Synthesis of AGVS by coloured-timed Petri nets

SUHUA HSIEH

Abstract. The paper presents a new AGVS modelling method using a class of command-place coloured-timed Petri nets (CP-CTPN). The CP-CTPN is capable of modelling many distinctive AGVS features, such as the functions of zone control and vehicle blocking, that are not generally available in most general-purpose commercial programmes. In the proposed new modelling method, the CP-CTPN is generated systematically in four steps by simply giving a floor-path layout and a management policy. These four modelling steps are; (1) to construct a PN-based floor-path model by assembling modular PN sub-models, (2) to incorporate coloured tokens and timed transitions into a CTPN, (3) to generate vehicle movement sequences by introducing relay functions and management commands into CP-CTPN, and (4) to compute reachable markings evolved from initial marking by incorporating the product of a coloured incident matrix and a firing sequence. Two floor-path examples, given by Hsieh and Sirinivasan, are presented to show the effectiveness of the method. Simulation results using the present method indicate that both the required fractions of time that a vehicle travels loaded and unloaded are somewhat higher than those of Sirinivasan’s. That is because the present method allows many functions to be modelled in detail that are not available in other approaches.

1. Introduction

Material handling is one of the most important tasks in an automated manufacturing system (AMS). Automated guided vehicles (AGV), travelling among work-stations and automated storage/retrieval systems (AS/RS), are the material handling tools commonly used in an AMS. Since the flexibility of an AGV system is much confined by its original design, it is often desirable to develop a computerized modelling method that can systematically generate an AGVS of great flexibility.

The design of an AGVS can generally be divided into three levels, namely, the levels of device, floor and management. In device design level, the hardware of vehicles is carefully planned, designed and fabricated. These tasks are usually done by manufacturers. A few technical papers are available in this area. In floor design level, a detailed layout of floor-path is carefully planned and implemented. It is usually done by the floor manager. Since it is application dependent, it cannot be generated independently without a priori system knowledge. Various floor-path layouts were proposed and carefully studied by researchers (Gaskins and Tanchoco 1987, Kaspi and Tanchoco 1990, Venkataramanan and Wilson 1991). The problem of locating pick-up and delivery stations was studied by Goetz and Egbelu (1990). A systematic floor-path Petri-net representation method was proposed by Hsieh and Shih (1992, 1994). In management design level, the management policy is translated into a set of dispatching and routing rules. The determination of a management policy is also application dependent. The problems of determining a required number of vehicles for a system were studied by Maxwell and Muckstadt (1982), Newton (1985), Egbelu (1984), Tanchoco et al. (1987), and Lin (1990). The problems of vehicle dispatching and routing were studied by Egbelu and Tanchoco (1984), Hodgson et al. (1987), Kim (1990) and Yim et al. (1993). A variety of management rules have been proposed in these studies.

Since there are too many application factors involved in an AGVS design process, a simulation model is usually used for study. Although there are several general-purpose commercial simulation programmes available on the market, they are too general to include specific AGVS features such as zone control and traffic blocking in the model. The AGVS models generated by general purpose commercial packages are less accurate since many features are neglected or ideally approximated by other means in their programmes. Gaskins and Tanchoco (1989) developed an AGVS design tool, AGVSim2, for free-ranging vehicle management. AGVSim2 focuses on the management strategies of an AGVS rather than on the construction of a simulation model. Since the construction of a simulation model is tedious and time-consuming, it becomes very difficult to obtain an optimal solution in an AGVS design process.

The objective of this paper is to develop a method that can systematically generate simulation models for an AGVS based on a proposed modelling strategy. Simulation models are constructed and solved by a
Compact computer program that converges to a solution very quickly. The modelling procedure is presented below.

2. PN representation of a given floor-path layout

Since the AGVS design process is application dependent, it is generally difficult to plan for a new AGV system simply by modifying an existing one. However, it is indeed possible to construct a new system by combining modular features of the system. Hsieh and Lin (1991) presented four basic flow-path modules of unidirectional AGVS in terms of Petri nets. The work was extended to model bi-directional systems by Hsieh and Shih (1992). Zone control nets were introduced to the flow-path model in their model. To assure the robustness of the model, several system properties were carefully examined. These properties include, (1) collision-free traffic flow, (2) conservation of vehicles and control signals, (3) floor-path reachability, and (4) deadlock-free structure. By the employment of invariant properties and reachability tree of Petri nets, the properties of safeness, boundedness, strict conservation, reachability, and liveness of each sub-net were carefully checked. By putting sub-net modules together, a complete robust floor-path model was constructed systematically in terms of Petri nets. In order to develop the proposed modelling method, some fundamental building block elements are defined and presented below.

