

Dynamic control systems for AGVs

by **A. J. Bostel and V. K. Sagar**

AGVs are automatic load carriers that transfer objects from one location to another in a factory environment. Due to the increasing complexity of factory floor environments coupled with the need for increased flexibility in AGV systems, it is becoming increasingly important to be able to control AGV fleets dynamically. In this article, a neural-network-based method for controlling AGVs is presented.

Autonomous guided vehicles (AGVs) are automatic load carriers that transfer objects from one location to another in a factory environment. The use of these systems in factory environments is on the increase and is projected to increase dramatically in the next decade. The requirements for AGV systems are becoming more demanding due to the increasing complexity of factory floor environments. This has resulted in a need for AGV systems with increased flexibility and reliability. In particular, it is becoming more important to be able to alter dynamically the AGV job queue and the AGV path.

The selection of an optimal set of jobs and the choice of the best route between jobs for a given AGV can be considered as a job scheduling problem. In this case the problem is to select the most efficient combination of loading points called station material buffers (SMBs) for an AGV to visit given the current combination of job requests. Job scheduling is actually a classic problem in operations research and is known to be an NP-complete* constraint-satisfaction problem.¹ The solution of such a problem generally involves finding the minimum value of a multivariable cost function describing the constraints in the problem. One method for solving problems of this type is to use an artificial neural network (ANN) model. In this article a new technique is presented which

incorporates an ANN to evaluate the best job assignment for an AGV so as to achieve better system efficiency. An AGV simulator was implemented to evaluate the functional architecture, the methodology and the techniques to be used in the new AGV system.

Functional architecture of AGV system

An AGV system can comprise a fleet of vehicles and a supervisory system which communicates with all vehicles and takes care of job control, traffic control, status monitoring, layout management, route and task planning, and interfacing to other systems. Fig. 1 illustrates the functional architecture of an AGV supervisory system.

The following sections illustrate each module of this supervisory system in detail.

Traffic control

A traffic control module is essential in an AGV system which consists of more than one vehicle. This module

List of abbreviations

AGV	automated guided vehicles
SMB	station material buffers
ANN	artificial neural network
VBP	vehicle boarding point
BAT	recharging point
TRASH	waste disposal point

*NP-complete problems cannot be completely solved for a deterministic solution within a reasonable time. It is only possible to find a solution which may not be the optimum one.

