CONTROL OF INTELLIGENT MATERIALS HANDLING EQUIPMENT IN TEMPORARY FACILITIES

Robert B. Blackmon
U.S. Army Construction Engineering Research Laboratory
Champaign, Illinois USA

INTRODUCTION

Initial interest in robotics was the result of several studies concerning construction of large training camps with limited resources and within a relatively short time after the construction decision. An analysis was made of these projects to determine how computer-aided equipment could be used to expedite the work. It was found that the need for autonomous vehicles was the most common issue to be addressed. Potential applications for such vehicles included moving materials in an automated warehouse, operating a storage yard, moving supplies from the warehouse/storage yard to the erection site, participating in the erection and finishing process, installing utility systems, and developing the general site. Within this range, warehouse applications appeared to be the simplest, and finishing activities the most complex tasks to incorporate robots on the sites.

Nelson[1] discusses the potential for computerizing warehouse operations in all areas other than materials handling equipment. The use of autonomous equipment and easily read bar code markers on materials will enhance Nelson's approach to optimizing warehouse operations.

Autonomous navigation is an essential characteristic for mobile robots performing material handling functions, especially where floors will not have a special finish or be equipped with embedded tapes or other navigation aids.

In any successful system, the robot must be programmed with a physical layout of the operational area as the global model of the world. As robots move through the world collecting data from sensors, an abstract sensor model of the space will be developed to replace the initial global map. This local map is dynamically modified as the moving robots identify barriers and obstructions, and travelway segments are restored from the global model as such barriers are removed. This mapping describes a series of pathways or travelways within the operating area controlled by a series of operating assumptions. The next step is to search for a path between the current location of a mobile robot and the target location, and optimize that path based on some criterion. Optimizing criteria may be minimum time, shortest distance, minimum cost of traversal, etc.

The emergency of mobile robots has proved to be revolutionary in material handing. Depending on the installed equipment and the intelligence of the controlling software, these devices can have such characteristics as machine learning, self-correcting ability, and unlimited

---

[1]Nelson, Raymond A., Computerizing Warehouse Operations, Prentice-Hall, Inc., 1985.

flexibility. This paper covers development of path planning and navigation algorithms for mobile robots operating in a temporary construction warehouse.

BACKGROUND

A modular warehouse design was selected as the basis for the material handling system. The layout consisted of racks to store pelletized items. The racks were grouped into modules separated by aisleways. The modules, Figure 1, measured 60 by 60 feet and were located back to back to define an area of 60 by 120 feet. The crosshatched areas shown in the figure represent a separation between the freeways and the alleys; in actual floor layout this may be a yellow strip on the floor. The modules were grouped to form a warehouse layout as shown in Figure 2. A high speed highway was provided along the perimeter of the warehouse to simplify movement. (The highway is shown widely separated from the modules for clarification only.) A system of travelways was designed to provide access to all storage modules. All travelways contained two lanes of traffic with direction of travel on each lane predetermined. Average travel speeds were set for each type of travelway. Our work was directed toward getting the robot from any starting point to the entrance to the target module. While the process can be extended to the bin location, our discussion will be limited to the module level.

Storage modules are identified in terms of location as shown in Figure 3. In this partial view of the warehouse, the one-way pathways are shown in all corridors. Each type of pathway can be assigned a different average velocity. The highway is given the highest speed and the alleys the lowest, which allows the robots to use the high speed highway to reach the most convenient entrance to the storage area (e.g., A, B, AA) in the shortest time.

The layout can be represented as a network using the intersections as nodes as shown in Figure 4. A robot entering the storage section at gate A would first contact A1, the entrance to module A, then A1A3, the exit corner of the module A. The robot would then cross the intersection to A3, the entrance corner for the module A3, and continue along the path in the same manner. In reading this chart, remember that the storage modules are back to back. Module entrance corners are designated with two descriptors: path (A, B, C, etc.) and module number (1,2, etc.).

Path planning can now be described as a series of nodes. For example, a path from entrance A to B5 would be:

A - B - BB1 - B1 - B1B3 - B3 - B3B5 - B5

Initial path planning will be accomplished before the robot starts to move. The system develops a decision tree, Figure 5, of the entire system then uses an A* search to find the optimal path. Obstructions known to the central computer will be used in the initial path planning. For example, on Figure 4, if path segment B1B3 - B3 is blocked, the robot could use the following path to avoid the blocked segment:

A - B - C - CC1 - C1 - C1C3 - C3 - C3C5 - C8 - B3B5 - B5 or

A - B - BB1 - B1 - B1B3 - B3 - C8C10 - C3 - C3C5 - C8 - B3B5 - B5

Collision avoidance between traveling robots is accomplished by establishing a Node-Arrival time chart for each operating robot. Every node in a robot's path has an associated arrival time based on the distance and the average velocity traveled. When arrival times of different robots at a particular node fall within a specific tolerance limit, it implies a change of collision at that node. Collisions are avoided by altering the arrival times of the robots based on assigned priorities. The robot with the lowest priority or, if of equal priority, the latest start time, will be delayed for a fixed interval and the arrival time table recalculated and analyzed for possible conflicts.

Landmarks are wall-hung bar-code strips, patterns, or signals that were assumed to be located at all decision points in the travelway network. They are shown as a series of X's on Figures 1 and 2. The landmarks were assumed to provide the information needed to confirm the location of the robot or to serve as the basis for correcting its course, adjusting its speed, determining its orientation, or performing other essential navigational functions. Use of landmarks reduce the need for sensors and intelligent control software on the mobile robot.

The planning method demonstrated above has been referred to as the alternate planning process. An alternative path is automatically planned as obstructions are identified. Robots move along the planned travelway and at each landmark check the next segment of the travelway to see if it is clear. If the segment is not clear, the robot will scan the segment again. If the obstruction has not changed distance from the robot, it is assumed to be a static obstruction and an alternate path must be planned. Alternate plans involve going around modules to gain a clear path. The alternate path will be coordinated with the central computer and the obstruction recorded for reference in future planning. Upon successful completion of path planning the robot will initiate movement along the path. The robots would perform this planning function as many times as necessary to reach the target module or to decide that it is impossible and return to the starting point.

If the obstruction has increased in distance from the robot, it is assumed to be another robot and the second robot will continue on its planned path.

Bypass Planning

It was quickly realized that an obstruction could suddenly appear in the pathway segment already occupied by a robot. This could happen if something were spilled or dropped from the preceding robot or fell from the storage module. To handle this situation, the conceptual robot had to be given more intelligence. The robot needs to periodically check the path ahead for the presence of an obstruction rather than just checking from the nodes. Upon detection of such an obstruction, the robot must stop, make several measurements to define the obstruction, and then take one of the bypass activities discussed below. The obstruction is reported

to the central computer and all other robots plan alternative paths to avoid the obstructed pathway segment.

The bypass activities are performed as follows:

1. The distance between the obstruction and the righthand wall is measured to determine if there is enough space for the robot to pass (Figure 6). This may mean changing the centerline of the robot's path to bring it closer to the wall and then repositioning the robot after it has passed the obstruction. If space is available, the robot will modify its course and continue to follow the planned path. It was assumed in this example that the robot is 2 units wide, the lane is 4 units wide, and the travelway is 8 units wide.

