A PATH PLANNING ALGORITHM FOR AUTOMATED CONSTRUCTION EQUIPMENT

By Seungho Lee[a] and Teresa M. Adams[b]

[a] *Ph.D. Candidate, Dept. of Civil and Environmental Engineering, University of Wisconsin, Madison, WI. 53706*
[b] *Associate Professor, Dept. of Civil and Environmental Engineering, University of Wisconsin, Madison, WI. 53706*

This paper presents a new sensor-based path planning algorithm, the ACE algorithm, in unknown environment with multiple moving obstacles. The ACE algorithm is designed for an automated construction equipment (ACE). The algorithm is based on the practical assumptions that fit to the construction environment. It is assumed that the robot can measure instantaneous velocity of obstacles in a range of vision and has a cache memory to memorize a generated path from a *tracking* point. The ACE algorithm is an extension of the Tangent algorithm [5, 6]. The ACE algorithm guarantees reachability in an unknown environment with not only multiple moving obstacles but also composite obstacles. If an obstacle that is being tracked moves and/or if the robot returns to the generated path after a *tracking* point in clockwise tracking direction, the ACE algorithm terminates *Tracking* mode and resumes *Toward Destination* mode.

Keywords: ACE algorithm, Multiple moving obstacles, Composite obstacle.

1. INTRODUCTION

Many construction robots have been developed and tested by large construction companies and research institutes all over the world since 1980 [1, 3, 4]. However, the navigation systems of current construction robots far from satisfying the intelligent navigation that is required to interact with the changing environment of a construction site [11,13]. In order to address the unstructured and changing construction environment, automated construction equipment (ACE) must be equipped with sophisticated collision-free path planning capabilities.

Because the environment of the construction site is unstructured and changing, it is reasonable to apply the path planning algorithm in an unknown environment to the construction site. The path planning algorithm in an unknown environment includes Bug1 [7], Bug2 [7], VisBug [8], Curv1 [12], Azimuth [9], Tangent [5, 6], and DistBug [2].

These algorithms assume that all obstacles are stationary. If an environment is populated by not only stationary but also moving obstacles, then those sensor-based path planning algorithms may fail. For example, in the Bug2 algorithm, the robot moves towards its destination along a straight line until it encounters an obstacle. It then follows the boundary of the obstacle in the local direction (right or left) until a certain leave condition is satisfied. If the obstacle is moving, the leave condition cannot be satisfied and thus the robot cannot leave the boundary of the obstacle [10]. This example is illustrated in Figure 1. While Moving from S to T, a robot R (shown as a small black disc) encounters obstacle M. After defining the hit point H, the robot starts moving around M in the local direction (Figure 1a). If while the robot moves around M, M moves to the new position, then the Bug2 algorithm would cause the robot to go around M indefinitely because the robot would never meet the line from S to T (Figure 1b).

Recently, Lumelsky and Harinarayan (1997) presented a new strategy, the Cocktail Party Model, for finding a path in an unknown environment with multiple moving obstacles. The Cocktail Party Model guarantees that the robot arrives at the destination. However, the Cocktail Party Model requires human level of intelligence to distinguish between a stationary obstacle and a moving obstacle, even though the moving obstacles is not moving. For example, the robot must be able to distinguish between a mannequin and a man at a glance. Thus, although the Cocktail Party Model can escape from an unknown environment with multiple moving obstacles, it is impractical to apply this model to automated construction equipment unless human level of image recognition is realized.

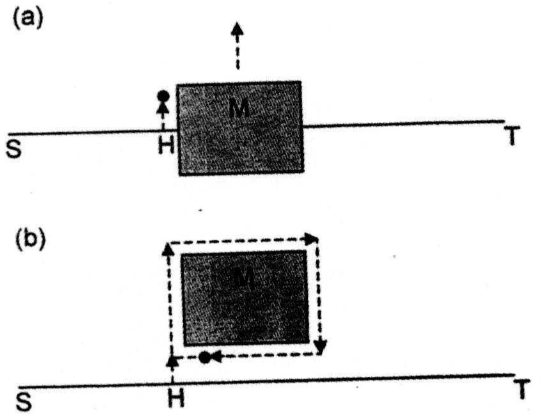

Figure 1. Problem of Bug style sensor-based path planning algorithm in an unknown environment with moving obstacles [10].

2. ASSUMPTION

This paper focuses on the development of a sensor-based path planning algorithm for unknown environment with multiple moving obstacles with practical assumptions that fits to the construction environment.

