A Grounded Layered Architecture with Integrated Reasoning for Autonomous Agents

Henry Hexmoor

Johan Lammens

Stuart C. Shapiro

Computer Science Department

226 Bell Hall

State University of New York at Buffalo

Buffalo, NY 14260

tel. (716) 645-3180

Topic area: intelligent autonomous agent architecture.

November 7, 1992

A Grounded Layered Architecture with Integrated Reasoning for Autonomous Agents

Topic area: intelligent autonomous agent architecture

Recently, behavior based AI [8, 21] has questioned the need for modeling intelligent agency using generalized cognitive modules for perception and behavior generation. Behavior based AI has demonstrated successful interactions in unpredictable environments in the mobile robot domain [7, 8]. This has created a gulf between "traditional" approaches to modeling intelligent agency and behavior based approaches. We present an architecture for intelligent autonomous agents which we call GLAIR (Grounded Layered Architecture with Integrated Reasoning) [15]. GLAIR is a general multi-level architecture for autonomous cognitive agents with integrated sensory and motor capabilities. GLAIR offers an "unconscious" layer for modeling tasks that exhibit a close affinity between sensing and acting, i.e., behavior based AI modules, and a "conscious" layer for modeling tasks that exhibit delays between sensing and acting. In this paper we will describe the principles of GLAIR and systems we have developed that demonstrate how GLAIR based agents exhibit a repertoire of behaviors at different cognitive levels. We will describe two different domains for GLAIR and our implementations as of this writing.

1 Overview of GLAIR

By an *architecture* we mean an organization of components of a system, what is integral to the system, and how the various components interact.[1] Which components go into an architecture for an autonomous agent has traditionally depended to a large extent on whether we are *building a physical system*, *understanding/modeling behaviors of an anthropomorphic agent*, or *integrating a*

[1] Our discussion of architecture in this paper extends beyond any particular physical or software implementation.

select number of intelligent behaviors. The organization of an architecture is also influenced by adopting various philosophical positions like Fodor's *modularity* assumption [11], or a *connectionist* point of view, e.g. [22], or an *anti-modularity* assumption as in Brooks's subsumption architecture [7]. The *modularity* assumption supports (among other things) a division of the mind into a *central system*, i.e. cognitive processes such as learning, planning, and reasoning, and a *peripheral system*, i.e. sensory and motor processing [9]. Our architecture is characterized by a three-level organization into a Knowledge Level (KL), a Perceptuo-Motor Level (PML), and a Sensori-Actuator Level (SAL). This organization is neither modular, anti-modular, hierarchical, anti-hierarchical, nor connectionist in the conventional sense. It integrates a traditional symbol system with a physically grounded system, i.e. a *behavior-based* architecture. The most important difference with a purely behavior-based architecture like Brooks's subsumption architecture is the presence of three distinct levels with different representations and implementation mechanisms for each, particularly the presence of an explicit KL. Representation, reasoning (including planning), perception, and generation of behavior are distributed through all three levels. Our architecture is best described using a resolution pyramid metaphor as used in computer vision work [5], rather than a central vs. peripheral metaphor.

Architectures for building physical systems, e.g. robotic architectures [2], tend to address the relationship between a physical entity like a robot and sensors, effectors, and tasks to be accomplished. Since these physical systems are performance centered, they often lack general knowledge representation and reasoning techniques. These architectures tend to be primarily concerned with the *body*, that is, how to get the physical system to exhibit intelligent behavior through its physical activity. These architectures address what John Pollock calls *Quick and Inflexible* (Q&I) processes [24]. These systems are not concerned with *consciousness*, as we see it. We define consciousness for a robotic agent operationally as being aware of one's environment, as evidenced by (1) having

some internal representations that are causally connected to the environment through perception, (2) being able to reason explicitly about the environment and one's own actions, and (3) being able to communicate with an external agent about the environment and one's own actions.

Architectures for understanding/modeling behaviors of an anthropomorphic agent, e.g. cognitive architectures [3, 24, 20], tend to address the relationships that exist among the structure of memory, reasoning abilities, intelligent behavior, and mental states and experiences. These architectures often do not take the *body* into account. Instead they primarily focus on the *mind* and *consciousness*. Our architecture ranges from body-dependent physical behavior to general knowledge representation and reasoning.

We are interested in autonomous agents that are embedded[2] in a dynamic environment. Such an agent needs to continually interact with and react to its environment, and exhibit intelligent behavior through its physical activity. To be successful, the agent needs to reason about events and actions in the abstract as well as in concrete terms. This means combining situated activity with acts based on reasoning about goal-accomplishment, i.e. deliberative acting or planning.

Figure 1 schematically represents our architecture. There are several features that contribute to its robustness:

- We differentiate conscious reasoning from unconscious Perceptuo-Motor and Sensori-Actuator processing.
- The levels of our architecture are semi-autonomous and processed in parallel.
- Conscious reasoning takes place through explicit knowledge representation and reasoning. Unconscious behavior makes use of several different mechanisms, e.g. Perceptuo-Motor Automata (section 3.2).

