Research article

On-line path planning strategy integrated with collision and dead-lock avoidance schemes for wheeled mobile robot in indoor environments

Ya-Chun Chang
Graduate School of Tokai University, Hiratsuka, Japan, and

Yoshio Yamamoto
Department of Precision Engineering, Tokai University, Hiratsuka, Japan

Abstract

Purpose – This paper aims to present a hybrid path planning algorithm which is designed for use of autonomous vehicles in indoor environments. The approach mainly contributes the ability of generating a safe and smooth collision avoidance path for attaining a desired position in an unknown and obstructed environment.

Design/methodology/approach – The hybrid planner is based on potential field method and Voronoi diagram approach, and it is represented with the ability of concurrent map building and autonomous navigation.

Findings – The possibility of controlling the look-ahead distance allows the mobile robot to smartly control the velocity for creating a smooth trajectory autonomously. The dead-lock problem is solved by defining necessary sub-goals between targets on the constructed map.

Originality/value – The system controller (look-ahead control) with the potential field method allows the robot to generate a smooth and safe path for an expected position. Only essential exploration of unknown environment is performed since the approach constrains the mobile robot to explore a safe and sub-optimal route towards a destination.

Keywords Robotics, Navigation, Control systems, Linear motion

Paper type Research paper

1. Introduction

Path planning has been studied extensively in recent years because of its importance to mobile robot control. Most of path planning is executed by calculating a path for a mobile robot to follow, so that it can move from one pose to another without colliding with obstacles. Moreover, in order to allow a robot to perform autonomous tasks in a wide and unknown environment, the ability of building a reliable map of its workspace becomes extremely important. At present, map building and path planning are always being discussed in different areas because of vastness of both fields. During exploration stage the robot localizes its position while generating a map model of its environment via sensory data simultaneously. Based on a reliable and consistent map the robot calculates a path to an expected position as well as updating its position in conjunction with performing a desired motion precisely during a long-term navigation task (Figure 1(a)).

The current issue and full text archive of this journal is available at **www.emeraldinsight.com/0143-991X.htm**

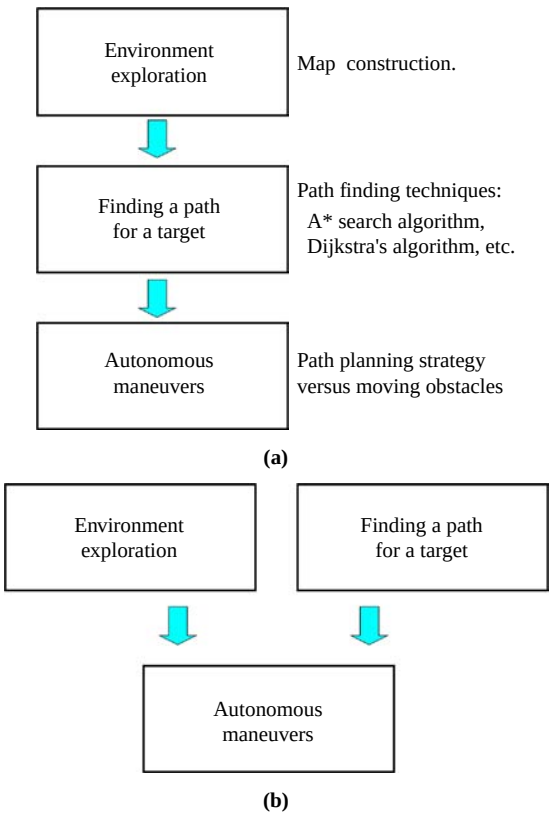
Industrial Robot: An International Journal
35/5 (2008) 421–434

[DOI 10.1108/01439910810893590]

In this paper, we propose a hybrid path planner model which achieves concurrent navigation and topological map building. The hybrid path planner is composed of potential field method (Khatib, 1986) (Volpe and Khosla, 1990) and Voronoi diagram approach. The notion of this approach is inspired by the fact that most of mapping and path search algorithms are not computationally viable over large-scale environment. In contrast, the proposed approach constrains the mobile robot to explore a safe and sub-optimal route to a destination and allows the robot to accomplish environment modeling and safe path finding parallelly when a position of where the robot should attain is explicitly (Figure 1(b)). As a result, computational burden of environment exploration is reduced to a safe path finding for reaching a target position and the approach benefits from saving time during environment exploration if a target position is explicitly defined.

The potential field-based path strategy is straightforward and guarantees a safe navigation under dynamic environment. However, it suffers from a certain limitation (Krogh, 1984) (Lau, 2003) in which the robot will be trapped in positions where attractive and repulsive forces negate each other. There are several conventional solutions for the dead-lock problem. (Connolly, 1992; Connolly and Grupen, 1993 utilized Laplace's heat equation to create a potential field without local minima. Koditschek and Rimon, 1990) developed a local minima free potential field method by designing potential functions that contain only one minimum which is

Figure 1 Notion of proposed hybrid path planner: (a) general procedure of autonomous navigation; (b) our approach



precisely at the goal position. Also there is a conventional recovery algorithm called wall following method (Borenstein and Koren, 1989) that can be used for solving the dead-lock problem. In our approach, we propose our potential field-based path planner within topological map building ability which enables the robot to perform an exploration towards a desired position and to build a reliable and compact map between targets (a start and goal), as well as helping the robot to avoid dead-lock positions existing in environment during a navigation task.

We also propose a recovery algorithm (Chang and Yamamoto, 2006) for dead-lock problem encountered while the robot is travelling along sub-goals. Three dead-lock avoidance schemes are taken into account in different consequences of positions of obstacles which surround a mobile robot while it is in a dead-lock position. In order to estimate the space between the obstacles and be applicable for modifying repulsive potential shapes, the shapes of the obstacles causing the robot halt are detected by adopting the Hough transformation (Hough, 1962).

The system controller (look-ahead control) used in this research is based on feedback linearization technique in which linearization is achieved between the control inputs and appropriate outputs. By controlling the output equation, the velocity of the mobile robot can be controlled to create a very smooth trajectory during obstacle avoidance. Within Look-ahead control framework, our path planner becomes more powerful and allows the mobile robot to easily maneuver in an environment where obstacles exist.

The paper is organized as follows. Section 2 introduces our system controller. In Section 3, we verify the control method in real experiments. Having covered the control system of the mobile robot, we go into the path planning of the robot in Section 4. Section 5 presents a recovery algorithm for dead-lock problem, and Section 6 concludes with final remarks.

2. System model and controller

2.1 Modeling of system

In this subsection we derive the motion equation of the mobile robot. Motion equation of the mobile robot which is two wheeled can be obtained from the approach proposed in (Yamamoto and Yun (1997). Suppose we choose $\mathbf{M}(q)$ as inertia matrix, $\mathbf{C}(q,\dot{q})$ as velocity vector, $\mathbf{G}(q)$ as gravity vector, $\mathbf{E}$, $\boldsymbol{\tau}$, $\mathbf{A}^{\mathrm{T}}(q)$ and λ as constant matrix, input torque vector, constraint matrix and Lagrange multipliers, respectively. Then we can write the motion equation as:

$$\mathbf{M}(q)\ddot{q} + \mathbf{C}(q,\dot{q}) + \mathbf{G}(q) = \mathbf{E}\tau - \mathbf{A}^{\mathrm{T}}(q)\boldsymbol{\lambda} \tag{1}$$

The system model of the mobile robot is shown in Figure 2 where b represents half the distance between two driving wheels, and r is the radius of the wheel.

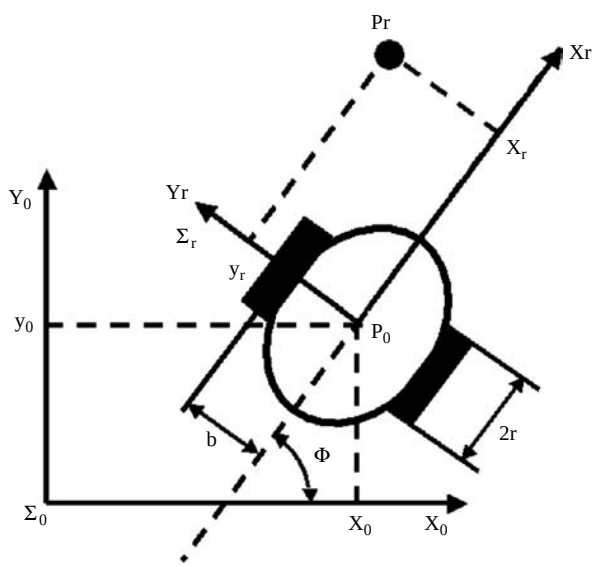
In this kind of mobile robot (two wheeled type), two kinematic constraints are imposed. First, no lateral slip is allowed between the wheels and the floor, and the second one is the no longitudinal slip constraint. According to these two constraints, the mobile robot is subject to two kinds of non-holonomic constraints which can be described by:

$$\dot{x}_0 \sin\phi - \dot{y}_0 \cos\phi = 0 \tag{2}$$

$$\dot{x}_0 \cos\phi + \dot{y}_0 \sin\phi = \frac{r}{2}(\dot{\theta}_r + \dot{\theta}_l) \tag{3}$$

where (x_0, y_0) is the coordinates of the center of the mobile robot, ϕ is the heading angle of the mobile robot measured from X-axis, θ_r and θ_l are the angular positions of the right wheel and the left wheel.

