SIMULATING THE USE OF AUTONOMOUS INTELLIGENT AGENTS ON CELLULAR MANUFACTURING PLANT FLOORS

Alcino Cunha **Carlos Biscaia** **Mário Torres**
Luis Sobral **Orlando Belo**

Departamento de Informática
Universidade do Minho
Largo do Paço
4709 Braga Codex
PORTUGAL

KEYWORDS

Artificial Intelligence in Simulation, Transportation, Pattern Recognition, Computer-aided Analysis and Discrete Simulation.

ABSTRACT

This paper presents the design and the implementation of a software system that allows the simulation of an *Automatic Guided Vehicle* (AGV) behaviour in a *Cellular Manufacturing* plant floor. The combination of *Multi-Agent Systems* technology, *Image Processing and Recognition*, and *Neural Networks* provides the means for the development of the main AGVs unit components in a single multi-agent system architecture. The AGV can be placed in a strange plant floor environment in order to explore and recognise its structures and specificities. The simulation platform emulates the AGV's behaviour in a virtual replica of the environment. This simulation process allows the validation and tuning of the AGV's behaviour and assists the decision making process related with plant floors' changes.

INTRODUCTION

In the last years the use of *Cellular Manufacturing* (CM) in the design of industrial plant floors as become widely accepted and one of the most implemented (Akhun and Durmusoglu 1994). The main reasons to this development are due to the general flexibility presented by CM. The flexibility of the work force, the modularity aspects provided by cell's structures and the possibility to improve maintenance services and production materials distribution, are some of the advantages that one can obtain through the implementation of a CM philosophy.

Very often, the migration of the materials among cells are performed by workers located on the production cells. Additionally, in some cases, they are also responsible to request to other cells production items (raw materials, components or even final products) that they need to accomplish their tasks and deliver their outputs in other cells . AGVs' facilities can be used to implement efficiently such kind of tasks. Moreover, AGV's abilities can be very useful and effective in stock areas management, helping workers on warehouse's space optimisation.

Whenever a new plant floor is designed or the configuration of existing production cells is changed, it is necessary, in most of the cases, to reprogram the AGVs units. Such operation requires to much time and money consuming, and surely it is not desired due to the directly effects in the global production costs. To overcome this problem, some research has been done in order to design and develop a generic AGV unit with self-adaptation abilities to unknown or changing production environments. The combination of *Multi-Agent Systems* technology (Wooldridge and Jennings 1995; Craig 1994), *Image Processing and Recognition* (Pitas 1993), and *Neural Networks* (Freeman and Skapura 1991) provided the means for the development of the main AGVs unit components in a single multi-agent system architecture.

The complexity of such adaptable behaviour and the costs of developing a real prototype requires extensive validation. This process can be achieved with the use of a simulation platform that emulates an AGV's behaviour in a virtual replica of a CM plant floor environment. Even when an AGV is already in use in a real production system, such simulation platform is also very useful to support the system managers in the decision process related with the plant floor's structures change. Every time a change occurs the AGV starts a learning and exploratory phase and its service rates may degrade. With the simulation platform such changes can be planed in order to prevent and minimise such degradation. In order to sustain the simulation platform, the multi-agent system that integrates the AGVs was enlarged with agents to control simulation processes and to feed the AGVs' sensors with images representing the external environment.

THE *AUTOMATIC GUIDED VEHICLE*

The AGV's Architecture

The AGV presents a typical multi-agent system architecture (figure 1). The system units are agents with specific tasks allocated. Agents are reactive entities responding in a timely manner to events occurred in the AGV's environment. They cooperate with each other providing the AGV with the aptness and performance to explore, recognise and navigate over a strange CM plant floor as an autonomous and intelligent entity.

