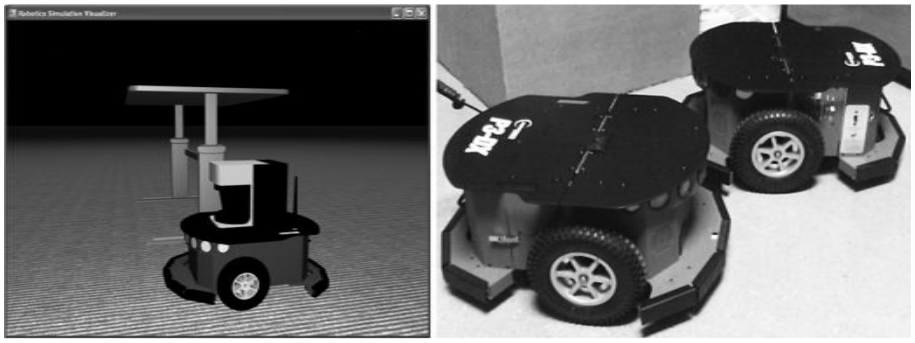

Figure 9. Simulated and real Pioneer 3DX robots.

Based on (mission-specific) internal map representations (see Figure 10), which are continuously updated with new sensed percepts, CERA core layer calculates an adaptive workspace modulation response as indicated by the implemented MCCM.

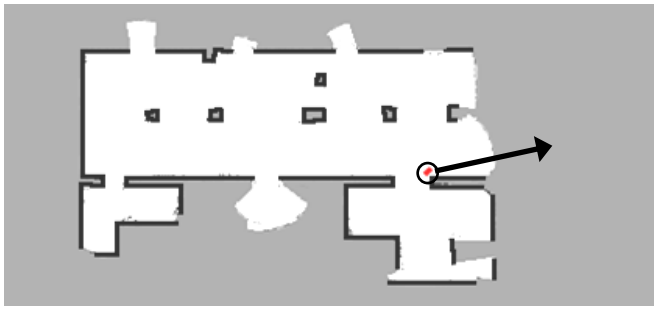

Figure 10. Mission-specific percept representing the map.

Each sensor provides different measurements of *j*-referents. As explained above, sonar range data is used for building mission percepts that represent walls or corridors. Given that sonar sensors usually provide noisy readings in real environments, contact sensors are also used for robustness

(especially when maneuvering too close to obstacles). For the sake of conciseness, in the following we only explain how single and complex percepts are built out of robot bumpers sensory information. For a detailed description of sonar percepts formation see (Arrabales et al., 2009a). Pioneer 3DX bumper arrays consist of five points of contact sensing arranged at angles around the robot (see Figure 11).

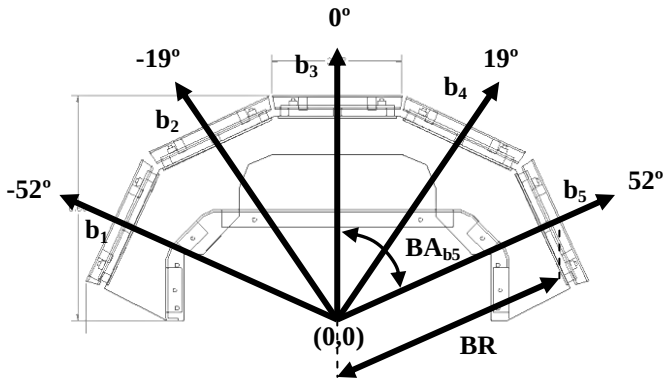

Figure 11. Pioneer 3DX frontal bumper array.

The *j*-referents of single percepts generated when the bump panels are pressed are calculated using equation 3. Where BR is the bump panel radius (or the distance from the origin of the robot spatial reference system to the bumper contact surface), and BA is the bump panel angle to the front of the robot (bump panels are located at angles -52º, -19º, 0º, 19º, and 52º).

$$j = (BR \cdot Cos(BA),\ BR \cdot Sin(BA)) \tag{3}$$

Additionally, two more vectors are calculated to be associated to a bumper single percept: the left-*j* referent and the right-*j* referent vectors (see Figure 12). These two vectors represent the dimensions of the percept (the width assigned to the collision). All these referent vectors plus the contact timestamp are used to build the *J*-index of the bumper single percept. The combination of the *J*-Index and the sensory data provided by the sensor constitutes the $N(\delta S_J)$. In the case of bumper single percepts sensory data is just the indication of a bump panel press or release. Other sensor modalities will include other types of data like range measurements or image bitmaps.

