Automation and Robotics in Construction X
G.H. Watson, R.L. Tucker and J.K. Walters (Editors)

MOTION PLANNING FOR AUTOMATED CONSTRUCTION

By: Raghavan Kunigahalli,[a] Jeffrey S. Russell,[b] and Miroslaw J. Skibniewski,[c]

[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.

[c] Assoc. Prof., School of Civ. Engrg., Purdue University, West Lafayette, IN 47907.

Abstract: Increasing research and development in the area of construction robotics and computer-aided-construction has resulted in a large number of automated construction equipment applications. Full potential of such construction equipment can be realized by providing the collision avoidance motion planning capabilities. Models used in motion planning can be categorized into two types: (1) planning with complete information and (2) planning with incomplete information. State-of-the-art general algorithms appropriate for the motion planning process are highlighted. Motion planning with complete information are typically NP-complete and hence are modeled by approximate approaches or by exploitation of special conditions. Motion planning with incomplete information requires sensory feedback. An algorithm for motion planning in an automated concrete placement operation that exploits the special conditions provided by structural engineering floor design theory is also provided.

1. INTRODUCTION

In the last five years, there has been an increasing trend in the application of automation technology to the construction industry. A large number of automated construction equipment applications have been developed for field operations such as concrete placement, concrete finishing, wall inspection, slurry wall construction, and tunneling. A survey of automation applications in the construction industry can be found in Skibniewski and Russell [1].

A significant challenge in construction automation is the ill-structured constantly changing environment of the construction site. In order to address this challenge, automated machines must be equipped with sophisticated collision avoidance motion planning capabilities. The collision avoidance motion planning is formulated as a static problem. Only the path (i.e., position and orientation information) is being sought, with no consideration for the dynamics along the path. There are two formulations of collision avoidance motion planning problem: (1) path planning with complete information describing the environment and (2) path planning with incomplete information for an uncertain environment. With complete information an optimal path can be obtained. With incomplete information, the path is computed continuously or piece by piece, based on the incoming information from sensors and hence an optimal solution cannot be found.

This paper describes the generic challenges associated with motion planning of automated construction equipment. A review of the classical formulation of the mover's problem is provided along with a discussion of its complexity. State-of-the-art of planning algorithms for complete and incomplete information are also described. A motion planning algorithm with complete information for an automated machine in concrete placement operation is also provided.

2. MOTION PLANNING WITH COMPLETE INFORMATION

In motion planning problems with complete information, the general approach is to compute the configuration space in which the moving object shrinks to a point, and obstacles are grown to compensate for the shrinking object. Any path that lies within the complement of the grown obstacle is feasible. General algorithms for motion planning with complete information are summarized below.

Generalized Mover's Problem. The classical formulation of mover's problem in d-dimensional Euclidean space has been provided by Rief [2]:

Input-- (O, B, P_s, P_t) where O is a set of polyhedral obstacles fixed in Euclidean space, and B is a rigid polyhedron with distinguished positions P_s and P_t.

Problem-- Can B be moved by a sequence of translations and rotations in d-space from positions P_s to P_t without contacting any element in O?

The mover's problem may be generalized to allow B, the object to be moved to consist of multiple polyhedra freely linked together at various distinguished vertices. The generalized mover's problem in 3-space is p-space hard, by a direct log-space reduction from the acceptance problem for polynomial space bounded by Turing machines. The generalized mover's problem is one of few known p-space complete combinatorial problems. Reif [2] has presented a polynomial time algorithm to the classical mover's problem by mapping the problem in lower dimension to a simpler problem in higher dimension.

Piano Mover's Problem. The general mover's problem has been reformulated by Schwarz and Sharir [3] in abstract algebraic terms so as to reduce it to the problem of decomposing certain algebraic varieties into their connected components. This reformulation has been referred to as the "Piano Mover's" problem.

Algebraic formulation of the problem includes one or more hinged bodies B. Each body B consist of a finite number of rigid compact subparts B_1, B_2,........, each bounded by algebraic surfaces. These subparts may be connected to each other by various types of attachments such as revolute and sliding.

Body B is required to be moved from initial to the final position in empty space bounded by a finite collection of walls, each of which is an arbitrary algebraic surface. The rotational component of body B's motion is a smooth 3-dimensional algebraic submanifold of 9-dimensional Euclidean space. If B is a single rigid body, its position is described by a Euclidean motion T that takes B from some standard position to its given position. This transformation $Tx = Rx + X_o$ is defined by a pair of $[X_o, R]$ consisting of a point X_o in 3-dimensional Euclidean space E^3 and of a 3x3 rotation matrix R. Hence T is a point in a smooth 6-dimensional algebraic submanifold G of 12-dimensional Euclidean space E^{12}. Irrespective of the manner in which subparts of the body are connected together, overall position of all its parts can always be defined by a point belonging to smooth algebraic manifold G lying in a Euclidean space of some appropriate dimension.

