A Control System Based on Reactive Skills for Autonomous Mobile Robots

M.J.L. Boada  F.J. Rodriguez  R. Barber  M.A. Salichs
Mechanical Dept.  System Engineering and Automation Dept.  System Engineering and Automation Dept.
mjboada@ing.uc3m.es  urbano@ing.uc3m.es  rbarber@ing.uc3m.es
Carlos III University
Avd. de la Universidad, 30 - 28911 - Leganés (Madrid) - SPAIN

Abstract

This paper presents a new control system based on reactive skills which allows a mobile robot to carry out its tasks safely. This control system is based on the human beings actuation capacity. The control system is formed by skills, which include the robot’s perception and action capacities, and reflex actions, which allow the robot to respond with priority to a specific stimulus. A skill can generate its own events and notify them to other skills previously registered at it in order to receive notification. In order to increase the robot's actuation capacity, it must exhibit a variety of different behaviors. For this, three different methods for generating complex skills from simple ones are also proposed.

1 Introduction

In the last years, one of the main challenges in robotics is to endow the robots with a grade of intelligence in order to allow them to extract information from the environment and use that knowledge to carry out their tasks safely. Every intelligent system must possess autonomy to be capable of operating without any help and having control over its actions and internal state. According to this idea, the robot must exhibit a variety of different behaviors [3]. Furthermore, the robot must be able to flexibly adapt its behaviors to react adequately to changes. For this, it is necessary for the robot to store its previous experiences.

Robots decision capacity to react to events in order to carry out its tasks safely depends on how its deliberation and reaction capacities are organized. Initially, the selection of the action was based on a hierarchical decomposition of functionality [2]. According to these systems, the robot has an internal representation of actions, goals and events which are used by a reasoner to predict the outcome of its actions. The main drawbacks of these systems are the lack of flexibility and responsiveness.

In order to solve these problems, behaviors-based systems were developed [8]. Instead of using world models they use direct information from the sensors. Different control systems produced different behaviors of the robot. Simple behaviors can be combined to obtain complex ones. The main advantage of these systems is the high responsiveness although they are not able to reach global goals.

Recently, the solution adopted by most researchers (Arkin [4], Firby [10], Bonasso et al. [7], Alami et al. [1]) is to combine the deliberation provided by the first systems with the reactivity provided by the last ones. These systems are called Hybrid Systems. In general, these architectures need at least two layers, one layer represents the deliberation and the other represents the reactivity. One of the main problems is how to coordinate the two layers. The architectures proposed by the authors mentioned above have an intermediate layer. In most cases, this layer is in charge of sequencing lower layer’s behaviors according to a plan provided by the higher layer. This layer has also to be capable of responding to events. Mainly, the deliberative layer of these architectures is in charge of planning but they do not have in mind another human deliberative skills such as the possibility of localizing itself or the map generation. R. Barber et al. [5] propose the architecture called AD (Automatic-Deliberative) which takes into account these capacities. One of the objectives of our research is to develop the Automatic level of this architecture. The Automatic level is formed by skills which include the robot’s perception and action capacities and by reflex actions which allow the robot to respond with priority to a specific stimulus.

2 Architecture AD

The architecture AD is inspired from the human being reasoning capacity and the actuation capacity. According to the theories of modern psychology of Shiffrin and Schneider [13], there are two mechanisms for processing information: controlled and automatic processes. These two levels of activity are related to how the reasoning capacity and the actuation capacity are distributed. Controlled processes are characterized because they require reasoning or decision capacity. Automatic processes are characterized because they require little intervention of
the attention. According to this idea, the architecture $AD$ is formed by two levels: Deliberative and Automatic level, see figure 1. The Deliberative level is associated with the reflective processes. This level is formed by deliberative skills which are each of the capacities of reasoning and learning which the autonomous system has; a long-term memory which contains information considered more stable through time; and a main sequencer which is in charge of managing the deliberative skills. The Automatic level is characterized because the actions are controlled automatically in it. That means, the Deliberative level decides what skills have to be activated or deactivated but it is not necessary conscious or do not pay attention how they are performed. Following, the Automatic level is going to be described with detail.