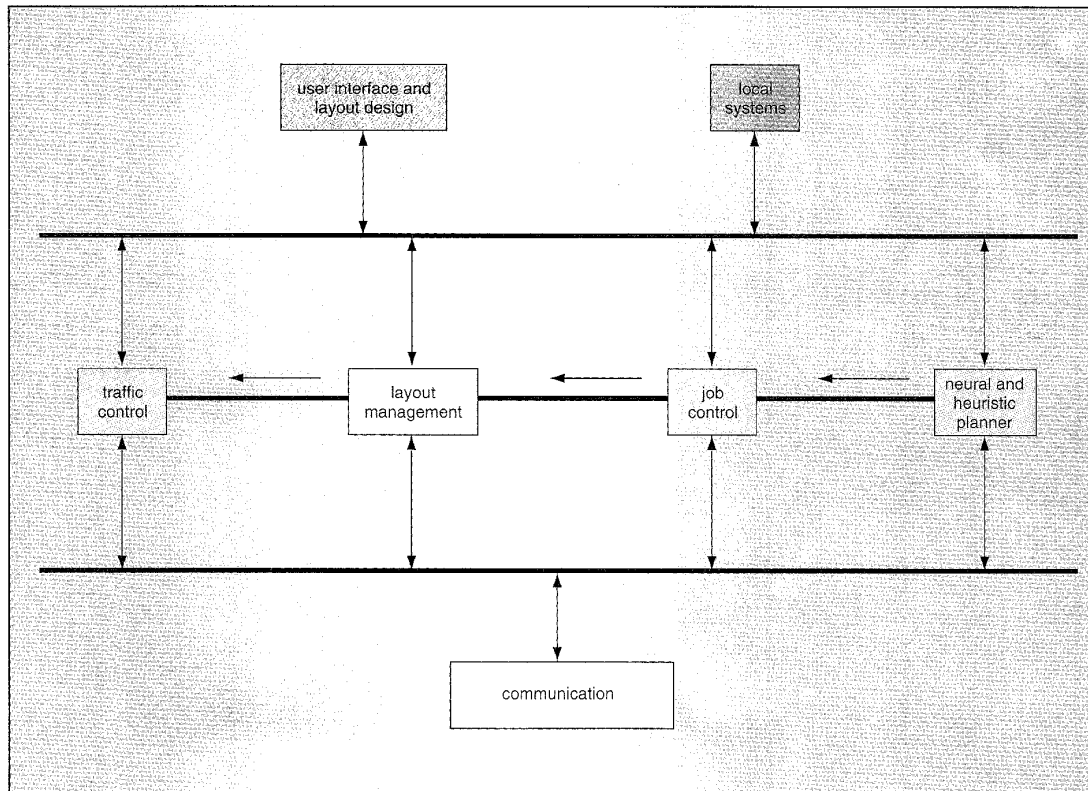


Fig. 1 AGV functional architecture

organises the movement of the AGV fleet about the factory. In our system, meeting zones are created at strategic points about the factory in order to regulate the vehicle movement and avoid any possible collisions in an analogous manner to automotive traffic lights. Any vehicle approaching a meeting zone will have to request for permission to enter the meeting zone. Permission will only be granted provided that no other vehicle is currently occupying that area. Hence the status of a meeting zone will either be FREE or OCCUPIED.

Layout management

The main component of this module is a layout map generator which contains the position, dimensions and functions of all elements that make up the working area, such as walls, paths, workstations, machines and store areas. This is an important feature for vehicle navigation. This map is created in the simulator using the layout drawing screen with each screen pixel representing a certain amount of the actual factory layout dimension. The layout drawing screen is illustrated in Fig. 2.

A menu-driven graphic user interface package has been developed which provides users with the necessary tools to design their very own factory layouts without too much difficulty. The newly created layout is converted and stored in binary format. In this format it is portable

and can be easily downloaded into the on-board computers of the AGVs.

A given layout should consist of the following elements: a vehicle boarding point (VBP), i.e. the place where an AGV collects new loads; a waste disposal point or 'TRASH'; and a number of station material buffers (SMBs). In addition, since most AGVs are battery powered, at least one recharging point or 'BAT' is usually necessary. In a valid factory layout these elements must comply with certain conditions. Each newly created layout is therefore subject to the following layout validation rules:

- only one VBP in the layout
- one one TRASH in the layout
- at least one BAT
- TRASH must be located before the VBP along the same path
- at least two stop points in a layout
- no stop points should have similar name (label)
- a stop point must NOT be placed at a junction
- paths must NOT be too close to the obstacles (vehicle size dependent).

A new factory layout must fulfil the above rules before a simulation based on that layout can be carried out.

Job control module

The fundamental responsibility of the job control module is to organise the fleet of AGVs so as to efficiently satisfy the transportation requirements of a facility. Transportation requests are generated by the local systems (SMBs) as their online inventory runs low. However, manual jobs are also possible in the simulator.

Neural and heuristic search module

The main function of this module is to find a route which results in the minimum distance travelled by an AGV. The source node of the route is always the vehicle boarding point (VBP), the destination node is the location at which the tasks are to be performed.

Status monitoring

The fundamental responsibility of this module is to provide messages to the user about the current status of the vehicles, the meeting zones or the material requests. These status messages should provide sufficient information for a user to evaluate the performance of the system and spot any malfunction instantly. For example, a material request message may be any one of the following: waiting, busy or delivered. A 'waiting' job is a material request waiting to be served, possibly by an available vehicle; a 'busy' status implies that the job has been taken up by an AGV and is on its way to the station; and a 'delivered' job indicates a job which has been successfully completed. This feature further enhances

the human-computer interaction of the program and could be a useful tool for troubleshooting.