2.1. One-dead-end bi-directional line structure

The bi-directional line structure is composed of two unidirectional line structures with opposite travelling directions. Some typical bi-directional path segments needed in an AGVS are the path segments to home, to rest areas, and to pickup/delivery point of the storage area. A bi-directional path segment usually leads to a dead-end and thus a one-dead-end bi-directional line net is usually used. Figures 1(a) and 1(b) show a typical one-dead-end bi-directional line structure and its associated Petri-net sub-model.

2.2. Station-zone macro structure

A station is usually located near an AGV path zone. It consists of one input buffer, one output buffer and one machine. When a vehicle stops at a station, it performs the task of loading/unloading to/from the input/output buffer. Sensors are equipped in buffers to detect the existence of parts. If the vehicle is loaded and the buffer is empty, the controller will issue an unloading command to the vehicle. If the vehicle is empty and the buffer is loaded, the controller will issue a loading command to the vehicle. The machine in the station-zone receives a part from the input buffer, processes it and then moves it to the output buffers once it is done. A typical station-zone macro net is given in figure 2. In this net, \( Z_p \) is a zone-place nearest to the station among all zones, where the subscript \( p \) is the index of a zone. \( MC_n \) is a timed-place showing the required processing time of a machine, where the subscript \( n \) is the index of a machine. \( IB \) is a place for input buffer. \( OB \) is a place for output buffer. \( JS \) is a place that issues 'job is done' signal to the controller. \( RS \) is a place that receives 'job request' signal from the controller.

![Figure 1(a). The one-dead-end bi-directional line structure.](image)

![Figure 1(b). The one-dead-end bi-directional line net.](image)

![Figure 2. The station-zone macro net.](image)
controller. Since the operations of loading and unloading usually take times, the transitions of \( t_{p1}, t_{p2}, t_{n1} \) and \( t_{n2} \) are timed-transitions.

The station zone macro net describes activities that are related to the station. The reason to introduce a station-zone macro net here is to enable the measure of average waiting time of a part in a station. The station-zone macro net is usually embedded in the associated zone-place. In other words, the time spent to process a specific part can be measured via the refinement of the station-zone place in the model. Whenever a zone-place in a floor-level model is a station-zone, the zone-place should be replaced by a station-zone macro place.

2.3. Combination rules

Modular basic flow-path nets can be combined to generate a complete net structure. Since two places cannot be connected directly, transitions are used to connect the output places of preceding net and the input places of the succeeding net. A dashed circle is used to represent the connected place. The combined structure is called the modular connecting net. Figures 3(a) – 3(e) show the modular connecting nets of the line structure, the divide structure, the merge structure, the intersection structure, and the one-dead-end bi-directional line structure. Here L stands for a line structure, D stands for a divide structure, M stands for a merge structure, I stands for an intersection structure, and BL stands for a one-dead-end bi-directional structure. The combination rules can be summarized as follows.

1. A dashed-line place in preceding net should be connected to a solid-line place in succeeding net.
2. A dashed-line (solid-line) place cannot be connected to a solid-line (dashed-line) place in the same net.

![Figure 3. The modular connecting nets of (a) line structure, (b) divide structure, (c) merge structure, (d) intersection structure, and (e) BL structure.](image-url)
2.4. PN modelling procedure

This section is devoted to the PN modelling of a given floor-path layout. To summarize the modelling process, a procedure of six steps is presented as follows.

1. Specify the basic flow-path types of a given AGVS floor-path layout.
2. Generate basic modular connecting nets for each flow-path type of line, divide, merge, intersection, and one-dead-end bi-directional line.
3. Combine modular connecting nets to obtain a complete floor-path net using combination rules.
4. Inspect whether the complete floor-path net is closed? If not, go back to step 3.
5. Replace a zone-place by a station-zone macro place if a station is present near a zone.
6. Stop.

3. CTPN representation of coloured tokens and timed transitions

The floor-path PN model constructed above can be made more comprehensive by incorporating coloured tokens and timed transitions/ places. Coloured tokens are introduced to represent the entities of vehicles and statuses of vehicles, machines and buffers. Timed transitions are used to measure the performance time of the vehicles. Timed places are used to measure the process time of the machine. After putting coloured tokens into places and adding time into some transitions and places of the PN model, the system can then be expressed in terms of a coloured-timed Petri net (CTPN) model.