2. If the lane is blocked and there is insufficient room for the robot to continue forward in its own lane, the system will determine the distance between the obstacle and the left wall. If adequate space is available, the robot will move into the left lane, go around the obstacle, and return to the proper lane. In Figure 7, there is adequate clearance in the lane and the robot can continue its planned path. In bypassing an obstacle in the left lane, the robot is traveling in the wrong direction in a lane of the travelway, so it must be able to return to its planned path after clearing the obstacle. To do this, it must be able to sense the presence of the obstacle during the bypass and scan forward to ensure a clear path during this operation. If the bypass is blocked, the robot must reverse its travel and return to the last node, Figure 8, and plan an alternate path.

Collision avoidance in the bypass operation depends on two activities. The bypass decision is immediately followed by a path search from the current positions of robots other than the bypassing robot to their respective targets. The general collision avoidance routine is used to compare identical nodes in the robot's paths and alter the arrival times if necessary. The bypass collision avoidance routine is then used to control traffic along the bypassing lanes to avoid possible collisions.

The various steps of the collision avoidance algorithm are explained in the flow chart, Figure 9. The following definitions are used in the flowchart.

BR = Robot that is carrying out the bypass operations; also referred to as the bypassing robot.

N(BR,I) = Node through which the robot (BR) bypasses; also referred to as the bypass node. For example, in Figure 10, B8 is the bypass node.

N(BR,I+1) = Node following the bypass node in the path of the bypassing robot (e.g., node B3B5 in Figure 10).

N(BR,I-1) = Node preceding the bypass node in the path of the bypassing robot (e.g., the position of the bypassing robot when the obstacle was traced, also referred to as Cur-pos).

SUCC(BR,I) = Successor node of the bypass node that lies in the same segment of the path as that of the bypassing robot. In the nodal representations of the layout, the relationship between the nodes is obtained by establishing pointers for each node. The nodes to which the pointer points are called the successors of the node in question. For the example, node B8 is the bypassing node in the path segment B8 - B8B10. The successors of B8 are B8B10 and A3A5; therefore SUCC(BR,I) is B8B10.

For the robot R under consideration,

N(R,J) = the current node in the path

T(R,J) = Arrival time at N(R,J).

The bypass algorithm can be summarized in the following steps:

1. The decision to bypass results in an additional node (the bypass node) added to the path of the bypassing robot. This in turn results in a recomputation of the arrival times at the nodes for the bypassing robot. Any alterations in the node-time chart for one robot may tend to cause collision at identical nodes in the paths of the robots. This situation is resolved by working the general collision avoidance algorithm at the beginning.

2. The priority of each robot is compared to that of the bypassing robot. If a robot's priority is higher than that of the bypassing robot, then alteration, if any, of the arrival times is done on the nodes of the bypassing robot. On the other hand, if a robot's priority is less than that of the bypassing robot, then the arrival time at the nodes of that robot is susceptible to alterations.

3. For each robot containing the bypass node, the nodes preceding and following the bypass nodes are compared to the nodes in the path segments of the bypassing robot that contains the bypass node. Rules mentioned in the flow chart, Figure 9, are then applied to avoid collision.

4. The execution of collision avoidance algorithm for bypassing may result in changes in the arrival times at the nodes in the path of the robots. Note that changes are made by comparing the path of the bypassing robot with each robot. No effort is made at this stage to check if the changes in arrival times at the nodes (bypassing node, nodes following and preceding the bypass node) can cause collision at the other identical nodes that may be present in the path of the robots. Thus the general collision avoidance algorithm is evoked at the end of the algorithm.

The collision-avoidance algorithm for bypassing is invoked only when bypassing operation has to be performed. It is always preceded and followed by the general collision avoidance algorithm as described earlier. Notice that in the collision-avoidance for bypassing, the bypassing robot is always penalized when higher priority robots are involved. This is because bypassing is a slow and complicated operation involving frequent stops and sensor interaction and thus is performed last.

Example of Collision Avoidance Algorithm for Bypassing

The performance of the collision avoidance algorithm for bypassing could be better understood using the following example. The various steps of the algorithm are illustrated in the flow chart given in Figure 9.

Consider a three-robot working environment. ROB1 starts from node B3 (Figure 10) and moves to the target node B7. ROB2 moves from node B6 to node B10 and ROB3 moves from node C6C8 to node B10. Let ROB2 have the highest priority (=3) and ROB3 the lowest priority (=1). The path planning algorithm yields the following paths for the robots after carrying out a A* from their start nodes to the goals.

ROB1: B3 → B3B5 → B5 → B5B7 B7
ROB2: B6 → B6B8 → B8 → B8B10 → B10
ROB3: C6C8 → C8 → B3B5 → B8 → B8B10 → B10

The general collision avoidance algorithm is executed next to modify the node-time chart for each robot.

Let the arbitrary node-time chart for a three-robot case generated by the general collision avoidance algorithm be:

ROB1 (Priority = 2)		ROB2 (Priority = 3)		ROB3 (Priority = 1)	
Node	Arr-Time	Node	Arr-Time	Node	Arr-Time
B3	30	B6	0	C6C8	20
B3B5	70	B6B8	40	C8	40
B5	90	B8	60	B3B5	60
B5B7	130	B8B10	100	B8	80
B7	150	B10	20	B8B10	120
				B10	140

(Note that the assignment of the arrival times to the node is based on the assumption that it takes 20 time units to travel from the intersection nodes (e.g., B3B5) to main modular nodes (e.g., B5) and 40 time units when traveling from main module nodes (e.g., B5) to intersection nodes (e.g., B5B7). Let ROB1 trace an obstacle while traveling between B3 and B3B5 and decide to bypass. The newly computed path for ROB1 is

Cur-Pos → B8 → B3B5 → B5B7 → B7,

(where Cur-Pos refers to the current position of the robot) while search for paths for ROB2 and ROB3 from their current positions to respective goals will yield paths for ROB2: Cur-Pos → B6B8 → B8 → B8B10 → B10 and for ROB3: Cur-Pos → C8 → B3B5 → B8 → B8B10 → B10. The collision-avoidance for bypassing algorithm is now called into operation (see the flowchart in Figure 9).

Step 1:

BR = ROB1
N(BR,I) = B8
N(BR,I+1) = B3B5
N(BR,I-1) = Cur-Pos.
SUCC(BR,I) = B8B10

The node-time chart after the execution of the general collision avoidance algorithm is:

ROB1 (Priority = 2)		ROB2 (Priority = 3)		ROB3 (Priority = 1)	
Node	Arr-Time	Node	Arr-Time	Node	Arr-Time
Cur-Pos	35	Cur-Pos	35	Cur-Pos	35
B8	70	B6B8	40	C8	40
B3B5	90	B8	60	B3B5	60
B5	110	B8B10	100	B8	80
B5B7	150	B10	120	B8B10	120
B7	170			B10	140

(Note that the assumption here is that obstacle was traced by ROB1 after traveling for five units from node B3. Hence arrival time at Cur-Pos is assigned 35.)

