Paper 114, IT302

Routing Map for Automated Robotic Part Delivery System in Manufacturing Environments

[1]Payam H. Matin,[1]Ali Eydgahi, and [2]Kenneth Burch

[1]Department of Engineering and Aviation Sciences
University of Maryland Eastern Shore

[2]TIME Center: An NSF sponsored ATE Center
School of Applied and Information Technology
Community College of Baltimore County
phmatin@umes.edu , aeydgahi@umes.edu , and kburch@ccbcmd.edu

Abstract

This paper presents a student research project as a supplement to the undergraduate curriculum for students in the area of Science, Technology, Engineering and Mathematics. The goal of the project presented in this paper is to provide students with experience in design of a routing path for an automated robotic part delivery system in manufacturing environments. Despite all advancement in automation of manufacturing processes, still there are many challenges that need attentions in automating part transformation between manufacturing stations on an unstructured plant. In this study, first a mathematical model of a plant layout containing static obstacles is constructed. A partitioning algorithm is then introduced that partitions the layout into obstacle-free regions. Next, a searching algorithm is utilized to yield all possible combinations of regions that can connect a starting point to a destination throughout the plant layout. Finally, physical paths are constructed by drawing line segments within the obstacle-free regions and through the intersections between the regions. A program in C language based on the proposed algorithm has been developed. Excellent level of success has been measured in the performance of the proposed algorithm and the designed program codes. This study could be potentially beneficial to the industry since it tends to remove the labor costs.

Introduction

Automated Guided Vehicles (AGVs) represent an integral component of today's manufacturing processes. They are widely used on plant floors for intra-factory transferring of parts between conveyors and assembly lines, loading, and unloading. The use of Automated Guided Vehicles aligns to the philosophy of flexible manufacturing systems, where 24-hour-a-day use of machines rather than human is utilized for performing repetitive tasks. This approach helps reduce manufacturing cost and increase efficiency in a manufacturing system [1]. Although the use of AGVs is helpful, operating them in an unstructured manufacturing environment could be technically challenging and economically very costly. Challenges associated with design and operation of AGVs may be named as routing, navigation and guidance, traffic management, load transfer, and system management [2]. Many authors have shed light on these key challenges [3]-[5]. The first challenge in design of an AGV is routing, which refers to system's ability to make decision in order to select optimum routs to specific destination [4],[6]. If optimum routs are selected using a routing system, the guided vehicle is supposed to follow the predetermined rout [7]. Navigation and guidance system is designed to make this possible. As an Automated Guided Vehicle navigates through a path, it should be able to avoid collisions with other operating vehicles and obstacles on the field [8]. Traffic management system addresses such important task [9]-[10]. Furthermore, since the system should be able to load and unload goods, a load transfer system should manage these operations. Finally, a system management is needed to have full provision over all components of the systems for efficiency in operation [11]-[12]. Despite all advancement made in AGV systems, many challenges still remain unsolved, which could be viewed as an opportunity for more advancement and development in this area. Also, as these systems are becoming more popular in today's industry, any developmental activity by students in this area that addresses existing challenges of various parts of the AGV systems would contribute to the educational and learning experience of students.

There are different types of AGV in today's industry. Wired AGV may be named as one of the original one. On one hand a wire sensor is placed on the bottom of the AGV, and on the other hand a wire is placed in a slot cut in the ground. The radio frequency is transmitted from the wire to the sensor. The sensor follows the path through which the wire has been installed. The path followed is permanent since the wire is installed permanently in the ground. Additionally, the wire needs to be energized. Tape AGV may be named as more flexible type of AGV. A colored or magnetic tape is placed on the ground. A compatible sensor installed on the bottom of the AGV follows the path of the tape. Tape AGV is more flexible since the tape can be removed or redirected. However, tape may be easily damaged particularly in high traffic areas. Laser navigated AGV is more advanced type that navigates using laser. The AGV transmits laser signals, and receives them back from so called retro-reflective tape on walls to calculate the current position for navigation purposes.

A comprehensive effort has started by a team of undergraduate engineering students to design and prototype a Robotic System that will completely automate part transformation between different manufacturing stations. To accomplish this task, real-world data from industry part transformation scenarios are used to design and prototype a Robotic Part Delivery System. This project helps students get exposed to a number of emerging areas in Mechanical Engineering, Electrical Engineering, and Computer Science. Graduate students with math and computer science background have also joined the team to help with mathematical, computational and programming needs of the project. They also gain experience in making a working prototype system, doing engineering analysis, and creating engineering documents/drawings as required for manufacturing of a Robotic System. All the activities mentioned are in support of the nation in STEM area.