3.1. Coloured and timed Petri nets

In a place-transition net, it is often necessary to have several identical sub-nets since a folding into a single subnet will destroy the possibility to distinguish different processes. In that respect, Genrich and Lautenbach (1981) developed predicate-transition nets (PrTNs). In PrTNs, information is attached to each token by a token-colour and each transition can occur in several ways represented by different occurrence colours. When a transition is fired, the relation between occurrence colours is defined by the expressions attached to the arcs. Restrictions on the possible occurrence colours can be imposed by a predicate that is attached to the transition. Inspired by PrTNs, Jensen (1981, 1983) has proposed Coloured Petri nets (CPNs). The relation between an occurrence-colour and a token colour involved in the firing of a transition is defined by the functions attached to the arcs. Moreover, CPNs explicitly attach a set of possible token-colours to each place, and a set of possible occurrence-colours to each transition. By so doing, the place-invariant can be interpreted and the elements of the incidence-matrix become functions instead of integers.

Since an event is completed after a sequence of activities are performed by the system. In other words a certain amount of transition time is elapsed between two consequent events. The time epoch can thus be associated with transitions. The choice of time associated with transitions is one of the popular topics in the study of Timed Petri-nets TPNs (Viswanadham and Narahari 1992). It is noted that Petri nets with timed transitions are equivalent to Petri nets with timed places. Several researchers (Ramchandani 1973, Sifakis 1977, Rama-moorthy and Ho 1980) investigated the type of timed Petri nets in which places or transitions were associated with deterministic time duration. The analysis of such timed Petri nets is, however, tractable only for some special classes such as marked graphs. The introduction of associating random time duration was first explored independently by Natkin (1980) and Molloy (1982). This study was the starting point for the emergence of Stochastic Petri nets (SPNs) that was later extended and used as a principle performance modelling tool for discrete event systems. Our focus in this paper is on the deterministic TPNs and SPNs. We will consider both transitions and places associated with time.

3.2. Coloured vehicles

Vehicles are identical in their functionality. It is the status attributes (e.g., loaded, empty with assignment, empty without assignment, etc.), the locations of the vehicle, and the priority of tasks that make them different from each other. For instance, if a part is waiting to be transferred from one station to another, vehicle A is free and it is also in shortest distance to the part among free vehicles, vehicle A is then given the highest priority for the job. Therefore, it is very important that each vehicle is assigned with an identification number and the vehicle status is being monitored from time to time. In this paper, we use coloured Petri nets to identify and to monitor vehicles. The notations of $V_i$-free, $V_i$-asgn$\{j\}$, and $V_i$-ld$\{j\}$ are ‘coloured tokens’ that represent different vehicles at different status. Here $V$ stands for ‘vehicle’, $i$ stands for the number of vehicles, $(i = 1, 2, \ldots, n)$, and $n$ is the total number of vehicles in the system. $\text{free}$ stands for the vehicle status that it is available for job assignment; $\text{asgn}$ stands for a vehicle that is assigned to $\text{job}$, and $\text{ld}$ stands for a vehicle currently executing $\text{job}$.
3.3. Colour tokens in station-zone macro net

In order to distinguish the vehicles from each other in the system, different colours are assigned to the vehicles to indicate different entities and status in different time periods. Similarly, different colours are also assigned to various machines and input/output buffer stands to indicate their status. To enumerate the concept, the colours are defined as follows.

1. The colour jobj stands for the situation that the part jobj is on the input or output buffer, or is being processed on the machine.
2. The colour empty stands for the situation that the machine is idle, or the input or output buffer is empty.

3.4. Colour matching

It should be noted that the loading/unloading actions can happen only when the colours of the vehicles and the buffers, or the colours of the buffers and the machines are matched. The relation between the colours and actions can be described as follows.