Route and task planning

From the application point of view, this is the most important operation of a supervisory system. The planning system consists of various modules that determine the shortest route to be taken, effectively manage the user memory, and intelligently assign and dispatch the AGVs.

The main function of this module is to find a route which results in the minimum distance travelled between the source and destination. Minimising the distance is one way of optimising the task. There are other parameters such as number of turning points for AGVs that can also be optimised, but for this preliminary study distance travelled was optimised as this is usually the main criterion for assessing the quality of the result. The source node of the route is the vehicle boarding point (VBP), the destination node is the location at which the tasks are to be performed. A heuristic algorithm, the A-STAR algorithm, has been adopted to find the shortest route between two points in a factory. This algorithm is described below.

The A-STAR algorithm

The A-STAR algorithm⁴ is an extension of the Branch-and-Bound method² and has an evaluation function f , which is defined so that its value $f(n)$ at any node n is an

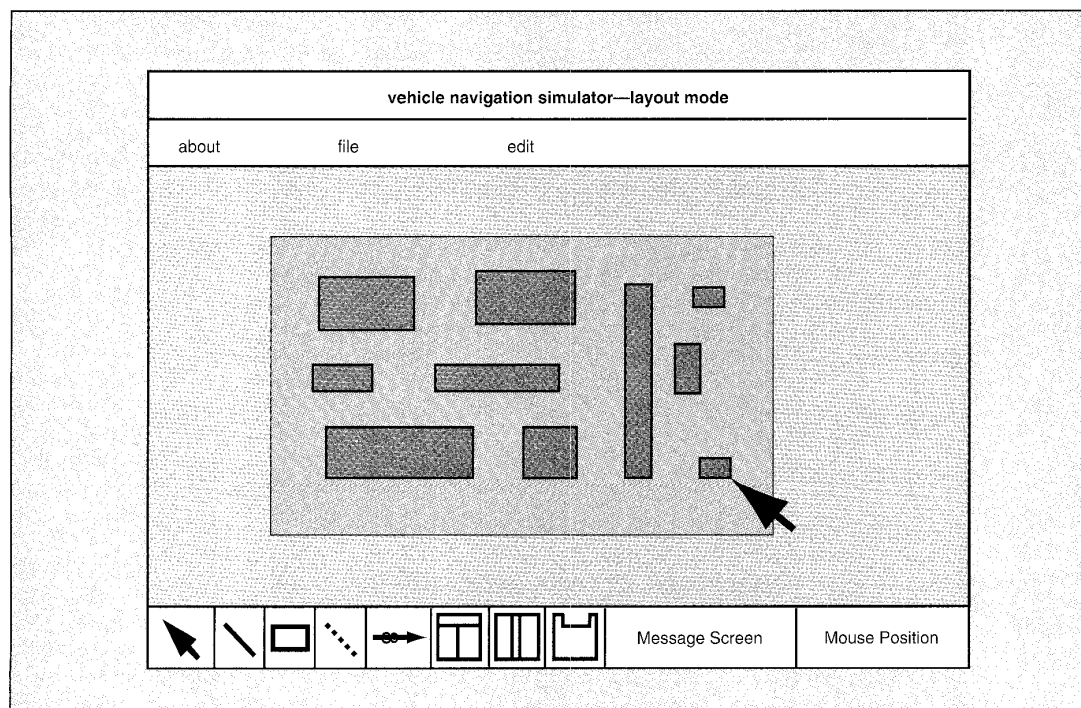


Fig. 2 Layout drawing screen

estimate of the cost of a minimal cost path from the start node s to node n plus an estimate of the cost of a minimal cost path from node n to a goal node. The function, $f(n)$, is an estimate of a minimal cost path constrained to go through node n . That node on the list having the smallest value of f is then estimated to be on a minimal cost path.