![Figure 1: Architecture AD](image1)

### 3 Automatic Level

The Automatic level takes over the low level control of the devices which constitute the robot. This level allows the robot to have the necessary reactivity to respond to changes quickly. Figure 2 shows the elements that form this level:

- **Automatic skills.** They are the sensorial and motor capacities of the system. The skill’s concept includes the basic and emergent behaviors’ concepts of the behaviors-based systems [11, 8].
- **Reflex actions.** They are involuntary and priority responses to a stimulus.

Automatic skills and reflex actions communicate with the robot’s hardware through virtual sensors and virtual actuators. All virtual sensors have defined the same interface, the same apply to virtual actuators. This allows to define a hardware independent architecture. The Automatic level must fulfill the following requirements:

- This level must be able to execute parallel processes, synchronize and communicate processes. These processes have to be able to react to events.
- It must allow to add new processes or modify them without affecting the rest of the system.
- It must allow to generate complex skills from current skills. A skill can be used by different complex skills. It allows to define a flexible architecture.
- It must allow the communication among processes implemented in different programming languages which can run in machines with different operating systems. In our case, the robot has two computers. One PC runs under Linux and it is dedicated to running the movement modules of the robot and the pan-tilt platform, and to execute the robot skills. The other PC runs under Windows NT and it is dedicated to images acquisition and processing.
- A skill must allow other skills to access to its processing results in order to be used as input data.

All elements, which form the Automatic level are server modules or server/client modules. Each module can contain different objects. These objects are separate units of software with their own identity, interfaces and states. The communication among objects is made through CORBA.

### 4 Virtual Sensors

Virtual sensors provide information from physical sensors for automatic skills and reflex actions. These sensors inform about the environment of the robot (eg. laser, sonars etc.) or its internal state (eg. odometry etc.). Each virtual sensor contains an active object and data objects. The active object has its own thread of control which is in charge of connecting to physical sensor and reading continuously data provided by it. Whenever a new reading is obtained, data are turned into a suitable format and stored in data objects in order to be used by other modules.

### 5 Virtual Actuators

Virtual actuators send movement commands from automatic skills or reflex actions to physical actuators. Virtual actuators allow reflex actions to have priority to send...
movement commands. When a reflex action occurs, it sends commands to the corresponding actuators inhibiting automatic skills from sending commands. When the reflex action disappears, automatic skills can send commands to actuators again.

An actuator not only receives movement commands but it can also receive commands to inform about its internal state (e.g., velocity with which it moves, position in which it stays etc.). In that case, the information provided by the actuator is stored in the data objects of its corresponding virtual actuator in order to be used by other modules.

6 Automatic Skills

Automatic skills are defined as the capacity of processing sensorial information and/or executing actions upon the robot’s actuators [6]. Skills are classified as perceptive and sensorimotor. Perceptive skills interpret the information perceived from the sensors, sensorimotor skills, or other perceptive skills. Sensorimotor skills perceive information from the sensors, perceptive skills or other sensorimotor skills and on the basis of that perform an action upon the actuators. All automatic skills have the following characteristics:

- They can be activated by skills situated in the same level or in the Deliberative level.
- A skill can only deactivate skills which it has activated previously.
- Skills have to store their results in order to be used by other skills.
- Skills can generate events. Skills only notify events to skills registered at them.
- Skills can register at other skills in order to receive notification of events. Skills have to indicate the event of which they want to receive notification.

Skills are server/client modules. They are server modules because they provide resources to other modules and they are client modules because they request resources from other modules. For example, the perceptive skill called Detect Door Laser requests data from the laser and provides the door’s position to other skills.

Figure 3 shows the generic structure of a skill. It contains an active object, an event manager object and data objects. The active object is in charge of processing. When a skill is activated, it connects to data objects of other skills or to sensors’ servers as required by the skill. Then, it processes the received input information, and finally, it stores the output results in its data objects. These objects contain different data structures depending on the type of stored data. The interfaces to access the data are similar. When the skill is sensorimotor, it can connect to actuators’ servers in order to send them movement commands. Skills which can be activated are represented with a circle such as the one shown in figure 3. There could be skills which are permanently active and in this case they are represented without a circle, see figure 2. During the processing, the active object can generate events. For example, the sensorimotor skill called Go To Goal generates the event GOAL REACHED when the required task is achieved successfully. Events are sent to the event manager object, which is in charge of notifying skills of the produced events. Only the skills that they have previously registered on it will receive notification. During the activation of the skill, some parameters can be sent to the activated skill. For instance, the skill called Go To Goal receives as parameters the goal’s position, the robot’s maximum velocity and if the skill can send velocity commands to actuators directly or not.

