Automation and Robotics in Construction XI
D.A. Chamberlain (Editor)

VISIBILITY GRAPH APPROACH TO DETAILED PATH PLANNING IN CNC CONCRETE PLACEMENT

By: Raghavan Kunigahalli[a] and Jeffrey S. Russell[b]

[a] Ph.D. Candidate, Dept. of Civ. & Env. Engrg., University of Wisconsin-Madison, Madison WI 53706.

[b] Asst. Prof., Dept. of Civ. & Env. Engrg., University of Wisconsin-Madison, Madison WI 53706.

Abstract: An algorithm that employs visibility graph approach to generate a minimal L_1-metric obstacle avoidance path in CNC concrete placement is presented. A set of pseudo-obstacles that represent nodes in a relational structure called Rectangle Adjacency Graph (RAG) is introduced during construction of visibility graph. Shrinking of pseudo-obstacles and growing of primary obstacles such as elevator shafts results in an efficient network for generating minimal L_1-metric paths from a given source location to a target location in concrete floors. Secondary obstacles such as column bars are avoided using geometric and topological relationships stored in the RAG.

1. INTRODUCTION

A "constrained path" problem is formulated as follows: there is a set of objects that are polyhedral in space or polygonal in plane called "obstacles." The problem is to find a possibly shortest path in the applicable metric between the given *source* and *target* points without intersecting the interior of any "obstacle." Variants in algorithmic approaches to solve "constrained path" problems for automated construction have been discussed in Kunigahalli et al. [1]. A relational structure called RAG has been developed by Kunigahalli et al. [2]. RAG stores the topological relationships among geometric entities required to solve motion planning problems in automated concrete construction. This paper describes a visibility graph approach to generate a local obstacle avoidance path between two nodes in the RAG.

A visibility graph G = (V, E) is an undirected graph whose nodes correspond to geometric entities, such as vertices or edges. Two nodes u and v

of a visibility graph are connected by an edge u, v ∈ E if the geometric entities associated with nodes can see each other. Analysis of visibility graphs is a challenging problem as combinatorial structure of visibility graphs is not fully understood. As of date, there is no polynomially bounded algorithm to decide whether or not a given graph is a visibility graph of some polygon (with or without holes) [3, 4]. Application of visibility graph approach to solve "constrained path" problem was first proposed by Wangdahl et al. [5], who used a variation of Dijkstra's algorithm to solve pipe routing in ships. This approach was adopted later by Lozano-Perez and Wesley [6] to navigate a robot vehicle. Lozano-Perez and Wesley also showed that shortest path for a moving polygon amidst convex polygonal obstacles can be solved in $O(n^2)$ time using Dijkstra's algorithm applied to certain visibility graphs.

Even though the shortest path could be obtained in $O(n^2)$ time, for many years the fastest algorithm known for constructing the visibility graphs had a bound of $O(n^2 \log n)$. In 1985, Wezl [7] exploited the special properties of arrangement of lines (dual of a set of given points) to reduce the complexity to $O(n^2)$.

A special case of "constrained path" problem related to rectilinear obstacles gained researchers attention due to its application in VLSI (Very Large Scale Integrated-Circuit) design. Lee and Preparata [8] proposed an $O(n \log n)$ algorithm to solve the constrained path problem among rectilinear barriers without explicit construction of the entire visibility graph. Their algorithm exploited the dual properties of triangulated simple polygons [8].

Another interesting case of "constrained path" problem is related to shortest path in L_1-metric among rectilinear obstacles. Larson and Li [9] proposed an algorithm that included: (1) construction of visibility graph in L_1-metric and (2) generating shortest path using a modified Dijkstra's algorithm. Wu et al. [10] proposed a different algorithm for the same problem using a grid-like structure called a tack graph. In this paper, we are proposing a modified Larson and Li [9] algorithm to generate a local collision avoidance path plan for concrete placement. This approach is specifically chosen because it alleviates the problems associated with: (1) consolidation and screeding crew and (2) path planning of other robots related to concrete construction.

2. GEOMETRIC PRELIMINARIES

Let n be the number of nodes in the Rectangle Adjacency Graph (RAG), $x = (x_s, y_s)$ be the source point, and $T = \{ (x_t, y_t) \in \Re^2 \mid 1 \le t \le 4 \}$ be the set of target points. The source point S corresponds to the current position of the placement pipe after completion of concreting rectangular partition represented by the node R_i of RAG. Target points correspond to vertices of rectangular partition represented by the node R_{i+1} of RAG.