Step 2: R = ROB2
Compare the nodes in the path of ROB2 with B8
B8 is a node in the path of ROB2: (R,J) = B8
Priority of ROB2 > Priority of ROB1

Step 3: N(R,J-1) = B6B8; N(R,J-1) $\neq$ N(BR,I+1)
N(R,J+1) = B8B10 = SUCC(BR,I)
T(R,J) = 60; T(BR,I) = 70; T(R,J) < T(BR,I): Collision is possible
N(R,J+1) = B8B10
T(R,J+1) = 100
N(BR,I-1) = Cur-Pos
Alter T(BR,I-1) to 100 + 5 = 105
Modified Node-Time Chart:

ROB1 (Priority = 2)		ROB2 (Priority = 3)		ROB3 (Priority = 1)	
Node	Arr-Time	Node	Arr-Time	Node	Arr-Time
Cur-Pos	105	Cur-Pos	35	Cur-Pos	35
B8	140	B6B8	40	C8	40
B3B5	160	B8	60	B3B5	60
B5	180	B8B10	100	B8	80
B5B7	220	B10	120	B8B10	120
B7	240			B10	140

Consider the path of ROB3:

R = ROB3; B8 is a node in the path of ROB3
Priority of ROB3 < Priority of ROB1: N(R,J) = B8
N(R,J-1) = B3B5 = N(BR,I+1)

T(R,J-1) = 60; T(BR,I+1) = 160
T(R,J) = 80 T(BR,I) = 140
T(R,J-1) < T(BR,I+1) and
T(R,J) < T(BR,I)
N(R,J+1) = B8B10 = SUCC(BR,I)
T(R,J) (=80) < T(BR,I) (=40)
N(R,J=1) = B3B5

Alter T(R,J-1) TO 140 + 5 = 145

Modify Node - Time Chart:

ROB1 (Priority = 2)		ROB2 (Priority = 3)		ROB3 (Priority = 1)	
Node	Arr-Time	Node	Arr-Time	Node	Arr-Time
Cur-Pos	105	Cur-Pos	35	Cur-Pos	35
B8	140	B6B8	40	C8	40
B3B5	160	B8	60	B3B5	145
B5	180	B8B10	100	B8	165
B5B7	220	B10	120	B8B10	205
B7	240			B10	225

T(R,J-1) = 145; T(BR,I+1) = 160
T(R,J) = 165; T(BR,I) = 140
T(R,J-1) < T(BR,I+1) and T(R,J) > T(BR,I)
Alter T(R,J-1) to 160 + 5 = 165

Modify Node-Time Chart:

ROB1 (Priority = 2)		ROB2 (Priority = 3)		ROB3 (Priority = 1)	
Node	Arr-Time	Node	Arr-Time	Node	Arr-Time
Cur-Pos	105	Cur-Pos	35	Cur-Pos	35
B8	140	B6B8	40	C8	40
B3B5	160	B8	60	B3B5	165
B5	180	B8B10	100	B8	185
B5B7	220	B10	120	B8B10	225
B7	240			B10	245

T(R,J) (=185) > T(BR,I) (=140)
No more robots.

Step 4:

Execute the general collision avoidance algorithm. Since the arrival times at the identical nodes are different, no alteration of arrival times is necessary. The node-time charts remain the same.

TESTING

The path planning and collision avoidance algorithms have been successfully tested in a simulation model using six robots moving within the same area of the warehouse and several starting simultaneously from the same point.

When two robots are moving along the same path and the first identifies an obstacle, the second robot will find an alternate path to its target destination. The obstacle will be reported to the central computer which will cause all other robots to avoid that segment of the path. Realizing that robots could continue to use the path if the obstacle was similar to that shown in Figure 4, it was decided to impose this rule to avoid having each robot entering the segment perform the same time-consuming sensor search and calculations.

FUTURE ACTIVITIES

The work is continuing on this project. Short-term goals include the development of (1) a user friendly front end for the program to simplify the process of describing the warehouse layout and milestone locations to the program, (2) algorithms for determining layout of the storage modules in an existing space, and (3) technical requirements for the sensor systems. Retrofitting materials handling equipment and prototype testing will follow in the outyears.

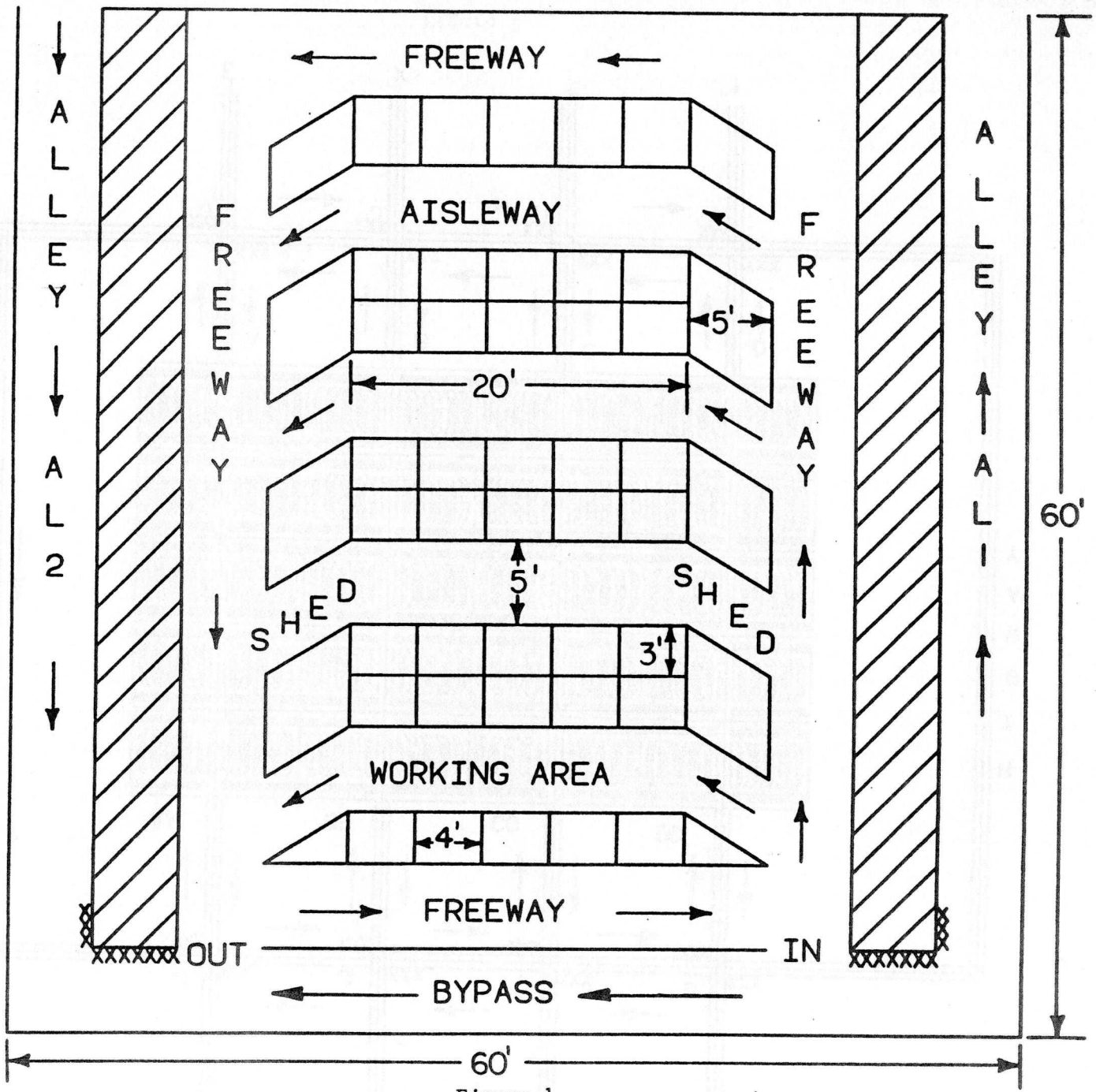
FREEWAY
ALLEY ALL 2
FREEWAY
AISLEWAY
FREEWAY
5'
20'
5'
SHED
SHED
3'
WORKING AREA
4'
ALLEY ALL 1
60'
FREEWAY
OUT
IN
BYPASS
60'