This paper focuses more on the modeling approach that has been taken to design the routing algorithm of a part delivery system. The objectives of this work include:

- To make a realistic model of a manufacturing environment layout,

- To develop an algorithm for routing purposes,
- To implement the algorithm to a number of realistic scenarios for validation purposes.

Modeling Approach

The routing system is an algorithm that makes decision in order to select optimum path to a specific destination from a starting location. Before the routing system can be discussed, a model of plant layout should be considered. A plant layout may be generally viewed as what has been depicted in Figure 1. The plant boundary is considered as a rectangle. The obstacles such as partitioning walls or machines are considered as vertical and horizontal lines located within the boundary. All obstacles are assumed to be static. No dynamic obstacle has been considered at this stage of the study. The routing system is an algorithm that makes decisions on what path the part delivery system should take to avoid the obstacles and reach the final destination from the starting location. For the plant layout model presented, such path may be shown in Figure 2 from the starting location to the final destination.

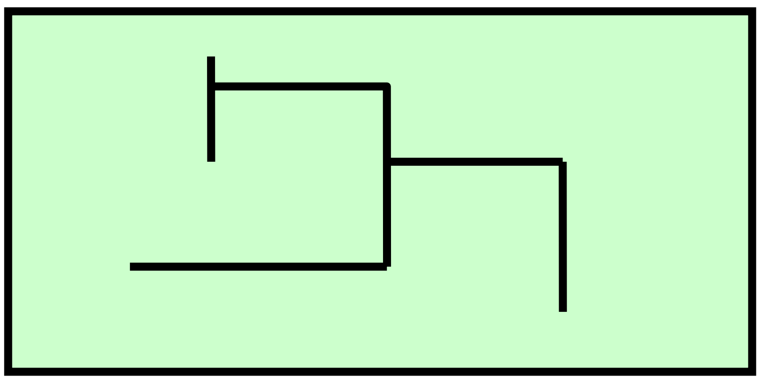

Figure 1: Plant layout model with static obstacles such as partitioning walls and machines

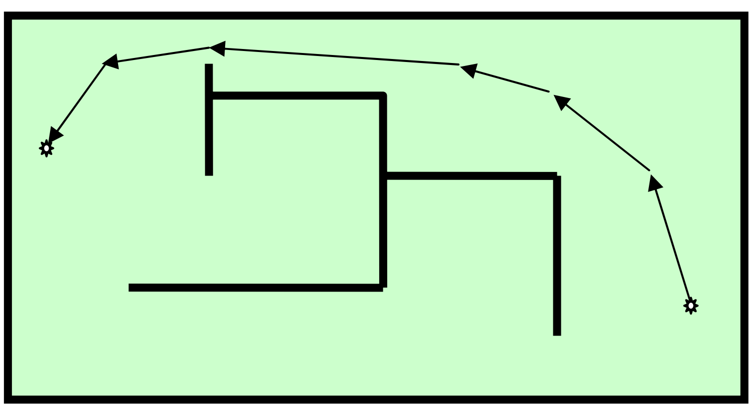

Figure 2: Obstacle-avoiding path from the staring location to final destination

The path selection problem would be fairly simple if no obstacles were located within the boundary. In such case, the path would simply be the straight line connecting the starting to the ending point as shown in Figure 3.

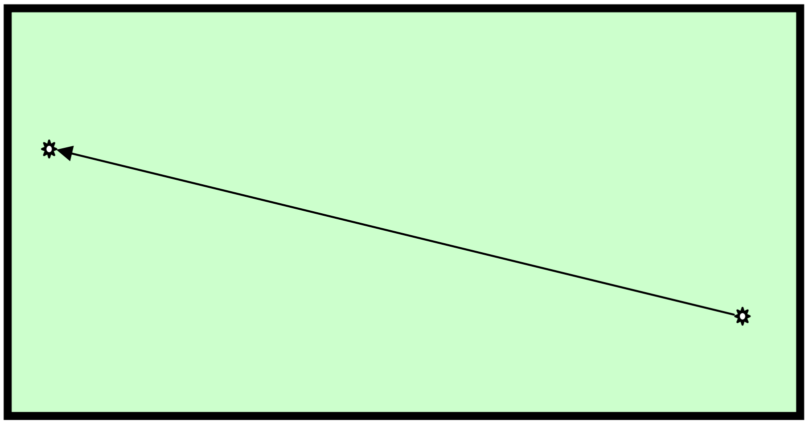

Figure 3: Path to the destination from the starting location in an obstacle-free layout