1. If the colour of a vehicle is Vi-assgndj, the colour of output buffer of a station is jobj, and two colours are matched, vehicle i will move toward the station and pick up the part from the output buffer. Once the loading action is done, the colour of the vehicle is changed to Vi-llddj and the colour of output buffer is changed to ‘empty’.
2. If the colour of a vehicle is Vi-llddj and the colour of input buffer of a job-j-next-operation workstation is ‘empty’, vehicle i will move toward that workstation, and co-ordinate with the input buffer to move the part. Once the unloading action is done, the colour of the vehicle is changed to empty, and the colour of the input buffer is changed to jobj.
3. If the colour of a machine is jobj and the colour of output buffer is empty, part jobj will be moved to the output buffer. Once the job is done, the colour of the machine is changed to empty, and the colour of the output buffer is changed to jobj.
4. If the colour of a machine is empty and the colour of input buffer is jobj, part jobj will be moved to the machine. Once part jobj is transferred to the machine, the colour of the machine is changed to jobj, and the colour of the input buffer is changed to empty.

3.5. Timed model

In order to evaluate the lead time of an AMS, the development of a detailed time model to include the vehicle travelling time, loading/unloading time and machine process time is very essential. On manufacturing floor, an AGV often travels at a variant speed in response to different traffic conditions. For example, when a traffic jam is encountered, the AGV stays awaited until the jam is cleared. Thus, the travelling time of a vehicle between two specific stations may not be always the same. The loading/unloading activities and the machining processes all take time. In a station macro, the parts are moved from the vehicle to the input buffer if the input buffer is empty, and then moved from the input buffer to the machine when the machine is available. Once the part is processed, it is moved from the machine to the output buffer if the output buffer is empty, and later moved from the output buffer to an available AGV. In this paper, timed transitions will be used to model the vehicle travelling and loading/unloading time, and timed places will be used to model the machine process time. □ will be used as timed transitions and squares will be used as timed places.

The travelling time, $T_k$, is the sum of time required for each transition $t_k$, where $k$ is the transition number. If the vehicle speed $v$ is a constant value and $T_k$ is a function of the zone length $d_m$ (where $m$ is the zone number), the time required for a transition is estimated by equation (1). If the vehicle speed is $v_\ell$ and $T_k$ is a function of $d_m$ and $v_\ell$ (where $\ell = 1$ indicates high, $\ell = 2$ median, and $\ell = 3$ low speeds), the time function can be estimated by equation (2). If a random delay, $\delta$, is introduced to the time equation $T_k$, the equation will become a stochastic function as given in equation (3).

$$T_k = \frac{d_m}{v}$$  \hspace{1cm} (1)
$$T_k = \frac{d_m}{v_\ell}$$  \hspace{1cm} (2)
$$T_k = (\frac{d_m}{v_\ell}) + \delta$$  \hspace{1cm} (3)

$T_{p1}$, $T_{p2}$, $T_{n1}$ and $T_{n2}$ are the loading/unloading time required for timed transition's $t_{p1}$, $t_{p2}$, $t_{n1}$ and $t_{n2}$ respectively in a station macro. Normally, they are constant values specified by the user. $T_{MC}$ is the machine process time associated with timed-place $MC_j$ in the station macro. The process time for each machine is also specified by the user.
4. CP-CTPN representation of movement command sequences

By extending a 2-D floor-path PN model to a 3-D CTPN model, all tokens of vehicles and parts are coloured. The elapse time for vehicle travelling, loading/unloading and machine processing can be measured by timed-transitions and/or timed places. Once the vehicle is assigned and the route is determined, the movement command is generated and delivered to the CTPN model. The command is then used to move the vehicle. Command places are thus introduced to receive the movement command in this paper. Coloured tokens in command places represent contents of the command. The CTPN model can then be expressed in terms of a command-place coloured-timed Petri net (CP-CTPN) model.

One of the most commonly used AGVS control methods is the PLC control technique. The underground-vehicle-guidance wires are connected to the input/output modules of the PLC. Once the route of the vehicle is selected by the system computer, the PLC will switch the relays to turn on the underground-vehicle-guidance wire segments of the selected routes and turn off other wire segments. A relay in AGVS is considered as an external signal that delivers management commands via the floor-path wires, and the vehicle is actuated by the commands to move around. The AGVS driving mechanism is composed of relays and wires. The wires are represented by the command places $m_i$, and the relays are represented by coloured tokens. A token appeared in a command place can turn on a wire to move the vehicle. The command sequences are executed by putting the command tokens to command places $m_i$.

The colours of external-command-tokens are represented by $V_n$, where $n = 1, \ldots, N$ and $N$ is the total number of vehicles. The wire-transition will fire as the colour of the token in input-zone-place $Z$ matches that of the command place $m_i$. In other words, the transition will fire only when the token colour of the zone place is $V_{i-x}$ and that of the command place is $V_i$, where $x$ stands for empty, asgndj or lddj. After the transition is fired, the external-command-token $V_n$ is removed from the command place and the vehicle-token will appear in the next zone-place.