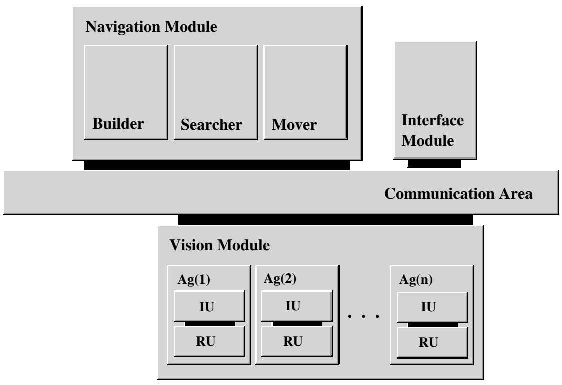

Figure 1: The AGV's Architecture

The AGV's agents are divided into three different modules:

- the **vision** module, that provides the vehicle with the ability to extract knowledge related with the structure and objects of its surrounding CM plant floor environment;
- the **navigation** module, that is responsible for the AGVs navigation procedures such as building the external plant floor model and finding and tracking paths through that model according to some user defined utility functions and constrains; and
- the **interface** module, that receives user's commands (translated into internal goals that the other system's modules try to fulfil) and provides general descriptions of the current system's state.

The AGV's agents communicate with each other through message passing. To support such interaction process, a special data structure was developed: the communication area. The communication area acts like a blackboard (Engelmore and Morgan 1988) structure ensuring information exchange and storing global AGV's knowledge. It is a fixed and dedicated structure as a way to reduce computational costs and to facilitate inter-agent communication. AGV's control is distributed through its agents, preventing the system against deadlock situations and avoiding contention.

The Vision Module

The vision module acts over specific visual domains, in order to get better definitions of the system's focus of attention. In a CM plant floor one may find enormous amounts of ambiguous information that can be collected by the AGV on the overall production system environment. Normally, such circumstances constitute the main difficulties that one may find in vision systems development (Boissier and Demazeau 1992). Due to the large diversity of information that must be recognised in a production environment (e.g. different kinds of cells, paths or other AGVs units), this module is organised in competence areas, each one associated to a specific agent. The vision agents produce descriptions of the recognised objects. The descriptions will be trusted by all other agents and must be relieved of uncertainty before posted in the communication area.

A vision agent is divided into two functional units:

- the **recognition unit**, that is responsible for the application of low-level recognition algorithms to the incoming raw data in order to produce the first object description; and
- the **intelligent unit**, that relieves the descriptions uncertainty that might exist; this task is achieved through the combination of the descriptions with the knowledge acquired in previous navigation processes and the internal rules related with the agent competence area; it is also responsible to filter the information relevant to high level navigation purposes.

Although the AGV's external environment is well defined as an CM plant floor, differences may exist between two different plant floors that require the change of the AVG's focus of attention. This can be achieved either by replacing the vision agents or by changing its internal rules. This flexibility is only possible due to the major division between the vision and navigation modules. The navigation module is generic and independent of the AGV's focus of attention.

Most of the vision agents are pattern recognition oriented, whose recognition unit is a feed-forward multi-layer neural network. Instead of training the neural network in the conventional way (once in its life time), it was adopted a dynamic retraining approach. Every time the intelligent unit decides that a pattern was wrongly identified and determines its correct identification, a feedback process occurs. Then, the involved pattern is returned to the recognition unit and the neural network training is reinitiated integrating the pattern in its training set.

The Navigation Module

There are two types of navigation procedures in the AGV:

- the **high-level navigation**, that is concerned with the relevant and permanent aspects of the CM plant floor (such as the cells or warehouses); a model of the environment must be maintained in order to be possible to determine high-level paths trough the plant floor; and

- the **low-level navigation**, that supports the navigation between two well defined points in a high-level path; all the objects of the environment are relevant in order to avoid collisions.

The high-level navigation tasks are performed by two agents:

- the **builder**, that is responsible to build, verify and maintain the plant floor's model according to the information sent by the vision module or, in some cases, by the interface module; this model stores the recognised objects and their relations, using a specific knowledge representation language; and

- the **searcher**, that searches high-level paths in the plant floor model respecting the environment characteristics and the user demands, and maximising a given utility function.