Figure 2 System model



Here, we choose $q = [x_0 \quad y_0 \quad \theta_r \quad \theta_l]^{\mathrm{T}}$ as the generalized coordinate vector. By following the standard approach for dimensional reduction, the motion equation (1) can be transformed as:

$$\mathbf{S}^{\mathrm{T}}\mathbf{M}\mathbf{S}\dot{\eta} + \mathbf{S}^{\mathrm{T}}\mathbf{M}\dot{S}\eta + \mathbf{S}^{\mathrm{T}}\mathbf{C} = \boldsymbol{\tau} \tag{4}$$

where a distribution matrix $\mathbf{S} = \{S_1, \ S_2\}$ satisfies $\mathbf{A}(q)\mathbf{S}(q) = 0$. Moreover, the velocity vector $\dot{q}$ also has to be in the same null space of $\mathbf{A}(q)$ while the vector $\boldsymbol{\eta} = [\eta_1 \ \eta_2]^{\mathrm{T}}$ represents an independent pseudo velocity vector which lives in the same null space.

Combining q with η to define $\mathbf{x}$ as the state vector of the mobile robot, from equation (4) we can write the differential of the state vector $\mathbf{x}$ as:

$$\dot{x} = \begin{bmatrix} \dot{q} \\ \dot{\eta} \end{bmatrix} = f(\mathbf{x}) + g(\mathbf{x})\boldsymbol{\tau} \tag{5}$$

where:

$$f(\mathbf{x}) = \begin{bmatrix} S\eta \\ -(\mathbf{S}^{\mathrm{T}}\mathbf{M}\mathbf{S})^{-1}(\mathbf{S}^{\mathrm{T}}\mathbf{M}\dot{S}\eta + \mathbf{S}^{\mathrm{T}}\mathbf{C}) \end{bmatrix}$$

and:

$$g(\mathbf{x}) = \begin{bmatrix} 0 \\ (\mathbf{S}^{\mathrm{T}}\mathbf{M}\mathbf{S})^{-1} \end{bmatrix}$$

2.2 Controller design

The control method using in this study is based on feedback linearization technique in which linearization is achieved between the control inputs and appropriate outputs. The control diagram of the mobile robot is shown in Figure 3.

We choose the output equation at the XY coordinates of the appropriate reference point P_r (refer to Figure 2). The output equation y is:

$$y = \begin{bmatrix} x_0 + x_r\cos\phi - y_r\sin\phi \\ y_0 + x_r\sin\phi + y_r\cos\phi \end{bmatrix} \tag{6}$$

To find the new linearization between the control input and output, we differentiate the output (6) with respect to time until a new input μ appears.

After the second time-differential of y, the new input μ appears while $\ddot{y}$ is written as:

$$\ddot{y} = \boldsymbol{\Phi}(x)\dot{\eta} + \dot{\boldsymbol{\Phi}}\eta \quad = \boldsymbol{\Phi}(x)\mu + \dot{\boldsymbol{\Phi}}\eta \tag{7}$$

where $\boldsymbol{\Phi}(x)$ is the decoupling matrix described as:

$$\Phi(x) = \begin{bmatrix} (r/2)\cos\phi - Lc\sin\phi & (r/2)\cos\phi + Lc\sin\phi \\ (r/2)\sin\phi - Lc\cos\phi & (r/2)\sin\phi - Lc\cos\phi \end{bmatrix} \tag{8}$$

where L is the distance between the reference point P_r and the center of the mobile robot, and $c = r/2b$.

Also the new input μ is given by:

$$\mu = \boldsymbol{\Phi}^{-1}(x)(u - \dot{\Phi}(x)\eta) \tag{9}$$

From equations (7) and (9), we obtain:

$$\ddot{y} = u, \quad u = \begin{bmatrix} u_1 & u_2 \end{bmatrix}^{\mathrm{T}} \tag{10}$$

Equation (10) is our new linearization between the control input and output. Moreover, for stabilizing the output y, the following equation is used:

$$u = \ddot{\hat{y}} + k_D(\dot{\hat{y}} - \dot{y}) + k_p(\hat{y} - y) \tag{11}$$

where $\ddot{\hat{y}}$, $\dot{\hat{y}}$ and $\hat{y}$ are the targets.

In addition, from (9) we know that the decoupling matrix has to be invertible, namely, $\boldsymbol{\Phi}$ has to be a non-singular matrix. Also from equation (8) we obtain:

$$\det(\boldsymbol{\Phi}) = -\frac{r^2}{2b}L.$$

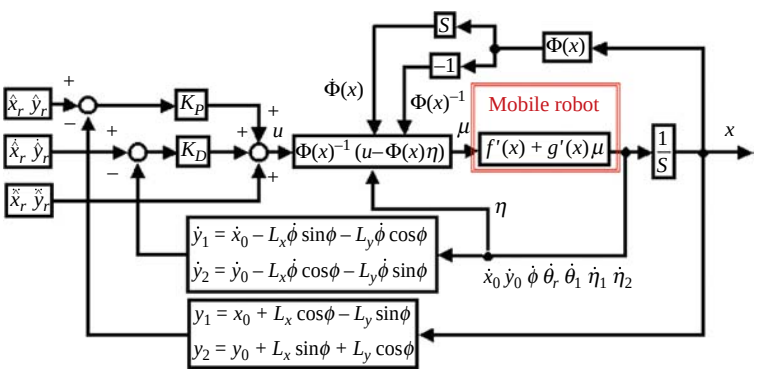
Therefore, $L \neq 0$ has to be satisfied because b and r always are positive.

By using this control system, the motion of the mobile robot is generated so as to compensate any deviation from the desired one, for example, in the sense of manipulability (Yamamoto and Yun, 1994). Moreover, because of the capability of controlling the distance between the center of robot and the reference point, the motion of the mobile robot can be more flexible and easy to control under dynamic environment where moving obstacles exist. In addition, for the sake of simplicity, we choose the reference point on the Xr-axis (Figure 2) in the following implementations.

3. Experimental setup

The mobile robot we use for experiments is an ActivMedia product, Pioneer 3-DX, which has two independent wheels

Figure 3 Control diagram



and eight sonars affixed under the front deck. In order to make our robot more autonomous, having the ability of gathering accurate information about environment becomes extremely important. Therefore, a range finding device, SICK laser, is equipped on our robot. Figure 4 shows Pioneer 3 with SICK laser loaded on the top.

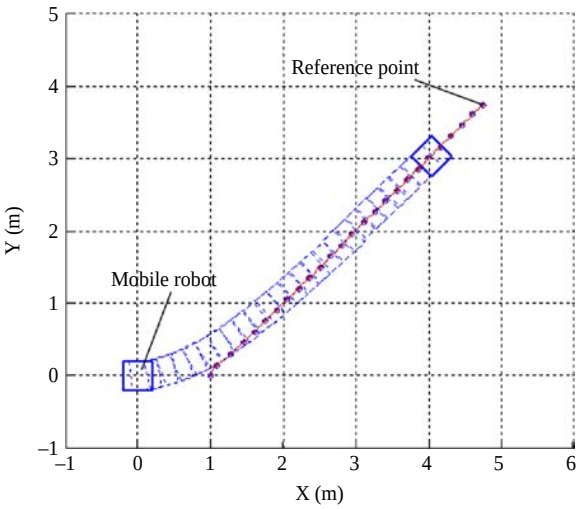
3.1 Implementation of look-ahead control system

In this subsection we verify our system controller in real robot experiments. There are two kinds of case studies conducted while the mobile robot is moving along a straight line 45 and 225° from a positive X-axis, respectively. In both studies, the mobile robot initially faces to a positive X-axis, and the length of L set in 1 $[m]$, also the velocity of the mobile robot is at 0.15 (m/s). Figure 5(a) and (b) show the experimental results of the trajectory of mobile robot which moves along a straight line 45 and 225° from a positive X-axis in processing time 25 and 30 s, respectively.

