International Journal on Artificial Intelligence Tools, Vol. 3 No. 3 (1994) 375–381
© World Scientific Publishing Company

MULTI-LAYERED HYBRID ARCHITECTURE TO SOLVE COMPLEX TASKS OF AN AUTONOMOUS MOBILE ROBOT

F. TIECHE, C. FACCHINETTI and H. HUGLI
Institute of Microtechnology, University of Neuchâtel,
Rue de Tivoli 28, CH-2003 Neuchâtel, Switzerland,
E-mail: {tieche,facchinetti,hugli}@imt.unine.ch

ABSTRACT

In this paper, we present the implementation of an autonomous mobile robot controller developed according to the principle of a multi-layered hybrid architecture. This architecture is composed of four layers: sensori-motor, behavioral, sequencing, and strategic. The paper describes its general structure and the function of its main elements. It further analyses the development of an example task presenting the advantages of the hybrid architecture.

Keywords: Mobile Robotics, Multi-layered Architecture, Behavior-Based Control

1 INTRODUCTION

The ability of a mobile robot to achieve reliably tasks in a real environment depends essentially on the architecture of its controller. We use a multi-layered hybrid architecture that combines the advantages of both the behavioral and the centralized architectures. This architecture distributes distinct competence levels on several layers: the top layer is responsible for symbolic planning, the intermediate layers are behavioral-based, and, finally, the bottom layer controls the robot. Our architecture extends the behavioral approach discussed in [1] to more complex tasks, by offering the possibility to define and execute the goals as sequences of simple behaviors. In order to evaluate our architecture, we choose a task where the robot has to tidy up chairs in a room, by pushing and aligning them, using sequences of simple vision-based behaviors. The architecture is realized in the form of a development environment called MANO (Mobile Autonomous robot system NOmad200). Its main features are (i) the possibility to control either a real robot or a simulated one, (ii) a set of concurrent processes implementing the various levels of the architecture, and (iii) a blackboard, handling information exchange between elements of the architecture.

2 RELATED WORK

Traditional architectures — such as centralized or hierarchical — split up the robot control in three modules responsible for: sensing, planning and acting. These architectures are convenient for high-level planning tasks. The sensing module builds a high-level representation from sensed data. Using this information, the planning module generates the robot actions executed in the acting module. Such architectures are not time-efficient and have difficulties to take into account some of the uncertainties issued from the real world. The subsumption architecture [1] separates the robot control in several layers of modules. Each module is responsible for the complete processing from sensing to control and interacts directly with the environment. These modules are organized hierarchically: the upper ones activate or deactivate the modules of the underlying levels. For real applications, it is often difficult to partition a global task in a set of elementary modules because (i) the decision element is distributed over several modules, and (ii) there is no model of the robot's world.

Both architectures share advantages and disadvantages depending on complexity of the task they have to realize. The hybrid architecture [2] [6] [7] integrates and organizes them to take advantage of each, resulting in a multi-layered hierarchical architecture. The lowest layers are organized according to the behavioral architecture. The topmost layer is a module responsible for the high-level planning based on a map representation of the world. At each level the sensor interpretation is used both for control at the same level and to feed the upper level. The layers are structured according to response time (quick reaction at lower level versus slower reaction at higher level), data abstraction (signal versus symbols), and locality of spatial information (local measures versus global map).

3 ARCHITECTURE

Our architecture is composed of four layers operating asynchronously with respect to each other: sensori-motor, behavioral, sequencing and strategic. The lowest one called sensori-motor is based on control theory and on signal processing. It is responsible for the elementary movements of the robot and processes data acquired by the sensors. The second layer is behavioral-based and controls the robot with respect to the environmental characteristics. Next, the sequencing layer implements tasks described as a sequence of behaviors. It acts by selecting the elementary behaviors that form the tasks. The strategic level has both global and symbolic knowledge of the world. It is used to define the long term strategy to reach a given goal.

The **sensori-motor layer** is characterized by fast interactions and is usually hardwired. The movements of the robot are controlled by servo loops, both for velocities and position. This layer is also used for the processing of sensor data which — at this level — are essentially local measures of the world. The sensori-motor layer interacts with the environment sending command signals to the actuators and receiving signal from the sensors. Measures of the world are send to the behavioral level.

