Time-Minimal Paths among Moving Obstacles

Kikuo Fujimura
Hanan Samet

Center for Automation Research and
Computer Science Department and
Institute for Advanced Computer Studies
University of Maryland
College Park, Maryland 20742

ABSTRACT

Motion planning for a point robot is studied in a 2-dimensional time-varying environment. The obstacle is a convex polygon which moves in a fixed direction at a constant speed. The point to be reached (referred to as the destination point) also moves along a known path. The concept of 'accessibility' from a point to a moving object is introduced, and it is used to define a graph on a set of moving obstacles. The graph is shown to exhibit an important property, that is, if the moving point is able to move faster than any of the obstacles, a time-minimal path is given as a sequence of edges in the graph. An algorithm is described for generating a time-minimal path and its execution time is analyzed.

Keywords and phrases : time-varying environments, visibility graphs, accessibility graphs, motion planning.

1. INTRODUCTION

The visibility graph [Ghosh87] has been an important combinatorial structure in planning paths among stationary polygonal obstacles and related problems [Loza79, Asan85]. The path planning problem is one of finding a path that connects a given start and destination points in an environment that contains a pre-defined set of obstacles so that the path is collision-free. For a survey of research on motion planning, see [Whit85, Yap86]. One important property of the visibility graph is that shortest paths are given as a finite sequence of edges of the graph.

In this paper, we expand on the concept of *accessibility*, which is a generalization of the visibility concept [Fuji88]. We make use of accessibility to represent moving objects for the purpose of planning the motion of a robot. The robot is assumed to move in a two-dimensional world in which polygonal obstacles, as well as the destination point, are in motion. The *accessibility graph* is a generalization of the visibility graph. Paths to the destination point are found as a sequence of edges of the graph. In fact, when all the obstacles have zero velocity - i.e., when they don't move, the accessibility graph becomes the visibility graph of these polygonal obstacles. Most importantly, we prove that the accessibility graph has a property which is analogous to the visibility graph, in that, time-minimal paths are constructed as a sequence of edges of the graph.

* The support of the National Science Foundation under Grant IRI-88-02457 is gratefully acknowledged.

Previous work dealing with moving obstacles includes the following. Reif and Sharir [Reif85] show that motion planning in a three-dimensional environment containing moving obstacles is PSPACE-hard given bounds on the robot's velocity, and NP-hard without such bounds. Canny and Reif [Cann87] show that motion planning for a point in the plane with a bounded velocity is NP-hard, even when the moving obstacles are convex polygons moving at constant linear velocity without rotation. These results indicate that motion planning with moving obstacles is computationally harder than motion planning with stationary obstacles. Nevertheless, there are some approaches at solving the problem. These approaches are successful in a limited domain.

Kant and Zucker [Kant86] decompose the problem of motion planning with moving obstacles into two parts. In the first part, they plan a path among the stationary obstacles while ignoring the moving obstacles. In the second part, a graph is used to define regions through which the robot may not pass when following the path computed in the first part. The positions of these regions influence the choice of the velocity. Erdmann and Lozano-Perez [Erdm87] make use of stacks (i.e., piles) of two-dimensional Configuration Spaces. These spaces are created each time some object changes its velocity. A path consists of a sequence of vertex-to-vertex transitions between two adjacent elements of the piles. None of these approaches deals with the case that the destination point can also be in motion.

We study the case where the robot can move faster than the obstacles. Given a velocity (i.e., a speed and a direction to proceed), a robot can travel a certain distance before it meets with some obstacle (or it may not meet any obstacle at all). Let us call such a point, if any, the collision point with respect to the velocity. The collision points around the start point, with respect to a certain speed, can be decomposed into subsets which we call *collision fronts*. They consist of a number of curve or line segments. Having computed collision fronts at the start point, the robot moves to an endpoint of one of the collision fronts, say E. At E, the robot is coincident with one of the vertices of the obstacles. Next, we compute another collision front with E as a start point. Now, the robot moves from E to one of the endpoints of the collision fronts that are generated about E. We repeat this until the goal point is finally reached.