While dealing with an obstacle-free layout is too unrealistic, partitioning the layout to obstacle-free regions (partitions) is possible and so helpful. If the layout is partitioned to obstacle-free regions, the path selection process will be as simple as straight line selections within each region. Traversing between the regions will be conducted through the intersections between the obstacle-free regions. Figure 4 shows the plant layout partitioned into obstacle-free regions based on the locations of the existing obstacles in the layout. The layout partitioning algorithm is discussed in details. Generally, the idea is to block the areas between the plant boundaries and the wall obstacles such that no obstacle is left within.

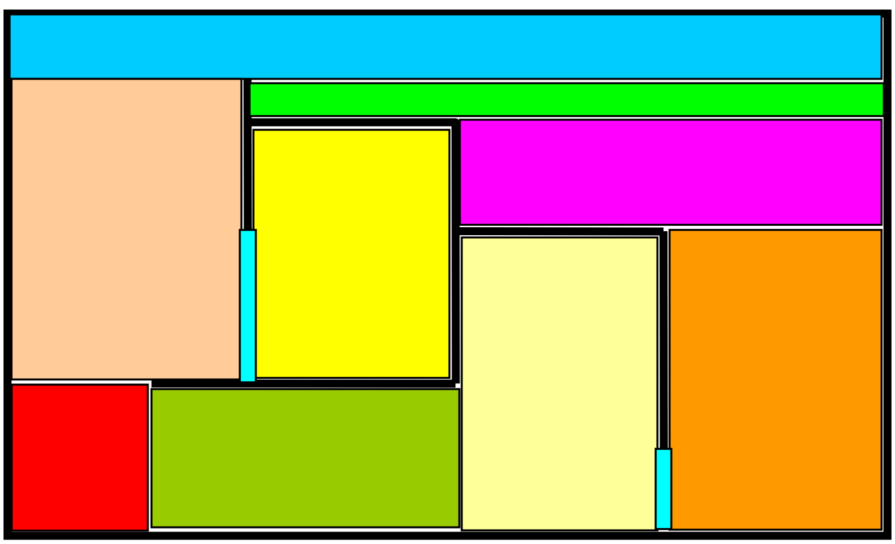

Figure 4: Plant layout partitioned to obstacle-free regions based on the locations of obstacles

To accommodate the partitioning algorithm, the plant layout should be discritized and meshed into rectangular grid as shown in Figure 5. The robotic system can only operate through the grid points on the layout excluding the ones on the boundaries and wall obstacles. These points may be referred as free points as opposed to close points on the boundaries and obstacles. Mathematically, free points may be viewed as 1s as opposed to close points that may be viewed as 0s. Thus, mathematical representation of the layout is achieved by constructing a binary matrix called layout matrix using grid information. As shown in Figure 6, in such matrix, 1s are placed in the corresponding locations of the open points on the layout. In contract, 0s are placed in the corresponding locations of the close points on the layout.