When the searcher can not find a path due to a lack of knowledge about a specific location, the AGV initiates a exploratory process where the searcher commands the vehicle to unexplored zones of the environment until the referred location is found. Such zones are labelled in the model through specific marks stored in previous navigation processes. The exploratory process also terminates if the searcher determines that the referred location do not exist, there is not enough time to find it or the user requires its termination through the interface agent.

The low-level navigation is done by the **mover** agent. This agent tracks the high-level paths (produced by the searcher) and rules the AGV mechanics accordingly. A tracking process starts automatically every time the searcher posts a high-level path into the communication area. If the mover detects a change in the environment, the tracking process stops, the AGV is immobilised and the builder agent is notified to correct the model. The process restarts with a new path finding by the searcher agent.

The Knowledge Representation Model

Adequacy and expressiveness are two of the most important characteristics that a knowledge representation language must exhibit. The diversity of the external environment requires a powerful semantic capable of expressing all the descriptions in a appropriate and effective way. Being the AGV operation a time critical task, the representation language must also be effective and reliable in path finding situations and in the updating process that occurs every time the plant floor changes.

Due to the flexibility and efficiency navigation and updating through well defined algorithms, the environment was modelled with an oriented graph. The notion of scenery was introduced to represent the graph's nodes. A scenery represents a particular view of an external environment's location. Different sceneries exist to represent the different directions that the AGV may assume when tries to reach a location. If all the information about a location is in the same node, the path finding algorithm would be slower because it is necessary to identify which objects are relevant in the present direction of the AGV. For instance, with the scenery notion, there are different sceneries to represent the same production cell according to the possible directions that the AGV may reach. The range of the possible directions must be reduced in order to limit the size of the system's model. The sceneries are represented by tuples whose fields express their identification, the characteristics of particular views and the links to adjacent sceneries. A special link field exists to represent a connection to an unexplored zone of the plant floor.

THE SIMULATION PLATFORM

A real CM plant floor may contain several AGVs in order to improve the overall service rate. The simulation can not be done independently to each AGV since the degradation induced by the collision avoiding may be significant. Thus, the simulation platform is composed by several AGVs (figure 2).

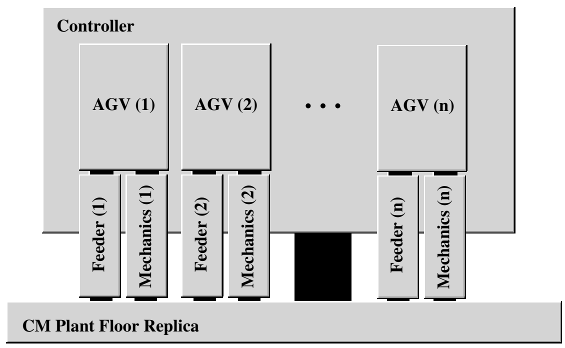

Figure 2: The Simulation Platform

The simulation process is managed by the **controller** agent. A configuration file, defined previously by the users, provides the simulation parameters, the desired statistics, the tasks that the AGVs must accomplish and characterise the CM plant floor to simulate. The controller interacts with the AGVs interface agents in order to transmit the required tasks and to collect the desired statistics. It is also responsible to modify the virtual replica of the plant floor during the simulation process according to the configuration file. This process enables the desired simulation of the vehicle adaptability to dynamic environments. The virtual replica is represented using the same structure of the AGV's internal model.

The integration of each AGV in the simulation platform is provided by two agents (figure 2):

- the **mechanics**, that emulates the AGV's mechanics, capturing the tracker orders into the communication area; its main goal is the detection of the AGV's present location (a scenery identification), which is achieved combining the received orders with the virtual replica of the plant floor; the location is itself stored in the environment replica; and
- the **feeder**, that transmits to the recognition units of the vision agents the image that characterises the respective scenery, according to the AGV's present location; these images are synthesised from a image database in order to reflect the scenery with all the objects that characterise it and with all the other AGVs that may be located in that zone.