Dealing with moving obstacles and a moving destination point is of current interest in robotics. The ability to avoid moving obstacles leads to an increase in the mobility of a robot for navigation. It also results in a higher productivity for factory manipulators. By allowing the destination point to move, we are able to consider a larger class of applications. For example, suppose that the robot arm must pick up an object that lies on a conveyer belt. As another example, consider an autonomous vehicle whose goal is to catch up with another vehicle that is in motion.

Throughout our paper, a goal point in motion is called the *destination point*, and the term *point robot* is used to refer to a point to be moved from the start point to the goal point.

The rest of this paper is organized as follows. In Section 2, we define the motion that obstacles are permitted to take, and introduce the concept of accessibility. Section 3 describes the procedure to find a path and its time-optimality. Section 4 contains an analysis of the execution time of the algorithm. Section 5 contains a few concluding remarks.

2. THE ACCESSIBILITY GRAPH

Throughout this paper, O, G and R are used to denote the start point, the destination point and the point robot, respectively. First, we define the motions of the obstacles, the point robot, and the destination point.

Motion of an Obstacle: An obstacle is a convex polygon which moves in a fixed direction at a constant speed. We call such a straight motion a *movement*. To ease the explanation, we treat the line-segments (edges) that constitute polygons as the basic units of our discussion. A *movement* is defined by a tuple (L,d_L,v_L) which represents the motion of line-segment L in direction d_L at speed v_L. We will consider an environment which contains a finite set of movements, $M=\{M_1,M_2,\ldots,M_n\}$, where each M_i represents a movement as defined above. Note that M corresponds to the motions of all the edges in the environment; not just one polygon. Hence, if a polygon P_i consists of l_i edges (thus vertices), then $n = l_1+l_2+\cdots+l_k$, where k is the number of polygons in the environment. Also, note that throughout this paper, stationary obstacles are treated as obstacles with zero velocity, i.e., we treat both stationary and moving obstacles uniformly.

Motion of a Point Robot: After leaving the start point at a start time, R can take any motion as long as its speed doesn't exceed a given maximum speed. We assume that the maximum speed of R is greater than that of any of the obstacles and that of the destination point.

Motion of a Destination Point: The motion of the destination point consists of a finite series of movements. Within a movement, it moves in a constant direction at a constant speed.

Now, let us define the concepts of accessibility, accessible vertex set, collision front, and accessibility graph.

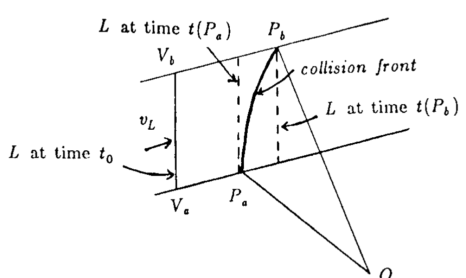

Figure 1

Accessibility: Consider a set of movements $M=\{M_1,M_2,\ldots,M_n\}$ and G, the destination point. Let R be a point robot located initially at O at time t_0. Suppose that R starts moving at time t_0 at a speed v. After it starts moving, the direction of the movement remains fixed. A point V (V is either the destination point or a vertex of a polygonal obstacle) is said to be *accessible* from O, if there exists a direction of the motion of R such that R meets V without a prior interception by any other movements. We say that V and R *meet* if there exists a location X through which both V and R pass at the same time t, where $t_0<t$. The location X is called an *accessible point* of V. t is called the *accessible time* of X with respect to V and is denoted by $t(V)$. The accessible point of a vertex varies for different values of the speed and the initial location of R. Note that the accessible point of a stationary vertex V is V itself, if applicable.

Accessible Vertex Set: Let VS be the vertices of the given polygonal obstacles in motion M, and let O, t_0, v be as in the definition of accessibility. The set of accessible points corresponding to vertices in VS with respect to O, t_0, v is called the *accessible vertex set* and denoted as $AVS(M,O,t_0,v)$. Since some vertices in VS may not be accessible from O, the size of AVS is at most $|VS|$.