Figure 1

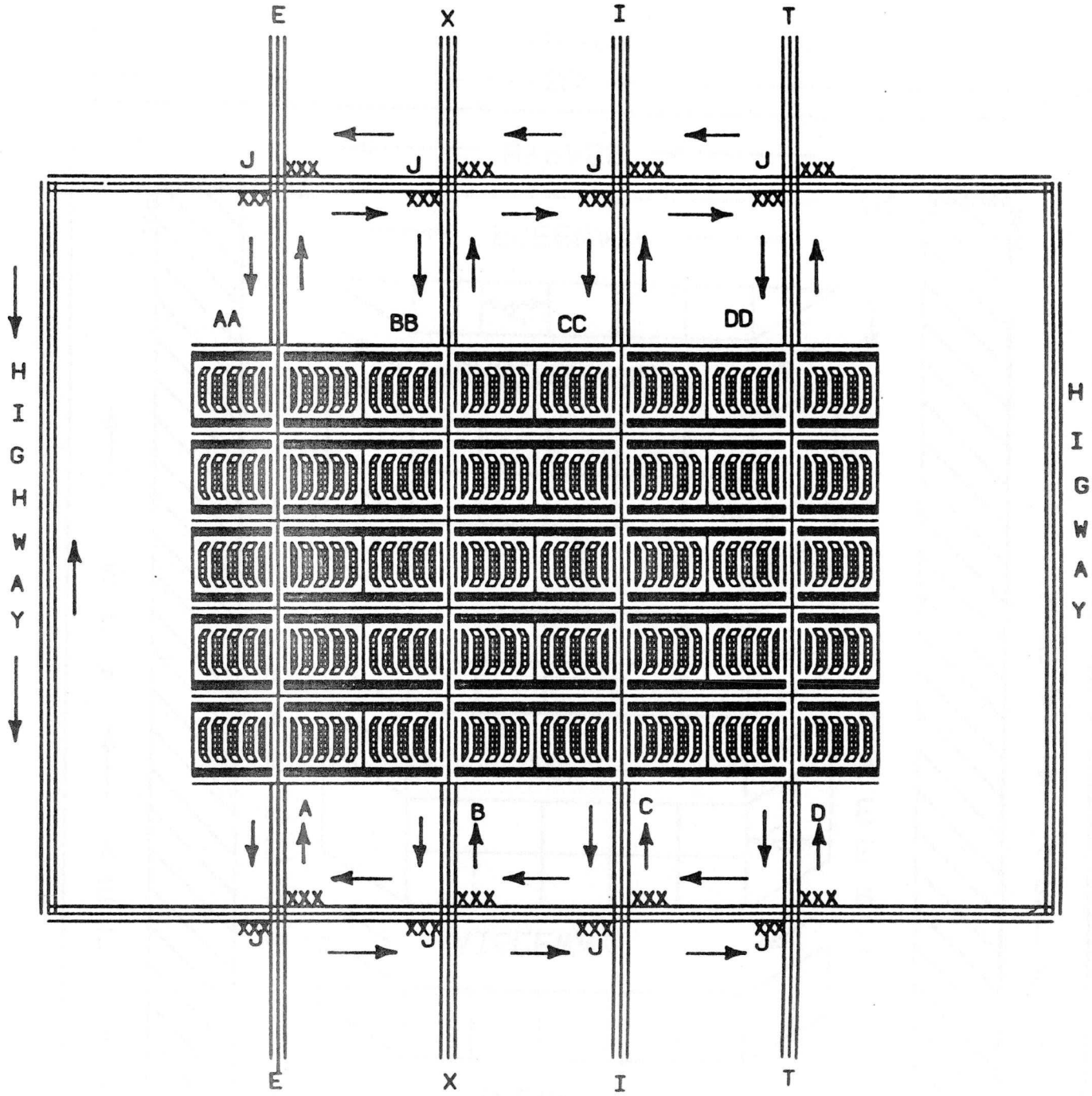

Figure 2

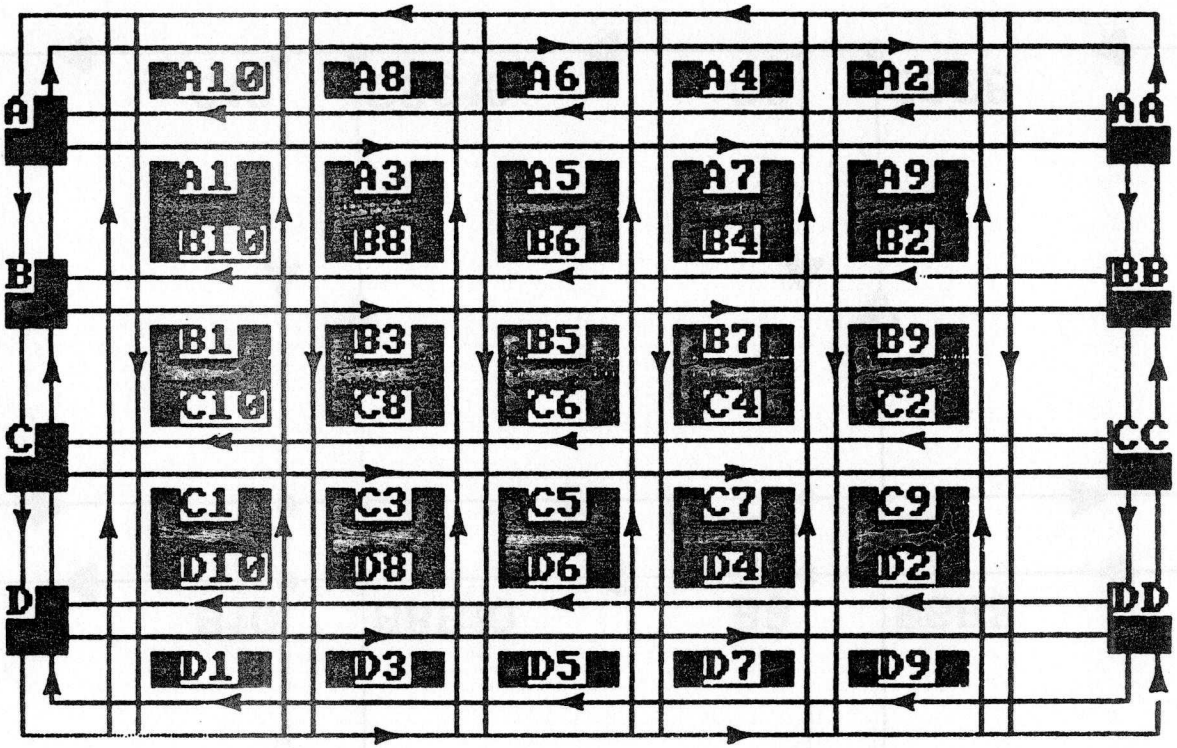
A10
A8
A6
A4
A2
A
AA
A1
B10
A3
B8
A5
B6
A7
B4
A9
B2
B
BB
B1
C10
B3
C8
B5
C6
B7
C4
B9
C2
C
CC
C1
D10
C3
D8
C5
D6
C7
D4
C9
D2
D
DD
D1
D3
D5
D7
D9