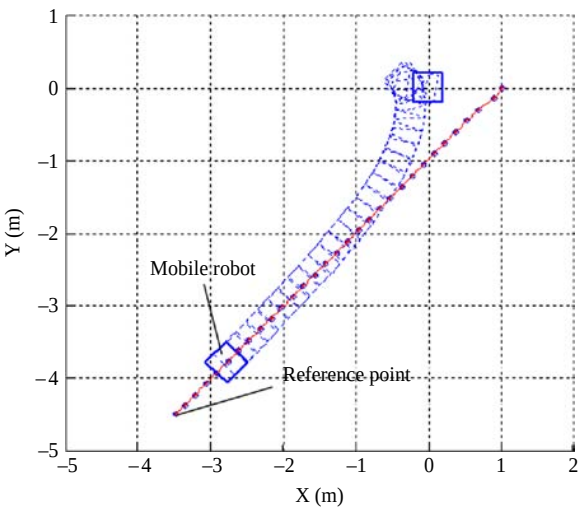
Figure 6 plots the heading angle of mobile robot with respect to time in two case studies. Figure 7 represents the occurring error during the mobile robot chases the reference point in both studies. From both studies, it is seen that by using look-ahead control system, the mobile robot will always keep chasing the reference point, which agrees with the three necessary conditions for asymptotic convergence discussed in Brockett (1983). Therefore, as to path planning, instead of scheming the path of the mobile robot directly we plan on calculating the path to reach the target position for the reference. As a result, the mobile robot will attain the target position as we expect by following the motion of the reference point.

Figure 4 Pioneer 3 with SICK laser

Figure 5 Experimental result: trajectory of mobile robot: (a) 45°; (b) 225°



(a)



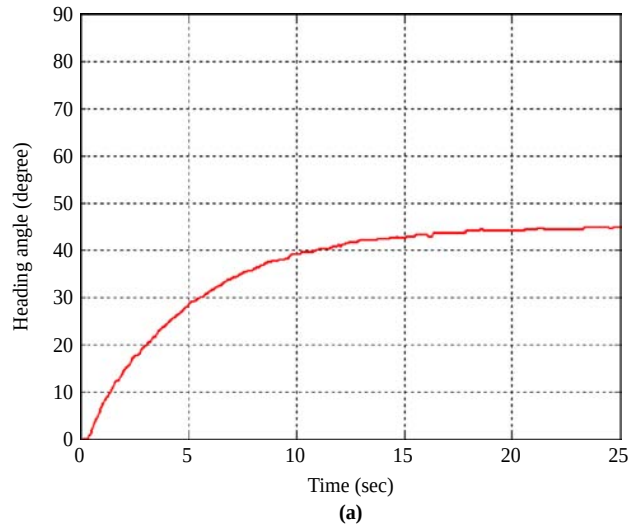
(b)

4. Concurrent navigation and map building system

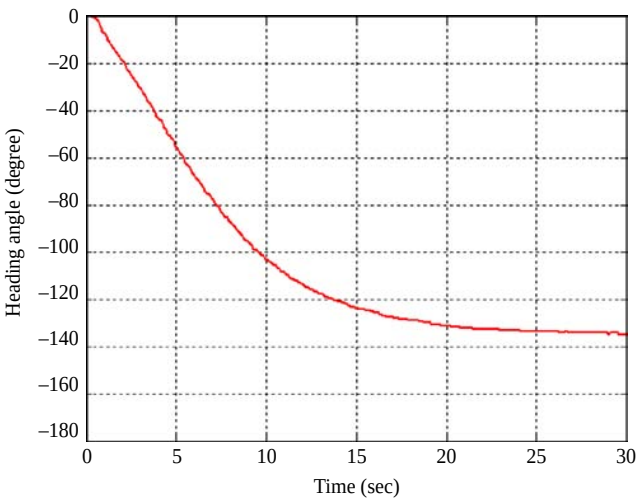
As mentioned previously, we propose our path planner with both navigation and map building abilities. Based on our approach, the a priori unknown environment does not need to be neither bounded nor static in terms of building a map model of robot's workspace since the robot is constrained to explore a route towards a destination and is capable of avoiding collisions with moving obstacles during exploration. Based on the constructed map, the robot is able to seamlessly start autonomous maneuvers between any targets in the explored area as well as starting a new exploration to an unvisited region.

The hybrid path planner is comprised of two modules. Section 4.1 introduces the obstacle avoidance module of the hybrid planner. Section 4.2 explains the map building module of the system and shows a demonstration result in real environment.

Figure 6 Experimental result: heading angle of mobile robot: (a) 45°; (b) 225°

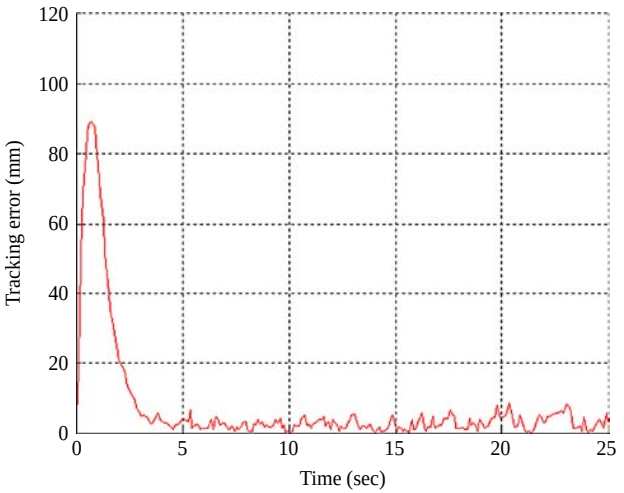


(a)

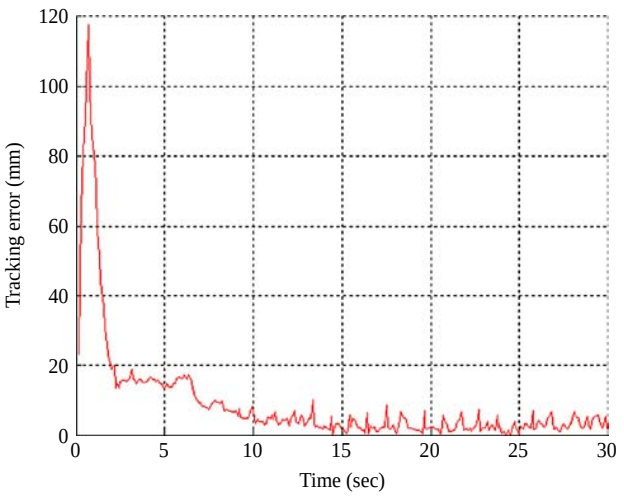


(b)

Figure 7 Experimental result: tracking error: (a) 45°; (b) 225°



(a)



(b)

4.1 Collision avoidance algorithm

Since the control system we use will cause the mobile robot keep chasing the reference point, instead of scheming the obstacle avoidance of the mobile robot directly, we focus on the path planning of the reference point. In this research, we adopt the potential field method for obstacle avoidance during navigation.

4.1.1 Potential field method

Potential field method is widely used in decades for autonomous mobile robot path planning because of its simplicity. An attractive potential which drives the mobile robot to its destination can be described by:

$$U_{\text{att}}(x) = A\sqrt{(x_r - x_g)^2 + (y_r - y_g)^2} \tag{12}$$

where A is a positive constant, (x_r, y_r) is the coordinates of the reference point, and (x_g, y_g) is the coordinates of the destination.

Also a repulsive potential well which prevents the mobile robot from collisions can be written as:

$$U_{\text{rep}}(K) = B\frac{e^{-\alpha K}}{K} \tag{13}$$

where B and α are positive constants. K is the variable which acts as a pseudo-distance from the obstacle, becoming zero at the surface and increasing with successive contours away from the surface.

From equations (12) and (13), we obtain the potential energy U which drives the reference point of the mobile robot to the destination, described by:

$$U = U_{\text{att}}(x) + U_{\text{rep}}(K) \tag{14}$$

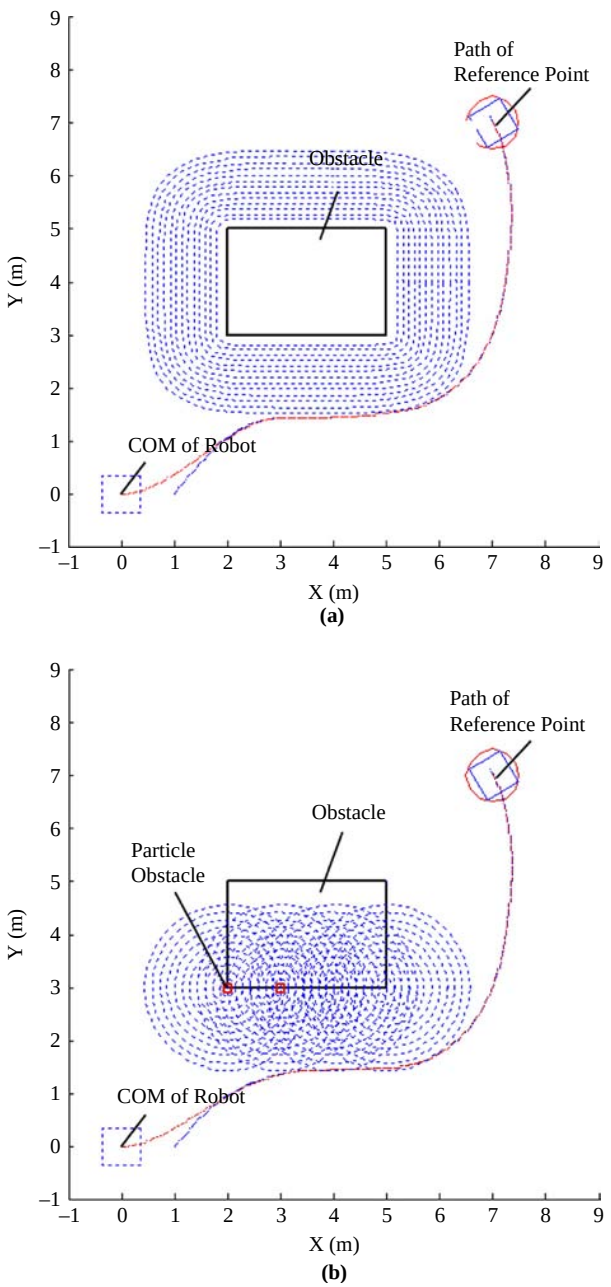
also the momentary force F which the reference point receives can be given by:

$$F = -\nabla U = \left[-\frac{\partial U}{\partial x} \quad -\frac{\partial U}{\partial y}\right]^{\text{T}} \tag{15}$$

In order to perform our avoidance algorithm in real-time, unlike the conventional potential field method we do not take into account whole existing obstacles in environment.