Collision Front: Let V_aV_b be an edge of an obstacle and let P_a and P_b be accessible points corresponding to V_a and V_b, with respect to R's initial location O, start time t_0, and speed v. It can be shown that if R keeps moving in a direction that places it within the angle formed by lines OP_a and OP_b at speed v, then R will eventually collide with edge V_aV_b at some point. The set of these collision points with respect to edge V_aV_b forms a (curve) segment, which is called a *collision front* of edge V_aV_b (Figure 1). On the other hand, when R keeps moving outside the angle formed by OP_a and OP_b at speed v, then R does not meet V_aV_b in motion.

Accessibility Graph: Let O be a start point (i.e., the point at which R is initially found), t_0 be a start time, and G a destination point. With each accessible point, X, we also associate a time value, say $t(X)$, to denote X's corresponding accessible time. We define a directed graph called the *accessibility graph*, or $AG(M,O,G,t_0,v)$, by the following construction:

(1) Insert O in the vertex set of AG, and set its default accessible time to t_0.

(2) For every newly added vertex V in the set of vertices in AG, consider the accessible vertex set with V as the initial point, i.e., $AVS(M,V,t(V),v)$. Insert the elements of this AVS in the vertex set of AG, and the edges from V to these points in the edge set of AG.

This graph can be infinite. Note that for a given set of M, O, and G, AG varies for different values of t_0 and v. For stationary M, AG is the same as the visibility graph of M.

Figures 2a and 2b demonstrate how to construct an accessibility graph. S and G are the starting point and the destination point, respectively. ABC and DEFG are obstacles moving towards the indicated directions. First, S is inserted in the vertex set of the accessibility graph (step (1)). B, C, and F are accessible from S at P, Q, and U, respectively, i.e., $AVS(M,S,t(S),v) = \{P, Q, U\}$. Therefore, P, Q, and U are inserted in the vertex set of the accessibility graph. Accordingly, SP, SQ and SU are inserted in the edge set of the accessibility graph (step (2)). Next, let us take Q as a newly added vertex in step (2) of the construction (Figure 2b). Vertices A, B, E, and F are accessible at X, Y, V, and W, respectively, i.e., $AVS(M,Q,t(Q),v)$ = {X, Y, V, W}. Thus, X, Y, V, and W are inserted in the vertex set of the graph, and QX, QY, QV, and QW are inserted in the edge set of the graph. This process is applied to other vertices such as P, U, etc.

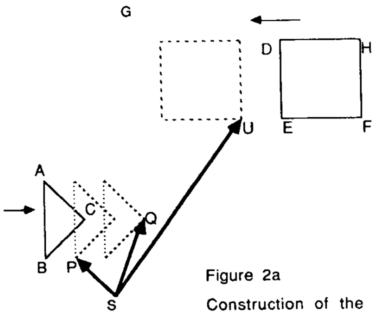

Figure 2a
Construction of the Accessibility Graph

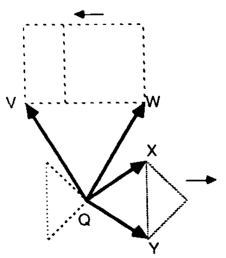

Figure 2b

3. TIME-MINIMAL PATHS

In this section, we first describe a procedure that finds a path to the destination point using AG as defined in the previous section. Next, we prove that the path found by the procedure is time-minimal. In this process, we use a priority queue of vertices where the accessible time of each vertex serves as the vertex's priority.

Procedure FindPath:

(1) Push the starting vertex O onto the queue.

(2) Pop a vertex, say V, whose associated accessible time is the youngest.

(3) If V in step (2) is G, then report the path and exit; otherwise, push all the vertices that are adjacent to V in AG onto the queue, and repeat steps (2) and (3).

Figure 2c is an example of planning a path using *FindPath*. S and G are the start and destination point, respectively. Triangle ABC is an obstacle moving towards the right, and rectangle DEFH is an obstacle moving towards the left. Points Q, V, and T are the accessible points of C from S, of E from Q, and of D from V, respectively. The dashed objects show the locations of the corresponding obstacles at the indicated times. Note that while the robot is moving between V and T, it is coincident with some point on edge ED of obstacle DEFH. SQVTG constitutes the final path.

