A Frame for an Urban Traffic Control Architecture*

Teresa de Pedro, Ricardo García, Carlos González, Javier Alonso, Enrique Onieva, Vicente Milanés, and Joshué Prez

Instituto de Automtica Industrial-CSIC. Arganda del Rey, 28500 Madrid, España.
{tere,ricardo,gonzalez,jalonso,onieva,vmilanes,jperez}@iai.csic.es

Abstract. Due to its potential for going into details or getting a global view of the system, agent architecture is a good frame to create an urban traffic control system. In fact, the agent architecture has allowed us to design a control system able of coordinating the traffic of a set of cars in certain scenarios, using, as initial core, the car control algorithms. In further steps, a higher level layer with the decision making systems and a lower level layer with the car control actuators have been added to the agents. Finally, the agent architecture can be extended with a higher level layers to control the traffic in critical areas or urban areas.

1 Introduction

If there is no doubt that multi-robot systems are an emerging topic in the field of Robotics research, the urban traffic systems are a topic even more challenging, and they also stay in the frontier of Robotics. On the other hand the techniques of multi-robot systems, agent systems and soft computing can contribute to improve traffic efficiency and reduce the number of fatalities.

Our ideas to improve the urban traffic efficiency come from our experience in the AUTOPÍA program, in which we have developed a control architecture to allow cooperation and coordination among automated cars. Our purpose here is to expose what we think about the way in which the AUTOPIA architecture could be extended and adapted to deal with the problem of urban traffic. Until now, the AUTOPIA architecture has been distributed among modules located in cars. The extended architecture would has to add, also, modules located in the elements of the infrastructure.

Nevertheless, improving the urban traffic by making an engineering effort in car and road safety is not the only way to approach the goal of a better urban traffic. Hans Monderman [1] pioneered the concept of the "naked street" by removing all the things that were supposed to make it safe for the pedestrian - traffic lights, railings, kerbs and road markings. He thereby created a completely open and even surface -the Monderman "shared space" model on which motorists and pedestrians "negotiated" with each other by eye contact. As a car lover, he

* This work has been supported by Spanish grants TRA2008-06602-C03-C01, CENIT MARTA 20072006 and Ministerio de Fomento. P9/08, GUIADE.

J. Mira et al. (Eds.): IWINAC 2009, Part II, LNCS 5602, pp. 399–407, 2009.

claimed that the car is part of the solution and is not part of the problem. He believed that a natural interaction between the driver and the pedestrian would create a more civilised environment. But, it was not until 1992, that the first big urban application of shared space was completed at the town of Makkinga, where every trace of road signs, markings and signals was removed. In 2001 his biggest urban schemes were completed, such as the La Weiplein junction in Drachten. In 2003 a European Union research project about shared space was launched and naked streets began to appear in Austria, Belgium, Germany, Sweden, Denmark and Switzerland. The concept has spread to the USA, Canada, Russia, South Africa, Australia, Japan and Brazil.

The vision of Tony Tether, Director of the DARPA Grand Challenge [2], is quite different from Monderman's view, but it's closer to our vision. He said: "driving accidents have both a human reason and a human victim. The solution, especially for the engineers in robotics, is obvious: to replace the easily distracted, readily fatigued driver with an ever attentive, never tiring machine".

In this paper we are going to explain our technological point of view to solve the problem of urban traffic, though the bio-mimetic approach. The technologies to achieve automatic pilots are available and there are many research projects in progress. And, among them, there are the projects of the Instituto de Automtica Industrial, CSIC, bracketed under the AUTOPÍA program. The core of the AUTOPÍA program is its control architecture, a hierarchical and modular architecture distributed between vehicles and elements of the infrastructure and able of coordinating the movements of different kinds of vehicles partially or totally automated.