[2] "Embedded agents are computer systems that sense and act on their environment, monitoring complex dynamic conditions and affecting the environment in goal-oriented ways." ([17] page 1).

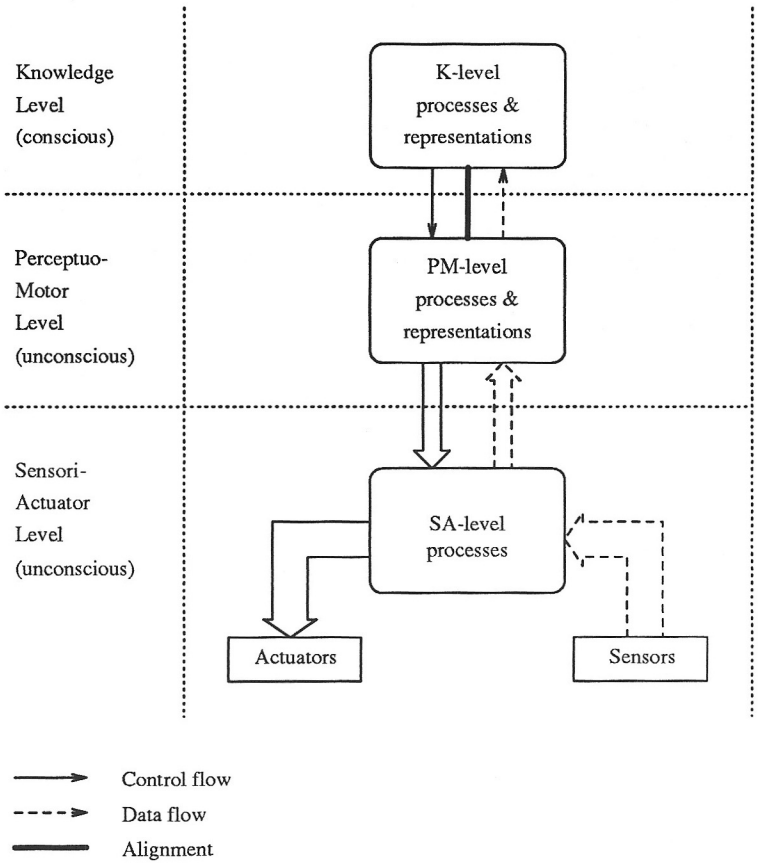

Figure 1: Schematic representation of the agent architecture. Width of control and data paths suggests the amount of information passing through (bandwidth). Sensors include both world-sensors and proprio-sensors.

- Conscious reasoning guides the unconscious behavior, and the unconscious levels, which are constantly engaged in perceptual and motor processing, can *alarm* the conscious level of important events, taking control if necessary. Control and generation of behavior are layered and not exclusively top-down.

- There is a correspondence between terms in the Knowledge Representation and Reasoning (KRR) system on one hand, and sensory perceived objects, properties, events, and states of affairs in the world and motor capabilities on the other hand. We call this correspondence *alignment.*

- The level at which any given behavior is generated and/or controlled is not fixed, but can vary in the course of learning, or depending on the particular goals and capabilities of the agent in question.

Our architecture is general (as demonstrated by the various implementations for different domains), scalable (in that high-level KL and PML mechanisms can easily be added and/or learned), taskable (aided by the KL set of concepts that allow for communication at the grain size of natural language), reactive (due to the distributed generation and control of behavior), and psychologically valid (in its distinction between conscious KL processes and unconscious PML and SAL processes).

2 Architecture Levels

2.1 Motivation

The levels of our architecture are semi-independent. While control flows mainly top-down and data mainly bottom-up, local control mechanisms at any level can preempt higher-level control, and these local mechanisms filter the data stream for their own purpose, in parallel with higher-level ones. Representations become coarser-grained from bottom to top, while control data becomes more fine-grained from top to bottom. The terms in the KL's KRR system model conscious awareness of the world (and the body), and the perception and motor capabilities in the other levels provide the grounding for an *embodied semantics* of the former. Routine activities are controlled by close coupling of perception with motor actions at the (unconscious) PML. This close coupling avoids having to exert control over these activities from the conscious level, as in purely top-down structured architectures with a symbol level at the top of the hierarchy. The low-level coupling provides for better real-time performance capabilities, and relieves the Knowledge Level of unnecessary work.

In general, we have multi-level layered representations of objects, properties, events, states of affairs, and motor capabilities, and the various levels are *aligned.* By alignment we mean a correspondence between representations of an entity at different levels. This organization contributes to the robustness and computational efficiency of implementations. The semi-autonomous nature of the levels allows for graceful degradation of system performance in case of component failure or situation-dependent incapacitatedness. Lower levels can function to some extent without higher-level control, and higher levels can function to some extent without lower-level input.