Let the function $k(n_i, n_j)$ give the actual cost of a minimal cost path between two arbitrary nodes n_i and n_j . If T is the set of goal nodes, then the cost of a minimal cost path from node n_i to a goal n_j is given by

$$h(n_i) = \min_{n_j \in T} k(n_i, n_j)$$

Hence, any path from node n_i to a goal node that achieves $h(n_i)$ is an optimal path from n_i to a goal n_j . The function h is known as the heuristic function.

For this application, we have designed our cost criteria to be:

- 1 the distance between two nodes
- 2 the angle of rotation along the path.

The angle of rotation is important because it adds to the difficulty of moving along the path. Therefore there is a penalty in time for meandering routes as it is more difficult for the vehicle to make any turn.

Based on these cost factors, we want to know the cost $p(s, n)$ of an optimal path from a given start node s to some arbitrary node n , including the angle of rotation along the path. A new function g is introduced for this purpose, where g is defined as follows:

$$g(n) = p(s, n) \quad \text{for all } n \text{ accessible from } s \quad (1)$$

The calculation of the cost g is illustrated in Fig. 3. The network shown in Fig. 3 consists of a start node s and three other nodes n_1, n_2, n_3 . The arcs are shown with arrowheads and costs. Starting with s , we obtain successors n_1 and n_2 . The estimates $g(n_1)$ and $g(n_2)$ are then 3 and 7, respectively. Suppose the algorithm expands to n_1 next with successors n_2 and n_3 . At this stage $g(n_3) = 3 + 2 = 5$, and $g(n_2)$ is ignored because it is more costly than the previous path found. The value of $g(n_1)$ remains equal to 3.

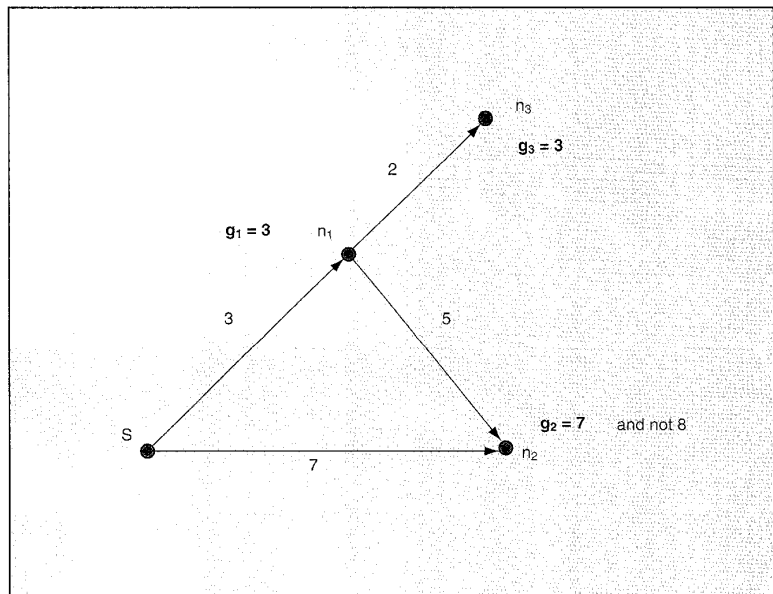
Finally, we define the function f so that its value $f(n)$ at any node n is the sum of the cost of an optimal path from node n to a goal node and the cost of an optimal path from node s to node n including the difficulty of moving along the path, i.e.

$$f(n) = h(n) + g(n) \quad (2)$$

The value of $f(n)$ is then the cost of an optimal path constrained to go through node n .

The A-STAR algorithm can evaluate the shortest route through a network in a comparatively short time. However, to implement this method in an AGV route planner would require that the algorithm is carried out each time a new job is required. Our system therefore employs the following strategy. Once a new layout has satisfied the validation rules it is subject to the heuristic search for the shortest routes to all stop points and a look-up table is generated for each layout design in the simulator. This look-up table, consisting of all possible stop points along a route, can then be used to determine the best job combination for that layout without the necessity of recalculating the shortest route each time.