Figure 3

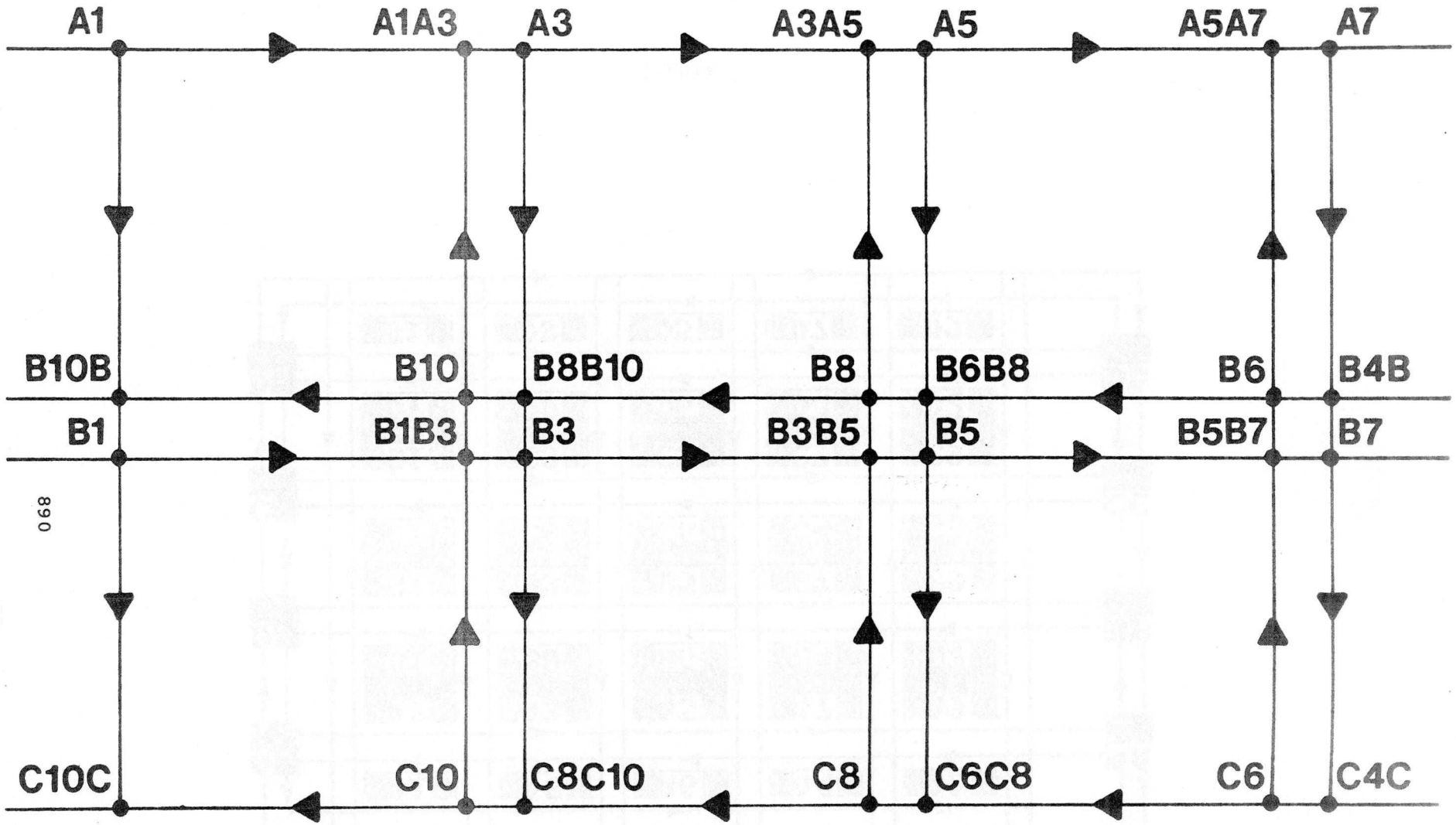
A1
A1A3
A3
A3A5
A5
A5A7
A7
B10B
B10
B8B10
B8
B6B8
B6
B4B
B1
B1B3
B3
B3B5
B5
B5B7
B7
068
C10C
C10
C8C10
C8
C6C8
C6
C4C