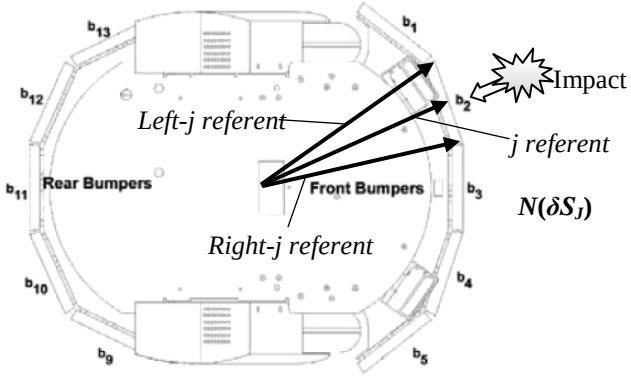

Figure 12. Referent vectors calculated to build the *J*-index of a bumper single percept.

Using time and relative location parameters as contextualization criteria single percepts can be related and associated forming complex percepts. For example, if the bumper service from the CERA sensory-motor services layer reports contacts in bump panels *b2*, *b3*, and *b4* simultaneously (see Figure 13), three single percepts are created by bumper sensor preprocessors in CERA physical layer. Then, these three single percepts can be associated by temporal and location active contexts by a percept aggregator.

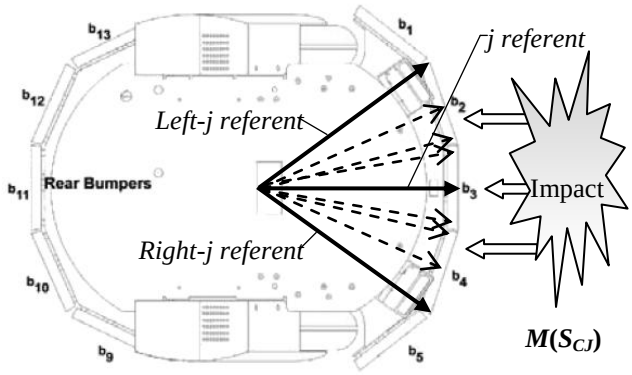

Figure 13. Calculation of the *J*-index of a bumper complex percept.

The newly created complex percept is assigned a new *CJ*-index, which is obtained as a combination of single percepts *J*-indexes (note that Figure 13 solid lines depict the complex percept *CJ*-index, while dashed lines represent the referent vectors of the old single percepts). As *b2*, *b3*, and *b4* are located side by side, the *CJ*-Index of the new bumper complex percept will have as left-*j* referent the left-*j* referent of single percept triggered by *b2*; analogously, complex percept right-*j* referent will be the same as right-*j* referent from single percept triggered by *b4*.

The way in which the *CJ*-Index of a complex percept is calculated depends on the nature (shape, dimensions, etc.) of the single percepts that take part in the context that gave place to it, and has to be calculated by the corresponding percept aggregator. The composition of *CJ*-indexes is trivial when all single percepts belong to the same modality. However, the composition can be much more complicated when different modalities are involved. See (Arrabales el al., 2009a) for details about the calculation of *CJ*-indexes for multimodal complex percepts.

Motor capabilities of the Pioneer 3DX robot are based on a two-wheel differential drive system. Robot movement control has been greatly simplified in the current setting, considering only two atomic actions: rotate in place and move straight. This means that any high level behavior will ultimately be represented in terms of sequences of these single actions. Attending to the relative direction specified by the active contextual *J*-index, an angle parameter is calculated for the rotate in place operation in order to set the robot heading towards the location that "called the robot's attention". Also a speed parameter is calculated as a function of the distance to the object.

In addition to the global exploration behavior, which is driven by the top-down contextual *J*-index, local obstacle avoidance behaviors are also implemented thanks to some mission-specific processors like *Nearest-Obstacle-Detector* and *Possible-Impact-Detector*. The first one monitors sonar percepts and creates mission percepts indicating the position of nearest obstacle; the second one reads sonar complex percepts and current active behavior in order to calculate the possibility of a collision. The output of these processors is in turn monitored by a reactive processor, which

will generate a high priority simple behavior for obstacle avoidance only when there is an obstacle close to the robot and the robot is approaching it (instead of steering away from it).

3.2 CERA Core Layer Design

Taking into account that minimizing exploration time is a requirement of the autonomous mapping mission, attention should be focused on those areas which have not been previously visited. This functionality could be implemented as some sort of attentional mechanism located in the CERA mission-specific layer. However, one of the key features of CERA-CRANIUM is that higher cognitive abilities, like attention, are not directly implemented as mission-specific procedures. In fact, higher cognitive abilities should be problem domain independent, and therefore implemented in the core layer. This means that the operation of the core layer is directed by *meta-goals* instead of mission-specific goals. Here is where the specific definition of a MCCM comes into play, because *meta-goals* are defined in terms of the particular cognitive model being implemented. For instance, discovering abstractions (detecting an invariant in a variance) is a possible definition for a mission-independent *meta-goal*.