Environment. The construction site is two-dimensional plane that is populated by obstacles. There are two types of the obstacles, stationary obstacles and moving obstacles. The stationary obstacles include buildings, foundations, and soil heaps, and have no constraints on their shape for generality. The moving obstacles represent vehicles and people (There are no other moving obstacles). The shape of all moving obstacles is convex. In addition, moving and/or stationary obstacles comprise a composite obstacle when those obstacles are simultaneously in contact and establish a obstacle. The composite obstacle may split later when the robot is following its boundary. The boundary of an obstacle is a simple closed curve that represents its shape.

Sensor. The input information includes start location, current location, destination, and sensor feedback. The robot is equipped with an appropriate positioning method like GPS and achieves local information about its environment via vision sensors as input. Perfect sensing and accurate position information are assumed. This means that sensor inaccuracies are considered to be independent of the planning algorithm [10]. In addition, the robot is equipped with range sensors to measure distances to any obstacles within the radius of vision and can measure an instantaneous velocity of obstacles within the radius.

3. APPROACH

Simple Algorithm. Because we assume that moving obstacles in the construction site are construction equipment and workers, a simple algorithm that navigates a robot directly toward its destination can be used to escape from them. Whenever the robot meets an obstacle, the robot follows the obstacle's boundary until a path towards the destination is clear. If the obstacle is convex, the simple algorithm can find a destination whether the convex obstacle is stationary or moving because this simple algorithm does not have any hit or leave condition. Intuitively, it is a same strategy that a person walks through a crowded place toward destination with avoiding collision to other people. However, the simple algorithm may not arrive at its destination in an unknown environment with stationary obstacles in some configurations even if the obstacles themselves are convex [7]. Two examples are shown in Figure 2. Although the size of each obstacle is finite, the algorithm causes the robot to follow the obstacles to infinity (Figure 2a) or to loop infinitely around the obstacle field (Figure 2b).

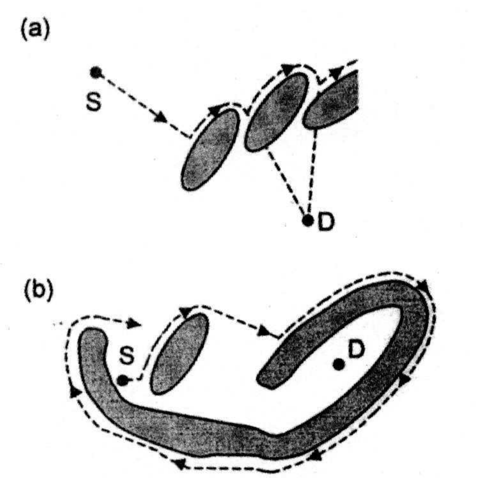

Figure 2. Situations when the Simple Algorithm Fails [7].

Tangent Algorithm. Lee et al [5,6] updated the simple algorithm and developed a sensor based path planning algorithm, the Tangent algorithm, that guarantees reachability in an unknown environment with stationary obstacles in any configuration by introducing a *tracking* point. The Tangent algorithm has three navigation modes: *Toward Destination,* *Searching*, and *Tracking*. The *Toward Destination* mode navigates toward the destination. When an obstacle is detected, *Searching* mode tries to find a clear path toward the destination while avoiding the obstacle. Together with *Toward Destination* and *Searching* modes comprises the simple algorithm and thus the Tangent algorithm can overcome a moving convex obstacle at a time. *Tracking* mode navigates the perimeter of an obstacle and enables the robot to escape from concave obstacles fields.

Composite Obstacle. The Tangent algorithm is not applicable to situations when several obstacles that are simultaneously in contact and establish a concave obstacle. The concave composite obstacle may split later when the robot is in *Tracking* mode. As shown in Figure 3, a robot R encounters a composite obstacle that consists of a stationary obstacle S, a moving obstacle M_1, and a moving obstacle M_2. R has only local information and cannot recognize the whole configuration of the composite obstacle. The composite obstacle is concave. R defines T as a *tracking* point on the boundary of the composite obstacle. If obstacle M_1 is splitting from the composite obstacle while R is tracking M_1 (Figure 3a), then R loses track and cannot meet a leave point L (Figure 3b).