Our architecture allows us to elegantly model a wide range of behaviors: from mindless, spontaneous, reflex-like, and automatic behavior, e.g. "stop if you hit an obstacle", to plan-following, rational, incremental, and monitored behavior, e.g. "Get in the car now and go to the travel agent, if you want to go to LA on Friday".[3]

In anthropomorphic terms, we identify the KL with consciously accessible data and processing; the PML with not consciously accessible processing and data involved with motor control and perceptual processing; and the SAL with the lowest-level muscular and sensor control, also not consciously accessible. The substrate of grounding and embodiment [12, 18, 30] of actions, concepts, and reasoning is mainly the PML. Grounding is accomplished through the alignment of the Knowledge and Perceptuo-Motor Levels.

2.2 The Knowledge Level

The *Knowledge Level* (KL) comprises a traditional Knowledge Representation and Reasoning (KRR) and planning system, using a relatively course-grained representation of objects, events (including actions), and states of affairs. Representations at this level are needed only for explicit reasoning about entities, and contain only the information necessary for doing so. In most cases,

[3]This example is suggested in [25].

this is limited to a nondescript intensional representation of objects.[4] Physical details of interactions with entities are normally handled at the PML only[5]. Some entities are represented at the Knowledge Level but not at the PML (abstract concepts, for instance). KL representations are needed for reasoning about entities; PML representations are needed for physically interacting with entities.

Traditional use of the concept of world modeling refers to building models of interactions between the agent and its environment at the conscious level. These models maintain internal states for the agent. The difference in our use of the term "world model" is that we do not intend to have a precise model of all objects in the environment. Instead, we want to model only the entities relevant to the agent's interaction with its world. This requires filtering out some details accessible at the PML as the entities are aligned with their counterparts on the KL. This is also known as perceptual reduction.

All knowledge about cognitive behaviors for reasoning, planning, learning, as well as knowledge about the agent itself, is uniformly represented in the KRR system. Knowledge about the agent includes knowledge about its capabilities, i.e. primitive and complex actions. For a discussion of agent capabilities see [28]. Cognitive behaviors may produce behaviors that need to be realized by the agent in the world, but realization is in the realm of the lower levels in our architecture.

As is pointed out in [28], from the perspective of a general KRR system, external entities are extensions of concepts in the agent's mind. However, we need not maintain an extension for each entity in the agent's mind. Instead, we need to identify objects in terms of how they relate to the agent and the agent's current task.

[4] See [29] for our use of "intensional representation".

[5] The exception being explicit theories of such interactions entertained by the agent, but these need not agree with the Perceptuo-Motor Level representations

2.3 The Perceptuo-Motor Level

The *Perceptuo-Motor Level* (PML) uses a more fine-grained representation of events, objects, and states of affairs. For instance, they specify such things as size, weight, and location of objects on the kinematic side, and shape, texture, color, distance, pitch, loudness, smell, taste, weight, and tactile features on the perceptual side. At this level, enough detail must be provided to enable the precise control of actuators, and sensors or motor memory must be able to provide some or all of this detail for particular objects and situations.

The representations at the PML are *embodied* (cf. [18]). That means that the representations depend on the body of the agent, its particular dimensions and characteristics. Robots may therefore have different representations at this level than people would, and different robots may have different representations as well. These representations are agent-centered and agent-specific. For instance, they would not be in terms of grams and meters, but in terms of how much force to apply to an object to lift it, relative to the agent's capacity, or what percentage of the maximum to open the hand to grasp an object. Weights of things in this kind of representation are relative to the agent's lifting capacity, which is effectively the maximum weight representable. An agent may have a conscious (KL) understanding and representation of weights far exceeding its own lifting capacity, but that is irrelevant to the Perceptuo-Motor Level. When it comes to lifting it, a thousand Kg object is as heavy as a ten thousand Kg one, if the capacity is only a hundred. Similarly, sizes are relative to the agent's own size. Manipulating small things is not the same as manipulating large things, even if they are just scaled versions of each other. A consequence of using embodied representations is that using different "body parts" (actuators or sensors) requires different representations to be programmed or (preferably) learned. While that may be a drawback at first, once the representations are learned they make for faster processing and reactive potential. Representations are direct; there is no need to convert from an object-centered model to agent-

centered specifications. This makes the computations at this level more like table lookup than like traditional kinematics computations, which can be quite involved. Learning new representations for new objects is also much simpler; it is almost as easy as trying to grasp or manipulate an object, and merely recording one's efforts in one's own terms. The same holds, mutatis mutandis, for perceptual representations.

At the PML, an agent has a close coupling between its behaviors (responses), and significant world states (stimuli). We observe that, for a typical agent, there are a finite (manageably small) number of primitive ("innate") behaviors available. As the agent interacts with its environment, it may learn sophisticated ways of combining its behaviors and add these to its repertoire of primitive behaviors. This is called *automaticity* in psychology. We discuss an implementation mechanism for these automated behaviors in section 3.2.