The rest of the paper is organized as follows: The epigraph 2 is dedicated to the general ideas of the AUTOPÍA program, the epigraph 3 is dedicated to the set of driving agents, the epigraph 4 is dedicated to the Occam as a specification language, the epigraph 5 is dedicated to the pilot, the epigraph 6 is dedicated to the decision agents -the copilot- and the epigraph 7 is dedicated to the traffic control agent. Finally we end with the conclusions.

2 AUTOPÍA Architecture: An Agents Architecture to Drive Cars

The directive idea in the AUTOPÍA program is that the serial vehicles can be driven automatically like robots by extending to cars the techniques used to control mobile robots. It is known that, among these techniques, the artificial intelligence plays a key role in robot navigation. In our case, we take the fuzzy logic from the artificial intelligence and the agent theory to organize the control architecture and to model the driver behaviour. The agent theory has been chosen because of its potential to allow a functional decomposition of the driving tasks into a hierarchical set of agents; and that decomposition is very convenient to grant a further develop of the AUTOPÍA program. The fuzzy logic has been chosen because of its potential to model the human way of driving with a simple set of rules.

According to the theory, a guiding agent can evolve from an initial core. An agent can grow up by integrating new agents, and can be broken down in simpler agents. The number and granularity of the agents can vary with the level of automation and with the complexity of the driving environment, but the system architecture can stay unchanged if it is open and modular enough to include or eliminate agents, and if it allows cooperation among them.

To understand the way in which AUTOPÍA architecture is defined, the simplest concepts of the agent theory are introduced: a) An agent behaves following the scheme: perception → action. In other words, there are two sequential states in the agent, in the first state it perceives the working area and, in the second state, the agent modifies the working area consequently. b) An agent can be split in simpler ones and several agents can be integrated in one more complex.

For an agent that is guiding a car, the scheme of behaviour is:

Driving agent = perception agent → action agent,

The perception agent can be split in a data-acquisition agent and a mapping agent. The data-acquisition agent can be split, on its turn, in several sensor agents, filtering agents -charged of cleaning the data by eliminating the false or redundant data- and map following agents.

In the same way, the action agent can be divided in a set of agents, depending on the application. In our case, to control the traffic of a set of vehicles, several kinds of agents can be identified in the action agent. For instance, one agent to take global decisions about all the vehicles involved in the traffic environment, such as it could be to establish a maximum speed, other agents can be assigned to take individual decisions about each car and finally other agents can be set to execute these decisions in each car. We can name the agent taking global decisions manager, the agents taking the individual car decisions copilots and the agents executing the manoeuvres decided by the co-pilots pilots. In order to facilitate a better understanding, we detail these agents in the inverse order that we have mentioned them.

3 An Agent to Drive Vehicles

From a theoretical point of view, the function of a driving agent in a car or in a robot is quite similar. The differences of working on cars or on robots lie in the requirements, much more committed in the first case (speeds, overpopulated environments, human passengers, etc.). The scenarios for driving cars are very dynamic and little structured, so a hybrid architecture that blends reactive modules and modules based on priority behaviour, has been designed. A reactive architecture is based on a functional and hierarchical task decomposition, in this way the agents are activated in a determined sequence, in which an agent fires the agents of lower level. An architecture based on behaviours [3] contains as many agents as potential behaviours there are in the system; all the agents are active at once, but only the agent with the highest priority takes the control at each moment.

The application task determines the design of the control architecture. The task of an automatic pilot is to control the speed and the direction of a car when

it tracks a determined trajectory while satisfying some conditions, for instance to maintain the speed under a maximal value, to maintain a security distance with the precedent car, to keep the car into the lane, etc.

As in a car there are only three control commands -the accelerator, the brake and the steering wheel- in a first approach a pilot can be divided in three actuator agents: the accelerator agent, the brake agent and the steering agent.

4 Key Words to Describe the Architecture

We have chosen the key words of the Occam [4] process constructors to be used as symbols to specify the architecture, so, before continuing, it is convenient to outline briefly the concepts of process constructors of the Occam language. To avoid confusion it has to be remarked that Occam language is not involved at all neither in the design neither in implementation of the driving agent, it is used only as specification language.