5. Marking evolution via transition firing

The characteristics of an AGVS are captured by the incidence matrix of the PN model. In the matrix, a row is used to stand for a place, and a column for a transition. Each column corresponds to the marking change when the associated transition is fired. It is noted that a negative element in the incident matrix stands for a situation that tokens are withdrawn from the associated place as the transition is fired. On the other hand, a positive element in the incident matrix stands for a situation that tokens are added to the associated place after a transition is fired. Therefore, the marking evolution of an AGVS depends on the incremental product of the incident matrix and the transition firing sequence. An enabled transition can fire at any time. When an enabled transition in a marking $M$ fires, a new marking $M'$ is reached according to the equation

$$M' = M + N \cdot S^T,$$

where $N$ is an $m \times n$ incident matrix, and $S$ is a $1 \times n$ firing vector. Since the change of marking, expressed by the product of incident matrix and transition firing sequence, is mainly induced by the system activities, the marking evolution in equation (4) is usually used to describe the system activity. Because the coloured incident matrix is such a formal representation of the place-transition-token relationship in a given AGVS, it is suggested to establish modular coloured incident sub-matrices for each flow-path sub-net and then assemble them into a complete floor-level coloured incident matrix. To make the assembly process efficient, it is required to develop a numbering scheme. The details of the procedure are presented below.

5.1. Coloured incident sub-matrices

The control place ($C_i$) in the floor-level model is developed so that the vehicle collision can be avoided. The command place ($m_i$) is used to transmit external commands to the floor level. The control-place token has one single colour and is always located at a local place. The command-place token is cleared after the associated transition is fired. Marking changes in both the command and the control places do not affect the vehicle status. To reduce the size of the incident matrix, one can consider the marking changes in the zone place portion alone. In other words, only the zone place portion of the coloured incident sub-matrix of each net can be constructed. The zone-place portion (which reflects the floor-path layout) of the coloured incident matrix is assembled. The size of the incident matrix thus obtained is reduced to at least one half of the original size. The zone-place portions of the coloured incident sub-matrices of five modular connecting nets (figures 3(a) – 3(e)) and the incident matrices of the station-zone macro net (figure 2) are constructed and illustrated as follows.
1. Coloured incident sub-matrix for the line net

\[ N = \frac{Z_1}{Z_2} \begin{bmatrix} t_1 & 0 & 0 & 0 \\ -V_{i-x} & V_{i-x} & -V_{i-x} & 0 \end{bmatrix}. \] (5)

2. Coloured incident sub-matrix for the divide net

\[ N = \frac{Z_1}{Z_2} \begin{bmatrix} t_1 & t_2 & t_3 & t_4 \\ V_{i-x} & 0 & 0 & 0 \\ -V_{i-x} & 0 & -V_{i-x} & 0 \end{bmatrix}. \] (6)

3. Coloured incident sub-matrix for the merge net

\[ N = \frac{Z_1}{Z_2} \begin{bmatrix} t_1 & t_2 & t_3 & 0 \\ V_{i-x} & V_{i-x} & -V_{i-x} & 0 \\ -V_{i-x} & 0 & 0 & 0 \end{bmatrix}. \] (7)

4. Coloured incident sub-matrix for the intersection net

\[ N = \frac{Z_1}{Z_2} \begin{bmatrix} t_1 & t_2 & t_3 & t_4 & t_5 & t_6 \\ V_{i-x} & V_{i-x} & -V_{i-x} & 0 & 0 & 0 \\ -V_{i-x} & 0 & 0 & 0 & 0 & 0 \\ 0 & -V_{i-x} & 0 & 0 & 0 & 0 \\ 0 & 0 & V_{i-x} & 0 & 0 & 0 \\ 0 & 0 & 0 & V_{i-x} & 0 & 0 \end{bmatrix}. \] (8)

5. Coloured incident sub-matrix for the one-dead-end bi-directional net

\[ N = \frac{Z_1}{Z_2} \begin{bmatrix} t_1 & t_2 \\ V_{i-x} & -V_{i-x} \end{bmatrix}. \] (9)

6. Coloured incident sub-matrix for the station-zone macro net

\[ MCn \begin{bmatrix} -\text{job} + \text{empty} & -\text{empty} + \text{job} \\ -\text{empty} + \text{job} & 0 \end{bmatrix} \]

\[ OB \begin{bmatrix} 0 & -\text{job} + \text{empty} \\ -\text{job} + \text{empty} & 0 \end{bmatrix} \]

\[ IB \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix} \]

\[ RS \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix} \]

\[ JS \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix} \]

\[ Z_p \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix} \]. (10)

Note: \(V_{i-x}\) stands for \(V_{i-empty}, V_{i-agndj}\) or \(V_{i-bdij}\).