The definition of goal types at different CERA layers can help illustrate the role of the core layer and the implemented cognitive model. Feedback loops depicted in Figure 5 can also be expressed in terms of different level goals: (a)-type loops or reflexes are the expression of physical level goals (*basic-goals*). These goals are assigned the higher priority and can trigger behaviors without the intervention of higher layers. (b)-type loops trigger behaviors directed to provide full or partial solutions to the specific problem domain (*mission-goals*). Finally, (c)-type loops are generated as a mean to achieve *meta-goals*. The precise definition of *meta-goals* and how they affect the operation of CRANIUM workspaces is defined by the cognitive model implemented in the core layer. While the available set of *mission-goals* defines the possible functionality of the robot, *meta-goals* shape the overall resulting behavior. In other words, CERA core layer provides the mean to orchestrate the operation of the whole control system using models for higher level cognitive features like emotions, attention, imagination, etc. Examples of the implementation of these features using CERA-CRANIUM are briefly described below.

According to the implemented MCCM, the CERA core layer is intended to periodically calculate a contextual *J*-index that represents current cognitive region of interest for the robot. In fact, this *J*-index represents the bias (workspace modulation) that core layer will induce in lower CERA layers. The input data processed by the cognitive model in order to calculate an adaptive workspace modulation consists of the set of mission and complex percepts received from the mission-specific layer (see Figure 14). These percepts provide information about contexts (*CJ*-indexes), activation, match/mismatch/novelty signals, *mission-goals* level of accomplishment, behaviors being currently executed, etc.

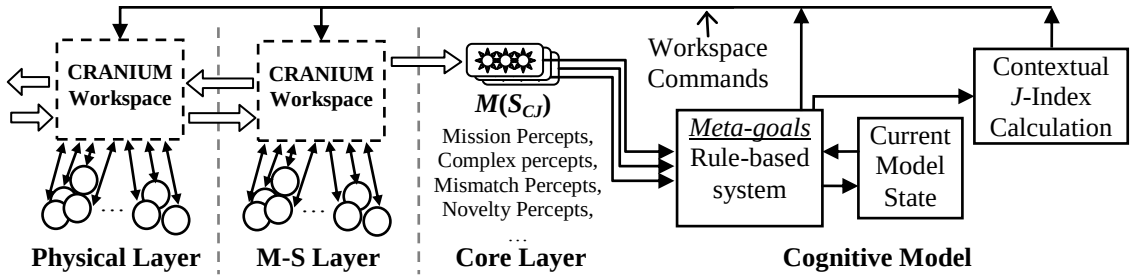

Figure 14. Generic implementation of a cognitive model in the CERA core layer.

The main task of the cognitive model implementation is to iteratively calculate an adaptive contextual *J*-Index. In the unknown environment exploration example, *J* is bi-dimensional, because only time and relative location are considered. Nevertheless, contexts could be defined using additional properties; consequently, *J*-Indexes can be defined as n-dimensional constructs.

From the point of view of the Global Workspace Theory, the core layer generates the contexts that outline the border of the metaphorical spotlight. From the point of view of the Multiple Draft Model, the core layer is in charge of selecting the winning version (reduced set of percepts) which will become the explicit content of the mind. In short, while physical and mission-specific layers provide the required environment for the composition of implicit percepts, the execution of the model implemented in the core layer induces a modulation which directs the generation and selection of explicit percepts. CERA core layer is designed to host any model able to generate a *J*-index as a function of the incoming percepts. Figure 14 represents a simplified implementation of a generic MCCM. In the illustration, a rule-based system takes incoming percepts and current state as input and generates an adaptive contextual *J*-index. Rules are expected to be defined based on the model *meta-goals*, and state can be maintained as indicated by rules. *Meta-goals* are defined as mission-independent goals, that is, they are defined exclusively in terms of MCCM parameters. For instance, if the model considers emotions, a possible *meta-goal* can be to keep a positive emotional state.