The Occam language was closely related to the "transputer" [5], a processor made up to design concurrent programs with sequences of instructions. The Occam was the first language that included the concept of parallel execution and provided tools to communicate and synchronize processes automatically, so is appropriated for specifying the architecture. The Occam constructors useful for our purposes are: The sequential, the parallel and the alternative (Figure 1).

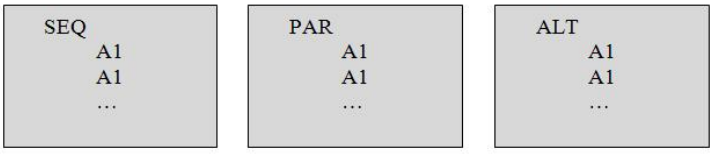

Fig. 1. Occam constructors

The sequential constructor, denoted SEQ, followed by a list of agents means that the agents are activated in the order of the list, thus a sequential process begins when the execution of first agent begins and ends only when execution of the last agent ends.

The parallel constructor, denoted PAR, followed by a list of agents means that the agents can be executed in parallel. Thus a parallel process begins when the execution of the first active agent begins and ends when the execution of the slowest agent ends:

The alternative constructor, denoted ALT, followed by a list of agents means that only one agent of the following list of agents is executed, the first active agent. Thus an alternative process begins when the execution of the first active agent begins and ends when the execution of this agent ends. This constructor allows also to assign priorities to agents and to execute the agent with the highest priority.

5 The Pilot

Though the global behaviour scheme of the pilot is sequential, there is no doubt that all kind of temporal dependencies can happen among the agents forming the pilot. For instance, the steering agent and the accelerator agent have to work in parallel, but the accelerator agent and the brake agent have to work alternatively. Other important idea, to determine and organize the agents forming the pilot, is that the human driver can be an agent in the architecture. In fact the named dual-mode cars can be driven manually or automatically. This is a key consideration because it allows to design only one architecture, and it can be used to guide autonomous cars or to assist the driver via Advanced Driving Assistant Systems (ADAS). Finally we take the human way of driving as model for the automatic pilot.

In a first approach, the scheme of the pilot (Figure 2.) contains two alternative agents: an automatic pilot and an assisted pilot, The automatic pilot contains two sequential agents, one to perceive the environment and other to act, while the assisted pilot contains two parallel agents a human driver and an ADAS agent.

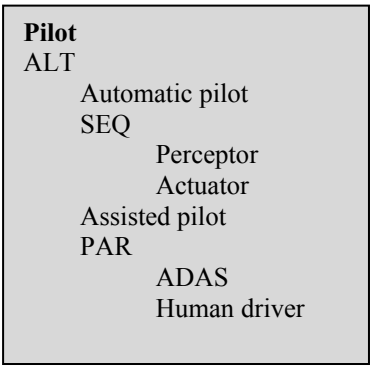

Fig. 2. Pilot scheme

According with this scheme a car can be driven by an automatic pilot or an assisted pilot. In the first case the human is not involved in the guidance, only the automatic pilot. It perceives its environment and after that, it moves the car. In the assisted pilot the driver and the ADAS work together, the driver moves the car taking into account the advertisements provided by the ADAS.

From our point of view, the automatic pilot and the ADAS are agents conceptually very similar. The only difference is that, in the ADAS, the actuator agent is substituted by an advertiser agent that provides information to the driver. But this information is the same that the automatic pilot needs to move the car. Taking this into account, the assisted pilot will not be developed in this paper further.

We have said already that the components of the perception agent are very dependent on the application, a lot of sensors and filtering algorithms can be used. So, to complete the scheme of the automatic pilot, we explain only the scheme of the actuator agent (Figure 3).