The theoretical existence of solutions to the "Piano Mover's" problem was shown by Schwartz and Sharir [3]. The techniques to obtain the solution were based on Tarski's algorithm for deciding statements in the theory of real numbers and Collin's algebraic decomposition of Euclidean spaces. An algorithm that solves the "Piano Mover's" problem by computing a connectivity graph with complexity $O((2n)^{3^{r+1}} . m^{2^r})$ was provided. In the complexity equation, r relates to the number of degrees of freedom for all subparts B_i of body B, n is the maximum degree of the polynomial defining a single geometric constraint and m is the number of polynomials defining the sets G and F; where, G is a smooth manifold in the Euclidean space E^r in which transformation T_i are given and F is the manifold describing forbidden configurations of body B due to non-collision requirement.

Visibility Graph Approach. An algorithm presented by Lozano-Perez and Wesley [4], for motion planning with complete information is closely related to an optimization approach. The constraints on the position of an arbitrary reference point on the moving object are computed. Polyhedral obstacles in two or three dimensions give rise to sets of polyhedral forbidden region. This transformation reduces the problem of finding a safe path for the polyhedron to the simpler problem of finding a safe path for a point. The last task is accomplished by finding a path through a graph called *visibility graph* VG(N, L) defined as follows: the node set N is $V \cup \{S, T\}$ where V is the set of all vertices of obstacles and L is the set of all links (n_i, n_j) such that a straight line connecting the ith element of N to the jth does not overlap any obstacle [4].

Configuration Space Approach. Another widely accepted approach presented by Lozano-Perez [5] is based on characterizing the position and orientation of an object as a single point in a configuration space in which each coordinate represents a degree of freedom in the position or orientation of the object. The configuration forbidden to this object, due to the presence of other objects, can then be characterized as regions in the configuration space called *configuration space obstacles*. Efficient algorithms for computing *configuration space obstacles* have also been developed by Lozano-Perez [5].

3. MOTION PLANNING WITH INCOMPLETE INFORMATION

In motion planning with incomplete information, an element of uncertainty is present and hence proof of convergence is essential. The missing data are typically provided by some source of local information through sensory feedback such as ultrasound range finder or a vision module. Hence, path planning becomes a continuous on-line computational operation. A universal lower bound for path-planning in an uncertain environment has been provided by Lumelsky and Stepanov [6]. The bound was formulated in terms of the length of the path (P) generated by the automaton on its way from the starting point (S) to target point (T). The bound is given by $P \geq D + \sum_i p_i - \delta$; where D is the distance between S and T and p_i refers to the perimeters of obstacles intersecting the disc of radius D centered at T.

Considering the motion planning with incomplete information model, when the environment can be modeled as a finite or infinite maze, Maze Searching Algorithms can be used to plan the path. Additional resources of input information are assumed to be the following: (1) automaton has a capability to mark corridors already traversed, (2) automaton knows direction in which traversal took place, and (3) corridors that led to an intersection for the first time.

Maze Searching Problem (MSP). Maze Searching Problem can be formulated as follows: describe a general algorithm that constructs a closed covering walk (W) in a connected graph G, such that in the course of constructing W, the algorithm can handle only local information available at any vertex reached by W [7]. There are many maze searching algorithms. Select algorithms are provided below.

Tarry's Algorithm. This algorithm works with local information only. The algorithm has the following hypothesis: At any vertex $v \in V(G)$ reached in the course of the construction of walk W, the set $E^o_{v,W} \subseteq E_v$ of edges already passed from v is known. Moreover, the edge $e_{in}(v)$ by which v has been reached for the first time is known. For the initial vertex v_o of W, the set $\{e_{in}(v_o)\} = \phi$. The algorithm can be formalized as follows [7, 8]:

Step 1: Set i=0 and choose $v_o \in V(G)$. Set $W = v_o$.

Step 2: Beginning at $v_i \in V(G)$ walk along an arbitrary
$e_i \in \widetilde{E_{v_i}} := (E_{v_i} \cup \{e_{in}(v_i)\})$. Set $W = W, e_i, v_{i+1}$ $(e_i \in E_{v_{i+1}})$.
Set $i=i+1$.

Step 3: Suppose $W=v_o, e_o, \ldots\ldots, e_{i-1}, v_i$ has been constructed. If $\widetilde{E_{v_i}} = \emptyset$, go to Step 2. Otherwise, go to Step 4.