Figure 4

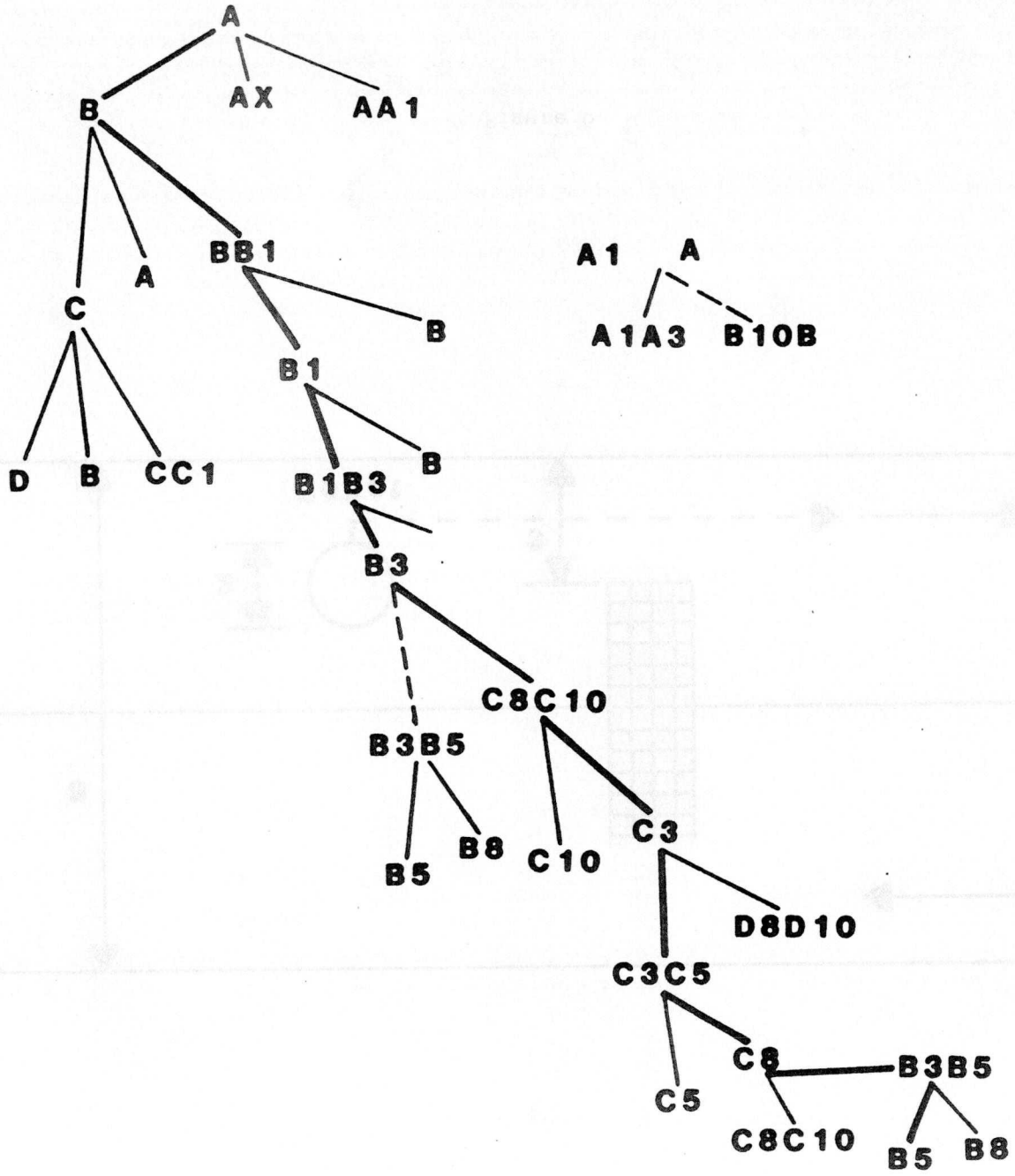
A
AX
AA1
B
BB1
A
C
B
B1
D
B
CC1
B1B3
B
B3
C8C10
B3B5
B5
B8
C10
C3
D8D10
C3C5
C8
B3B5
C5
C8C10
B5
B8
A1
A
A1A3
B10B