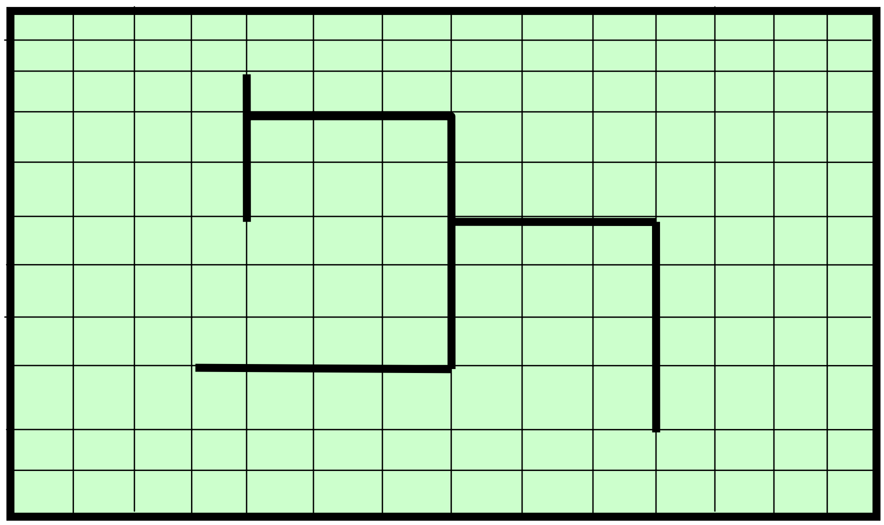

Figure 5: Plant layout meshed into rectangular grid

0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	1	1	1	1	1	1	1	1	1	1	1	1	1	0
0	1	1	1	0	1	1	1	1	1	1	1	1	1	0
0	1	1	1	0	0	0	0	1	1	1	1	1	1	0
0	1	1	1	0	1	1	0	1	1	1	1	1	1	0
0	1	1	1	0	1	1	0	0	0	0	1	1	1	0
0	1	1	1	1	1	1	0	1	1	0	1	1	1	0
0	1	1	1	1	1	1	0	1	1	0	1	1	1	0
0	1	1	0	0	0	0	0	1	1	0	1	1	1	0
0	1	1	1	1	1	1	1	1	1	0	1	1	1	0
0	1	1	1	1	1	1	1	1	1	1	1	1	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 6: Plant layout matrix; 0s on the sides represent the plant boundary; 0s within the boundaries represent obstacles;1s represent free grid points

Layout matrix contains all the information about the boundaries, obstacles, and free points on the plant layout. From now on, rather than working with physical geometry of the layout, the algorithm is managed by processing such binary matrices leading to more simplicity. In order to partition the layout matrix to free-obstacle regions, the areas which only contain 1s and are restricted by 0s on sides should be selected and blocked as shown in Figure 7. To accomplish this, following steps should be taken:

1- For the very top available row (the first row below the plant top boundary), successive 1s are blocked into one dimensional horizontal blocks called region-heads that are free of obstacles. If there is no zero on the row, the whole row is blocked into just one single region-head (as shown in Figure 7 by red dash block). If there were 0s on the row, multiple region-heads would be obtained that would belong to this particular row and be separated by the 0s.
2- Once the region-heads of the very first available row are obtained, in order to construct the free-obstacle regions, they have to be extended downward to the levels that restrict them by 0s. (In Figure 7, the very first available row can not be extended any downward since there is a 0 in the row below; Thus the region-head itself is selected as a region)
3- The information on the regions selected is stored using column and row numbers of the sides of the regions (that belong to the first available row). The regions are also numbered globally.
4- Steps 1 through 3 are repeated for the next row below the very top available row and so on and so forth as shown in Figure 7. It should be noted that every time that step 1 is repeated for a new row, the algorithm escape from the regions that have been already blocked, and only construct the region-heads where is still free of regions.
5- After all the steps explained are implemented, the layout matrix is partitioned into obstacle-free regions with global numberings shown in Figure 8. This partitioned matrix is the mathematical representation of the plant layout shown in Figure 4.

0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	1	1	1	1	1	1	1	1	1	1	1	1	1	0
0	1	1	1	0	1	1	1	1	1	1	1	1	1	0
0	1	1	1	0	0	0	0	1	1	1	1	1	1	0
0	1	1	1	0	1	1	0	1	1	1	1	1	1	0
0	1	1	1	0	1	1	0	0	0	0	1	1	1	0
0	1	1	1	1	1	1	0	1	1	0	1	1	1	0
0	1	1	1	1	1	1	0	1	1	0	1	1	1	0
0	1	1	0	0	0	0	0	1	1	0	1	1	1	0
0	1	1	1	1	1	1	1	1	1	0	1	1	1	0
0	1	1	1	1	1	1	1	1	1	1	1	1	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 7: Layout partitioning algorithm; Region heads depicted in red dash-lines; Free-obstacle regions depicted in rectangular block;