In the case of our autonomous explorer robot, a simple model has been implemented considering emotions. A reduced set of basic emotions has been considered, as well as their associated rules that contribute to the final calculation of an adaptive contextual *J*-index. For instance, *curiosity* is defined as an emotion that directs attention toward selected contents. Therefore, an incoming *CJ*-index associated with a novelty percept will trigger the curiosity rule, and contribute to direct the next contextual *J*-Index towards the novelty *CJ*-index. Novelty percepts are generated in the mission-specific layer by specialized processors able to perform scans over map mission percepts. The example illustrated in Figure 10 shows a map mission percept and also a 22.5º inclination *j* referent vector generated as a novelty percept. The application of the curiosity rule will likely produce a contextual *J*-Index in the core layer that will make the robot move in that particular direction. As the robot moves new single percepts are generated, then combined into complex percepts, map mission percepts, match/mismatch/novelty percepts, etc. Using just the selected percepts produced in the mission-specific workspace (explicit content), the MCCM implemented in the core layer will issue a new context command, thus closing the explicit control loop.

3.3 From Sensory Data to Qualia

How the proposed CERA-CRANIUM framework can account for qualia? Adopting an engineering approach, as proposed by Haikonen (2008), we believe to inspect the world through explicit percepts that appear in the system as qualia. Consequently, conscious perception is not possible without qualia. In terms of CERA, lower layers provide the required mechanisms for sensory data acquisition, processing, composition, and selection, which take place covertly. Only a winning selection of complex or mission percepts is overtly available for explicit reasoning. These explicit percepts, although grounded and adapted to the reality thanks to the CERA underlying mechanisms, do not represent the real qualities of the outside world, but an impression created by the multi-layer sensory system. In short, the input of the core layer is not built upon external stimuli, but based on lower layers reactions to these stimuli. At this level, the functional role of explicit percepts and their associated *CJ*-indexes is to provide the required information to

create the illusion that they directly represent the outside world qualities. However, as pointed out by Haikonen (2008), the qualities of generated qualia do not necessarily match with physical world properties.

CERA architecture is designed to integrate exteroceptive and proprioceptive sensing in such a way that *J*-indexes can be estimated for each percept as illustrated above. Thanks to the spatial localization properties derived from the *j* referent vectors, percepts can be processed as if they were located in the outside world. We argue that this ability to process percepts as if they were qualities of the outside world, instead of inspecting directly the sensory data, can be the base for the creation of qualia in the machine. For instance, the CERA implementation for the robot exploration task is able to build sonar single percepts out of sonar transducers range data. These percepts are representations of qualities of the outside world based on sonar sensor current operation parameters, robot position, and the actual range data (see Figure 15).

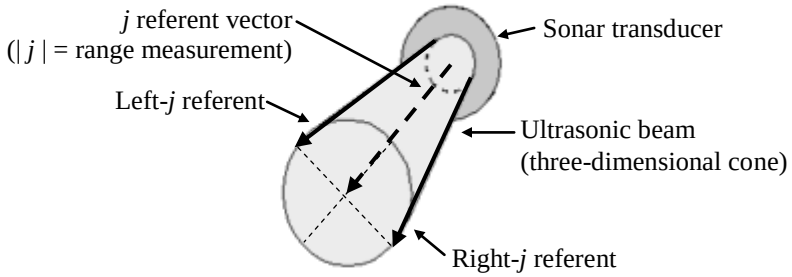

Figure 15. Single sonar percept representation.

4. Conclusions and Future Work

In this work we have presented a framework that enables the comparative study of different cognitive models of consciousness. As illustrated in the robot exploration application scenario that has been analyzed, particular implementations of CERA layers can be combined in an instantiation of the proposed framework. Having different implementations of CERA layers would enable the development of experiments like the following: evaluation of the same MCCM in different environments, evaluation of the same MCCM in the same environment but using different agents and different missions, and comparative study of different MCCMs by confronting them to the same environment and mission (this would only require modifications in CERA core layer).

The ultimate aim of these sorts of experiments is to discover what workspace modulation techniques and what cognitive models are best suited to produce conscious-like behaviors and rich adaptive perception. Furthermore, the proposed experimentation framework can be extended with classical artificial intelligence search and optimization approaches in order to either improve existing models or even generate completely new cognitive models of consciousness.

In addition to the former considerations, another benefit to take into account is the existing decoupling between the control architecture itself and the physical or simulated agent being controlled. Moving from one particular agent to another would only imply to adapt CERA sensory-motor services and physical layers to new sensors and actuators, while keeping the rest of the architecture unaffected by the change. Obviously, if new sensor modalities are added that could be specifically used in the mission-specific cognitive processing, CERA mission-specific

layer would need to be enhanced with this additional capability. The same implication applies in analogous terms for new actuators and their associated agent physical abilities.

Enhancing CERA-CRANIUM with flexible mechanisms for long-term memory and learning are the challenges we are currently facing. In order to have a noise-free and rich experimentation environment for the development of these cognitive capabilities we are currently developing an instantiation of CERA adapted for the control of video game characters (Arrabales et al. 2009b).