Fig. 3 Example calculation of estimate g



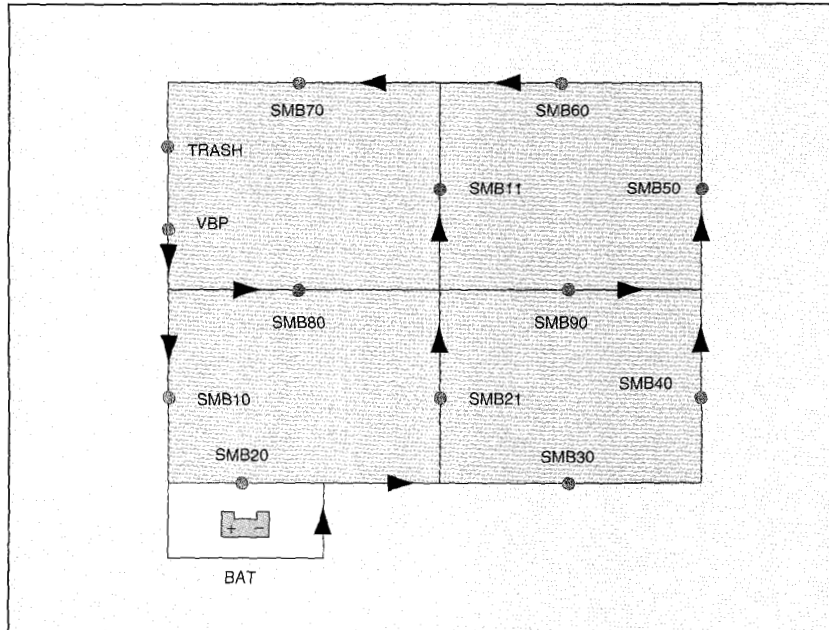


Fig. 4 Example of factory layout

Generation of a look-up table

As described above, the look-up table is designed to reduce the processing time of the system. A look-up table is associated with each particular layout design. This layout, having fulfilled the validation rules, is subject to the heuristic search for the shortest routes to all stop points. This information is then used to form the look-up table. This look-up table avoids the necessity of going through the A-STAR algorithm each time a new job is requested.

Even though the use of the look-up reduces the processing time of the system when dealing with job requests, the generation of the look-up table itself could still take a long time, particularly if the factory is large or has many SMBs. In view of this, a new algorithm called 'forward-reverse search' has been used to reduce the amount of search required for a particular layout. The algorithm works as follows:

- 1 Start with heuristic search using A-STAR from VBP to the TRASH.
- 2 Eliminate all the stop points visited in this route from the stop point list. This will be the shortest route for these stop points, as jobs are to be picked up from the VBP and delivered to the stop point (SMBs), while the TRASH are to be transferred from the SMBs to the TRASH.
- 3 Select any remaining stop point from the stop point list:
 - 3.1 Perform A-STAR search.

3.2 Locate last stop point which has not been visited by any previous routes.

3.3 Perform another heuristic search from VBP to this point and back to TRASH. Apply the following conditions:

If similar path found accept this route;
Else locate first stop point visits and Goto step 3.3.