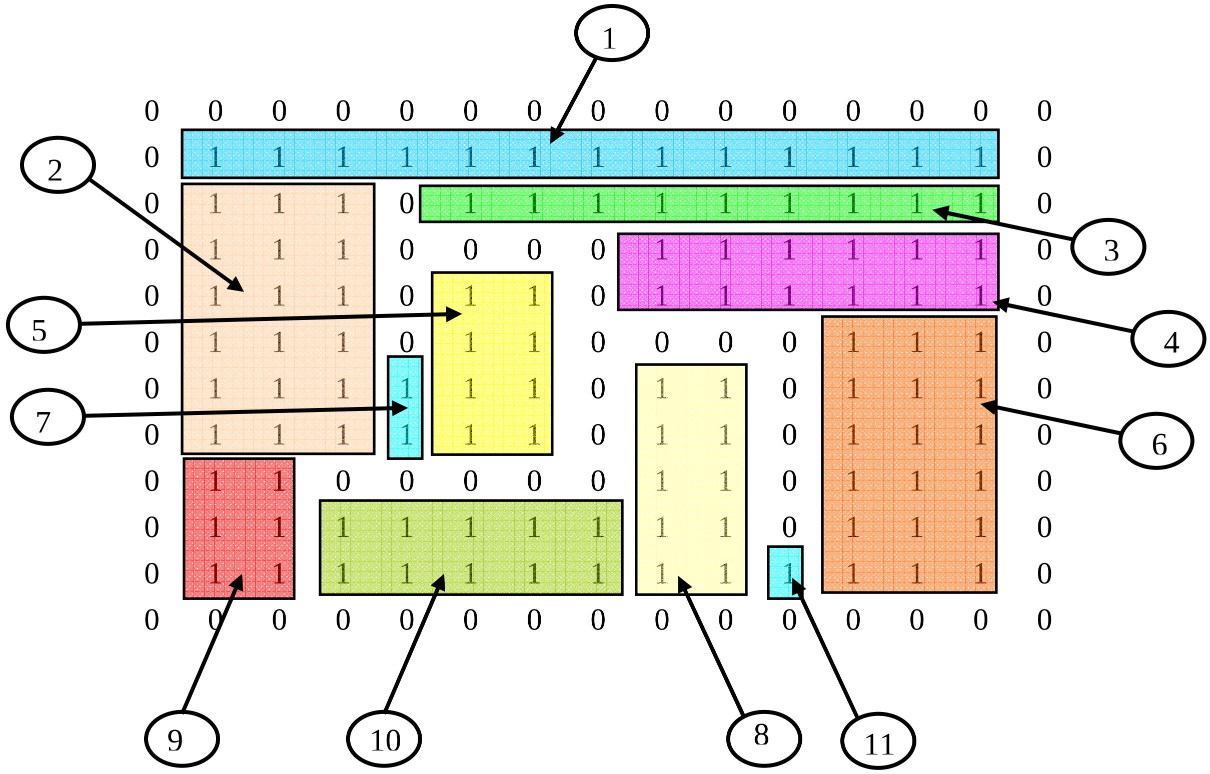

Figure 8: Layout matrix partitioned with free-obstacle regions along with their global numberings

Since the robotic system traverses from a region to another through the intersections between the regions, it is important to extract the information on the intersecting regions from the layout matrix. For this purpose, the possibility of intersection is considered for each and every region in the layout matrix with the other regions. For instance, an intersection on the lower side of region # i may be identified if there is a region such as region # x that satisfies the following conditions:

(The lower row number of region # i) +1 = (The upper row number of region # x)
And
Left column number of region # i <(left column number of region # x)< right column number of region # i
OR
Left column number of region # i <(right column number of region # x)< right column number of region # i

Similar conditions may be set up to identify the intersection on all other 3 sides of region # i. The information about intersection is completed as theses conditions are checked one by one for all the existing regions in the layout. At the end, it will be known what regions intersecting what other regions and where the intersecting points are as far as row and column numbers are concerned. A part of this information is stored in a binary matrix called intersection matrix shown in Figure 9. Intersection matrix indicates whether or not each region intersects with the other obstacle-free regions in the layout.

	1	2	3	4	5	6	7	8	9	10	11
1	0	1	1	0	0	0	0	0	0	0	0
2	1	0	0	0	0	0	1	0	1	0	0
3	1	0	0	1	0	0	0	0	0	0	0
4	0	0	1	0	0	1	0	0	0	0	0
5	0	0	0	0	0	0	1	0	0	0	0
6	0	0	0	1	0	0	0	0	0	0	1
7	0	1	0	0	1	0	0	0	0	0	0
8	0	0	0	0	0	0	0	0	0	1	1
9	0	1	0	0	0	0	0	0	0	1	0
10	0	0	0	0	0	0	0	1	1	0	0
11	0	0	0	0	0	1	0	1	0	0	0

Figure 9: Intersection matrix representing the info on the intersecting free-obstacle regions

In intersection matrix, 1 in ij location is the indication of intersection between region # i and region # j. In contrast, 0 in ij location indicates that there is no intersection between these regions. Since the intersection matrix contain the information on the intersecting regions, and since the only way of traversing between the regions is through the intersections, the intersection matrix may be used to find all the possible combinations of regions that construct a path from a starting region to a destination region. That could be accomplished by repeated