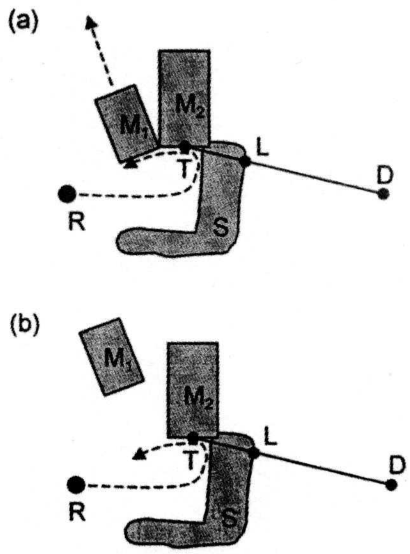

Figure 3. If an obstacle that is being tracked moves, a robot cannot meet the leave point.

There are two possible scenarios for R. R can continue to track M_1. In this case R will track M_1 indefinitely. Otherwise, R can stop tracking and resume *Toward Destination* mode again. In this case R will again encounter the combined obstacle and escape from the composite obstacle. Thus, to overcome this problem, the Tangent algorithm should be revised to resume *Toward Destination* mode whenever an obstacle that is being tracked moves.

As shown in Figure 4, R is tracking a composite obstacle and one obstacle component M_2 moves (Figure 4a). Because R has only local information, R is not aware that M_2 moves. Consequently, R will track M_1 infinitely (Figure 4b). To overcome this difficulty, the Tangent algorithm should be revised to record the path from T to the current location. If the current location overlaps with the memorized path, then the robot resumes *Toward Destination* mode. However, this is not enough to overcome this situation as shown in Figure 4c. If the robot tracks in counterclockwise direction and the current location overlaps with the memorized path, then the above strategy causes the robot trapped in the obstacle because configuration is not changed even though the robot returns to the memorized path.

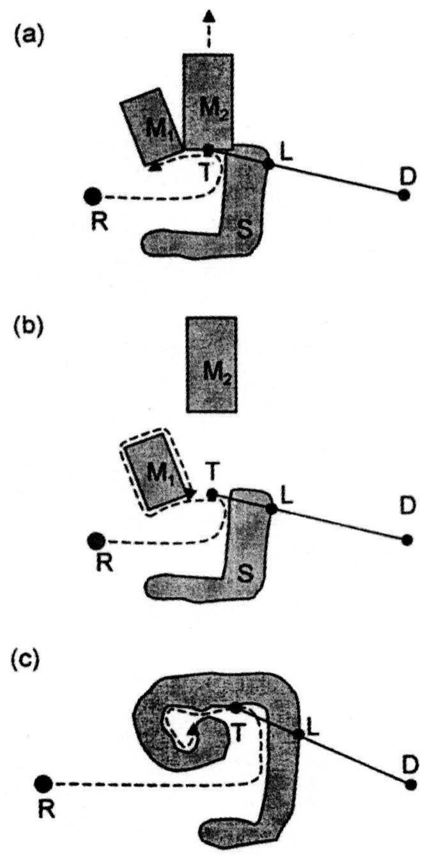

Figure 4. If an obstacle that was tracked moves, the robot may be trapped in an infinite tracking loop.

The difference between two cases is a direction of tracking. When the robot tracks in clockwise direction and meets the memorized path, the memorized path embeds the obstacle (Figure 4b). When the robot tracks in counterclockwise and meets the memorized path, the obstacle embeds the memorized path (Figure 4c).

Table 1. Parameters of the ACE Algorithm

Parameter	Description
Θ	Angle in global coordinates between the forward direction of the robot and the destination.
Θ_{start}	Initial value of Θ.
$\Theta_{current}$	Value of Θ at any position on the navigation path.
Θ_{ref}	Known direction for navigation in *Toward Destination* mode. The initial value of Θ_{ref} is Θ_{start}. At a *leave* point, $\Theta_{ref} = \Theta_T$ for the corresponding *tracking* point.
P	Memory for recording x-y coordinate from a *tracking* point to a current location.
Tracking point	Point on the obstacle perimeter where the robot, in *Searching* mode, begins to move away from the destination, i.e., $\lvert\Theta_{current} - \Theta_{ref}\rvert > 180°$.
Θ_T	Value of Θ at *tracking* point, T.
Leave point	Point on the line from a *tracking* point to the destination where the algorithm, in *Tracking* mode, resumes *Toward Destination* mode.