The **behavioral layer** (Figure 1) is made of a set of concurrent behaviors, reacting with the environment. We call the closed loop formed by the world, the sensor, the behavior module, and the actuator *external behavior*. By analogy, we call the module responsible for processing of measures of the world, used to update the internal database, *internal behavior*. The set of external behaviors defines the capability of the robot to interact with its environment. Each behavior extracts specific world characteristics from the measures provided by the sensori-motor layer: we call them *sign patterns*. Each time an expected sign pattern appears, the behavior is stimulated. It then controls the robot so that the sign pattern remains present. Each behavior informs the sequencing layer on its internal state using so-called *stimuli* signals. We distinguish two kind of behaviors: (i) simple behaviors with a two-values stimulus (not stimulated and stimulated), and (ii) goal-driven behaviors that stop when a expected configuration of sign patterns appears. Their stimuli take the values: not stimulated, stimulated, satisfied, and failed.

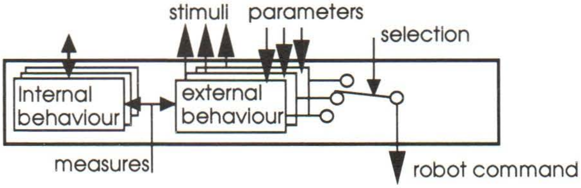

Figure 1: Behavioral layer.

While each behavior solves a small part of a robot task, the **sequencing layer** composes them to achieve a more complex one. According to a given strategy, this task parametrizes and selects one-by-one the suitable behavior, depending also on the stimuli it receives. At this level, the world representation reflects the current states of the behaviors and consists of the set of stimuli at a given time.

The aim of the **strategic layer** is to achieve a task using knowledge-based reasoning. It solves tasks such as map building, map validation, navigation. It needs a global world representation providing spatial relationship between objects. Long term goals are achieved by scheduling individual tasks according to information provided by the map and, retroactively, by the tasks.

4 DEVELOPMENT ENVIRONMENT MANO

MANO is the development environment [4] for our mobile robot. It implements the principle of our hybrid architecture (Figure 2). The core of this environment is composed of a virtual robot unit and of a blackboard handling the communication between the different layers. The four layers of the hybrid architecture are connected to these central elements. The sensori-motor layer is implemented on dedicated hardware located in the robot itself and on additional external units. The three other layers together with the blackboard and the virtual unit are distributed over a network of SUN work stations.

The virtual robot unit links the robot and the blackboard. It offers an interface with equivalent access to both the real and a simulated robot. The transition from real robot to simulated robot is possible at any time by a simple switch. In addition to the simulator, the virtual robot interface provides extended capabilities to monitor the robot, sensor data, commands, position etc.

The blackboard is the communication channel between the virtual robot, the behavioral layer and the sequencing layer. It acts as a server, using a TCP/IP connection protocol.

Clients can connect from any point of the network.

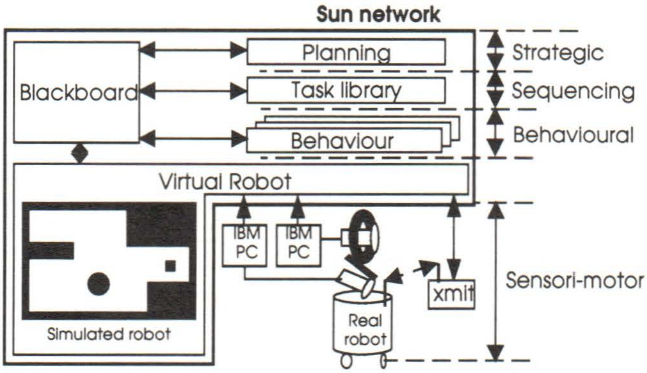

Figure 2: Development architecture MANO.

The robot Nomad 200 — from Nomadic Technologies [5]— is a one-meter-tall robot moved by a three wheel synchro-drive motion system; its upper body can be rotate around its vertical axis. It provides sensors of different types: 16 sonars, 16 infrared range-sensors and 20 tactile sensors. The communication between the robot and the virtual robot is established via a serial radio link. The **sensori-motor** layer is implemented on a number of PC-boards: the servo loops controlling the robot are on board while some vision processing is currently performed remotely.