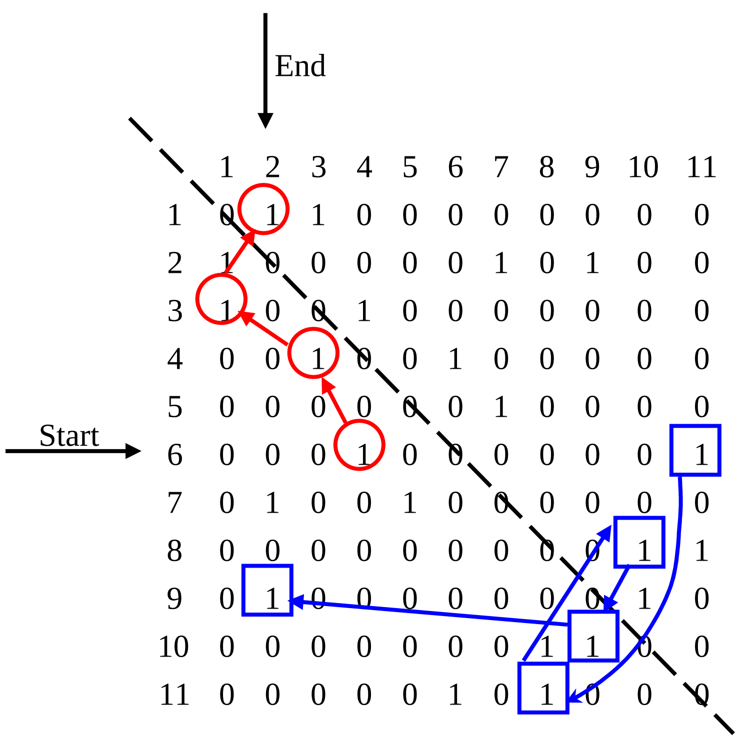

Figure 10: Repeated search algorithm to find all possible combinations of regions constructing a path from a starting region to a destination region

searching of 1s from a starting region to a destination region as shown in Figure 10. In this figure, the intension is to start from region # 6 and end up at region # 2. Starting from row # 6, two 1s are identified. The first one is on the 4th column and the other one is on the 11th column. These 1s are indicating that region # 6 has intersection with region # 4 and # 11. (This can also be seen in the partitioned layout matrix shown in Figure 8.) Each of these regions may be viewed as a starting region of an independent path that may be preceded by other regions to the final destination region. For instance, let's consider the path starting with region # 4. The repeated search should continue by looking at row # 4 identifying region # 3 as the next intersection region (which can be also observed in Figure 8 again). The searching algorithm continues all the way to the final destination, region # 2. The order of the regions found for both independent possible paths is shown in Figure 10. Alternatively, these paths are shown as a tree in Figure 11.

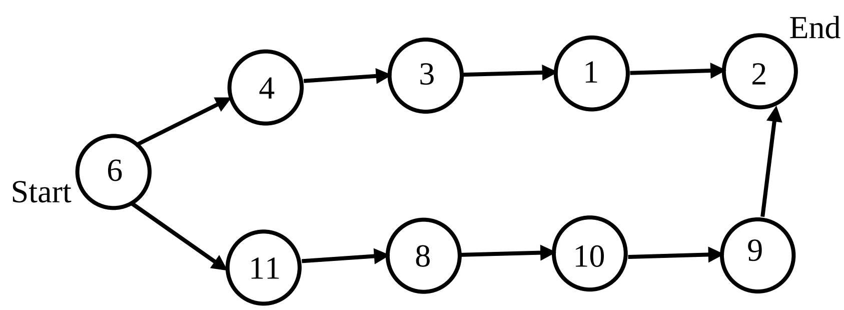

Figure 11: Path tree summarizing all possible paths from a staring region to destination

Now that the order of traversing the regions from the starting region all the way to the final destination is obtained, the physical path that connects the regions from a starting point in the starting region to the destination point in the destination region may be constructed. For this purpose following notes should be considered for path construction:

1- The Path is constructed of different line segments
2- Line segments are either within the free-obstacle regions or crossing over the intersections
3- Line segments are straight lines connecting only 1s (open points on the grid)

Considering these points, following steps should be taken to construct a path from the starting point to the destination as shown in Figure 12:

1- A line segment is constructed from the starting point located at the starting region to a targeted point on the intersection with the following region. The targeted point on the intersection is the point having shorter distance to the destination relative to the other points on the intersection. (Targeted point still belongs to the starting region).
2- A line segment is constructed by crossing over the intersection to a new targeted point in the following region. This new targeted point is located immediately next to the

point (on the intersection) which corresponds to the previous targeted point, and has shorter distance to the destination.