2.4 The Sensori-Actuator Level

The *Sensori-Actuator Level* (SAL) is the level of primitive motor and sensory actions (as opposed to primitive *behaviors* at the PML), for instance "move from $\langle x, y, z \rangle$ to $\langle x', y', z' \rangle$ or "look at $\langle x, y, z \rangle$". At this level, there are no object representations as there are at the Knowledge Level and the Perceptuo-Motor Level. There are no explicit declarative representations of any kind, only procedural representations (on the actuator side) and sensor data (on the sensory side). Primitive motor actions may typically be implemented in a robot control language like VAL, and some elementary data processing routines may be implemented in a sensory sub-system, like dedicated vision hardware. At this level, we also situate *reflexes*, which we consider to be low-level loops from sensors to actuators, controlled by simple thresholding devices, operating independently of higher-level mechanisms. We see reflexes as primitive mechanisms whose main purpose is prevention of damage to the hardware, or to put it in anthropomorphic terms, survival of the organism. As such

they take precedence over any other behavior. When reflexes are triggered, the higher levels are made "aware" of this by the propagation of a signal, but they have no control over the reflex's execution, which is brief and simple (like a withdrawal reflex seen in people when they accidentally put their hand into a fire). After the completion of a reflex, the higher levels regain control and must decide on how to continue or discontinue the activity that was interrupted by the reflex. Reflex-like processes may also be used to attract and shift the attention of the KL.

3 Interaction between behaviors

As an example of the interaction between reflexes and other types of behavior, consider a mobile robot going down a hall. Keeping to the middle of the hall is implemented as a Perceptuo-Motor Automaton, while obstacle avoidance is a reflex. In case the robot comes too close to an obstacle, the reflex takes control, stops the robot, and changes its orientation slightly. Simultaneously a signal is sent to the PML and the KL to signify that a reflex has occurred. The Perceptuo-Motor Automaton that was in control of the robot's motion before the reflex occurred regains control after its completion. It can then choose to take the reflex into account and try to circumnavigate the obstacle, or ignore it and proceed as before. The same goes for the KL, which might decide to abort execution of the Perceptuo-Motor Automaton altogether. A good choice of reflex behavior (the orientation change in this case) may result in obstacle avoidance even if the reflex is ignored, in most cases (after repeated trial and error, somewhat like a mechanical toy with a simple sensor avoids falling off the edge of a table). Compared to Brooks's approach, the main difference is that the subsumed behavior (motion) is notified of the event, and may or may not take that into account subsequently. This is more psychologically realistic in our opinion, as well as a more flexible control strategy.

3.1 A Repertoire of Behaviors

Our architecture provides a natural framework for modeling four distinct types of behavior, which we call reflexive, reactive, situated, and deliberative. Reflexive and reactive behaviors are predominantly unconscious behaviors, whereas situated and deliberative actions are conscious behaviors.

Reflexive behavior[6] occurs when sensed data produces a response, with little or no processing of the data. A reflex is immediate. The agent has no expectations about the outcome of its reflex. The reflexive response is not generated based on a history of prior events or projections of changing events (like a gradual temperature rise). Instead, reflexive responses are generated based on spontaneous and sudden changes in the environment of the agent. In anthropomorphic terms, this is innate behavior that serves directly to protect the organism from damage in situations where there is no time for conscious thought and decision making, e.g. the withdrawal reflex when putting one's hand into a fire. Reflexive behavior does not require conscious reasoning or detailed sensory processing, so our lowest level, the SAL, is charged with producing these behaviors.

Reactive behavior requires some processing of data and results in *situated action* [30]. However, its generation is subconscious. *Situated action* refers to an action that is appropriate in the environment of the agent. In anthropomorphic terms, this is learned behavior. An example is gripping harder when one feels an object is slipping from one's fingers, or driving a car and tracking the road. We use the term *tracking* to refer to an action that requires continual adjustments, like steering while driving. Examples of this type of reactive behavior are given in [23, 4]. Reactive behaviors are situated at the PML.

Situated behavior requires assessment of the state the system finds itself in (in some state space) and acting on the basis of that. In anthropomorphic terms, this is learned behavior that requires

[6] e.g. visual reflexes in [26]: Here responses are generated to certain visual stimuli that do not require detailed spatial analysis.

only shallow reasoning. An example is the Micronesian behavior described in [30]. Situated action is used in *reactive planning* [1, 10, 27]. Reactive planners and systems of situated behaviors share the following properties:

- Applicability of only one action is decided upon at any one time.
- Applicability of an action is decided based on the current situation and not on the history of what has happened.
- No predictions are made about what will be true after completing an action.

Situated behaviors are situated partly at the KL and partly at the PML.

Deliberative behavior requires considerable processing of data and reasoning which results in action. In anthropomorphic terms, this is learned behavior that requires deep reasoning, for example explicit planning and action.

Deliberative behavior is situated entirely at the KL.