We use the closest sensory measurement from the obstacle surface, and define a smallish particle obstacle at the closest location which generates a small repulsive potential as well. As a result, the generated repulsive potential will vary with time even though there are only static obstacles existing in environment, and the momentary potential fields from smallish particle obstacles will be gradually generated in discrete time which eventually sum up to create almost as the same shape as repulsive potential generated from the original obstacle. Figure 8 shows the comparison between the result of conventional potential field method and the result based on our new approach.

Figure 8 Simulation results for comparison: (a) conventional approach; (b) our approach

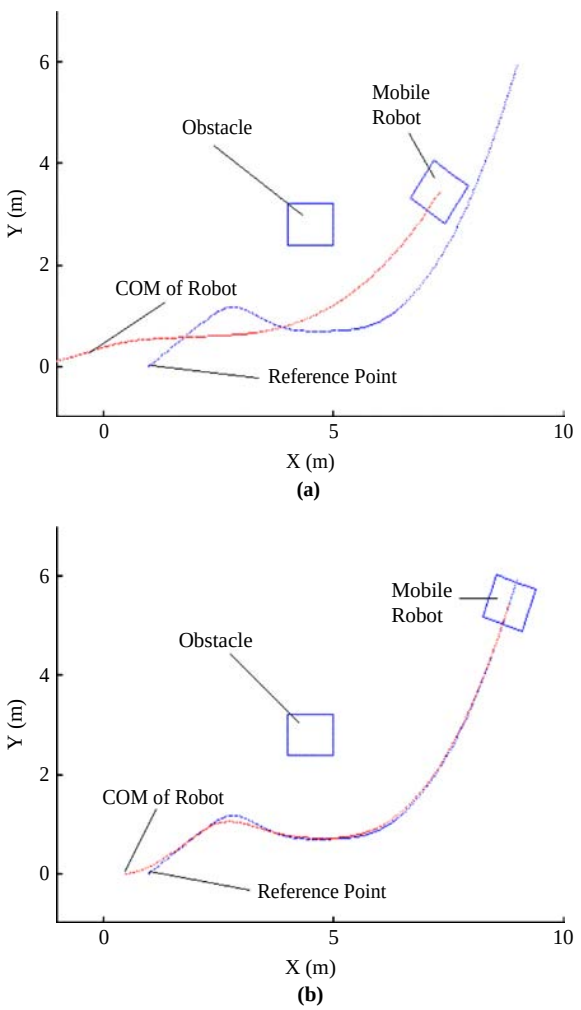


4.1.2 Dynamic control of output equation

As for creating a smooth trajectory in conjunction with obstacle avoidance, we vary the output equation of the control system to control the velocity of the mobile robot. Figure 9(a) and (b) represent the obstacle avoidance path of the mobile robot moving with a large and small look-ahead distance (L), respectively. From both results it can be seen that although the path of the reference point is the same, the motion of the mobile robot can be totally different, and the mobile robot may possibly collide with the obstacle if L is too large (Figure 9(a)). On the other hand, if we always keep L small, the mobile robot takes almost the same path as the reference point at the expense of lack of smoothness, and in general it takes more time for the mobile robot to reach the destination (Figure 9(b)). The term "dynamic control" should not be confused with "dynamic compensation" or "dynamic extension" used in the context of differential geometric control (Isidori, 1995; Nijmeijer and van der Schaft, 1990).

The natural choice is to keep L large when the mobile robot is away from the obstacle, and make it automatically become smaller while the mobile robot is being closer to the obstacle. Moreover, referring to Figure 3, we know that L cannot be 0 because the decoupling matrix $\mathbf{\Phi}(x)$ has to be invertible.

Figure 9 Simulation with different look-ahead distances: (a) large L; (b) small L



Since L cannot become 0, we realize that the mobile robot actually cannot attain the exact target position as we expect. Therefore, we propose:

$$L = \left(L_m + C\frac{\partial U_{\text{rep}}}{\partial K}\right)e^{-1/K_g} + \alpha \qquad (16)$$

where L_m and C are positive adjustable constants, and K_g is the distance between the center of the mobile robot and the target position. L changes along with $\partial U_{\text{rep}}/\partial K$ which becomes smaller as the mobile robot comes closer to the obstacle. The exponential function reduces L_m to become as close as 0 while the reference point attains the goal point, and also α keeps L would not become 0 and allows the mobile robot to attain the target position as closely as possible.

4.1.3 Under static and dynamic environment

Two experiments are conducted to verify the proposed path planning method under static and dynamic environment. In both experiments, the mobile robot initially faces to a positive X-axis, the length of the initial look-ahead distance (L) set in 1 [m], and the mobile robot moves at a constant velocity of 0.4 (m/s). Tables I and II show the condition settings of two experiments, respectively.

Figure 10 represents the obstacle avoidance path of the mobile robot while its look-ahead distance varies along with equation (16). Figure 11 shows the look-ahead distance with respect to time. L becomes short when robot is close to an obstacle, and increases gradually to the initial setting when robot is away from obstacle, and eventually varies almost to a constant when the robot is trying to attain the goal position.

Figure 12(a) to (e) thoroughly plot the obstacle avoidance path of the mobile robot in discrete time while it is maneuvered in a dynamic environment where three rectangular shapes of obstacles are moving in different directions. Figure 13 shows the change of the look-ahead distance with respect to time, as well as controlling the velocity of the mobile robot for creating a smooth obstacle avoidance path during a navigation task.

From these results, it is seen that based on our collision avoidance algorithm by predefining start and goal positions the mobile robot can find a safe and smooth path for attaining the target position autonomously whether in a static or dynamic environment.

Table I Experimental conditions of obstacle avoidance (static case)

Initial robot position	Origin
Initial robot direction	0°
Initial reference point position	$x_r = 1.0[m]$, $y_r = 0.0[m]$
Goal position	$\text{Goal}_x = 7.0[m]$, $\text{Goal}_y = 7.0[m]$
Obstacle position	$\text{Obs1}_x = 3.0[m]$, $\text{Obs1}_y = 5.0[m]$ $\text{Obs2}_x = 4.0[m]$, $\text{Obs2}_y = 2.0[m]$ $\text{Obs3}_x = 6.0[m]$, $\text{Obs3}_y = 4.5[m]$
Obstacle shape	Width = 1.0[m], Height = 0.8[m]
Processing time	30 s

Table II Experimental conditions of obstacle avoidance (dynamic case)

Initial robot position	Origin
Initial robot direction	0°
Initial reference point position	$x_r = 1.0[m]$, $y_r = 0.0[m]$
Goal position	$\text{Goal}_x = 7.0[m]$, $\text{Goal}_y = 7.0[m]$
Initial obstacle position	$\text{Obs1}_x = 0.0[m]$, $\text{Obs1}_y = 2.0[m]$ moving direction and velocity moving in a positive X-axis direction at 0.2 (m/s) $\text{Obs2}_x = 8.5[m]$, $\text{Obs2}_y = 5.5[m]$ moving in a negative X-axis direction at 0.2 (m/s) $\text{Obs3}_x = 6.0[m]$, $\text{Obs3}_y = 6.0[m]$ moving in a negative Y-axis direction at 0.2 (m/s)
Obstacle shape	Width = 1.0[m], Height = 0.8[m]
Processing time	30 s

Figure 10 Collision avoidance (static case)