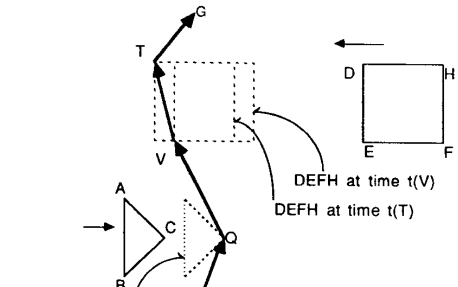

Figure 2c

In the rest of this section, let v_{max} denote R's maximum velocity, and O and G be the start point and the destination point, respectively. We subdivide the paths from O to G into two groups, i.e.,

Group 1 - paths at velocity v_{max} in a constant direction through a point in $AVS(M,O,t_0,v_{max})$.

Group 2 - the other paths.

It is assumed that velocities along paths in both groups don't exceed v_{max}. In the rest of this paper, we assume the following two conditions.

(A1) v_{max} is greater than that of any of the obstacles,

(A2) The obstacles do not touch among themselves, and the obstacles and the destination point do not touch.

Proposition 1. Given a set of movements M and a destination point G, suppose that at time t_1 point R and vertex V, the common endpoint of edges UV and VW, are at location X. Let M_i and M_j be the movements of UV and VW, respectively, and let M_k be the movement corresponding to the motion of a new edge UW. If there exists a path starting from X at time t_1 and terminating at G at time t_2 ($>t_1$) in $M-\{M_i, M_j\}+\{M_k\}$ (i.e., an environment formed by removing two edges UV and VW and adding UW), then there exists a path in M, starting from X at time t_1 and terminating at G at time t_2.

Proof: Let α be a path from X to G in $M-\{M_i, M_j\}+\{M_k\}$. There are two cases depending on whether or not α collides with M_i (or M_j) in M.

(Case 1): *α doesn't collide with M_i* nor M_j in M.

α is a collision-free path in M because it cannot collide with M_k without first colliding with either M_i or M_j which is impossible. Thus α satisfies the proposition.

(Case 2): *α collides with M_i* (or M_j) in M.

Without loss of generality, let L denote the edge with which α collides. We construct an alternative path β such that β is collision-free in M in the following manner. Let Y be the location of the occurrence of the last collision of α with L (See Figure 3). Starting at time t_1, define β as the shortest path followed by R from X to Y. This means that until reaching Y, R is always coincident with some point on L. This path is shorter than the corresponding path-segment of α. To see this, note that the shortest distance between two points is a straight line. After Y, R follows the remaining

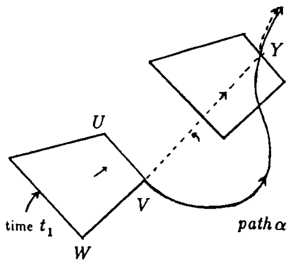

Figure 3

points is a straight line. After Y, R follows the remaining part of path α. R reaches G at the same time regardless of whether path α or path β is used. Along path β, R moves at a slower speed than v_{max} from X to Y. To see this, consider two paths from X to Y, namely α and β, both starting and arriving at the same time. Since path β is shorter than path α, the velocity of R between X and Y must be less than v_{max}. ■

As a special case of this Proposition, we consider a case in which an obstacle is just a single edge. We have the following proposition, and its proof is similar to the proof of Proposition 1.

Proposition 1′. Assume the same environment as in Proposition 1 except that UV is a single edge obstacle. Suppose that at time t_1 point R and vertex V are at location X. If there exists a path starting from X at time t_1 and terminating at G at time t_2 in M-$\{M_i\}$, then there exists a path in M, starting from X at time t_1 and terminating at G at time t_2.

Proposition 2. Let π_2 be a collision-free path from O to G. Let π_2 be in Group 2. There exists a collision-free path from O to G in Group 1, say π_1, which will be identical to π_2 starting at some point J. The speed along some portion of π_1 before J is less than v_{max}.

Proof: Let A be a region such that every point in it is accessible from O when R moves at velocity v_{max}, and let I be the complement of A (i.e., its points are not accessible from O when R moves at v_{max} (Figure 4a). Let Z be a point at which path π_2 crosses from A to I (Figure 4b).