5.2. Numbering rules for transitions and places

Six coloured incident sub-matrices for six basic flow-path and station-zone macro nets are presented above. The assembly of coloured sub-matrices is a process very similar to the assembly of modular sub-models. Once the floor-path layout is given, a complete incident matrix can be constructed by placing equivalent sub-matrices into system model one by one in compliance with the layout. Since the element location in incident matrix is determined by the numbering sequence of zone places and transitions, it is important for us to define a numbering rule as follows.

1. The first net is assumed the leftmost element in the first row of the input floor-path layout. The succeeding net element is located to the right of the preceding element in the same row, or the leftmost element of next row.

2. Since a one-dead-end bi-directional line net is attached to its mother net, its input sequence is the same as that of its mother net.

3. The number of first transition in the succeeding net is the number of last transition in the preceding net plus one.

4. The number of first place in the succeeding net is the number of last place in the preceding net plus one.

5. The number of the place or transition of a one-dead-end bi-directional line net may follow those places or transitions of the mother net.
After numbering places and transitions, the coloured incident matrix of a complete system model can be constructed by assembling modular sub-matrices of basic flow-path nets. The coloured incident matrix can be used to synthesize the system activities.

6. Model validation

A small manufacturing plan (Hsieh and Shih 1994) is taken as an example to show the effectiveness of the proposed method in this section. A more sophisticated case (Srinivasan et al. 1994) will be given in the next section for comparative study. The floor-path layout of the small manufacturing plant is given in figure 4. The figure gives two D nets, two M nets, one I net, two BL nets and several unmarked nets that are L nets. There are 26 zones, \( Z_1 - Z_{26} \), in the system. Stations \( S_1 - S_6 \) are located at \( Z_7, Z_5, Z_1, Z_{18}, Z_{10}, \) and \( Z_{26} \), respectively. The rest areas \( R_1 \) and \( R_2 \) of the vehicle are located at \( Z_6 \) and \( Z_8 \), respectively. The system is modelled as follows.

1. Specify the modules of a given floor-path layout, and generate the complete floor-path PN.
2. Specify station-zone places, \( Z_7, Z_5, Z_1, Z_{18}, Z_{10} \) and \( Z_{26} \), and replace them by macro places.
3. Specify vehicle numbers, job types, job schedules, zone distance, vehicle velocity, loading/unloading time, machine process time, deterministic or stochastic transition time, etc.
4. Construct and display floor-level CP-CTPN, as shown in figure 5.
5. Specify management policy and generate dispatching and routening rules to interface with the model via command places.
6. Generate coloured incident matrices, as shown in figure 6, and compute marking evolution.
7. Save animated graphs and simulation results.

The resulting floor-level CTPN is given in figure 5. It is noted that all transitions in the system are timed except \( t_{16}, t_{17}, t_{22}, t_{23}, t_{27}, \) and \( t_{28} \). These exceptional transitions are control transitions that have nothing to do with physical zones. \( Z_7, Z_5, Z_1, Z_{18}, Z_{10}, \) and \( Z_{26} \) are station-zone macro places. \( C_1, C_2, C_3, C_4, C_5 \) and \( C_6 \) are

![Figure 4. The floor-path layout of the plant.](image-url)
control places. \( C_2, C_4 \) and \( C_6 \) are superimposed with pseudo-zone or composite-pseudo-zone places (Hsieh and Shih 1992) \( Z_{12}, Z_{17} \) and \( Z_{22} \). Tokens in each control-loop place are zone-control signals. Places \( m_1 - m_{31} \) are command places that command the vehicle to move. It is noted that the command places, \( m_i \), are numbered after that of the transitions. This is because the main function of the command place is to turn the transition on and off, and therefore, the command place is associated with the transition.