Table 2. Pseudo code for the Modes of the ACE Algorithm

Mode	Pseudo Code
Toward Destination	Set $\Theta_{re} = \Theta_{start}$, P = empty Move toward the destination until one of the following occurs: • An obstacle is encountered. Go to *Searching* mode. • The destination is reached. The procedure stops.
Searching	Follow the perimeter of the obstacle. Compute $\Theta_{current}$ at each step. Continue until one of the following occurs: • The direction toward the destination clears. Go to *Toward Destination* mode. • $\lvert\Theta_{current}-\Theta_{ref}\rvert > 180°$. Go to *Tracking* mode.
Tracking	Define the current point as *tracking* point T and store Θ_T. Calculate the line from the *tracking* point to the destination. Follow the perimeter of the obstacle until one of the following occurs with memorizing a generated path after T in *P*: • If the obstacle is moving. Go to *Toward Destination* mode. • If the current location $\in$ *P and tracked clockwisely.* Go to *Toward Destination* mode. • The robot meets a *leave* point along the line from the *tracking* point to the destination. Reset $\Theta_{ref} = \Theta_T$. Go to *Toward Destination* mode. • The robot returns to T. The procedure stops. There is no solution because a closed curve along the obstacle is completed.

Thus, if the robot tracks a boundary in clockwise direction and meets the memorized path, then a configuration of the obstacle changes and the robot resumes *Toward Destination* mode. This revision to the Tangent algorithm will to avoid infinite loops.

4. *ACE Algorithm*

Table 1 and 2 contain the formulation of the ACE algorithm for sensor-based path planning in an unknown environment with multiple moving obstacles for an automated construction equipment (ACE). The ACE algorithm is an extention of the Tangent algorithm.

The parameters of the ACE algorithm are defined in Table 1. Pseudo code for the ACE algorithm is listed in Table 2. The ACE algorithm begins by setting $\Theta_{ref} = \Theta_{start}$. The algorithm always starts in *Toward Destination* mode. The robot navigates toward the destination. If the robot meets an obstacle, the algorithm changes to *Searching* mode, otherwise the destination is reached and the algorithm stops. In *Searching* mode, the robot follows the perimeter of the obstacle. At each position in *Searching* mode, the algorithm tests whether the direction toward the destination clears, or $|\Theta_{current} - \Theta_{ref}|$ is greater than 180°. If the robot clears the obstacle, the algorithm changes to *Toward Destination* mode. If $|\Theta_{current} - \Theta_{ref}|$ is greater than 180°, the algorithm changes to *Tracking* mode. In *Tracking* mode, the algorithm defines the current position as a *tracking* point, stores $\Theta_T = \Theta_{current}$ and memorizes its path in ***P*** while *Tracking* mode until it resumes *Toward Destination* mode. If the robot realizes that the currently tracking obstacle is moving, then the algorithm resumes *Toward Destination* mode. Otherwise, the robot tracks the perimeter of the obstacle until it meets a *leave* point (along the line from the *tracking* point to the destination), the *tracking* point, or the memorized path ***P*** in clockwise tracking direction. If the robot meets a *leave* point, the algorithm resets $\Theta_{ref} = \Theta_T$ and changes to *Toward Destination* mode. If the robot meets the *tracking* point again, the algorithm stops because the destination cannot be reached. If the robot meets the memorized path ***P*** in clockwise tracking direction, then the algorithm resumes *Toward Destination* mode.

5. *EXAMPLES*

Figure 5 shows a path generated by the ACE algorithm in an unknown environment with multiple moving obstacles. Two stationary obstacles and four moving obstacles (M_1, M_2, M_3, and M_4) populate the environment. M_1, M_2, M_3, and a stationary obstacle consists of a composite obstacle. Although M_1 is a moving obstacle, it does not move. The robot R starts toward the destination. From points A to B, the robot follows a path generated by *Searching* mode. At B, $|\Theta_{current} - \Theta_{ref}| > 180$ degrees, thus, B is defined as a *tracking* point and starts to save a generated path on P. The algorithm changes to *Tracking* mode. Point D is the leave point because D is along the line between B and the destination. After R passes C, M_2 and M_3 split away and R is not aware of it. When R returns to C, the algorithm resumes to *Toward Destination* mode because the current location is belong to ***P*** in clockwise tracking direction. From C to E, the robot follows a path generated by *Toward Destination* and *Searching* modes. From E to F, the robot follows a path generated by *Searching* mode. At F, the algorithm resumes *Tracking* mode and changes *Toward Destination* mode at G. At H, the robot meets M_4 and resumes *Searching* mode and overcomes M_3. After passing M_4, the algorithm resumes *Toward Destination* mode and arrives at the destination.