In the first part of the proof, let us assume that there is only one collision front about O. Let V_1 and V_2 denote the two endpoints of the collision front (Figure 4b). The inaccessible area is bounded by the collision front formed by V_1 and V_2 and the two infinite lines extending from O to V_1 and O to V_2, respectively. Without loss of generality, let Z lie on the line that passes through V_1. The case that Z lies on the collision front V_1V_2 can be handled in a similar manner. Consider path π_1. On this path, R first moves from O to V_1 at v_{max}. Next, let R move along a straight line from V_1 to some point J on path π_2 (denoted as a dotted line in Figure 4b). We now show that there exists a point J on π_2 such that R can move from V_1 to J at a speed less than v_{max}. Note that the path length from O to Z along π_2 is longer than OZ. Therefore, when R's speed along π_2 is fast (e.g., v_{max}), a point J near Z will satisfy this property. When R's speed along π_2 is slow, it is possible that path length OJ is even shorter than OV_1J along path π_1. Now, all that remains is to show that the subpath of π_1 between V_1 and J is collision-free. Suppose that it is not collision-free. Therefore, there exists a point C on an obstacle initially in I which will intersect the line V_1J at some point Y. Since the path on the infinite line OV_1 is collision free, C can only intersect the extension of line OV_1 at some point Q at a time later than $\frac{V_1Q}{v_{max}} + t(V_1)$ where $t(V_1)$ is the accessible time of V_1. The speed of R along π_1 from V_1 to J can be chosen so that R will pass Y before C reaches it. This can be seen by appealing to the triangle inequality (with respect to QV_1Y). Recall that the motion of all the obstacles (such as C) are known a priori and that the speed of the obstacle is less than v_{max}. Hence, we have a contradiction and thus the subpath of π_1 between V_1 and J is collision-free.

Next, we consider the case where there is more than one collision front generated at O. If there are n collision fronts, then there are at most n inaccessible areas. Let us use I_i to denote an inaccessible area behind collision front CF_i (Figure 4c). There are $2n$ choices for path π_1 - two for each collision front. For each of the collision fronts, CF_i, there is a range of speed values $[v_{max}\text{-}\epsilon_i,\ v_{max})$ such that the corresponding path π_{1i} joins path π_2 at point J_i. Pick a value of the velocity v_ϵ such that v_ϵ is in each of $[v_{max}\text{-}\epsilon_i,\ v_{max})$. Choose the path π_{1i} such that J_i is reached at the earliest time. Now we must show that this path is collision-free. Suppose that it is not. In this case, the obstacle must come from some inaccessible area I_j $(j \neq i)$. This means that a path that exits from an accessible point of CF_j will join path π_2 at an earlier time, which is a contradiction. ■

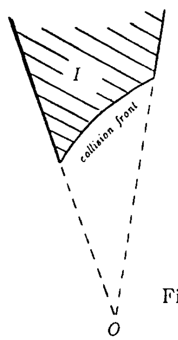

Figure 4a

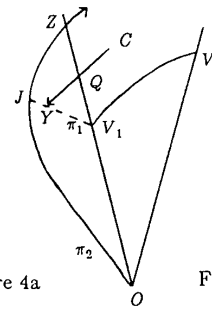

Figure 4b

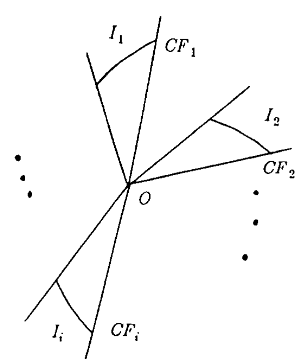

Figure 4c

Theorem: A time-minimal path is given as a sequence of edges of $AG(M,O,G,t_0,v_{max})$.

As a result of this theorem, R moves at the maximum velocity along a time-minimal path. In other words, if a path contains some portion during which R moves at a slower speed, then that path is not time-minimal. We prove the theorem by induction on n, the total number of movements in the given environment.