7 Complex Skills

Skills can be combined to obtain complex skills and these, in turn, can be recursively combined to form more complex ones. Owing to the modular characteristic of the skills, they can be used to build skills’ hierarchies with higher abstraction levels. Skills are not organized a priori; they are, rather, used depending on the task being carried out and on the state of the environment. In the present work, we proposed three different methods to generate complex skills called sequencing, output addition and data flow.

In the sequencing method the complex skill is formed by a sequencer which is in charge of deciding what skills have to be activated in each moment avoiding the simultaneous activation of other skills which may act upon the same actuator, see figure 4(a).

In the output addition method the resultant movements commands are obtained by combining the movement commands of each skill, see figure 4(b). In this case, skills act upon the same actuator and are activated at the same time. Contrary to the previous method, simple skills do not connect to actuators directly. They have to store their results in the data objects in order to be used by the complex skill. When a skill is activated it does not know if it has to send command to actuators or store its results in its data object. In order to solve this problem, one of the activation parameters sent to the skill determines if the skill has to connect to actuators or not.

In the data flow method, the complex skill is made up of skills which send information one to the other as shown in figure 4(c). The difference with the above methods is that the complex skill does not have to be responsible for
activating all skills. Simple skills activate skills of which they need their data.

The three methods are not exclusive; they can occur in the same skill. A generic complex skill must have a structure which allows its generation by one or more of the methods described above.

8 Reflex Actions

A reflex is the simplest form of animal behavior. It is an automatic and fast involuntary response triggered by a particular stimuli [14, 12]. The duration and intensity of the reflex response depends on the duration and intensity of the stimulus. Reflex actions are fast with high priority and do not involve the brain although this can receive information concerning the location and nature of the stimulating event. Once the brain is conscious of the event it can alter or inhibit the reflex action.

Each reflex action is associated with a sensor system. This means, there would be a reflex action associated with the sonars, other ones associated with the vision system, laser etc.. In most cases, a reflex action occurs when the sensor readings exceed a threshold, for example, if sonar readings indicate the robot is situated to a dangerous distance from the wall, the reflex action will make the robot move away. The escape velocity depends inversely on the distance to detected object. Depending on the perceived stimulus, the reflex action can act upon one or more actuators. For example, when a painful stimulus is applied to the human hand it causes a reflex movement of the hand. If the stimulus is stronger it can cause a reflex movement of the arm.

Figure 5 shows the generic structure of a reflex action. It contains an active object and an event manager object. The active object is in charge of checking the sensor readings. If the perceived data exceed a threshold, the active object enters the reflex state. The reflex action connects to virtual actuators in order to send movement commands inhibiting automatic skills from sending commands. At the same time, the Deliberative level receives notification from the reflex action. In that moment the Deliberative level can alter or inhibit the reflex action through the performance orders. The reflex action finishes when the reflex stimulus finishes or when the reflex action is inhibited by the Deliberative level.

9 Experimental Results

In order to test the proposed control system, different skills have been implemented on a RWI-B21 mobile robot equipped with a vision system and a laser sensor. An implemented skill is called Go To Goal Avoiding Obstacles. This skill allows the robot to be capable of going towards a given goal without colliding with any obstacle. It is a complex skill formed by a sequencer which is in charge of sequencing the following simple skills adequately depending on the received events:

-**Detect Obstacle Sonar.** It is a simple and perceptive skill. This skill estimates the distance to an obstacle from the sonars’ readings. Its output data are the minimum distance and the identification of the sensor which provides the minimum distance. Events generated by the skill are OBSTACLE IN FRONT, OBSTACLE ON THE RIGHT SIDE, OBSTACLE ON THE LEFT SIDE and NO OBSTACLE, which indicate the whereabouts of the obstacle in relation to the robot. One of the skill’s activation parameters is the distance at which the robot starts to detect the obstacle.

-**Go To Goal.** It is a simple and sensorimotor skill. This skill estimates the robot’s velocity in order to go in a straight line to the goal. The input that the skill receives is the robot position obtained from the base actuator. The parameters that the skill receives when it is activated are the goal’s position, the robot’s maximum velocity and if the skill can send velocity commands to actuators directly or not. In this case, the skill has to connect to the base actuator. If the robot reaches the goal, the skill gen-
erates the event **GOAL REACHED**.