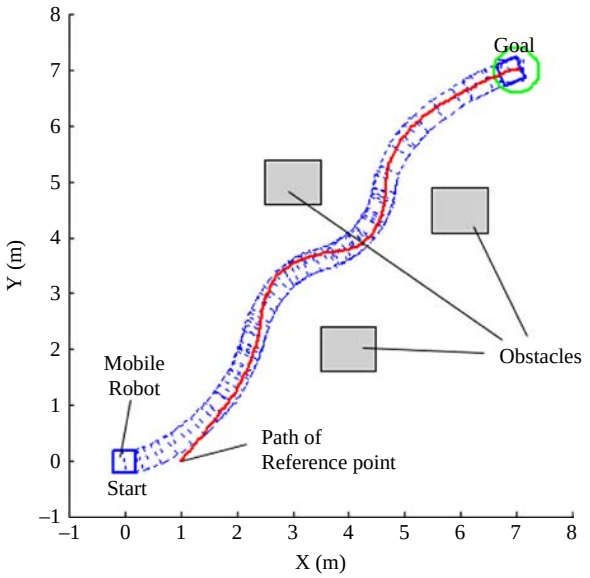


Figure 11 Look-ahead distance (static case)

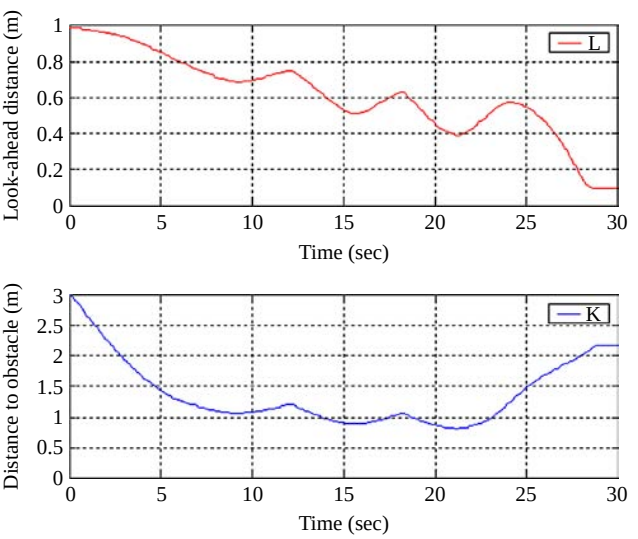
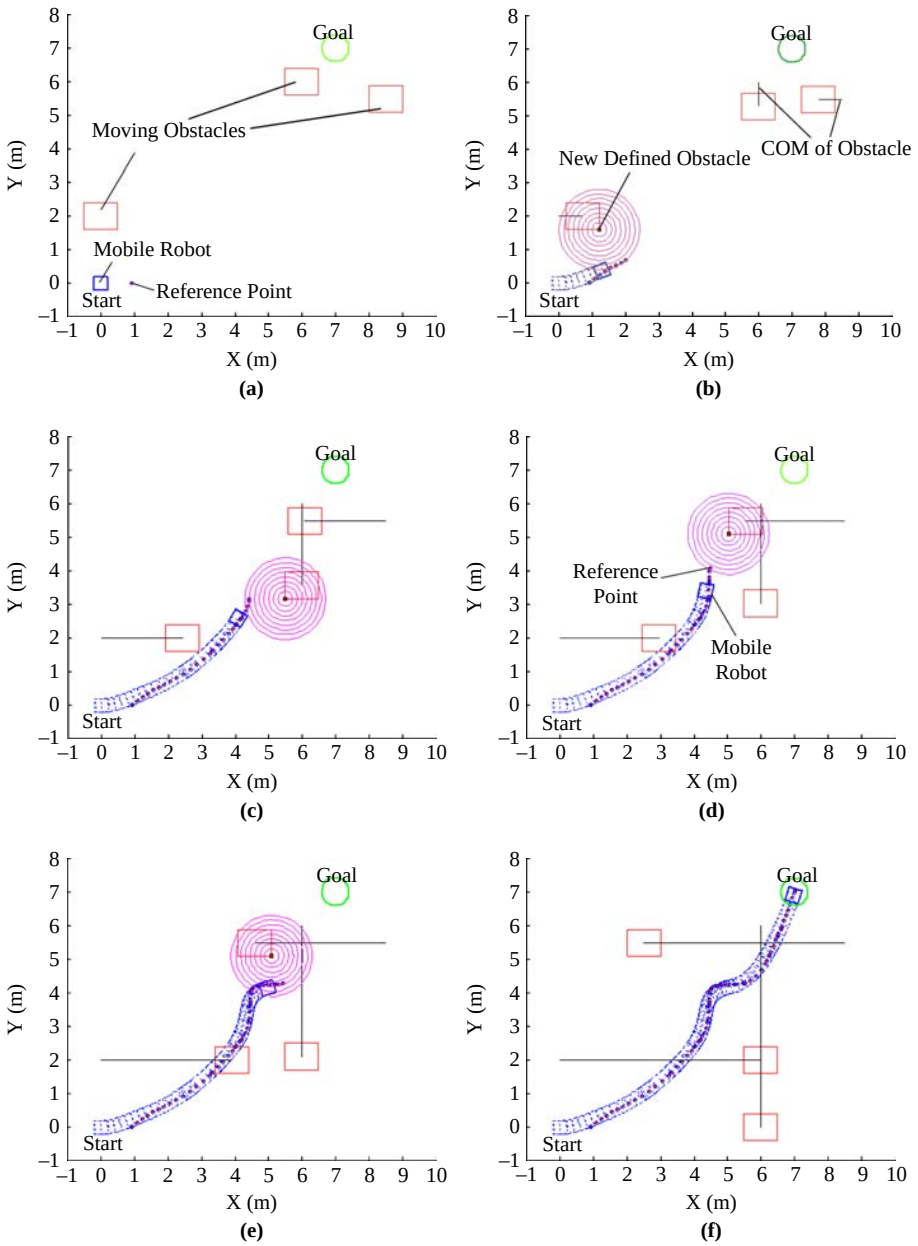


Figure 12 Collision avoidance (dynamic case): (a) 0.0 s; (b) 3.6 s; (c) 14.8 s; (d) 12.0 s; (e) 19.5 s; (f) 20.0 s

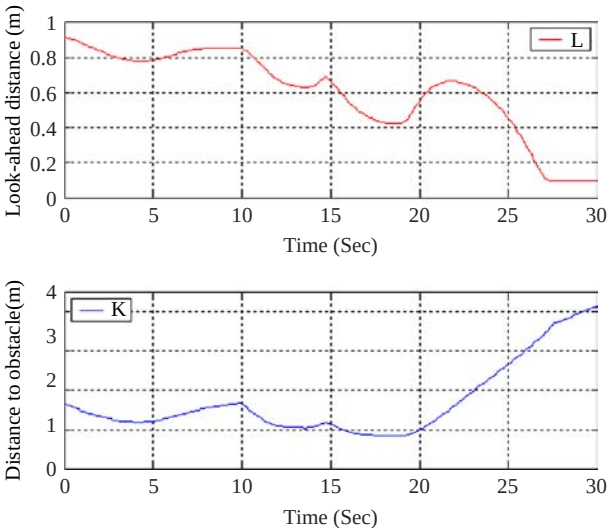


4.2 Simultaneous map building

In terms of modeling a configuration space of robot's environment during navigation, we adopt Voronoi diagram approach to enable the robot to generate a map of its workspace while it is moving towards a destination. Topological structure of environment representation can benefit from its compact model that achieves fast route planning and reduces the amount of sensory data. Usage of such technique for environment modeling has been demonstrated in previous approaches (Thrun, 1998; Mahkovic and Slivnik, 2000; Choset and Nagatani, 2001). However, their approaches have three common limitations:

1 First, size of the exploration space has to be set up as either a bounded area or a close environment in order to terminate the exploration when all the edges have been explored.
2 Secondly, their approaches do not support the obstacle avoidance ability. It limits the robot to trace the constructed Voronoi edges from one to another in a certain range, in order to allow the robot to follow a safe path while it is tracing towards Voronoi edges. As a result, it slows down the exploration process, as well as limiting the robot to perform exploration tasks in only static environments.
3 After an exploration process is finished, a graph search algorithm is required to calculate a path for performing autonomous maneuvers between targets defined in the existing exploration. However, most of graph search algorithms are not computationally viable over large-scale environment.

Figure 13 Look-ahead distance (dynamic case)



To cope with these limitations, the proposed hybrid path planner features:

- Efficient environment exploration: Instead of performing an exploration in a bounded area, the exploration is constrained to finding a safe path for attaining a target position. Moreover, the mobile robot does not have to trace the generated Voronoi edges one by one during exploration phase since this approach allows the mobile robot to simultaneously avoid collisions with obstacles and generate a smooth obstacle avoidance path while it is moving towards an edge position.
- Map building in conjunction with finding a sub-optimal path for attaining a desired position: It is unnecessary to search a safe path on the graph for attaining a target position. After an exploration towards a target, the robot can perform autonomous maneuvers immediately by setting up sub-goals on the constructed map.

4.2.1 Approximate Voronoi diagram

In order to reduce the computational burden of the Voronoi edge calculation, despite of the richness of the information obtained by a laser range finder, we approximate the sensory data to allow the robot to generate an approximate Voronoi structure such as a representation of free space of its environment. Before getting into the details of determining edges for tracing towards a target during an exploration, we explain the algorithm of approximate Voronoi diagram computation in this approach. Figure 14 shows the pseudo-code description for generating approximate Voronoi diagram from the sensory information.