Proof: (Base Step): In the base case in which $n=0$, G is accessible from O, the start point, since there are no obstacles (stationary or moving) in the scene. Recall that we treat stationary obstacles as obstacles with zero velocity. Thus, G's accessible point, X, is in $AVS(M,O,t_0,v_{max})$, and the procedure terminates with path OX along which R moves at v_{max}. Now we show that path OX is time-optimal. Suppose that it is not. Let γ be a time-optimal path and Y be the point at which R reaches G. Path γ can be extended to point X such that R is always coincident with G from Y to X. Now, there are two paths from O to X, i.e., OX and γ plus its extension, both of which reach X at the same time. This is a contradiction since R moves at v_{max} along straight line OX.

(Inductive step): Next, we consider an environment which contains n movements. We assume that the theorem holds for an environment with $n-1$ movements. If G is accessible from O, then the theorem obviously holds as in the case of $n=0$. Therefore, assume that G is not accessible from O. The theorem follows from the following two properties which are proved below. Let V be a point in $AVS(M,O,t_0,v_{max})$. First, we prove that a time-minimal path from V to G in M consists of a sequence of edges in $AG(M,V,G,t(V),v_{max})$. Second, we prove that for any path in Group 2, a better path in Group 1 exists. Thus a path in Group 2 is never time-minimal.

(Proof of the first property):

Let V be an accessible point (from O) of a vertex in movement M_i. Note that at time $t(V)$, R is on a vertex of some polygon. Suppose that L_i and L_j are two edges that are incident on the vertex. We remove L_i and L_j from the polygon and instead introduce a new edge L_k which connects two endpoints of L_i and L_j that are not incident at the vertex. This results in a polygon which has one less edge than before. By the induction hypothesis, a time-minimal path γ exists from V to G in $M-\{M_i,M_j\}+\{M_k\}$, which consists of the edges of $AG(M-\{M_i, M_j\}+\{M_k\}, V, G, t(V), v_{max})$, where M_i, M_j, and M_k are the movements of L_i, L_j, and L_k, respectively. We show that this path γ is also a collision-free path in M. In other words, γ doesn't collide with M_i or M_j.

Assume that γ collides with M_i. Since R is located at V, we can use Proposition 1 to construct a path γ' in M which terminates at G at the same time as γ in $M-\{M_i,M_j\}+\{M_k\}$. In case M_i is a single edge, then Proposition 1' is applied and $M-\{M_i\}$ is used in place of $M-\{M_i, M_j\}+\{M_k\}$ in the following argument. γ' must be one of the time-minimal paths from V to G in $M-\{M_i, M_j\}+\{M_k\}$, as it terminates at G at the same time as γ. However, γ' contains a path-segment, along which R moves at a slower speed than v_{max} (as noted in the proof of Proposition 1). Therefore, by the induction hypothesis, γ' cannot be a time-minimal path from V to G in $M-\{M_i, M_j\}+\{M_k\}$. Since γ and γ' terminate at G at the same time, γ is also not a time-minimal path, which is a contradiction (recall our initial assumption was that γ is time-minimal). Thus, path γ doesn't collide with M_i and is a collision-free time-minimal path from V to G in M as well. This can be seen by noting that if a collision-free path is time-minimal $M-\{M_i, M_j\}+\{M_k\}$, then it is also time-minimal in M.

We now show that a path from V to G in M, say π, containing a segment during which R moves at a velocity less than v_{max} will terminate at G at a time later than γ. By the induction hypothesis, in $M-\{M_i,M_j\}+\{M_k\}$, π terminates at G later than γ. Since γ is also a path in M, as we showed above, π terminates at G later than γ in M, and thus π is not time-minimal in M. Thus our first property has been proved.

(Proof of the second property):

Assume that there exists a path α in Group 2. We show that there exists a path in Group 1 which reaches G earlier than α. Let β be a path from O to G constructed in the following manner. After leaving O at time t_0, β moves to one of the vertices in $AVS(M,O,t_0,v_{max})$ (let us call this vertex V) at velocity v_{max}, and then turns to rejoin path α at a velocity v less than v_{max}. Proposition 2 assures that this rendezvous is possible. After rejoining α, the remaining parts of path β are identical to path α, and β terminates at G at the same time as α. On the other hand, there exists a time-minimal path γ from V to G in M, as described in the first property. Since β has a path-segment along which R moves at a slower speed than v_{max}, β is not time-minimal from V to G and β terminates at G later than γ. This implies that we can construct a path in M that terminates at G earlier than α by concatenating paths β and path γ at V. Thus our second property has been proved.