3.2 Perceptuo-Motor Automata

We now briefly discuss an implementation mechanism for behaviors at the Perceptuo-Motor Level. To recapitulate, we suggest that at this level, the behaviors resulting in physical actions specify an automaton, which we will call a PM-automaton (PMA) [16]. PMA are implementation mechanisms for routine activities at an "unconscious" level. A PMA is a finite state machine represented by <Actions, Sensations, Transitions, Internal States>. Actions are associated with states of a finite automaton. Sensations are labels of arcs in the finite automaton. Sensations are atomic (i.e., nondecomposable) inputs to PMA. Transitions are directed arcs between two states. An arc may have several sensations on it. The set of sensations on a transition is a *situation.* Arcs in a machine corresponding to a PMA are situations that the agent perceives in the environment. Since

perception is a continuous function over time, the arcs are designed to be *anytime* algorithms. In contrast to *one-shot* algorithms (the algorithm is uninterruptible once it receives its input), these algorithms improve over time. When a PMA arc emanating from a state becomes active, it behaves like an asynchronous interrupt to the act in execution in the state. This causes the PMA to stop executing the act in the state and to start executing the act at the next state at the end of the arc connecting the two states. This means that in our model, the agent is never idle, and it is always executing an act.

The action at the head of the incoming arc is said to be triggered when any one of the sensations of the arc holds in the environment, i.e., is sensed. Since sensations are not unique to a transition, when a sensation is present, all actions with transitions containing the sensation are triggered. A state receiving a trigger does not become *active* until all sensations on the incoming arc hold. Each state also contains an auxiliary part we call the Internal State (IS). The IS contains a domain dependent test such as a test on the previous state visited. Most often, the IS serves as a simple memory device as in push-down automata.

In each PMA, two distinguished states are used to correspond to "no-op" and "exit" actions to start the PMA and halt the PMA. In addition to arcs between these actions and other actions in the PMA, we provide for two arcs that reach each of the two states from outside the PMA. The origin of these arcs are unknown to the PMA. Sensations used on these arcs are beyond the scope of the PMA and they are used as hooks by outside mechanisms to control starting and halting of the PMA operation. In this way, a PMA can be interrupted when the sensations of the arc leading to "exit" hold.

The primary mode of acquiring a PMA is by converting plans in the Knowledge Level into PMAs by a process described in [16]. A PMA may become active by an intention to execute an action at the Knowledge Level. Once a PMA becomes active, sensory perception will be used by the

PMA to move along the PMA arcs (i.e. PMA transitions). The sensory perceptions that form the situations on the arcs as well as subsequent actions on the PMA may be noticed at the KL, since the same sensory data is available to it. In general, all sensory information is filtered in parallel streams.

4 Applications of GLAIR

4.1 A Simulated GLAIR Based Agent

We have written a program, Air Battle Simulation (ABS), that simulates world war I style airplane dog-fights. ABS is an interactive video-game where a human player plays against a computer driven agent. The game runs on SparcStations and starts up by displaying a game window and a control panel, see Figure 2. The human player's plane is always displayed in the center of the

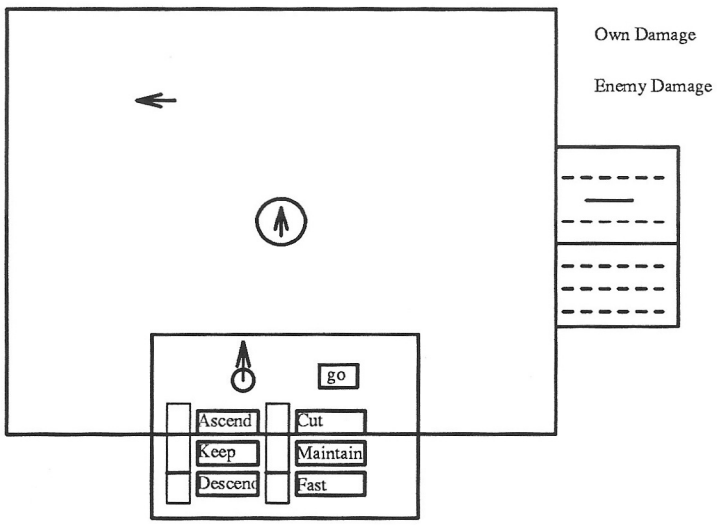

Figure 2: Air Battle Simulation game window

screen. The aerial two-dimensional position of the enemy plane is displayed on the screen with the direction of flight relative to the human player's plane. The human player looks at the game screen

to determine his airplane's position and orientation with respect to the enemy's plane. (S)he then uses the control panel to choose a move. A move is a combination of changing altitude, speed, and direction. When the human player presses the go button, the computer agent also selects a move. The game simulator then considers both moves to determine the outcome, and updates the screen and the accumulated damage to planes. ABS simulates simultaneous moves this way. If a player's plane is close in altitude and position to the enemy plane, and the enemy is in frontal sight, the latter is fired on automatically (i.e., firing is not a separate action). The levels of damage are recorded in a side panel, and the game ends when one or both of the two player's planes are destroyed.