First, the robot gathers information of its vicinity and creates a finite set P of points which approximates the sensory data from each laser scan. Clustering of each point is done by perceiving the change of distance between two successive laser readings. Based on each local scan, the robot accumulates the approximate sensory data for incrementally building an approximate Voronoi structure of its workspace while moving towards a target.

To allow the robot to create only useful Voronoi edges for exploration, the radius of Delaunay circle is constrained as:

$$\text{diameter of robot} < R < \text{a certain limitaion} \quad (17)$$

This constraint guarantees not only that safe edges allow mobile robot to pass through a narrow passage, but also that none of unnecessary edge caused by a large concave environment is created (Figure 15). Figure 15(a) represents approximate Voronoi diagram which is defined by equation (17). Figure 15(b) shows the motion of mobile robot which traces towards explored Voronoi edges as:

- access an unvisited Voronoi edge which is closest to the goal in the sense of exploring towards goal position;
- during an exploration if no Voronoi edge can be found, the mobile robot automatically switches to the motion of attaining the goal position;
- restart to trace an edge if any new and unvisited Voronoi edge is perceived; and
- the mobile robot avoids colliding with obstacle simultaneously while accessing an edge position.

4.2.2 Tracing edge determination

Figure 16 shows the notion of tracing Voronoi edges incrementally towards a target. By selecting an edge point which is unvisited and is closest to a predefined goal as an edge for tracing, the mobile robot is able to accomplish an exploration towards the target. However, there is a possibility for the robot to fall into a dead-lock while the robot is accessing an edge which is closest to goal and is created by an occlusion (Figure 17).

In order to avoid being trapped in a dead-lock position while the mobile robot is accessing an uncertain Voronoi edge, we allow the robot to access a three-equidistant point which is recorded as a node with departing edges if a node is perceived. In addition, in this approach the node point is created by points belonging to large groups in which the amount of each group is large enough and each group contains over a certain number of members. It allows the mobile robot to perform an exploration task in environment where moving obstacles exist, since a node created by a moving obstacle is unnecessary to be recorded and it may cause unexpected problem if the robot accesses a node generated by a moving obstacle despite of ignoring the motion of the moving obstacle. According to the definition of the node point, it is seen that none of node point is generated by an environment shown in Figure 15.

Once the mobile robot attains a node position, the mobile robot then selects one unexplored edge of the defined node which is closest to the goal and continues tracing. During an exploration if the mobile robot reaches a dead-end, the robot returns back to the closest visited node and restarts to trace another unexplored Voronoi edge which is closest to the goal, as well as starting to explore a new route for attaining the target position.

4.2.3 Sub-goal setting

The proposed hybrid path planner is demonstrated in real environment. The mobile robot initially faces to a positive X-axis, and starts at the origin (0, 0). The position of the goal is predefined as:

$$\text{Goal}(15.0[m], \quad -37.5[m])$$

Figure 18 shows the real experimental result in which (a) and (b) represent the motion of the mobile robot while it is tracing Voronoi edges towards a target and the accessed Voronoi edges of approximate Voronoi diagram, respectively. It is seen that the mobile robot autonomously generates a smooth and safe

Figure 14 Computation of approximate Voronoi diagram

```
Compute Approximate Voronoi Diagram (Input: sensory data)
01. for each local scan
02.   approximate sensory data and save as P(x,y, scanning degree)
03.   Clusterrize (P)
04.   write new data on data list and save data list as DataList.
05. if first scan from laser range finder
06.   divide DataList into different groups and save as group list (GroupList).
07. else
08.   divide DataList into different groups and save as new group list (NewGroupList).
09. Compare groups of GroupList to groups of NewGroupList
10.   if distance between two group are small enough
11.     add new data to group of GroupList.
12.   else
13.     add group of NewGroupList as new group on GroupList.
14. Triangulate (GroupList)
15. calculate center of circumcircle for each triangle
16. if radius of circumcircle is in a certain range (here we set as 1000[mm] < radius < 3000[mm])
17.   Generate AVD (GroupList)
```

Figure 15 Approximate Voronoi diagram defined by constraints: (a) approximate Voronoi diagram; (b) motion of mobile robot

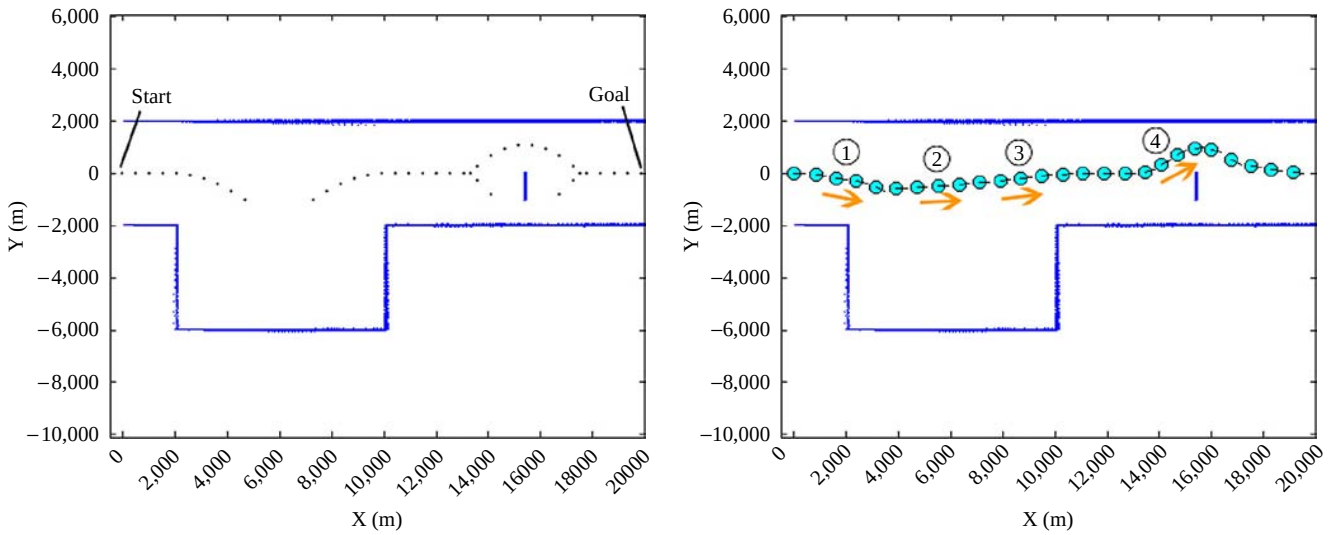
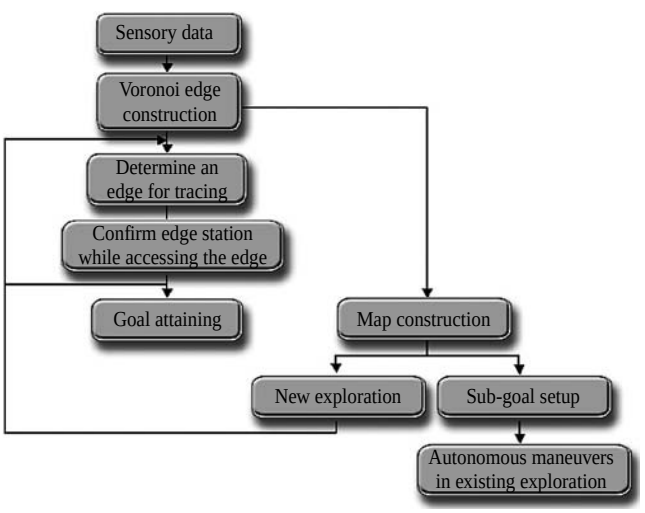


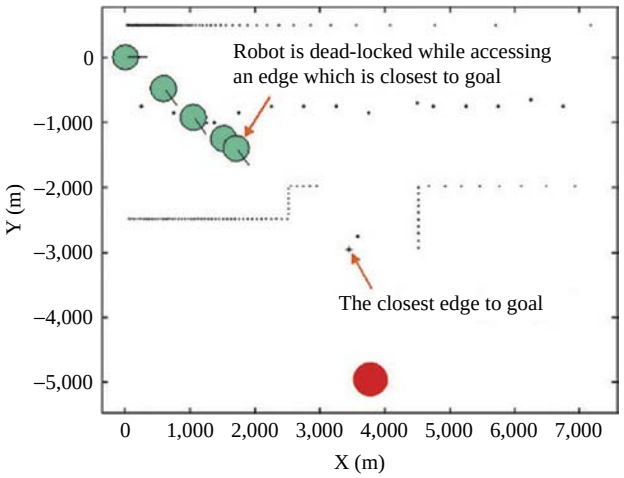
Figure 16 Procedure of autonomous navigation



trajectory by receiving attractive and repulsive forces while it is travelling along the accessed edge positions, and simultaneously avoids possible dead-locks caused by concave spaces existing in environment by exploring and tracing Voronoi edges.