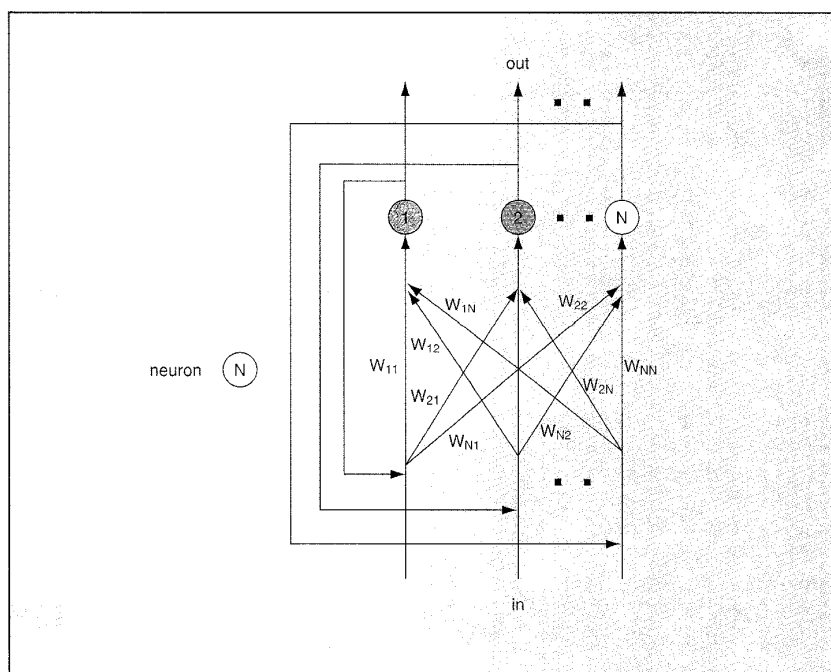
4 Repeat step 3 until all stop points are covered.

This algorithm is integrated with the A-STAR algorithm in order to form the shortest route look-up table. Its efficiency increases as the complexity of the layout increases. The forward-reverse search algorithm is applied to an example factory layout shown in Fig. 4.

Firstly: Perform A-STAR path search from VBP to TRASH. This results in the first shortest route to be stored in the look-up table. This visits SMB80, SMB11 and SMB70. Delete these three SMBs from the SMB list.

Secondly: Choose any SMB from the remaining SMB list, repeat A-STAR search from VBP to this node, and back to TRASH, e.g. take SMB40. The heuristic search results in a minimum distance route which covers SMB10, SMB20, SMB30, SMB40, SMB50, SMB60 and SMB70. However, SMB70 has been covered by the previous route, which implies that SMB70 can be reached by other shorter alternatives. Searching through the SMB list, we noticed that SMB60 is the last one in this route which has not been covered previously.

Fig. 5 Hopfield Model



Thirdly: Repeat A-STAR search from VBP to SMB60, then to TRASH.

Obviously, this search will end up with a route which covers SMB80, SMB90, SMB50, SMB60 and SMB70. This is not the ultimate solution, however, since we still have to search for a route to the first SMB on this route which has not been previously covered, namely SMB90. The result of this search provides a series of nodes which are similar to that covered by the route from VBP to SMB60. Hence, we fix that route to be the shortest one to reach SMB90, SMB50 and SMB60. These three SMBs are then removed from the SMB list. This process continues until all the SMBs have been covered. Going through the list, we end up with a look-up table with five routes covering a series of SMBs. This look-up table is shown in Table 1. The number of iterations required to generate the full look-up table depends on the selection of the search point. The example in Table 1 required 11 iterations to cover all stop points, whereas 22 (2×11 SMBs) would be required if the conventional method was used.

Table 1 List of possible routes

route number	stop points visit
1	SMB80, SMB11, SMB70
2	SMB80, SMB90, SMB50, SMB60, SMB70
3	SMB10, SMB20, SMB21, SMB11, SMB70
4	SMB10, SMB20, SMB30, SMB40, SMB50, SMB60, SMB70
5	SMB10, BAT, SMB21, SMB11, SMB70

Job assignment routine

After the job controller receives the material transportation request, the job assignment routine is responsible for generating the detailed transport orders for the AGVs. This process involves selecting a good combination of SMBs for a given AGV to visit given the current status of the SMBs and the other AGVs.

If more than one material request is awaiting attention, some means of selection is required. Priority rule is most commonly used in the AGV systems. A priority is a measure of the relative importance of the processes in a system. However, due to dynamic materials requests, the priority system may face the problem of not being able to serve the lower priority stations effectively. As the material request is dynamic and the pace of each production worker is different, the situation can become worse if the production line has not been balanced (a balanced production line is one in which a material is made available at the work place when it is required). Should this problem arise, the user may set similar priority for all stations, making it first-come first-served basis, thus defeating the purpose of priority control.