The computer agent is developed in accordance with the principles of the GLAIR architecture. Figure 2 schematically represents the structure of the computer agent. We run the agent in one of two modes, which we refer to as the long loop and the short loop. In the long loop, the computer agent uses conscious level reasoning, i.e., SNePS rules [29], to decided which move to make. In this mode the PMA is disabled. In the short loop, the computer agent uses a PMA for deciding its next move. We are planning to integrate these two modes by having the agent use the long loop if there is no PMA. By adding learning strategies, a PMA is developed that caches moves decided at the knowledge level for future use. Once a PMA is built, most of the time only the short loop is used. Learning can be used to mark PMA moves that prove unwise and to reinforce moves that turn out to be successful. We are exploring these learning issues. [14] describes ABS and related learning techniques. The computer agent particularly demonstrates real time behaviors and the inter-level alignment mechanism.

4.2 A Robotic GLAIR Based Agent

We are developing an experimental setup in which a robot arm will set a dinner table in various configurations, guided by input from a color camera and controlled by a host computer. The physical setup includes a dinner table, a supplies table containing kitchenware, a Puma 260 robot arm, a CCD camera, a PC-based color frame grabber, and a Sun 4/260 workstation host computer. In a later phase of this project, we will replace the Puma 260 robot arm with a larger Puma 560 robot arm.

The human operator instructs the the table setting agent implemented on the host computer to set the table for dinner, breakfast, etc., specifying by color which objects to choose from the supplies table (color is not a sufficient feature to recognize objects, as each type of object is available in several colors). The camera is mounted in a fixed position overlooking both tables and the robot. This testbed provides a dynamic, yet controllable, environment in which an agent is placed so as to facilitate empirical studies of its behavior.

We call the computer agent for this project the Robot Waiter (RW). RW is being developed in accordance with the principles of the GLAIR architecture. Figure 3 schematically presents the structure of our computer agent. The categorizer uses domain constraints to determine what objects are in the visual field. It can also be told to look for a certain object, e.g., *a red cup*. The sensory memory acts as an attentional register, keeping track of the object that is being manipulated or is to be inspected.

Actions are transmitted from KL to PML, e.g., *look for a red apple* and *pick it up*. Once the sensory memory contains an object matching the object desired, actions involving that object can be understood at the PML. For instance, once a red apple is discovered and recorded in the sensory memory, an action like *pick it up* is expanded by PMAs into robot tasks to reach for the apple, grasp it, and lift it. Each task involving a robot motion is subsequently submitted to the path constructor.

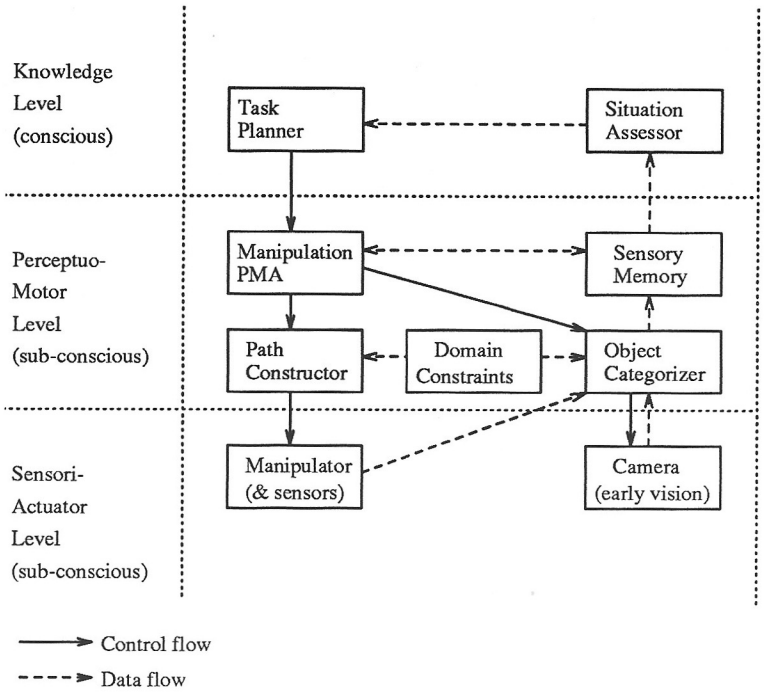

Figure 3: Schematic representation of the Robot Waiter GLAIR-agent.

Some motions may have to be decomposed to visit intermediate points in order to avoid fixtures in the environment. The path constructor generates path segments for each robot motion. Each path segment generated by the path constructor is transmitted to the SAL for execution.