Implementation Issues

With the exception of the vision agent, all the agents were implemented using Prolog (Carlsson and Widen 1993) due to its abilities to model complex behaviours in short development periods and to support high level knowledge representation. The AGV communication area and the CM plant floor virtual replica were implemented using Linda (Carriero and Gelernter 1989).

The vision agents are hybrid entities due to the different computational requirements of its units (figure 3):

- the recognition unit requires intensive computation and robust interfaces with the sensorial systems which had lead to an implementation in a distributed memory parallel machine using Parix (an extension to C that supports parallel programming);
- the intelligent unit requires fast inference abilities which can be easily obtained in a regular workstation using Prolog; some C components were added in order to facilitate the communication between the intelligent unit and the vision unit.

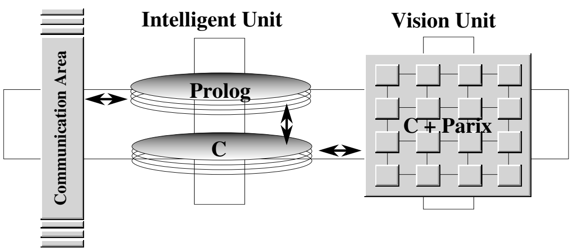

Figure 3: Vision Agent Implementation

CONCLUSIONS

The main objective of the present simulation platform is the study of the AGV's service rates related with production orders deliverance or even materials transportation among cells after changes in the plant floor. The system can be configured to collect different statistics such as: the overall or individual average service rates of the AGVs; the average waiting times of the cells; or the influence of the collision avoiding in the average service rate.

The developed simulation platform provides a mechanism to effectively validate an AGV behaviour prior to its real implementation. Its behaviour is encapsulated into specific agents with a well defined communication protocol allowing the reuse of almost every system's components. The use of a dynamic virtual replica of the surrounding environment enables the fine tuning of the AGV unit and the prediction of it's response to a changing plant floor. These predictions can be used to support the decision making process about future changes planing. The clear separation between the AGV's agents and the environment simulation agents simplifies a future implementation of a real AGV.

REFERENCES

Akhun M. and M. Durmusoglu. 1994. "Worker allocation problem in cellular manufacturing: A case study." In *Proceedings of the European Simulation Symposium, Vol. II* (Istanbul, Turkey), A. Kaylan, A. Lehmann and T. Oren, eds. 105–109.

Boissier, O. and Y. Demazeau. 1992. "A distributed artificial intelligence view on general purpose vision systems." In *Decentralized A.I.-3*, W.E. and Y. Demazeau, eds. Elsevier Science Publishers B.V., 311–330.

Carlsson, M. and J. Widen. 1993. *SICStus Prolog User's Manual*. Swedish Institute of Computer Science.

Carriero, N. and D. Gelernter. 1989. "Linda in Contex." *Communications of the ACM* 32, no.4 (Apr.): 444–458.

Craig, I. 1994. "A perspective on multi-agent systems." Research Report CS-RR-273, Department of Computer Science, University of Warwick, Coventry, UK.

Engelmore, R. and T. Morgan. 1988. *Blackboard Systems*. Addison-Wesley Publishing Company, Inc.

Freeman, J. and D. Skapura. 1991. *Neural Networks - Algorithms, Applications and Programming Technics*. Addison-Wesley Publishing Company, Inc.

Pitas, I. ed. 1993. *Parallel Algorithms for Digital Image Processing, Computer Vision and Neural Networks*. Jonh Wiley & Sons.

Wooldridge, M. and N. Jennings, eds. 1995. *Intelligent Agents, Proceedings of the ECAI-94 Workshop on Agent Theories, Architectures, and Languages*. Lecture Notes in Artificial Intelligence (890), Springer-Verlag.