Two active vision-based sensors have been added on top of the robot: a *vision by landmark* sensor and *laser-range* sensor [3]. The former uses a light source coupled to a video camera to enhance the contrast of reflecting landmarks distributed in the environment. The bright landmarks are detected, labeled and tracked in a dedicated Transputer system. The latter uses the principle of triangulation to measure the distance of objects in the robot environment. The specific range sensor we use applies triangulation between a plan of light and the line of sight relative to a pixel of the camera. The plane of light of the laser intersects with the environment in a profile line which geometry is finally obtained.

The **behaviors** are fully independent and run concurrently as individual UNIX processes. They are client of the blackboard server and read from it (i) the sensed data provided by the sensori-motor level and (ii) the parameters provided by the sequencing layer. The behaviors write their robot commands and their stimuli on the blackboard.

The **sequencing** tasks are implemented in form of state machines and are placed in a library. They are connected to the blackboard in order to exchange the selected behavior and the stimuli with the behavioral layer. The sequencing tasks use function parameters to exchange information with the strategic layer.

Currently, the **strategic** level is realized as a single UNIX process calling the sequencing tasks. It receives informations from the sequencing layer, and calls then the adequate tasks to reach a given goal.

5 APPLICATION: Tidying up chairs

As an example of a task implemented on MANO, we describe here *TidyUpChairs* in a room. It illustrates how a specific task is ported onto our multi-layered hybrid architecture. The robot has to detect chairs which are located arbitrarily in a room, and to push them up to a tidying up area which is a virtual line defined with respect to a fixed position of the environment. This fixed position, called *home*, is defined by two landmarks. The virtual line is parallel to the line supporting the two landmarks. This task needs a minimal world representation in form of the homing position and the virtual line position.

Figure 3 shows the *TidyUpChairs* task decomposed in a sequence of simple behaviors. First, the robot performs a wander around behavior (WA behavior) until the homing (HO) is stimulated by the two homing landmarks. Then the robot executes the homing (HO). When the homing point is reached, the current position of the robot is stored (GP) for further use. From this point the robot searches chairs by looking around (SC). If a chair is found, it goes towards the selected chair (GC). Then, the robot turns around the chair until it is positioned on the side of the chair opposite to the virtual line (AC) and pushes the chair (PC) until the line is reached. Finally it returns to the homing area (RH) and adjusts its position (HO). The task ends if no more chairs are detected (SC).

Two vision sensors are used: *vision by landmark* detects chairs marked with reflective material and homing landmarks, while *vision by structured light* detects obstacles in front of the robot. Odometry is used to move the robot to the virtual line and to bring the robot back to the homing area.

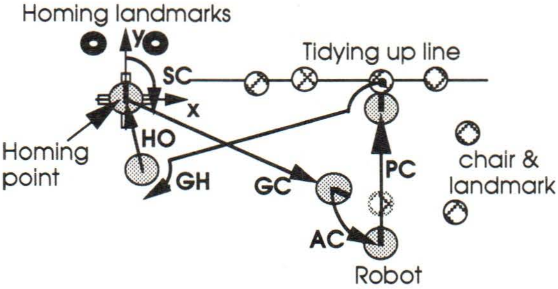

Figure 3: TidyUpChairs decomposed by behaviors.

The behaviors needed to tidy up chairs are describe below:

• Wandering around (WA): this behavior moves the robot forward and uses the ring of infrared sensor to detect a possible obstacle. If an obstacle is detected, the robot turns away from it and starts moving forward again.

• Homing (HO): based on vision by landmarks, the homing behavior brings the robot in a fixed configuration with respect to two landmarks. The behavior is stimulated as soon as two landmarks are visible and it is satisfied when the defined configuration of the landmark appears.

• Searching a chair (SC): searching a chairs is a behavior which is stimulated when a chair landmark is visible. It then turns the robot in the direction of the nearest landmark.

- Going to a chair (GC): this behavior moves the robot forward and servoes its orientation by centering the center most landmark in the image.
- Aligning on the chair (AC): this behavior moves the robots around a chair, until it is oriented perpendicular to the virtual line.
- Pushing the chair (PC): this behavior moves the robot forward. Using the odometry, it stops the robot when the tidy up line is reached.
- Returning home (RH): Using the odometry, this behavior brings the robot back to its home position.
- Getting position (GP): this internal behavior returns the current robot position.