Let $O = \{ O_i \mid 1 \le i \le m \}$ be the set of obstacles of orthogonal polygonal shape and $((x_u, y_u)_i, (x_v, y_v)_i)$ be a line segment of obstacle i. Further, $Q = \{ R_j \mid 1 \le j \le n \}$ be the set of rectangular partition corresponding to nodes of

RAG and $\widehat{O}$ be the set of obstacles that are grown to accommodate the size of the placement pipe.

We define $\widehat{Q} = \{ \widehat{R_j} \mid 1 \le j \le n \}$ as the set of *pseudo-obstacles* that results from shrinking all rectangular partitions of a given floor. Shrinking process can be performed by adopting an approach similar to growing of obstacles proposed by Lozano-Perez and Wesley [6]. Shrinking parameter doubles whenever boundary of a rectangular partition overlaps with the boundary of an obstacle. As the partitioning of the floor corresponds to non-overlapping rectangles that excludes obstacles [2], intersection of sets S, T, $\widehat{Q}$, and $\widehat{O}$ is a null set.

A rectilinear path P in $\Re^2$ having 2(z+1) steps is specified by a sequence of points in $\Re^2$ given by $\{ (x_o, y_o), (x_1, y_o), (x_1, y_1), (x_2, y_1), (x_2, y_2), \ldots\ldots (x_z, y_z), \ldots\ldots (x_{z+1}, y_{z+1}) \}$ such that the path P is given by Equation (1).

$$P = \{\{ (x, y_w) \in \Re^2 \mid x_w \le x \le x_{w+1} \vee x_{w+1} \le x \le x_w \text{ for } 0 \le w \le z\} \cup \{ (x_w, y) \in \Re^2 \mid y_{w-1} \le y \le y_w \vee y_w \le y \le y_{w-1} \text{ for } 0 \le w \le z\}\} \quad \ldots\ldots\ldots (1)$$

Further, rectilinear length of a path P is given by Equation (2):

$$L_p = \sum_{w=0}^{z} (| x_{w+1} - x_w | + | y_{w+1} - y_w |) \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots (2)$$

Let P_{ij} denote a feasible rectilinear path between any two points $i \in S$, $j \in T$. Clearly, there exists at least one rectilinear path P_{ij} for all elements $i, j \in \{ S \cup T \}$ such that $P_{ij} \cup O = \varnothing$.

Now consider the merged set of points $M = \{ S \cup T \cup \widehat{Q} \cup \widehat{O} \}$. Any two points m(i), m(j) $\in$ M with coordinates $(x_m(i), y_m(i))$ and $(x_m(j), y_m(j))$ are said to *communicate,* if there exists at least one feasible path P_{ij} such that its rectilinear length is given by $(| x_{w+1} - x_w | + | y_{w+1} - y_w |)$ [9]. There can be more than one *communicating* path between two points as shown in Figure 1 and hence minimal feasible paths are considered. For example, paths *abhc*, *aeflhc*, *aeijkgc*, and *aeidkgc* have the same rectilinear lengths and are minimal.

3. CONSTRUCTION OF VISIBILITY GRAPH

Construction of visibility graph begins first by including all the edges due to *simply communicating vertices* belonging to the set $\{ S \cup T \cup \widehat{Q} \cup \widehat{O} \}$ using *ray shooting* technique. Two vertices m(i) and m(j) are said to be *simply communicating* if one of the following three conditions are met: (1) m(i) and m(j) are adjacent vertices in the set $\{ \widehat{Q} \cup \widehat{O} \}$, (2) m(j) is one of the end points of the

line segment $\overline{s_k} \in \{ \hat{Q} \cup \hat{O} \}$ that intersect a ray from m(i) along one of the principal directions x or y, and (3) rays from m(i) or m(j) intersect at a point $b \notin \{ S \cup T \cup \hat{Q} \cup \hat{O} \}$.