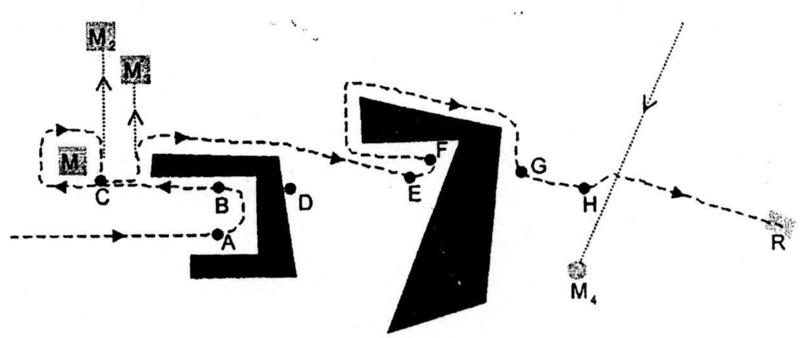

Figure 5. Example of Algorithm Performance

6. CONCLUSION

This paper presents a new sensor-based path planning algorithm, the ACE algorithm, for unknown environment with multiple moving obstacles for an automated construction equipment (ACE). It is assumed practical assumptions that fit to the construction environment; the construction site is populated by stationary, moving, and composite obstacles.

None of existing sensor based path planning algorithms except the Cocktail Party Model in an unknown environment guarantee reachability in those environment. Although the Cocktail Party Model can escape from an unknown environment with multiple moving obstacles, it is impractical to apply this model to the construction equipment until human level of image recognition is realized. A prerequisite of the Cocktail Party Model is the ability to distinguish between a stationary and moving obstacles, even though the moving obstacles are not moving.

The ACE algorithm does not require human level of image recognition and guarantees reachability in the construction site environment. The ACE algorithm can escape from stationary maze-like obstacles as well as moving convex obstacles because it is based on the Tangent algorithm. The ACE algorithm improves the Tangent algorithm to overcome composite obstacles.

In addition the ACE algorithm interacts with the construction site environment via sensors that are equipped with automated construction vehicle in real time. The local information via sensor feedback is directly used to plan a path and thus can confront with the changing environment of the construction site appropriately. Thus, the ACE algorithm can be applied to develop an intelligent navigation system for automated construction equipment (ACE) that is required to interact with the changing environment of a construction site.

REFERENCE

[1] Cousineau, L and Miura, N. (1998). *Construction Robots: The Search for New Building Technology in Japan*, ASCE PRESS, Reston, Virginia.

[2] Kamon, I. and Rivlin, E. (1997). "Sensory-Based Motion Planning with Global Proofs," *IEEE Transactions on Robotics and Automation*, 13(6):814-822.

[3] Kangari, R. and Yoshida, T. (1989). "Prototype robotics in construction industry," *Journal of Construction Engineering and Management*, 115(2):284-301.

[4] Kangari, R. (1990). "Automation in construction," *Robotics and Autonomous Systems*, vol. 6. pp. 327-335.

[5] Lee, S. H., Ryoo B. Y, and Adams. T. M. (1995). "A fuzzy navigation system for mobile construction robots," *Proceedings of 2nd Congress on Computing in Civil Engineering, ASCE, Atlanta, Georgia*, pp. 719-726.

[6] Lee, S. H., Adams. T. M., Ryoo B. Y. (1997). "A fuzzy navigation system for mobile construction robots," *Automation in Construction*, 6:97-107.

[7] Lumelsky, V. J. (1987). "Algorithmic and complexity issues of car in an uncertain environment," *Journal of Complexity*, 3:146-182.

[8] Lumelsky, V. J and Skewis, T. (1988). "A Paradigm for Incorporating Vision in the Robot Navigation Function," *Proc. of the 1988 IEEE International Conference on Robotics and Automation*, Philadelphia, PA, pp. 734-739.

[9] Lumelsky, V. J and Tiwari, S. (1994). "An Algorithm for Maze Searching with Azimuth Input," *IEEE* pp. 111-116.

[10] Lumelsky, V. J and Harinarayan, K. R. (1997). "Decentralized Motion Planning for Multiple Mobile Robots: The Cocktail Party Model," *Autonomous Robots*, 4:121-135.

[11] Poppy. W. (1994). "Driving forces and status of automation and robotics in construction in Europe," *Proceedings of 11th International Symposium on Automation and Robotics in Construction (ISARC)*, Brighton, U.K., pp.215-223.

[12] Sankaranarayanan, A. and Masuda, I. (1992). "A New Algorithm for Robot Curve-Following amidst Unknown Obstacles, and a generalization of Maze-Searching," *Proceedings of the IEEE International Conference on Robotics and Automation*, Nice, France, pp. 2487-2494.

[13] Warszawski, A. and Navon, R. (1998). "Implementation of robotics in building: current status and future prospects," *Journal of Construction Engineering and Management*, 124(1):31-41.