Figure 19 shows the constructed map after the robot explores a route for attaining the goal position. From this result, it can be seen that instead of an exploration of whole free space of the bounded environment, only a safe route exploration is performed when a desired goal position is given. Moreover, in order to reduce the computational burden, the mobile robot stops calculating Voronoi edges and performs autonomous maneuvers in a manner of fully depending on its obstacle collision avoidance ability after an exploration process is accomplished. As a result, after constructing and modeling the environment of robot's workspace, sub-goals are set up at positions of visited node points on the map for immediate applications such as travelling between targets (start and goal positions) in the existing exploration.

Figure 17 Robot is dead-locked while tracing an edge created by an occlusion



5. Recovery algorithm for dead-lock

Through previous section, we explain the notion of our hybrid path planner. We propose a potential field-based avoidance algorithm within map building ability which allows our robot to be able to trace explored Voronoi edges for attaining a goal position simultaneously while avoiding local minimum problem caused by concavities in environment. However, there is still a possibility for the robot to be trapped in a dead-lock position because of the presence of changes to the environment while it travels along setup sub-goals between targets. Here, we propose a recovery algorithm for solving the dead-lock problem. Figure 20(a) and (b) shows the iso-potential contours of two repulsive potential fields in both *XY* and 3D where the deadlock problem occurred. From Figure 20, we understand that the dead-lock problem is occurred when multiple potential contours are overlapping, and sum up as one big potential field with concavity that causes the mobile robot facing uphill from the repulsive potential well when an undesirable minimum is created.

5.1 Obstacle detection

In order to conduct our dead-lock avoidance technique for solving such type of deadlock problem, we need to acquire the actual shapes of obstacles that cause the robot to stop. Thus, there is a preliminary requirement that the obstacle recognition ability for dead-lock avoidance has to be taken into account. In this paper, we adopt the Hough transformation for it. For a simple example, suppose there is a rectangular shape of obstacle in environment. The detection of this rectangular obstacle can be done by the line detection. Because it requires that the desired feature be specified in parametric form, a measurement of the obstacle with coordinates x and y can be written as:

$$\rho = x\cos\theta + y\sin\theta \tag{18}$$

where ρ and θ are variables used in parameter space.

After mapping all measurements into parameter space, the shape of the obstacle can be extracted by finding local maxima in the parameter space (i.e. local peaks). Figure 21 shows the result of a real obstacle detection based on the LRF measurements by using this procedure.

5.2 Dead-lock solution

We propose a dead-lock recovery algorithm which is stated as the following procedure:

- Set a criterion, that is, if the mobile robot moves under a limit of speed (almost to zero) except in a goal position, the robot will declare itself to be in a dead-lock position.
- Define shapes and positions of obstacles that surround the robot in a certain area by adopting the obstacle detection

Figure 18 Experimental demonstration of proposed path planner: (a) motion of mobile robot; (b) accessed Voronoi edges

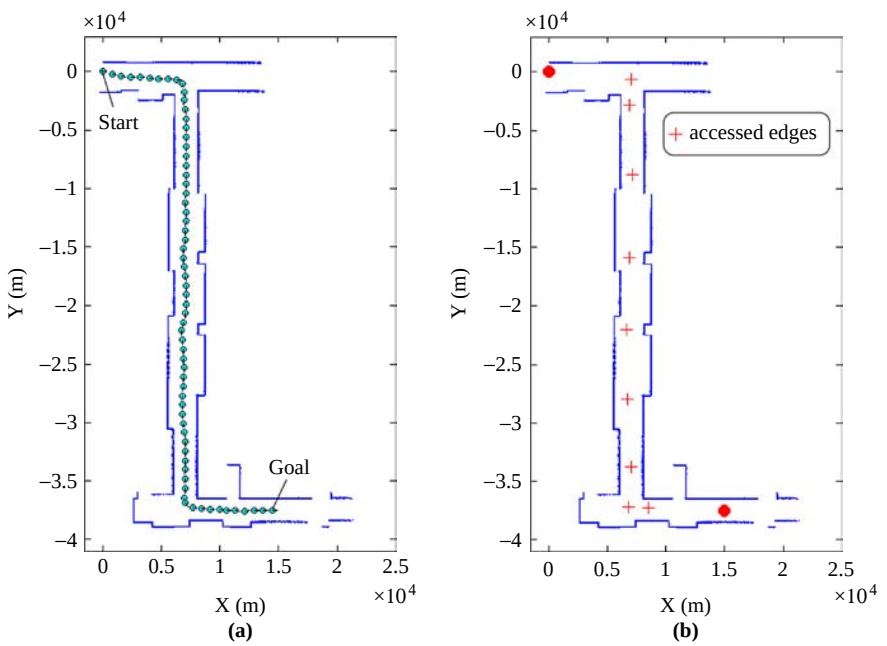
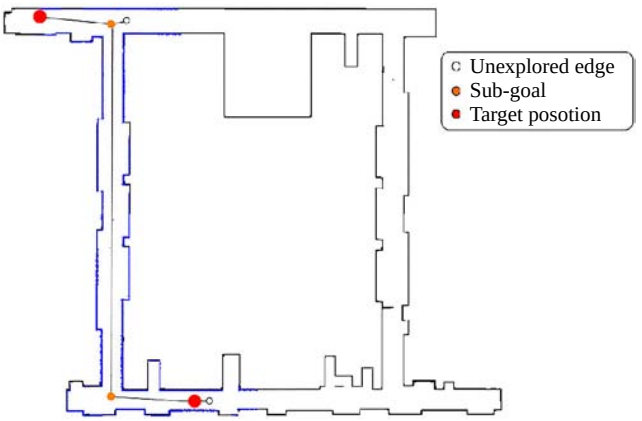


Figure 19 The topological map constructed by the proposed path planner



technique mentioned in previous subsection. Figure 22(a) shows a circular area which is drawn from the center of the robot where r is the radius of the circular area.

- Pick up two closest obstacles and define two reference lines to divide the area into three regions. Line 1 is a reference line that is across the centers of two obstacles. Line 2 is perpendicular to Line 1 and goes straight across the center of the robot. Areas 1 and 2 indicate the areas that are on the right side and the left side of Line 2, respectively. Area 3 represents the area that is behind the two closest obstacles.
- Based on obstacle information extracted from laser range finder measurements, the robot itself can estimate the space between the obstacles, and choose a suitable avoidance scheme to proceed by going through the decision-making process shown in Figure 22(b).

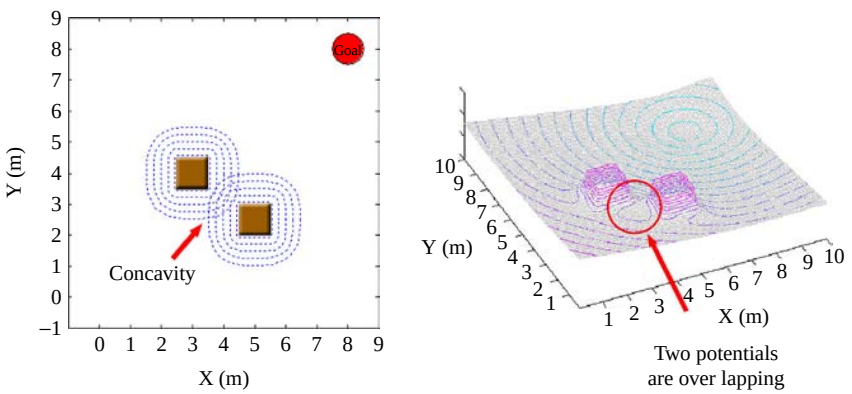
Three available dead-lock avoidance schemes that are taken into account as decisions:

A. Reduce the repulsive potential energy to create smaller potential contours for the robot to pass through (Figure 23(a)).

B-1. Combine the two obstacles into one pseudo-obstacle to make the mobile robot move to the downhill side of the obstacle and take a detour to escape from a dead-lock position. As a result shown in Figure 23(b), to make the mobile robot turn right and detour around a new potential field yielded from the pseudo-obstacle.

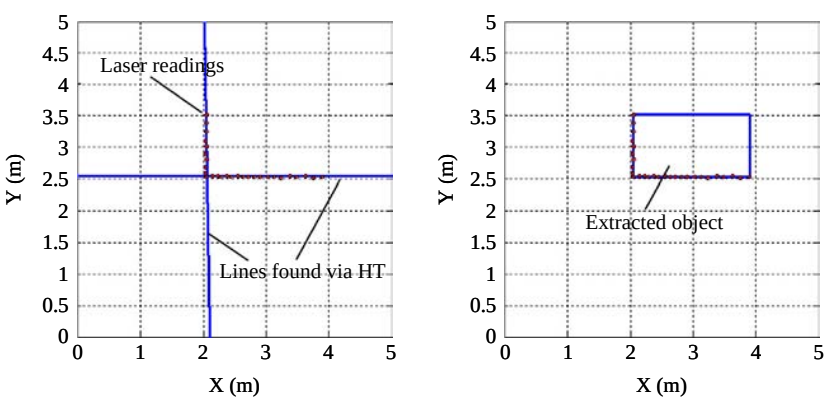
B-2. Tilt the merged obstacle for adjustment to let the robot to make a left turn and detour around a new potential field contour.