From the first property, we have that the time-minimal path from V to G consists of edges in $AG(M,V,G,t(V),v_{max})$ which is a subgraph of $AG(M,O,G,t_0,v_{max})$. From the second property, a time-minimal path from O to G must contain edge OV, which is an edge of $AG(M,O,G,t_0,v_{max})$. Thus the time-minimal path is a sequence of edges of $AG(M,O,G,t_0,v_{max})$. ■

4. EXECUTION TIME

We now analyze the time required to generate a collision-free path using the algorithm described in Section 3. First, we show that the shape of the collision front is either a parabola, hyperbola or an ellipse. Without loss of generality, we consider a line segment L lying parallel to the y-axis. Let (x_0,y_0) be the location of one of L's endpoints at the start time, l be the length of L, v_L be the velocity of L, v be the velocity of the robot, and θ be the angle formed by the direction of the movement and the x-axis (Figure 5). Suppose that a point P at (x_0,Y) in L is accessible from the origin (0,0) at point A at (x,y). x and y must satisfy:

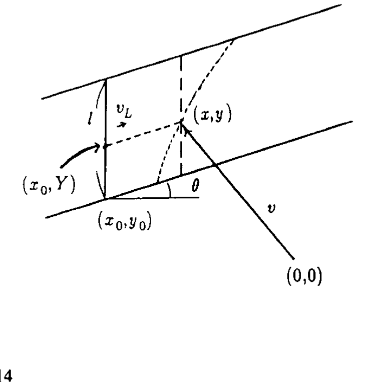

Figure 5

$$t(A)=\frac{\sqrt{x^2+y^2}}{v_{\max}}=\frac{\sqrt{(x-x_0)^2+(y-Y)^2}}{v_L} \quad (1)$$

$$y=(x-x_0)\tan\theta+Y \quad (2)$$

$$y_0 \leq Y \leq y_0+l \quad (3)$$

The points (x,y) that satisfy (1) and (2) form the collision front. These equations define a quadratic relationship - i.e., the collision front lies either on a parabola, hyperbola, or ellipse. (The curve degenerates to a straight line when the direction of the movement is parallel to L.) Also, we can show that the origin is on the same side as one of the foci of the curve. Generally, the two endpoints of a collision front (accessible points) can be computed by setting Y to y_0 and y_0+l in equations (1) and (2).

Having computed two endpoints of the collision front of a single movement, we must now compute all the endpoints of the collision fronts (AVS) when the robot is at a given start point with a given velocity v. For a set of polygonal obstacles with n vertices, there are n candidate points. However, some points may not be accessible from the start point since some other movement intercepts the accessibility of that point. We now show that it takes $O(n\log n)$ time to compute AVS. The set of accessible points is determined by using a similar technique to that used to compute the visibility of a given set of line segments. This is also known as a plane-sweep algorithm [Prep85]. This is a two-step process. The first pass sorts all the vertices, say in a clockwise direction with respect to O. This sorting process takes $O(n\log n)$ time. The second pass rotates a line about O. It halts each time the line intersects a vertex, and checks whether or not the collision front associated with the vertex is accessible from O. This process can be achieved in $O(\log n)$ time by using a 2-3 tree [Aho74] to maintain the active collision fronts based on their distance from O. In this way, the closest collision front is marked as accessible from O. Since it takes at most $O(n\log n)$ time to build the initial 2-3 tree and $O(\log n)$ time for each update at a vertex, the determination of accessibility of n candidate points takes $O(n\log n)$ time. Therefore, given a start point and a set of candidate points, it takes $O(n\log n)$ time to compute accessible vertices from the start point.

At this point, note that as a result of our theorem, a path that meets with the same vertex of an obstacle more than once is not time-minimal. As there are n vertices in the environment, an AVS needs to be generated at most n times before a time-minimal path reaches the destination point. This observation leads to an asymptotic computation time of $O(n^2\log n)$ to compute a path using procedure *FindPath*. Note that for stationary polygonal obstacles, a shortest length path can be computed using the visibility graph in $O(n^2)$ time [Welz85] and $O(E+n\log n)$ time [Ghos87], where n is the total number of vertices in all the obstacles and E is the number of edges in the visibility graph.