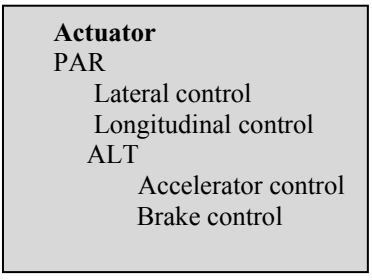

Fig. 3. Actuator scheme

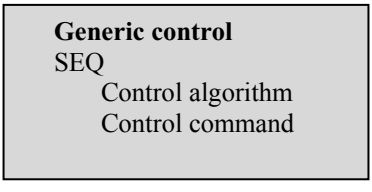

Fig. 4. Generic control scheme

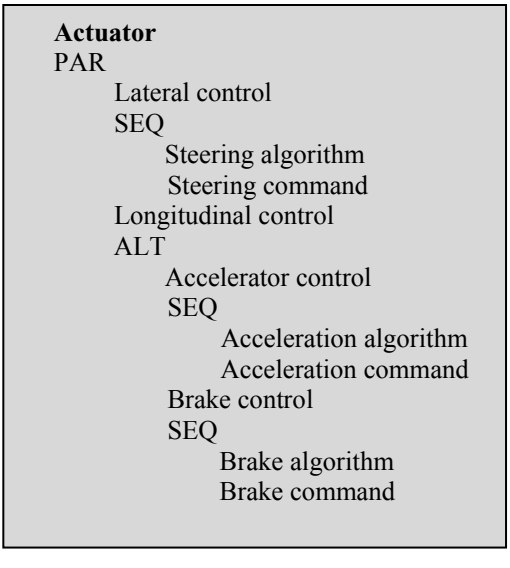

Fig. 5. Detailed actuator scheme

On the other hand, we can split both, the lateral [6] and the longitudinal [7] control, in two agents. One one of them to calculate the control values and the other to act on the commands -steering-wheel, accelerator and brake- according to these values. So a general scheme for a generic control is:

And breaking down in their components the lateral -or steering- control and the accelerator and the brake control, the actuator agent of the automatic pilot can be schematized as in the scheme of the figure 5.

To conclude with the actuator module of the automatic pilot we summarise its behaviour. In the actuator there are two agents working in concurrence, the lateral -or steering- control and the longitudinal -or speed- control. The lateral control is formed by two sequential agents, one to determine the value of the direction angle and other to turn the steering-wheel to adjust this angle. The longitudinal control is formed by two alternative agents, to control the accelerator and the brake pedal respectively. Each of these lower level controls is formed by two agents, the first to determine the acceleration or the braking value and the second to press the corresponding pedal.

6 The Co-Pilot

As we have already mentioned, the mission of the copilot agent is to take individual car decisions in a traffic environment. In this way, the pilot executes the instructions that its copilot sent to it. For instance, the copilot, based in the data acquired or received from the environment knows the traffic scenario and decides whether it has to follow the precedent car or to overtake it. Once the copilot decides what to do, it fires the suitable kind of lateral and longitudinal controllers. (In fact the pilot has different controllers for different situations, i.e. a CC to maintain

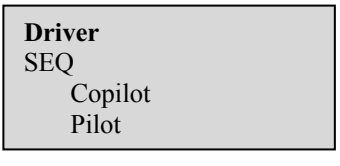

Fig. 6. Driving agent scheme

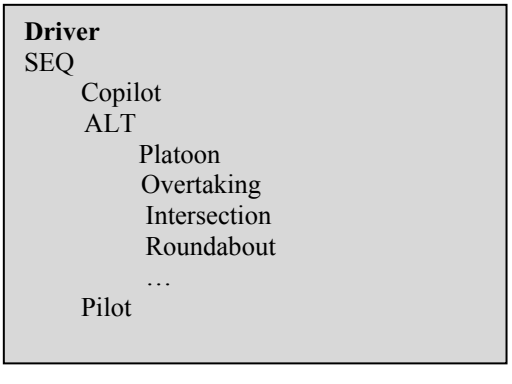

Fig. 7. Detailed driving agent scheme

the speed or an ACC to maintain a safety between two consecutive cars). Thus an agent to drive a car can be schematized as in the figure 6.