In our system, we make use of the look-up tables and adopt a neural net approach to solve the job assignment problem. There are several types of ANN models which can be used to solve this type of problem.¹ The choice of model to use in this system was governed by the practical constraints of the problem and the requirements of the application. In most cases, this particular application only requires a 'good' solution, i.e. one with a better

combination of jobs than a purely random choice of routes, so that it is not really necessary to calculate the 'optimum' solution. In addition, this type of system really requires the solution to be found rapidly and with the minimum of computational expense. The Hopfield Model neural network³ was selected as a prime candidate for solving the job scheduling problem because of its ease of implementation and the speed with which a solution is reached.

Job allocation network

The particular problem considered here is to pick a route for each AGV that allows it to be as fully loaded as possible at all times i.e. to minimise the time the AGV spends carrying less than its maximum capacity. This involves choosing a subset of the SMBs requesting service. The criterion adopted in this system is to select those SMBs which are nearest to the VBP, i.e. the place where jobs are collected, thus reducing the time the AGV spends returning 'empty handed'. An additional constraint in this case is the 'accessibility' of each SMB, i.e. the number of possible routes which visit the SMB. Obviously an SMB is a good candidate for including on the AGV's route if it is 'on the way' to several other SMBs also requesting jobs or with waste products ready to be taken to the TRASH. The TRASH is usually positioned close to the VBP, near the end of the route, so that an AGV will nearly always visit this location on the return journey regardless of precise details of the chosen route. A separate neural network system similar to the job scheduling scheme is used to process the trashing requests and this information is incorporated into the final decision for the route of the AGV. The following 'cost' function H can therefore be defined for this particular problem:

$$H = A \sum_{i=1}^N \sum_{j=1}^N d_{ij} x_i x_j - B \sum_{i=1}^N \sum_{j=1}^N (U_i + U_j) x_i x_j - C \sum_{i=1}^N z_i x_i \quad (3)$$

where A, B, C are arbitrary constants, d_{ij} is the minimum distance required to navigate between SMB_{*i*} and SMB_{*j*} found from the A-STAR search algorithm, z_i (=1 or 0) is used to indicate whether SMB_{*i*} is requesting a job, U_i is the accessibility of SMB_{*i*} and x_i (=1,0) indicates whether SMB_{*i*} is on the proposed route. This function has its minimum value when the route contains highly accessible, near neighbouring SMBs which are close to the VBP.

This cost function can be mapped into a Hopfield style neural network by letting x_i correspond to the output of the i th neurone in the network. The constraints in the problem are then incorporated into the system by setting the connection weight between neuron i and neuron j equal to

$$w_{ij} = -Ad_{ij} + B(U_i + U_j) \quad (4)$$

This system then acts to find a set of SMBs which satisfy these constraints by allowing the set of neurones to settle into a stable configuration. Thus, given an initial binary input vector $Z = (z_1, z_2, \dots, z_N)$, representing the set of SMBs requesting jobs, the network generates a binary output vector $X = (x_1, x_2, \dots, x_N)$, representing a good combination of SMBs to visit given the current state of the job queue. The system then iterates until a stable set of SMBs is generated. This set minimises the value of the cost function. The shortest route to these SMBs can then be determined by referring back to the look-up table storing the set of minimum distance routes around the factory.

This network is shown schematically in Fig. 5.

Simulator output

An AGV simulator has been developed around the above techniques to illustrate the system. It can also be used as a tool to evaluate the performance of an AGV system. Some example outputs from the system are illustrated in Figs. 6-8. The information contained in these screens would help a user to design an optimal AGV system for a given factory environment.