5. CONCLUDING REMARKS

Time-optimal trajectory planning has been an important subject in robotic control [Saha84, Bobr85, Cann88]. We have studied the problem of moving a point robot among the obstacles moving at constant velocities in the plane. We have discussed a case where the robot can move faster than the obstacles and the destination point. Making use of the concept of accessibility, we have demonstrated an $O(n^2\log n)$ algorithm to find a path, and proved that the path found is time-minimal. Because of its time optimality, the accessibility graph is expected to be an important tool in motion planning in dynamic environments, just as the visibility graph played a key role in the studies of motion planning among stationary obstacles. However, an extension to a path finding problem for a finite-size robot in dynamic environments doesn't seem to be straightforward as is the case with stationary obstacles.

REFERENCES

[Aho74] - A. V. Aho, J. E. Hopcroft, and J. D. Ullman, *The design and analysis of computer algorithms,* Addison-Wesley, Reading, Massachusetts, 1974.

[Asan85] - T. Asano, T. Asano, L. Guibas, J. Hershberger, and H. Imai, Visibility of disjoint polygons, *Algorithmica 1,* 1(1986), 49-63.

[Bobr85] - J. Bobrow, S. Dubowsky, and J. Gibson, Time-optimal control of robot manipulators, *International Journal on Robotics Research 4,* 3(1985 Fall), 3-17.

[Cann88] - J. Canny, B. Donald, J. Reif, and P. Xavier, On the complexity of kinodynamic planning, *Proceedings of the 29th Annual IEEE Symposium on Foundations of Computer Science,* White Planis, NY, October 1988, 306-316.

[Cann87] - J. Canny and J. Reif, New lower bound techniques for robot motion planning problems, *Proceedings of the 28th Annual IEEE Symposium on Foundations of Computer Science,* Los Angeles, CA, October 1987, 49-60.

[Erdm87] - M. Erdmann and T. Lozano-Perez, On multiple moving objects, *Algorithmica 2,* 4(1987), 477-522.

[Fuji88] - K. Fujimura and H. Samet, Accessibility: a new approach to path planning among moving obstacles, *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition,* Ann Arbor, MI, June 1988, 803-807.

[Ghos87] - S. K. Ghosh and D. M. Mount, An output sensitive algorithm for computing visibility graphs, *Proceedings of the 28th Annual IEEE Symposium on Foundations of Computer Science* Los Angeles, CA, October 1987, 11-19.

[Kant86] - K. Kant and S.W. Zucker, Toward efficient planning: the path-velocity decomposition, *International Journal of Robotics Research 5,* 3 (Fall 1986), 72-89.

[Loza79] - T. Lozano-Perez and M.A. Wesley, An algorithm for planning collision free paths among polyhedral obstacles, *Communications of the ACM 22,* 10(October 1979), 560-570.

[Reif85] - J. Reif and M. Sharir, Motion planning in the presence of moving obstacles, *Proceedings of the 26th Annual IEEE Symposium on Foundations of Computer Science,* Portland, OR, October 1985, 144-154.

[Saha84] - G. Sahar and J. Hollerbach, Planning of minimum-time trajectories for robot arms, AI Memo, No. 804, MIT, November 1984.

[Shil88] - Z. Shiller and S. Dubowsky, Global time optimal motions of robotic manipulators in the presence of obstacles, *IEEE International Conference on Robotics and Automations,* Philadelphia, PA, April 1988, 370-375.

[Welz85] - E. Welzl, Constructing the visibility graph for n line segments in $O(n^2)$ time, *Information Processing Letters 20,* 4(May 1985), 167-171.

[Whit85] - S. H. Whitesides, Computational geometry and motion planning, in *Computational Geometry* (Ed., G. T. Toussaint), North-Holland, Amsterdam, The Netherlands, 1985, 377-427.

[Yap85] - C. K. Yap, Algorithmic motion planning, in *Advances in Robotics Vol. 1* (Ed., J. T. Schwartz and C. K. Yap), Lawrence Erlbaum, NJ, 1986, Chapter 3.