Step 4: If $\{e_{in}(v_i)\} \subseteq E^o_{v_i, W}$ go to Step 5. Otherwise, set $e_i = e_{in}(v_i)$,
$W = W, e_i, v_{i+1}$ $(e_i \in E_{v_{i+1}})$, $i=i+1$; go to Step 3.

Step 5: W is a bidirectional double tracing in G.

Fraenkel's Algorithm. This algorithm is an improvement over the Tarry's algorithm. It has the following hypothesis in addition to the hypothesis mentioned in Tarry's algorithm: a bijection c: $\{W\} \to N \cup \{0\}$ is given, where $\{W\}$ denotes the set of walks produced algorithmically, such that $c(v_o)=1$, and for W and $W'=W, e_i, v_{i+1}$ we have $|c(W) - c(W')| = 1$. The algorithm can be formalized in the following steps [7, 9]:

Step 1: As long as $c(W)>0$ proceed as in Tarry's algorithm.

Step 2(a): If $v_{i+1} \neq v_j$ $0 \leq j < i$, set $c(W')=c(W')+1$.

Step 2(b): If $v_{i+1}=v_j$ for some $j<i+1$ and if $|E_{v_{i+1}} - E(W)| \geq 1$, while
$|E_{W'o}| \leq 1$ for the same v_{i+1}, set $c(W') = c(W)-1$.

Step 3: Suppose $c(W)=0$, If $E_{W,o} \neq \phi$ at v_j, proceed as in Tarry's algorithm. Otherwise, set $e_i = e_{in}(v_j)$ if $v_i \neq v_o$.

The outcome of Fraenkel's Algorithm is the final W, a closed covering walk such that $\lambda_W(e) \leq 2$ for every $e \in E(G)$.

Pledge's Algorithm. In addition to the assumptions made in the earlier algorithms for motion planning with incomplete information, Pledge's algorithm assumes that the automaton is equipped with a compass that allows a specific direction, say north, to be maintained. The automaton is equipped with a counter that is activated when it encounters an obstacle and begins integrating the turning angles. The algorithm is given by the following steps [10]:

Step 1: Walk straight toward the north until an obstacle is encountered or target (T) is reached.

Step 2: Turn left leaving obstacle on the right.

Step 3: Follow obstacle boundary until the value of the counter is zero. Go to step 1.

The outcome of the Pledge's algorithm guarantees eventual escape from any finite maze, but its path length performance is not bounded.

Bug1. In addition to the assumptions mentioned in the earlier algorithms, the Bug1 algorithm developed by Lumelsky and Stepanov [6] assumes that the mobile automaton (MA) knows its own coordinates and those of the target at all times. The goal of the algorithm is to generate a path from Start (S) to Target (T). When meeting an ith obstacle, MA defines a hit point H_i for i = 1,2,....... When leaving the ith obstacle, to continue its travel toward the target, MA defines a leave point L_i; initially, i=1 and L_o=S. The procedure uses three registers R_1, R_2, and R_3 to store intermediate information. All three are reset to zero when a new hit point H_i is defined. R_1 is used to store the coordinates of the current point, Q_m, of the minimum distance between the obstacle boundary and T, R_2 integrates the length of the obstacle boundary starting at H_i, and R_3 integrates the length of the obstacle boundary starting at Q_m. The algorithm consists of following steps:

Step 1: From point L_{i-1}, move toward Target (T) along a straight line until one of the following occurs:
- (a) T is reached and the procedure stops.
- (b) An obstacle is encountered and a hit point H_i is defined. In this case, go to Step 2.

Step 2: Turn left and follow obstacle boundary. If T is reached, then stop. After having traversed whole boundary and having returned to H_i, define a new leave point $L_i = Q_m$, and go to Step 3.

Step 3: Based on contents of registers R_2 and R_3, determine shorter way along boundary to L_i. After having defined a point L on an obstacle, if MA discovers that the straight line segment (L, Target) crosses the obstacle at point L, then target is not reachable.

Bug1 tends to be overcautious and never covers less than full perimeter of the obstacle. Another algorithm Bug2, takes advantage of simple situations, but may become inefficient in more difficult cases.