Figure 5

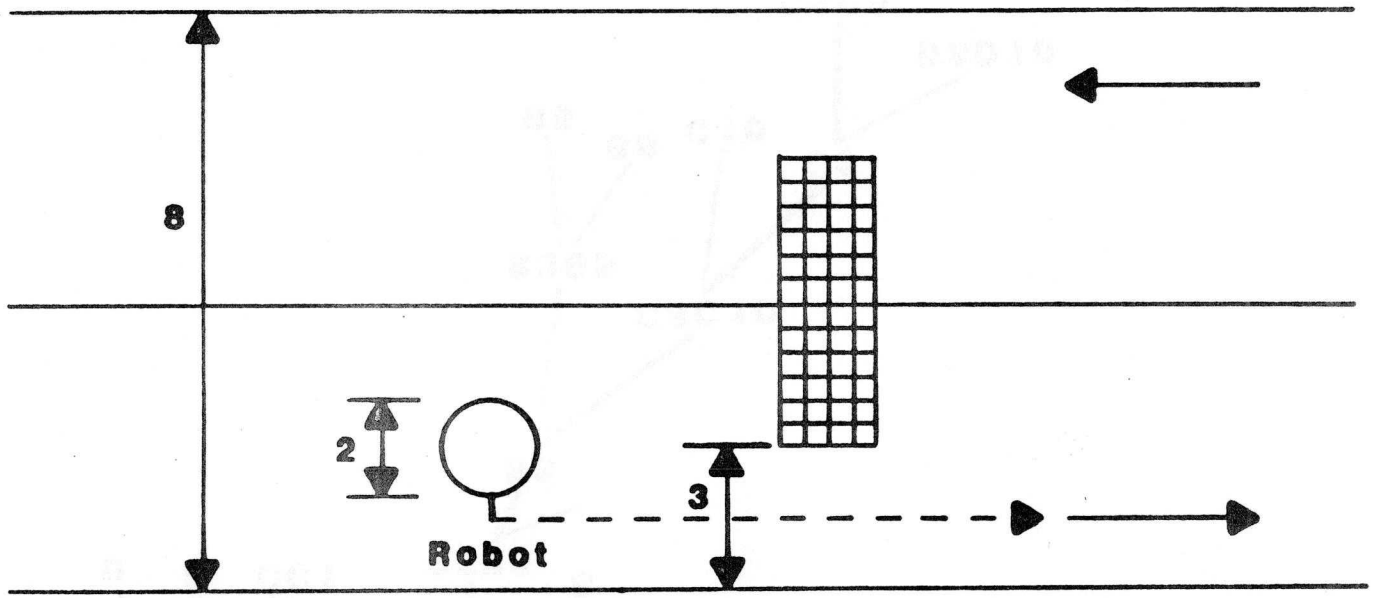

Figure 6

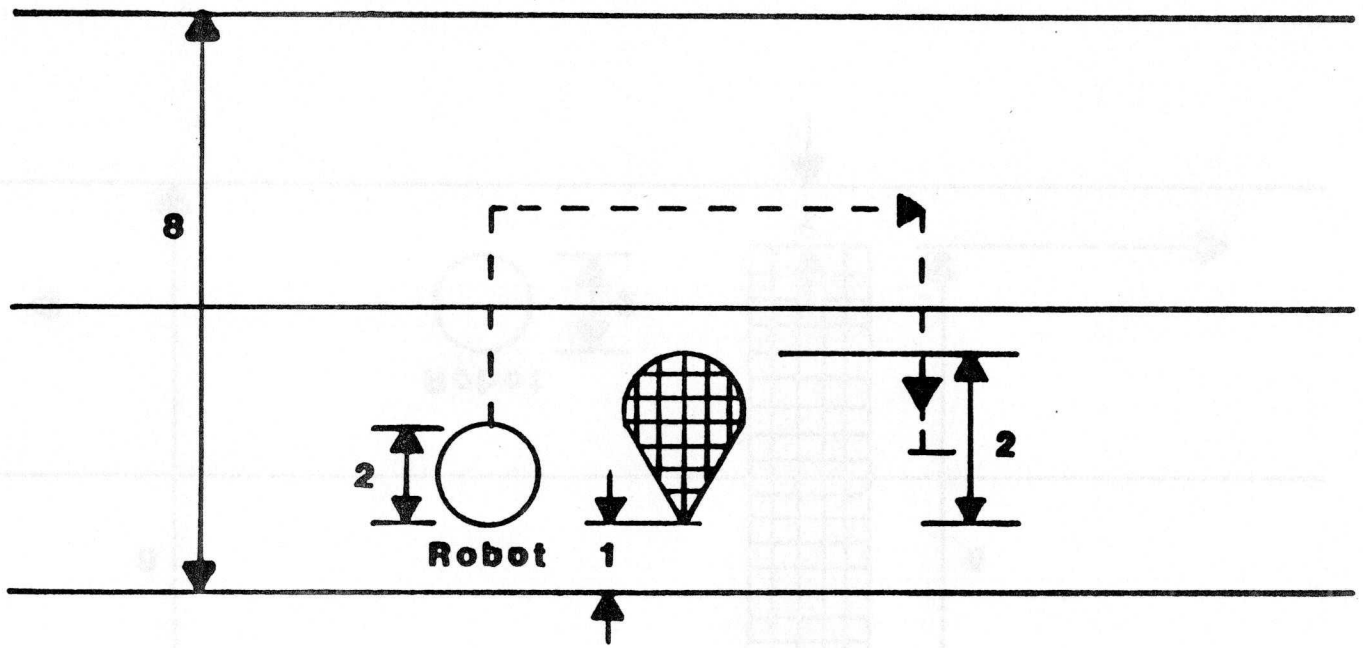

Figure 7

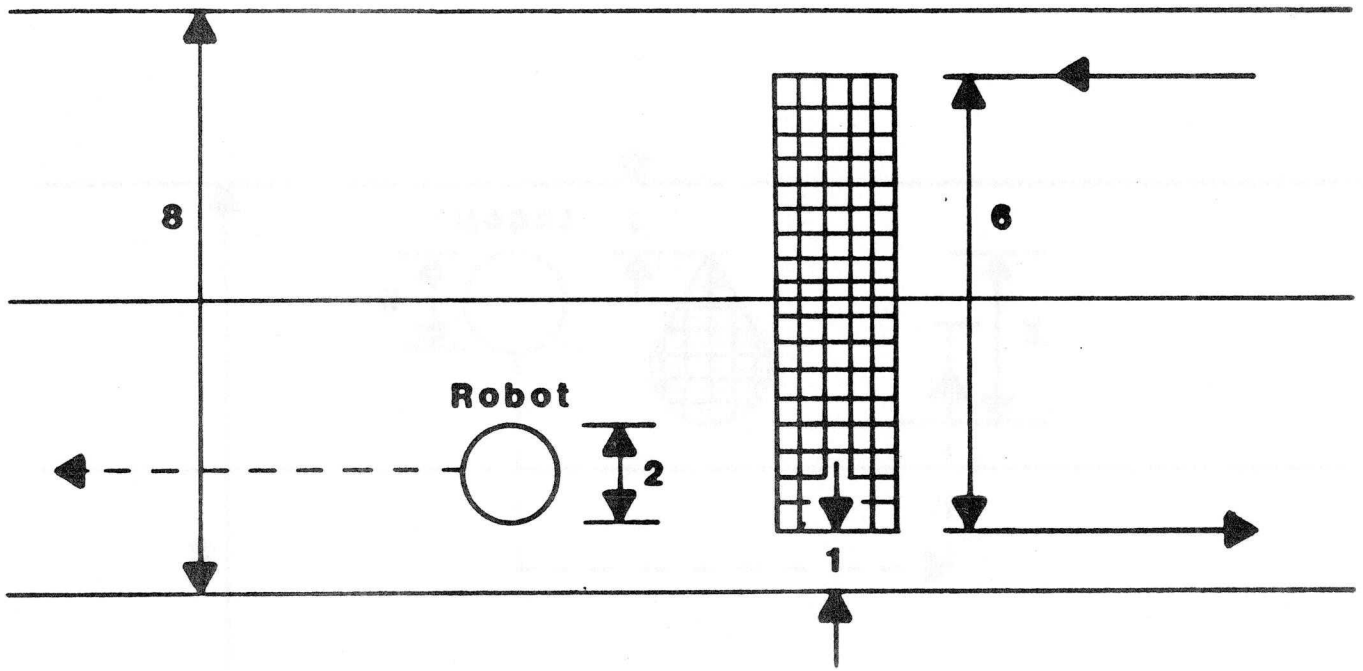

Figure 8

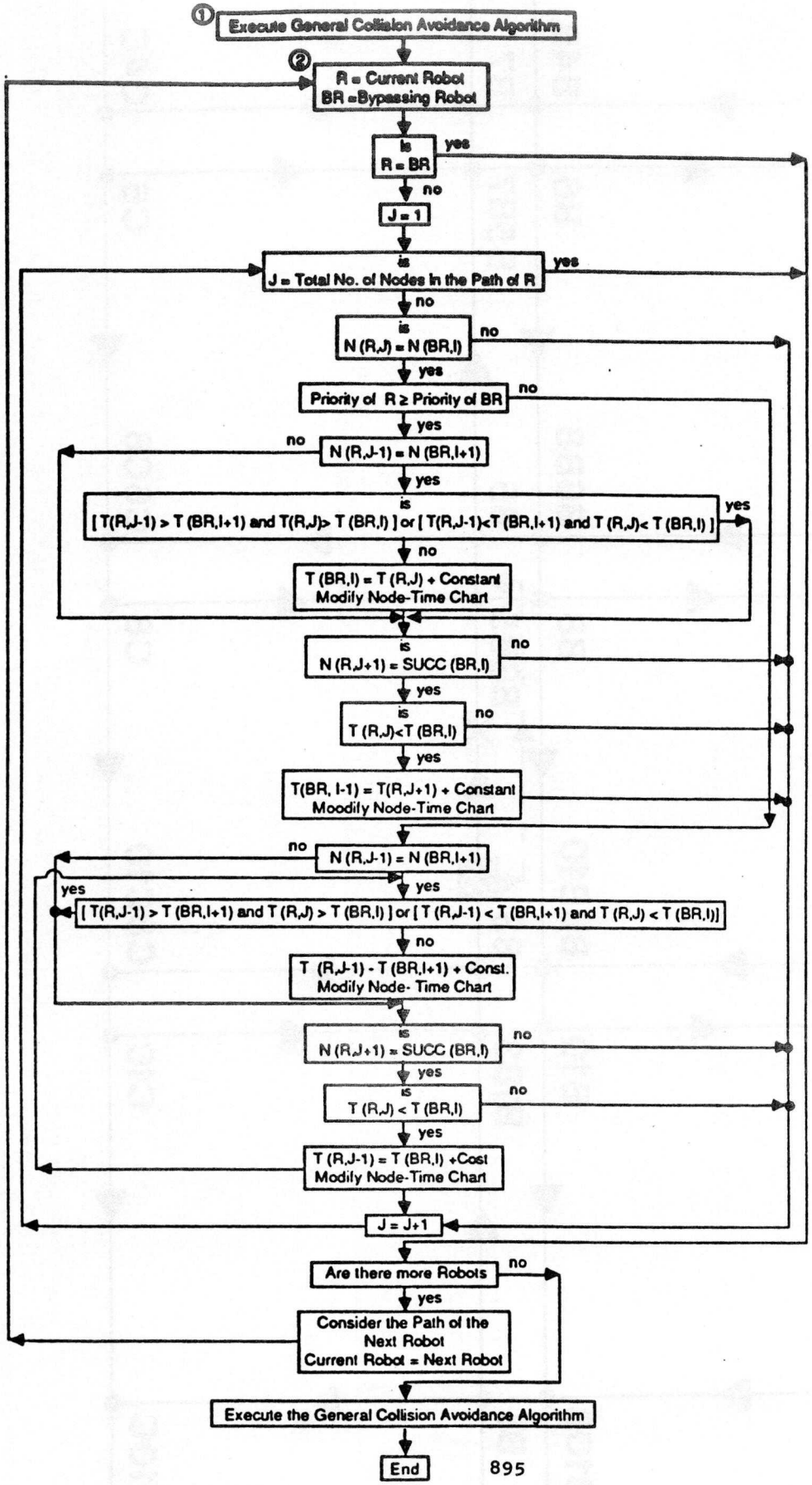
1
Execute General Collision Avoidance Algorithm
2
R = Current Robot
BR =Bypassing Robot
Is
R = BR
yes
no
J = 1
is
J = Total No. of Nodes in the Path of R
yes
no
is
N (R,J) = N (BR,I)
no
yes
Priority of R ≥ Priority of BR
no
yes
no
N (R,J-1) = N (BR,I+1)
yes
is
[T(R,J-1) > T (BR,I+1) and T(R,J)> T (BR,I)] or [T(R,J-1)<T (BR,I+1) and T (R,J)< T (BR,I)]
yes
no
T (BR,I) = T (R,J) + Constant
Modify Node-Time Chart
is
N (R,J+1) = SUCC (BR,I)
no
yes
is
T (R,J)<T (BR,I)
no
yes