3- Steps 1 and 2 are repeated for the following regions until getting into the region immediately before the final destination region.
4- A line segment is constructed to the point on the intersection that provides shortest distance to the final destination.
5- Final line segment to the final destination is obtained.

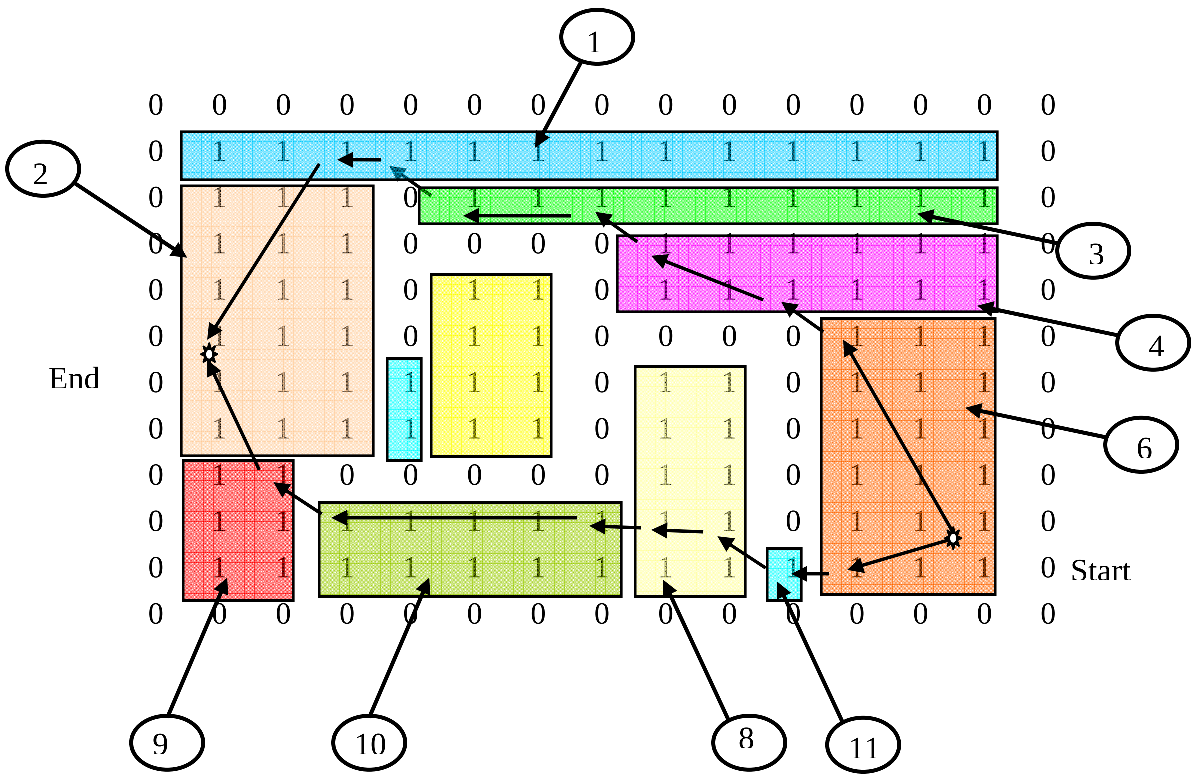

Figure 12: Paths constructed from the starting point to the destination

The path construction algorithm is repeated for all possible paths summarized in path tree. Once all physical paths are determined, the optimum path may be selected. The selection process could be based on shortest distance criterion. In such case, the total path length could be calculated by calculating the length of each line segment. The selection process could also be based on shortest time criterion. That involves more complexity. The average velocity of the robotic system along with starting acceleration, and ending deceleration should be considered for each line segment. The angles between the line segments and their subsequent effects on the velocity should also be considered. More complexity may even come into the play if the path selection process is based on both distance and time. That involves solving an optimization problem with a clearly defined objective.