Acknowledgements

This research has been supported by the Spanish Ministry of Science and Innovation under CICYT grant TRA2007-67374-C02-02.

References

Aleksander, I. and Dunmall, B. 2003, "Axioms and Tests for the Presence of Minimal Consciousness in Agents", *Journal of Consciousness Studies*, vol. 10, no. 4-5.

Arrabales, R., Ledezma, A., and Sanchis, A. 2009a, "A Cognitive Approach to Multimodal Attention", *Journal of Physical Agents*, vol. 3, no. 1, pp. 53-64.

Arrabales, R., Ledezma, A., and Sanchis, A. 2009b, "Designing Human-like Video Game Synthetic Characters through Machine Consciousness". In press.

Arrabales, R. and Sanchis, A. 2008, "Applying machine consciousness models in autonomous situated agents", *Pattern Recognition Letters*, vol. 29, no. 8, pp. 1033-1038.

Arrabales, R., Ledezma, A., and Sanchis, A. 2007, "Modeling Consciousness for Autonomous Robot Exploration", *Lecture Notes in Computer Science*, vol. 4527, pp. 51-60.

Arrabales, R. and Sanchis, A. 2006, "A Machine Consciousness Approach to Autonomous Mobile Robotics", *5th International Cognitive Robotics Workshop. AAAI-06.*

Atkinson, A.P., Thomas, M.S.C., and Cleeremans, A. 2000, "Consciousness: Mapping the Theoretical Landscape" in *Trends in Cognitive Sciences*. vol. 4, no. 10, pp. 372-382.

Baars, B.J. 2002, "The conscious access hypothesis: Origins and recent evidence", *Trends in Cognitive Science*, no. 6, pp. 47-52.

Baars, B.J. 1997, "In the Theatre of Consciousness: Global Workspace Theory, A Rigorous Scientific Theory of Consciousness", *Journal of Consciousness Studies*, no. 4, pp. 292-309.

Dennett, D.C. 1991, *Consciousness Explained*, Little, Brown and Co, Boston.

Goertzel, B. 2008. "Achieving Advanced Machine Consciousness though Integrative, Virtually Embodied Artificial General Intelligence", *Proceedings of the Nokia Workshop on Machine Consciousness*. pp. 19-21.

Haikonen, P.O.A. 2007, *Robot Brains. Circuits and Systems for Conscious Machines*, John Wiley & Sons, UK.

Haikonen, P.O.A. 2008, "The Role of Qualia in Machine Consciousness; Nothing or Everything? (The Answer is YES). *Proceedings of the Nokia Workshop on Machine Consciousness*, pp. 44-45.

Hofstadter, D. 1995, "The Copycat Project: A model of mental fluidity and analogy-making" in *Fluid Concepts and Creative Analogies* Basic Books.

Holland, O. 2007, "A Strongly Embodied Approach to Machine Consciousness", *Journal of Consciousness Studies*, vol. 14, pp. 97-110.

Microsoft Corp. 2006, *Microsoft Robotics Studio. http://msdn.microsoft.com/robotics/*.

Minsky, M. 1988, *Society of Mind*, Simon & Schuster, New York.

Nielsen, H.F. and Chrysanthakopoulos, G. 2006, *Decentralized Software Services Protocol - DSSP*, Microsoft Corporation.

Nii, H.P. 1986, "Blackboard Application Systems, Blackboard Systems and a Knowledge Engineering Perspective", *AI Magazine*, vol. 7, no. 3, pp. 82-107.

Ramamurthy, U., Baars, B., D'Mello, S.K., and Franklin, S. 2006, "LIDA: A Working Model of Cognition", *Cognitive Modeling*, eds. F.D.M. Danilo Fum & Andrea Stocco, Edizioni Goliardiche, pp. 244.

Richter, J. 2006, "Concurrent Affairs: Concurrency and Coordination Runtime", *MSDN Magazine*, no. 10.

Shanaham, M. 2005, "Consciousness, Emotion, and Imagination. A Brain-Inspired Architecture for Cognitive Robotics", *AISB Workshop Next Generation Approaches to Machine Consciousness*.

Shanahan, M. 2006, "A cognitive architecture that combines internal simulation with a global workspace", *Consciousness and Cognition*, vol. 15, no. 2, pp. 433-449.

Shanon, B. 2008, "A Psychological Theory of Consciousness", *Journal of Consciousness Studies*, vol. 15, pp. 5-47.

Singh, M.P. & Huhns, M.N. 2005, *Service-Oriented Computing. Semantics, Processes, Agents*. John Wiley and Sons.