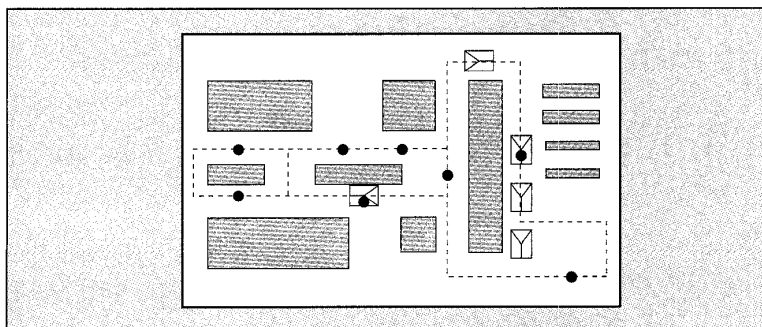








Fig. 6 Sample simulation display window

CONTROL SYSTEMS

Fig. 7 Sample vehicle status screen

vehicle status			
vehicle		status	dest
1		moving	VBP
2		handling	SMB10
3		moving	SMB21
4		rotating	SMB20
5		moving	TRASH
6		idling	N/A

The output of the simulator is a screen of four windows showing the simulation display, job control, vehicle status and job request. Three of them are shown in Figs. 6–8.

The simulation display window illustrated in Fig. 6 shows the AGVs moving from one location to another performing specific tasks. The simulation starts with the AGVs lining up along the VBP's path awaiting for job assignment. AGVs perform the jobs they have been assigned. There are two types of jobs: job requests and TRASH requests.

Job Control				
	request	job type	priority	status
1	SMB20	new part	2	busy
2	SMB11	new part	2	delivered
3	SMB21	new part	2	waiting
4	SMB11	TRASH	3	delivered
5	SMB10	TRASH	3	waiting

Fig. 8 Sample job control window

The job control window (Fig. 8) indicates the list of jobs that need to be served. Indicated next to each job is the requesting station, the priority level associated with the station and the status of the job.

The vehicle status window (Fig. 7) shows the status of the vehicle and the destination heading by each and every AGV. There are six different types of status that can relate to a vehicle: moving, rotating, handling, idling, low battery or faulty.

The job request window, which is not shown, enables manual job input. The user can specify jobs to be carried out by the AGV fleet in real time (as and when they occur). This could be used to evaluate the performance of a given fleet of AGVs on a factory floor. The user can then vary the parameters in order to optimise the system.

Conclusion

This article describes the application of a heuristic search algorithm and a neural network strategy for job allocation for automated guided vehicles in factory environments. This technique has been validated using a newly developed AGV simulator. The results of these simulations indicate that the system can perform better than many conventional AGV systems in various aspects, most notably in providing increased flexibility and efficiency.

In the course of this research we have developed a working simulation program for AGV systems. This simulator contains the basic requirements for layout creation and modification, the file system and the simulation mode. Some useful features of this simulator are auto-route creation and overtaking. Since the obstacles and the various stopping points in the factory are included in the layout specification of any factory floor, the system will automatically generate the best possible routes for that layout. In addition, the simulation facilities allow an AGV to deal with the situation in which an AGV breaks down or an unexpected obstacle blocks the path of a working AGV.

References

- 1 ZHOU, D. N., CHERKASSKY, V., BALDWIN, T. R., and OLSON, D. E.: 'A neural network approach to job shop scheduling', *IEEE Trans. Neural Networks*, 1991, 2, (1), pp.175–179
- 2 CHABRIS, C. F.: 'Artificial intelligence and turbo C' (TopPan Company, 1991, ISBN 1 55623 129 6)
- 3 HOPFIELD, J. J., and TANK, D. W.: 'Computing with neural circuits: a model', *Science*, 1986, 233, pp.625–633
- 4 NILSSON, N. J.: 'Principles of artificial intelligence' (Tioga Publishing Company, 1982, ISBN 3 54011 340 1)

© IEE: 1996

The authors are with the Department of Electronic Systems Engineering, University of Essex, Wivenhoe Park, Colchester, Essex CO4 3SQ, UK. Dr. Sagar is an IEE Associate Member and Mr. Bostel is an IEE Associate.