Figure 6 shows the coloured incident matrix of the model. The functional value of each element in incident matrix represents the status of the vehicle in the system. Blanks that appear in figure 6 indicate that no direct relation exists between the transition and the place. It is noteworthy that there exists one pair of non-zero elements in each transition. This implies that every vehicle will eventually leave the place some time after it enters the place.

7. Numerical simulation and comparison

A floor-path layout (Srinivasan et al. 1994) is employed to generate the CP-CTPN model for simulation by using the proposed method. The floor-path layout for numerical experiment is given in figure 7. Since the modelling procedure is tedious and the resulting model is complicated, it will not be presented here. However, the simulation results are presented in table 1, where \( \alpha_e, \alpha_w, \alpha_l, \alpha_b \) represent the fractions of time that are required by a vehicle working in four different conditions, namely, travelling loaded, travelling unloaded, waiting in an idle state, and blocking
Figure 6. The coloured incident matrix of the zone-place portion.
Table 1. Simulation results of the four vehicle dispatching rules from the CP-CTPN model.

<table>
<thead>
<tr>
<th>Throughput</th>
<th>No. of vehicles</th>
<th>FCFS</th>
<th>MFCFS</th>
<th>STTF</th>
<th>MOD FCFS</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>$\alpha_f$</td>
<td>$\alpha_e$</td>
<td>$\alpha_l$</td>
<td>$\alpha_b$</td>
</tr>
<tr>
<td>DAT1</td>
<td>8</td>
<td>0.236</td>
<td>0.209</td>
<td>0.555</td>
<td>0.0188</td>
</tr>
<tr>
<td>DAT2</td>
<td>8</td>
<td>0.335</td>
<td>0.298</td>
<td>0.367</td>
<td>0.0315</td>
</tr>
<tr>
<td>DAT3</td>
<td>8</td>
<td>0.472</td>
<td>0.421</td>
<td>0.107</td>
<td>0.0494</td>
</tr>
</tbody>
</table>

Table 2. Simulation results of the four vehicle dispatching rules from the paper of Srinivasan et al. 1994 (N/A means 'not available', Srinivasan et al. do not consider vehicle blocking time).

<table>
<thead>
<tr>
<th>Throughput</th>
<th>No. of vehicles</th>
<th>FCFS</th>
<th>MFCFS</th>
<th>STTF</th>
<th>MOD FCFS</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>$\alpha_f$</td>
<td>$\alpha_e$</td>
<td>$\alpha_l$</td>
<td>$\alpha_b$</td>
</tr>
<tr>
<td>DAT1</td>
<td>8</td>
<td>0.220</td>
<td>0.193</td>
<td>0.587</td>
<td>N/A</td>
</tr>
<tr>
<td>DAT2</td>
<td>8</td>
<td>0.315</td>
<td>0.276</td>
<td>0.409</td>
<td>N/A</td>
</tr>
<tr>
<td>DAT3</td>
<td>8</td>
<td>0.443</td>
<td>0.389</td>
<td>0.168</td>
<td>N/A</td>
</tr>
</tbody>
</table>
during travelling. For comparison purpose the simulation results of Srinivasan et al. (1994) are listed in table 2. It is noteworthy that both fractions of time computed by the proposed method for a vehicle travelling loaded and unloaded are higher than those of Srinivasan’s. This is because the detailed functions of zone control and vehicle blocking are included in the present model. Therefore, the mean waiting time obtained by the present method is somewhat higher than that of Srinivasan’s.

8. Conclusions

This paper presents a modelling method to automatically construct and solve an AGVS floor-level model using a given floor-path layout. The physical activity of the AGVS is modelled by CPNs, while the system performance measure is modelled by TPNs. In the process of developing the AGVS model, a computer-aided environment was established. In the environment, the AGVS floor-level model and coloured incident matrices are constructed systematically with many detailed features that are not available in most general-purpose commercial programmes. By employing the model, one can evaluate the design alternatives of an AGVS quickly and reliably. The modular model concept is extended to build coloured incident matrices. The coloured incident sub-matrices for different floor-path sub-nets are developed. The complete coloured incident matrix for an AGVS can generally be constructed by the developed computer programme. Two examples are presented to validate the proposed method. Further study on developing a top-level AGVS management model is now in progress.

References


Lin, J. T., 1990, Determine how many AGVs are needed. Industrial Engineering, 22, 53–56.


Newton, D., 1985, Simulation model calculates how many automated guided vehicles are needed. Industrial Engineering, 17, 68–78.