RW incorporates an embodied model of color perception and color naming, modelled after the physiology of human color perception. This model allows the agent to (1) name colors shown to it, and express a typicality judgement, (2) point out examples of named colors in its environment, and (3) learn new names for colors. This model provides the perceptual grounding for a set of basic color terms [6] represented at the Knowledge Level. The color domain was chosen as a case study for embodied perception because of the relative abundance of psychological, psychophysical and neurophysiological data in the literature [19]. It is a complex enough area to allow the usefulness of embodiment for computational models of perception to be demonstrated, yet feasible enough to be implemented in actual autonomous agents.

4.3 A Mobile GLAIR Based Agent

We will implement a computer agent, Hall Rover (HR), to run around hallways, delivering packages. HR will learn the layout of hallways as it traverses them. It will learn particular routes to offices and will remember landmarks by visual inspection. HR will reflexively avoid bumping into people and objects in the hallways. This domain will serve to illustrate behaviors generated at three different levels, corresponding to GLAIR levels. We will use this platform to contrast our approach to that of Brooks's subsumption architecture [7]. As in Brooks's systems, HR will exhibit mindless, reflexive behaviors. In addition to reflexive behaviors, HR will be able to quickly choose which routes to take in visiting offices. This exhibits unconscious behavior. HR will also be able to consciously organize its delivery errands. HR will decide in which order to visit offices. We will demonstrate how naturally the physiology of a mobile robot like HR and its behaviors fit our GLAIR levels.

HR will be developed in accordance with the principles of the GLAIR architecture. Whereas the reflexive mechanisms generate rapid reflexes for avoiding obstacles, at the percptuo-motor level HR will decide when and where to to turn to visit offices. A location finder will determine where in the hall HR is located. At the knowledge level, the order of office visits is determined. The three levels will operate independently and provide information for one another. Of course, the task of delivering packages imposes some structure in the sequence of operations and is independent of how HR's levels can function. The order of office visits is decided before planning any routes. But as soon as the first office to be visited is determined, HR can start heading in the direction of the first office. Meanwhile, the knowledge level continues to determine the order of the remaining offices to be visited.

5 The LIMBS Lab

We will develop a software simulation, Limbs Lab (LL), to simulate an agent that learns to use its limbs for performing complex actions. The agent integrates sensations of position of body parts from its "nervous system" and sensations of the external world from its "eyes". LL is a graphics simulation environment where an agent (a stick-figure) is drawn on the screen and its primitive physical abilities (i.e., ranges of joint motion) are defined by the user. Subsequently, the user puts the agent in an initial configuration and gives it a task. The task can be as complex as walking or swimming or a simple as pushing a block around on a table. Goals are specified as conditions, e.g. do not sink while swimming, and accomplishments, e.g. swimming a lap or walking a distance. Unlike ABS, this system has a single player, the computer agent, and the object of the game is learning. We plan to explore embodied representations for manipulation tasks and learning that takes place at the conscious and unconscious layers.

The computer agent will be developed in accordance with the principles of the GLAIR architecture. A *Where is my limb?* module will determine the position and orientation of a limb like the hand, with respect to a position in the agent's body. This is done through proprioception. A *Where is the object?* module will determine the location and orientation of external objects and relationships among objects with respect to the agent's body. A sensory memory module maintains percepts derived from these *Where-* modules. Initially, a basic skills PMA will consist of a PMA for each joint. The Knowledge Level will also contain concepts for joints and ways an agent may move each one separately. Also at the Knowledge Level we will encode domain axioms. As the agent starts its interaction in the environment, it reasons about its actions with respect to domain axioms and tries combining its joint motions for subtasks. Actions about joint actions are submitted to the PM level where corresponding PMAs start to execute. We plan to explore three different

learning techniques for this problem [13]. These learning techniques will address (1) how conscious reasoning is replaced by PMAs, (2) how connections in a PMA become stronger, (3) how concepts about new, nonprimitive actions are learned and represented.

6 Summary and Conclusion

We have proposed an architecture that models "conscious" and "unconscious" processes for behavior generation and learning for an intelligent agent. We distinguish three levels in our architecture: Knowledge Level, Perceptuo-Motor Level, and Sensori-Actuator Level. Generation and control of behaviors is distributed over all levels, and each level has independent access to sensors (via perceptual reduction) and (to some extent) actuators. We distinguish between deliberative, reactive, and reflexive behaviors. As we move down the levels, computational and representational power is traded off for better response time and simplicity of control. GLAIR agents learn from their interactions with the environment. A video-game agent and a robotic agent are being developed that exhibit the principles used in GLAIR. The video-game agent demonstrates real time behaviors and the inter-level alignment mechanism. The robotic agent demonstrates how perception and action are integrated in the architecture. Both agents display a variety of integrated behaviors. A mobile robotic agent and a second simulated agent will be developed that illustrate interactions between behaviors, and learning.

References

[1] Agre, P. E., and Chapman, D. Pengi: An implementation of a theory of activity. In *Proceedings of AAAI-87, Seattle, Wa.* (July 1987), pp. 268–272.

[2] Albus, J., Barbera, A., and Nagel, R. Theory and practice of hierarchical control. In *23rd International IEEE Computer Society Conference* (1981), pp. 18–38.