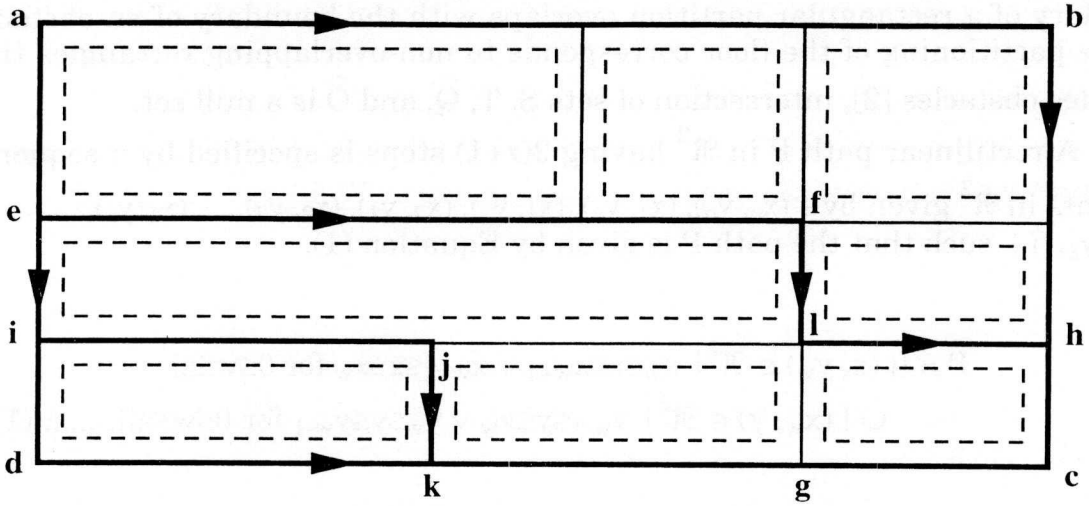

Figure 1. Multiple Minimal Rectilinear Paths

The next step in the construction of visibility graph requires identification of all vertices *communicating* with the source vertex. This step includes edges due to all paths that are *monotone* with respect to x and y-axis. A path P along the points $p_1, p_2, ...p_k$ is said to be a *monotone path* with respect to x and y-axis if ordering of the projections of $p_1, p_2, ...p_k$ on to the x and y-axis is same as that of the path itself. Identification of all vertices $m(j) \in \{ S \cup T \cup \hat{Q} \cup \hat{O} \}$ that *communicate* with the source vertex s can be performed iteratively by using information related to: (1) *orientation* of node m(j) with respect to s and (2) *permissible direction* that provide orientation about m(j) that includes *monotonic paths* through m(j) to s. The remaining edges of the visibility graph correspond to *non-monotonic paths* and can be determined by modified Dijkstra's algorithm described in the next section [4, 9].

4. MINIMAL RECTILINEAR DISTANCE

In graph-theoretic approach, a shortest-path problem is formulated as follows: given a directed graph G = (V, E) and weight function w that maps edges to real valued weights, the weight of a path w(p) is the sum of the weights of its constituent edges. Shortest-path weight d(i, j) from node i to node j is defined as:

$$d(i,j) = \begin{cases} \min w(p) & \text{if a path exists from i to j} \\ \infty & \text{otherwise} \end{cases} \quad \text{.....(3)}$$

The shortest path from node i to node j is then defined as any path p with a weight w(p) = d(i, j).

The technique used by a shortest path algorithm is called *relaxation*. *Relaxation* is a process that repeatedly decreases an upper bound on the actual shortest-path weight of each vertex until the upper bound equals the shortest path weight [11].

The weight function considered by minimal distance rectilinear path algorithm corresponds to the *penalty travel distance* that occurs due to *non-monotonic paths* between any two vertices. The *penalty travel distance* is mainly due to *doubling back* in the projections to principal axes as the path is traversed [4]. At any stage of this algorithm, two sets of vertices belonging to visibility graphs exist. Vertices whose minimal distance paths have already been determined are grouped under a "closed" set and will not be examined further. The vertices belonging to complement of this *closed* set are grouped under a *open* set. At the beginning of the minimal distance rectilinear path algorithm all vertices connected to the source vertex by a *monotonic path* are assumed to have attained the minimal paths and are included in the *closed* set as the *penalty travel distance* between them is zero.

At each *relaxation* step, all the vertices in the open set that *simply communicate* with at least one vertex in the *closed* set are examined and their minimum penalty distances to the source vertex obtained. Following this, the least penalty path that corresponds to a minimal penalty among all paths that *simply communicate* between a node in the *open* set and a node in the *closed* set is added to the *closed* set.