Results

A program in C language has been developed that accommodates the algorithm discussed. The plant layout, the starting point and the destination point are the inputs to the program. The program partitions the layout, and comes up with all possible combinations of regions that a path could be constructed through the regions. The shortest path is routed and outputted by the program. A number of layout containing different shapes of obstacles have been tried out. Excellent level of success has been measured in the performance of the designed algorithm and program code.

Future work

Future work involves incorporating in-angle, round, and dynamic obstacles in the algorithm. It also involves improving path construction algorithm such that the path can be updated and smoothened between each line segment.

Conclusion

A comprehensive effort has started by a team of engineering, math and computer science students to design and prototype a Robotic System that will completely automate part transformation between different manufacturing stations. A routing system has been designed by layout partitioning algorithm that partitions the layout to obstacle-free regions. The algorithm has delivered all possible combinations of regions that can connect the starting point to the destination. The algorithm also constructs the physical paths by drawing line segments within the obstacle-free regions and through the intersections. A program has been developed in C language to implement the algorithm introduced. The program exhibited an excellent amount of success.

Acknowledgement

The work was supported by the TIME Center “Technology and Innovation in Manufacturing and Engineering” an NSF sponsored ATE Center.

References

[1] The basic of Automated Guided Vehicles, AGV System, Seimens, March 2006.
[2] Flexible manufacturing systems, University of Kentucky, March 2006.
[3] T. Le-Anh, M. De Koster, “A review of design and control of automated guided vehicle systems”, European Journal of Operational Research, Volume 171, Issue 1, May 2006, Pages 1-23.
[4] A. Asef-Vaziri, M. Goetschalckx, “Dual track and segmented single track bidirectional loop guidepath layout for AGV systems”, European Journal of Operational Research, Volume 186, Issue 3, May 2008, Pages 972-989.

[5] I. Vis, "Survey of research in the design and control of automated guided vehicle systems", European Journal of Operational Research, Volume 170, Issue 3, May 2006, Pages 677-709.
[6] T. Nishi, S. Morinaka, M. Konishi, "A distributed routing method for AGVs under motion delay disturbance Robotics and Computer-Integrated Manufacturing", Volume 23, Issue 5, October 2007, Pages 517-532.
[7] Z. Katz, J. Asbury, "On-line position recognition for autonomous AGV navigational systems", Journal of Manufacturing Systems, Volume 12, Issue 2, 1993, Pages 146-152.
[8] C. Lin, L. Wang,"Intelligent collision avoidance by fuzzy logic control", Robotics and Autonomous Systems, Volume 20, Issue 1, April 1997, Pages 61-83.
[9] J. Evers, S. Koppers, "Automated guided vehicle traffic control at a container terminal", Transportation Research Part A: Policy and Practice, Volume 30, Issue 1, January 1996, Pages 21-34.
[10] F. Wang, J. Lin, "Performance evaluation of an automated material handling system for a wafer fab", Robotics and Computer-Integrated Manufacturing, Volume 20, Issue 2, April 2004, Pages 91-100.
[11] Ü. Bilge, G. Esenduran, N. Varol, Z. Öztürk, B. Aydın, A. Alp, "Multi-attribute responsive dispatching strategies for automated guided vehicles", International Journal of Production Economics, Volume 100, Issue 1, March 2006, Pages 65-75.
[12] T. Cheng, "A simulation study of automated guided vehicle dispatching", Robotics and Computer-Integrated Manufacturing, Volume 3, Issue 3, 1987, Pages 335-338.

Biography

PAYAM MATIN is currently an assistant professor at the department of Engineering and Aviation Sciences at the University of Maryland Eastern Shore. Dr. Matin's research interests are manufacturing simulation, manufacturing processes, mobile robots, experimental mechanics and computational mechanics.

ALI EYDGAHI is a Professor and Chair of Engineering and Aviation Sciences Department, Director of the Center for 3-D Visualization and Virtual Reality Applications, and Technical Director of Space Vehicle Mission Planning Laboratory at the University of Maryland Eastern Shore. Dr. Eydgahi has received a number of awards including the Dow Outstanding Young Faculty Award from American Society for Engineering Education in 1990. Dr. Eydgahi has served as a session chair and a member of scientific and international committees for many international conferences and has published more than hundred papers in refereed international and national journals and conference proceedings.

KENNETH BURCH is currently the director of the TIME Center at CCBC. TIME is an NSF sponsored Advanced Technology Education Center whose focus is to partner with businesses, schools, and colleges to meet the technical and engineering needs of Maryland manufacturing.