[3] Anderson, J. R. *The Architecture of Cognition.* Cambridge: Harvard University Press, 1983.

[4] Anderson, S., Hart, D., and Cohen, P. Two ways to act. In *ACM SIGART Bulletin.* ACM publications, 1991, pp. 20–24.

[5] Ballard, D. H., and Brown, C. M. *Computer Vision.* Prentice-Hall, Englewood Cliffs, NJ, 1982.

[6] Berlin, B., and Kay, P. *Basic Color Terms: Their Universality and Evolution*, first paperback ed. University of California Press, Berkeley CA, 1991 (orig. 1969).

[7] Brooks, R. A robust layered control system for a mobile robot. Tech. Rep. 864, MIT AI Labs, MIT, 1985.

[8] Brooks, R. A. Elephants dont play chess. In *Designing Autonomous Agents*, P. Maes, Ed. MIT Press, 1990, pp. 3–15.

[9] Chapman, D. Vision, instruction, and action. Tech. Rep. Technical Report 1204, MIT Artificial Intelligence Laboratory, MIT, 1990.

[10] Firby, R. J. An investigation into reactive planning in complex domains. In *Proceedings of AAAI-87* (1987), pp. 202–206.

[11] Fodor, J. *The Modularity of Mind.* MIT Press, 1983.

[12] Harnad, S. The symbol grounding problem. *Physica D 42*, 1-3 (1990), 335–346.

[13] Hexmoor, H. Representing and learning successful routine activities. Tech. Rep. Unpublished Proposal, Dept. of Computer Science, SUNY at Buffalo, New York, 1992.

[14] HEXMOOR, H., CAICEDO, G., BIDWELL, F., AND SHAPIRO, S. Air battle simulation: An agent with conscious and unconscious layers. Tech. Rep. Forthcoming, Dept. of Computer Science, SUNY at Buffalo, New York, 1992.

[15] HEXMOOR, H., LAMMENS, J., AND SHAPIRO, S. An autonomous agent architecture for integrating perception and acting with grounded, embodied symbolic reasoning. Tech. Rep. CS-92-21, Dept. of Computer Science, SUNY at Buffalo, New York, 1992.

[16] HEXMOOR, H., AND NUTE, D. Methods for deciding *what to do next* and learning. Tech. Rep. AI-1992-01, AI Programs, The University of Georgia, Athens, Georgia, 1992. Also available from SUNY at Buffalo, CS Department TR-92-23.

[17] KAELBLING, L., AND ROSENSCHEIN, S. Action and planning in embedded agents. In *Designing Autonomous Agents*, P. Maes, Ed. MIT Press, 1990, pp. 35–48.

[18] LAKOFF, G. *Women, Fire, and Dangerous Things: What Categories Reveal about the Mind.* University of Chicago Press, Chicago, IL, 1987.

[19] LAMMENS, J. M. A computational model of color perception and color naming: a case study of symbol grounding for natural language semantics. Dissertation proposal, SUNY/Buffalo CS department, June 1992.

[20] LANGLEY, P., MCKUSICK, K., AND ALLEN, J. A design for the icarus architecture. In *ACM SIGART Bulletin*. ACM publications, 1991, pp. 104–109.

[21] MAES, P. Situated agents can have goals. In *Designing Autonomous Agents*, P. Maes, Ed. MIT Press, 1990, ch. 4, pp. 49–70.

[22] McClelland, J. L., Rumelhart, D. E., and Hinton, G. E. The appeal of parallel distributed processing. In *Parallel Distributed Processing*, D. Rumelhart, J. McClelland, and the PDP Research Group, Eds. MIT Press, Cambridge MA, 1986, ch. 1, pp. 3–44.

[23] Payton, D. An architecture for reflexive autonomous vehicle control. In *Proceedings of Robotics Automation* (1986), IEEE, pp. 1838–1845.

[24] Pollock, J. *How to Build a Person.* MIT Press, 1989.

[25] Pollock, J. New foundations for practical reasoning. In *Minds and Machines* (1992).

[26] Regan, D., and Beverly, K. Looming detectors in the human visual pathways. In *Vision Research 18.* 1978, pp. 209–212.

[27] Schoppers, M. J. Universal plans for unpredictable environments. In *Proceedings 10th IJCAI* (1987), pp. 1039–1046.

[28] Shapiro, S. C., Kumar, D., and Ali, S. S. A propositional network approach to plans and plan recognition. In *Proceedings of the AAAI-88 Workshop on Plan Recognition* (1988), Morgan Kaufmann.

[29] Shapiro, S. C., and Rapaport, W. J. Sneps considered as a fully intensional propositional semantic network. In *The knowledge frontier: essays in the representation of knowledge*, N. Cercone and G. McGalla, Eds. Springer, New York, 1987, pp. 262–315.

[30] Suchman, L. A. *Plans and Situated Actions: The Problem of Human Machine Communication.* Cambridge University Press, 1988.