Vision with Motion Planning. Lumelsky and Skewis [11] provided two motion planning algorithms labelled VisBug-21 and VisBug-22 that incorporate vision information. Due to space limitations, a brief overview of only VisBug-21 algorithm is presented. VisBug-21 consists of the *Main Body* that does the proper motion planning along the path, and a procedure called *Compute*T_i*-21*, that performs the test for target reachability and produces the next intermediate target T_i for the given current position C of the automaton. The main body consists of two steps. The first step S_1 consists of moving towards T_i, while executing *Compute*T_i*-21* and performing the following tests: (1) target reached/not reached, (2) target reachable/not reachable, and (3) intermediate target is reached/not reached. The second step S_2 is executed only in those cases when the automaton is moving along a (locally) convex boundary of an obstacle and so it cannot use vision for defining the next intermediate target. Step S_2 consists of moving along the obstacle boundary while executing *Compute*T_i*-21* and performing similar tests to step S_1 [11].

4. PATH PLANNING ALGORITHM FOR AUTOMATED CONCRETE PLACEMENT

An algorithm to plan the path of a concrete placement pipe in a computer-controlled concrete placement system has been developed by Kunigahalli et al. [12]. Apart from providing an efficient placement pipe path, the algorithm integrates design data with the construction operation by obtaining input from a 2-D wire-frame CAD model. Wire-frame data corresponding to a line diagram of the floor plan is processed by employing computational geometry techniques. Processed data are then stored in a hypergraph G=(V, E, H), similar to the *face adjacency graph* structure normally used in Boundary Representation (B_Rep) schemes. Thus, the information stored in the hypergraph is more complete in terms of adjacency relation among beams and rectangular slabs of the given orthogonal polygonal floor.

The algorithm obtains information on beams above, below, to the left, and to the right of any given beam at any given beam to beam intersection. Based on this information, obstructing columns along the beams are identified. The algorithm then performs a rectangular partitioning of the orthogonal polygonal floor excluding obstacles such as elevator shafts. The information regarding obstructing columns combined with partitioned rectangular adjacency information is stored in a hypergraph G = (V, E, H). An efficient traversal of the hypergraph along with direction parallel path within each node of the graph can provide a detailed path plan for the automated concrete placement pipe.

5. CONCLUSION

Motion planning issues with complete and incomplete information have been discussed. The computational complexity of algebraically formulated mover's problem is double exponential in degrees of freedom of the moving body. Hence, approximate algorithms or exploitation of special condition and circumstances are required to solve the motion planning problem with complete information. Proof of convergence and sensory feedback for local information are essential for motion planning with incomplete information. A path planning algorithm with complete information for an automated concrete placement system has been described.

6. ACKNOWLEDGMENT

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.

7. REFERENCES:

1. Skibniewski, M. J. and Russell, J. S., "Robotic Applications to Construction," *Cost Engineering*, Vol. 31, No. 6, (1989), pp. 10-18.
2. Reif, J. H., "Complexity of the Mover's Problem and Generalizations," *Proc., 20th IEEE Symp. on Foundation of Computer Science*, (1979), pp. 421-427.
3. Schwartz, J. T. and Sharir, M., "On the "Piano Movers" Problem II: General Techniques for Computing Topological Properties of Real Algebraic Manifolds," *Advanced Applied Mathematics*, Vol. 4., (1983), pp. 298-351.
4. Lozano-Perez, T. and Wesley, M. A., "An Algorithm for Planning Collision-Free Paths Among Polyhedral Obstacles," *Communications of the Association of Computing Machinery*, Vol. 22, No. 10, (1979), pp. 560-571.
5. Lozano-Perez, T., "Spatial Planning: A Configuration Space Approach," IEEE Transactions on Computers, Vol. c-32, No. 2, (1983), pp. 108-120.
6. Lumelsky V. J. and Stepanov, A. A., "Path-Planning Strategies for a Point Mobile Automaton Moving Amidst Unknown Obstacles of Arbitrary Shape," *Algorithmica*, Vol. 2, (1987), pp. 403-430.
7. Fleischner, H., Eulerian Graphs and Related Topics: Part 1, Vol. 2, North-Holland, Amsterdam, (1991).
8. Ore, T., Theory of Graphs, American Mathematical Society, Providence, Rhode Island, (1962).
9. Fraenkel, A. S., "Economic Traversal of Labyrinths," *Mathematics Magazine*, Vol. 43, 1970, pp. 125-130.
10. Abelson, H., and diSessa, A. A., Turtle Geometry, MIT Press, Cambridge, (1980).
11. Lumelsky, V. and Skewis, T., "A Paradigm for Incorporating Vision in the Robot Navigation Function," *Proc., Int. IEEE Conf. on Robotics and Automation, Philadelphia*, (1988), pp. 734-739.
12. Kunigahalli, R., Russell, J. S., and Veeramani, D., "Path Planning Algorithm for Computer-Controlled Concrete Placement," *ASCE Journal of Computing in Civil Engineering*, (1993), under review.