The construction of visibility graph in the previous section, enabled the algorithm to avoid obstacles such as elevator shafts. However, projecting column bars must also be avoided by the placement pipe path. The relational structure RAG proposed in Kunigahalli et al. [2], contains sufficient information to avoid projection of column bars as described in the subsequent section.

5. AVOIDING COLUMN BARS

Every segment along the minimal rectilinear path P_{ij} between the *source* vertex s and the *target* vertex t is either a subset or a superset of $\{ \bigcup_{k=1}^{4} (u_k, v_k) \in R_i \mid R_i \in Q \text{ for } 1 \le i \le n \}$. Each element R_i in the set Q corresponds to a node in the RAG. Formation of a node in RAG includes *2-dimensional range search* within the range [*width* x *span*] of beams enclosing the rectangular partition represented by the node. The 2-D range search can be performed using either a *range-tree* or a *k-d tree*. The 2-D range search identifies all columns intersecting with beams enclosing a rectangular partition. Geometric information related to all intersecting columns are stored along each

edge of the partitioning rectangle that is represented as a single node in RAG [12]. During traversal of path P_{ij}, a local replacement of sub-paths along P_{ij} occurs at all intersections of P_{ij} with columns of the floor.

6. CONCLUSION

An algorithm to construct visibility graphs for collision avoidance path planning in concrete construction has been described. The algorithm employs *ray shooting technique* and utilizes information related to orientation of each node with respect to source node in order to generate edges of the visibility graph. The rectilinear path-planning approach adopted by this algorithm alleviates problem associated with coordination of construction crew and other robots operating on job-site.

7. ACKNOWLEDGEMENTS

The second author wishes to thank the National Science Foundation for Grant No. MSM-9058092, Presidential Young Investigator Award, for financial support of this effort.

8. REFERENCES

1. Kunigahalli, R., Russell, J. S., and Skibniewski, M. J., "Motion Planning for Automated Construction," *Proceedings, Tenth International Symposium on Automation and Robotics in Construction*, May 24-26, (1993), Houston, Texas, pp. 407-414.

2. Kunigahalli, R., Russell, J. S., and Veeramani, D., "An Algorithm to Extract Topological Relationships from a Wire-Frame CAD Model of Concrete Floors," *ASCE Journal of Computing in Civil Engineering*, (1993), under review.

3. Abello, J. and Egecioglu, O., "Visibility Graphs of Staircase Polygons with Uniform Step Length," *International Journal of Computational Geometry and Applications*, Vol. 3, No. 1, (1993), pp. 27-37.

4. O'Rourke, *Art Gallery Theorems and Algorithms*, Oxford University Press, Inc., New York, NY, (1987).

5. Wangdahl, G. E., Pollack, S. M., and Woodward, J. B., "Minimum-Trajectory Pipe Routing," *Journal of Ship Research*, Vol. 18, No. 1, (1974), pp. 46-49.

6. Lozano-Perez, T. and Wesley, M. A., "An Algorithm for Planning Collision-Free Paths Among Polyhedral Obstacles," *Communications of ACM,* Vol. 22, No. 10, (1979), pp. 560-571.

7. Wezl, E., "Constructing the Visibility Graph for n-Line Segments in $O(n^2)$ time," *Information Processing Letters*, Vol. 20, (1985), pp. 167-171.

8. Lee, D. T. and Preparata, F. P., "Euclidean Shortest Path in the Presence of Rectilinear Barriers," *Networks*, Vol. 14, (1984), pp. 393-410.

9. Larson, R. C. and Li, V. O. K., "Finding Minimum Rectilinear Distance Paths in the Presence of Barriers," *Networks*, Vol. 11, (1981), pp. 285-301.

10. Wu, Y., Widmayer, P., Schlag, M. D. F., and Wong, C. K., "Rectilinear Shortest Paths and Minimum Spanning Trees in the Presence of Rectilinear Obstacles," *IEEE Transactions on Computers*, Vol. c-36, No. 3, (1987), pp. 321-331.

11. Cormen, T. H., Leiserson, C. E., and Rivest, R. L., *Introduction to Algorithms*, The MIT Press, Cambridge, MA, (1990).

12. Kunigahalli, R., Russell, J. S., and Skibniewski, M. J., "Motion Planning for Automated Construction," to appear in a special issue of *International Journal of Automation in Construction*.