T(BR, I-1) = T(R,J+1) + Constant
Moodily Node-Time Chart
no
N (R,J-1) = N (BR,I+1)
yes
yes
[T(R,J-1) > T (BR,I+1) and T (R,J) > T (BR,I)] or [T (R,J-1) < T (BR,I+1) and T (R,J) < T (BR,I)]
no
T (R,J-1) - T (BR,I+1) + Const.
Modify Node- Time Chart
is
N (R,J+1) = SUCC (BR,I)
no
yes
is
T (R,J) < T (BR,I)
no
yes
T (R,J-1) = T (BR,I) +Cost
Modify Node-Time Chart
J = J+1
Are there more Robots
no
yes
Consider the Path of the
Next Robot
Current Robot = Next Robot
Execute the General Collision Avoidance Algorithm
End

Figure 9

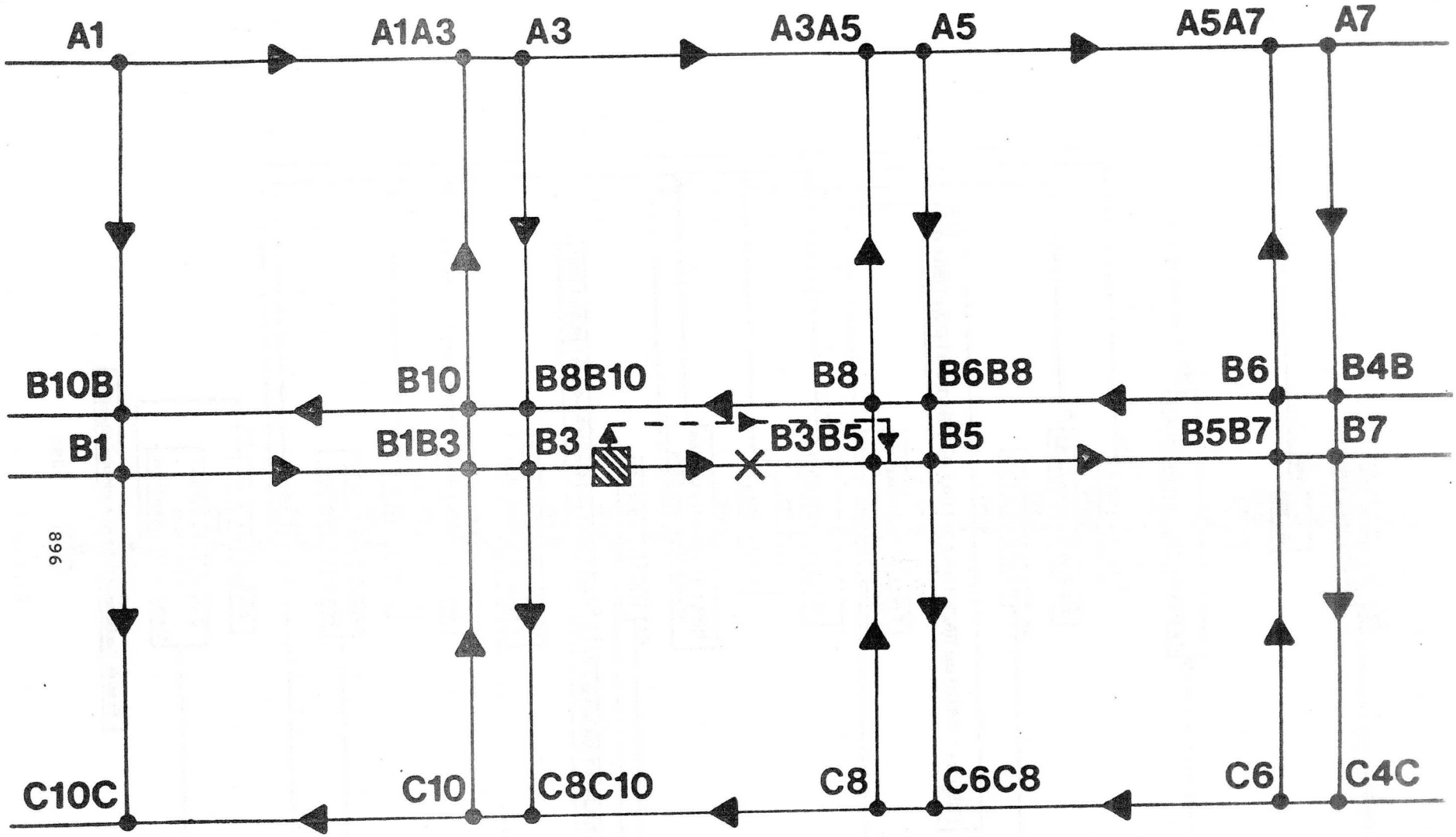
A1
A1A3
A3
A3A5
A5
A5A7
A7
B10B
B10
B8B10
B8
B6B8
B6
B4B
B1
B1B3
B3
B3B5
B5
B5B7
B7
C10C
C10
C8C10
C8
C6C8
C6
C4C

Figure 10