- **Right Contour Following Sonar** and **Left Contour Following Sonar**. These skills allow the robot to follow the contour of an obstacle situated on the robot’s right hand side or left hand side. The activation parameters are: distance to the obstacle, the maximum velocity and if the skill can send velocity commands to actuators directly or not. In this case, these skills have also to connect to the base actuator.

The complex skill’s architecture is shown in figure 6. The Deliberative level decides dynamically the skills of which the complex skill is formed according to the state of the environment and the skills’ reliability to perform their tasks. The Deliberative level has to choose among skills that perform the same task. For example, in order to follow a contour the robot can choose the laser or the sonars sensor. At a given moment, the robot will use the skill called **Contour Following Sonar** and at another moment the robot will use the skill called **Contour Following Laser**. The Deliberative level communicates its decision to the complex skill through the activation parameters. If the Deliberative level does not send these activation parameters, the complex skill will choose automatically the skills to activate.

Figure 7 shows the experimental results obtained during the execution of the complex skill called **Go To Goal Avoiding Obstacles**. The robot has to go towards the goal situated at (8, 0.1) meters. When an obstacle is detected, the robot is able to avoid it and keep going to the goal once the obstacle is behind it.

The implemented complex skill allows the robot to avoid obstacles situated at a distance from it. If the robot detects an object situated at a distance less than a **safety distance**, the robot will not be able to react quickly and end up colliding with the object. In order to resolve this problem a reflex action associated to the laser sensor is implemented. The reflex action will act with priority if the robot detects an obstacle situated to a distance less than the **safety distance**. The robot performs the skill **Go To Goal Avoiding Obstacles** until an object is detected at a distance lower than the **safety distance**. In that moment, the reflex action acts with priority notifying it to the Deliberative level and inhibiting the movements commands from the skill to the base’s actuator. The reflex action makes the robot move away from the obstacle. When the obstacle disappears, the reflex action stops and the skill continues sending commands to the actuator. It could have happened that the Deliberative level had inhibited the reflex action and activated other skills which allow the robot to surround the obstacle once it had been conscious that the reflex action had happened.

In previous works other complex skills have been implemented such as **Cross Door**, **Corridor Travelling** from sequencing method [9] and **Visual Tracking** from data flow method [6] obtaining good experimental results.

10 Summary and Conclusions

This paper proposes a new control system based on reactive skills for autonomous mobile robots. The defined generic structure for skills allows the system to be modular and flexible. New skills can be added without affecting the other system’s components. One of the main differences that the architecture **AD** presents with regard to other architectures is the way of generating and administrating the generated events. So, in the architecture **3T** proposed by Bonasso et al. [7] the events are special skills which receive inputs from other skills notifying to the sequencer when the desired state has been achieved. On the other hand, in the architecture proposed by Chatila et al. [1] each module can send a report to the decisional level when its activity has finished because the task has been over or an error has occurred. In the proposed architecture **AD** each skill generates its events. The skill itself is in charge of notifying the events to skills that have previously registered on it. Another difference is the way in which the processing results are stored. In the presented architecture, each skill stores its resultant data in its own data objects. Skills that require these data connect to the corresponding objects through CORBA.
The Deliberative level decides what skills have to be active in each moment depending on the task being carried out and on the state of the environment. It can occur the robot is not able to react quickly to changes. Reflex actions allows the robot to respond fast and with priority to these changes in order to carry out its task safely.

One idea all the authors share is the necessity of generating more complex behaviors from simpler ones. In the architecture proposed by Arkin et al. [4] the resultant movement commands are obtained from the weighted sum of the motor schemas output vectors. On the other hand, in the architectures proposed by Firby [10] and Chatila et al. [1] the resultant movement commands are obtained from the behavior which is active in that moment. Unlike the previous architectures, in the architecture AD the complex skills can be generated by three different methods. Skills are not organized \textit{a priori}, they can be used dynamically by one of these methods to generate complex skills.

Finally, the paper shows the results obtained from performing the complex skill \textit{Go To Goal Avoiding Obstacles}. The robot is able to execute adequately this skill to reach its goal in a dynamic environment.

Acknowledgments

The authors gratefully acknowledge the funds provided by the Spanish Government through the MCYT projects TAP1999-214 and DPI2002-00188.

References