At the sequencing layer, this task has a pre-programmed structure described by a state automata (figure 4). The circle indicates the selected behavior, and the arrows show the result of the executed behavior. The concentric circles are final states: success or failure. All the behaviors used are goal-driven. Only the *satisfied* stimulus provides a "success" output. The three other values of stimuli or a stimulus providing by an obstacle detection behavior give a "failed" output.

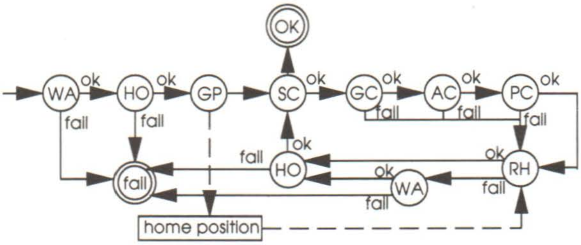

Figure 4: TidyUpChairs seen as an state automata.

The *TidyUpChairs* task performs as desired. Figure 5 shows the robot while performing the *TidyUpChairs* task. Many tests have been run with various chairs and homing landmark configurations: the programmed sequence invariably leads to the kind of path shown in figure 3 (path: HO-SC-GC-AC-PC-RH-HO).

6 CONCLUSION

In this paper we present a multi-layered hybrid architecture composed of four layers: sensori-motor, behavioral, sequencing and strategic, and describe their structure and interaction. In particular, we define the task to be performed by the robot as a state automaton responsible for the sequencing of the behavior activity. We illustrate and demonstrate this architecture in a development environment called MANO that runs on a network of workstations, a Nomad200 mobile robot and dedicated vision hardware. It also encompasses various sensing devices at the sensori-motor layer and a large set of behavior at the behavioral layer. To illustrate the architecture functionality, we present the *TidyUpChairs* task that is expressed in terms of the state automaton.

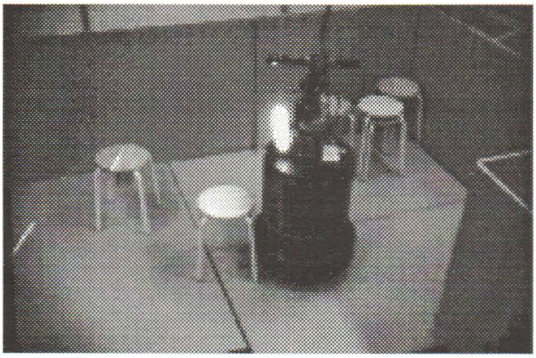

Figure 5: Robot while performing the TidyUpChairs task.

The experiment demonstrates the successful implementation of this task using this approach. It shows the advantage of this sequencing approach to describe tasks, which can hardly be expressed in a conventional behavioral architecture.

Acknowledgments

This work is bound to project 4023-027037 of the Swiss National Research Program "Artificial Intelligence and Robotics" (NRP23), conducted in collaboration with the Institute of Informatics and AI of the University of Neuchâtel, Switzerland.

References

[1] R. A. Brooks, *A Robust Layered Control System for a Mobile Robot System*, IEEE Journal of Robotics and Automation, Vol. RA-2, No 1, (1986), pp. 14-13.

[2] J. H. Connell, *SSS: A Hybrid Architecture Applied to Robot Navigation*, Proc. IEEE Int. Conf. on Robotics and Automation, Nice, France, (1992), pp. 2719-2714.

[3] H. Hügli, G. Maître, F. Tièche and C. Facchinetti, *Vision-based behaviors for robot navigation*, Proc. 4th Annual SGAICO Meeting, Neuchâtel, Switzerland, (1992).

[4] H. Hügli, F. Tièche, F. Chantemargue and G. Maître, *Architecture of an experimental vision-based robot navigation system*, Proc. Swiss Vision, Zürich, Switzerland, (1993), pp. 53-60.

[5] Nomadic, *Nomad 200 User's Guide*, Nomadic Technologies, Mountain View CA (1992)

[6] M. G. Slack, *Autonomous Navigation of Mobile Robots for Real-World Applications*, in Interdisciplinary Computer Vision, SPIE, Vol. 1838, (1992), pp. 101-109.

[7] C. E. Thorpe, *Point-CounterPoint: Big Robots vs. Small Robots*, in Interdisciplinary Computer Vision, SPIE, Vol. 1838, (1992), pp. 78-88.