Besides, the copilot has to be split in a lot of agents for dealing with different scenarios like intersections, platoons, overtakings, roundabouts, etc. All these agents have to be active but one will be executed each time. So we can extend the scheme showed in the figure 6 like the figure 7 shows.

7 Traffic Control

It is out of the scope of this paper to detail each of these agents. They deal with complex cooperative manoeuvres and are very dependent of the application domain. What we can say is that we have implemented agents to control some cases of platoons, overtakings and intersections. In general the copilot decisions are "which" and "when" to fire the active agents of the pilot. To fix ideas, if an overtaking decision has been taken, the instructions sent to the pilot related to the lateral control are the moments in which: 1) the controller to move to the left lane fires, 2) the controller to keep the lane fires and 3) the controller to move to the right lane fires.

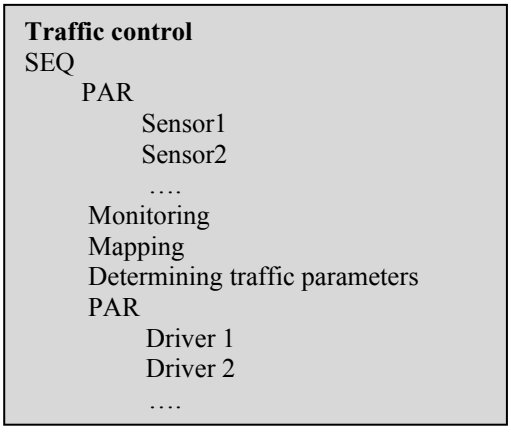

Fig. 8. Traffic control scheme

Finally the architecture would have to be completed with a traffic control agent. This will be the subject of future works. What we can say is that the traffic control agent will be the highest level agent and it will be decomposed according to the scheme of the figure 8.

8 Conclusions

From our point of view, the agent architecture has the capacity of folding and extending the forming agents. This capacity is a useful tool for design and develops progressively a traffic control system. In the case of the AUTOPÍA program

we have verified it. In fact, the AUTOPÍA architecture has evolved from a core formed by individual car controllers -lateral and longitudinal-. Now this first version has been extended including decision agents and actuator agents. The decision agents are specific for each type of manoeuvre but can be the same for all vehicles. By the other hand the actuator agents are different for each vehicle.

With the designed agent architecture we achieved in first step a full control of cars, in later steps we have achieved to control several vehicles doing cooperative manoeuvres like following a car, maintaining a determined clearance distance, overtaking a car, with or without traffic in the opposite direction, or coordinate the movements of cars in intersections. Finally we think that this architecture will allow us to do more complex functions as to control the traffic in local areas.

References

1. Monderman, H., Clarke, E., Baillie, B.H.: Shared Space -: the alternative approach to calming traffic. Traffic engineering & control 47(8), 290–292 (2006)
2. Journal of Field Robotics, special DARPA Urban Challenge (August, September 2008)
3. Arkin, R.C.: Integrating Behavioral, Perceptual and world Knowledge in Reactive Navigation. Robotics and Autonomous Systems 6, 105–122 (1990)
4. Hoare, C.A.R.: Communicating Sequential Processes. Prentice Hall International, Englewood Cliffs (1985)
5. Inmos Reference Manual: transputer
6. Rosa, R.G., de Pedro, T., Eugenio Naranjo, J., Reviejo, J., y Javier Revuelto, C.G.: Fuzzy Logic Based Lateral Control for Gps Map Tracking. In: IEEE Intelligent Vehicles Symposium, pp. 397–400 (2004)
7. García, R., de Pedro, T., Eugenio Naranjo, J., Reviejo, J., González, C.: Frontal and Lateral Control For Unmanned Vehicles In Urban Tracks. IEEE Intelligent Vehicles (2002)