Figure 20 Potential fields causing a dead-lock: (a) *XY*-plane; (b) elevated view



(a) **(b)**

Figure 21 Obstacle detection example: (a) line detection; (b) extracted obstacle



(a) **(b)**

Figure 22 Making decisions for dead-lock avoidance: (a) a dead-lock position; (b) decision-making process

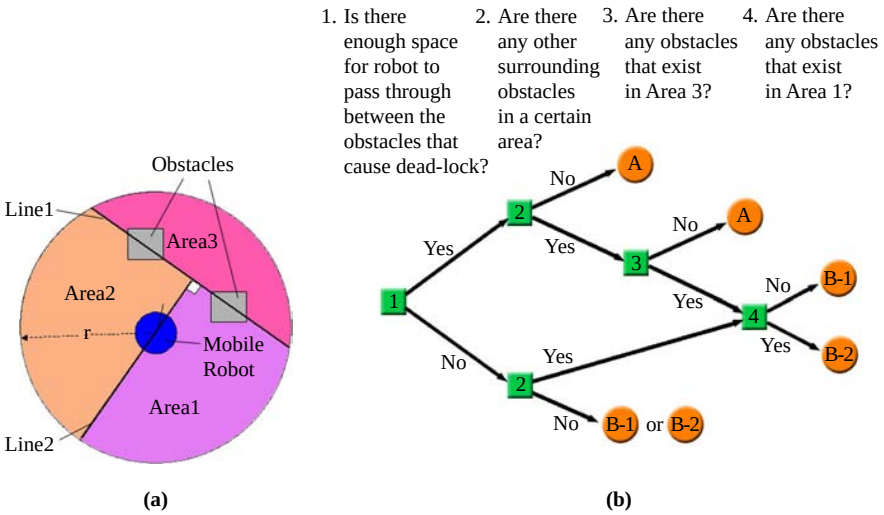
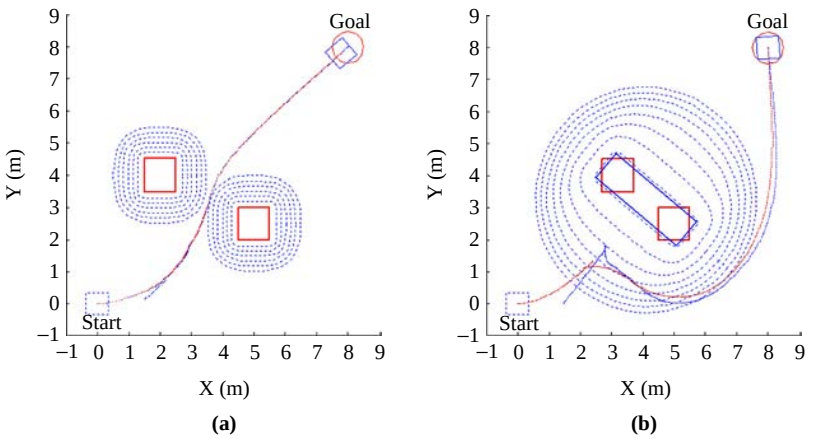


Figure 23 Dead-lock solutions: (a) reduce potential fields; (b) merge of potential fields



5.3 Real experimental results

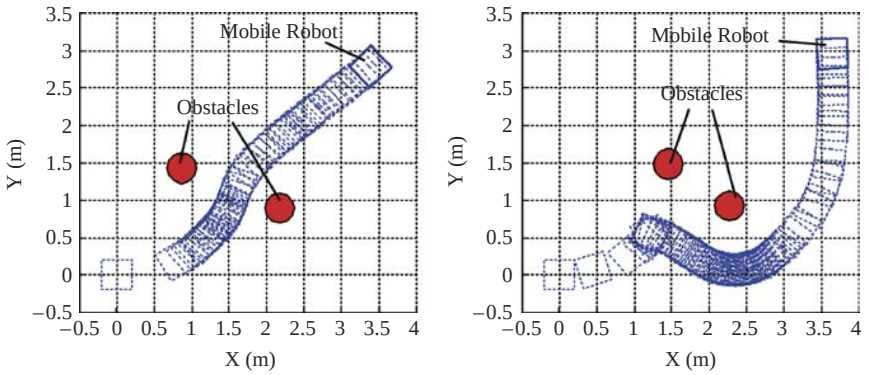
The proposed dead-lock avoidance technique is implemented on the real robot in an environment where two circular obstacles are located as to generate artificial potentials which stop the mobile robot at a dead-lock position. Figure 24 shows the results of applying our technique in different dead-lock conditions (different distances between obstacles causing dead-lock problem). From both results, we can verify that our dead-lock avoidance technique works successfully, and it gives the robot ability of escaping from a dead-lock position and choosing a suitable path for attaining the goal autonomously.

6. Conclusion

A hybrid path planner comprised of ability of concurrent navigation and map building is proposed. The collision avoidance algorithm presented has demonstrated its flexibility in both dynamically controlling the output equation and changing the shape of the repulsive potential field. By using Look-ahead control as the system controller, the possibility of controlling the look-ahead distance (L) allows the mobile robot to define the suboptimal path and smartly control the velocity for creating a smooth trajectory autonomously. The dead-lock problem is solved by defining necessary sub-goals between targets on the constructed map and a recovery algorithm is proposed for dead-lock avoidance while the robot travels between sub-goals.

Instead of giving a limitation such as setting up a bounded area for autonomous exploration during a navigation mission, we combine our path planner with map building ability to enable the mobile robot to explore a safe path towards a target as reducing the exploration as finding a safe route for attaining the target. As a result, only an essential exploration of unknown environment is performed and it allows the mobile robot to benefit from time saving during exploration for autonomous navigation. Based on collision avoidance algorithm the constructed map information is immediately applicable to autonomous maneuvers for helping the mobile robot to travel between predefined start and goal positions as well as travelling between any targets on the existing map. It should be mentioned that a new Voronoi structure will be naturally

Figure 24 Dead-lock avoidance on real robot: (a) able to pass through; (b) unable to pass through



merged to existing one and active area for further expansion. Thus, the map information can be augmented if the robot starts a new exploration to an unvisited area.

References

Borenstein, J. and Koren, Y. (1989), "Real-time obstacle avoidance for fast mobile robots", *IEEE Transactions on Systems, Man and Cybernetics*, Vol. 19 No. 5, pp. 1179-87.

Brockett, R.W. (1983), "Asymptotic stability and feedback stabilization", *Differential Geometric Control Theory*, Vol. 7 No. 11, pp. 1163-73.

Chang, Y-C. and Yamamoto, Y. (2006), "Dynamic decision making of mobile robot under obstructed environment", *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems, Beijing, China*.

Choset, H. and Nagatani, K. (2001), "Topological simultaneous localization and mapping (slam): toward exact localization without explicit localization", *IEEE Transactions on Robotics and Automation*, Vol. 17 No. 2, pp. 125-37.

Connolly, C.I. (1992), "Applications of harmonic functions to robotics", *Proceedings of IEEE International Symposium on Intelligent Control*, pp. 498-502.

Connolly, C.I. and Grupen, R.A. (1993), "On the application of harmonic functions to robotics", *Journal of Robotic Systems*, Vol. 10 No. 7, pp. 931-46.

Hough, P.V.C. (1962), "Method and means for recognizing complex patterns", US Patent 3,069,654.

Isidori, A. (1995), *Nonlinear Control Systems*, 3rd ed., Springer, New York, NY.

Khatib, O. (1986), "Real-time obstacle avoidance for manipulators and mobile robots", *International Journal of Robotics Research*, Vol. 5 No. 1, pp. 90-8.

Koditschek, D.E. and Rimon, E. (1990), "Robot navigation functions on manifolds with boundary", *Advances in Applied Mathematics*, Vol. 11, pp. 412-42.

Krogh, B.H. (1984), "A generalized potential field approach to obstacle avoidance control", *Proceedings of SME Conference on Robotics Research, Bethlehem*.

Lau, H. (2003), "Behavioural approach for multi-robot exploration", *Proceedings of Australasian Conference on Robotics and Automation, Brisbane, Australia*.

Mahkovic, R. and Slivnik, T. (2000), "Constructing the generalized local Voronoi diagram from laser range scanner data", *IEEE Transactions on Systems, Man and Cybernetics*, Vol. 30 No. 6, pp. 710-9.

Nijmeijer, H. and van der Schaft, A.J. (1990), *Nonlinear Dynamical Control Systems*, Springer, New York, NY.

Thrun, S. (1998), "Learning metric-topological maps for indoor mobile robot navigation", *Artificial Intelligence*, Vol. 99 No. 1, pp. 21-71.

Volpe, R. and Khosla, P. (1990), "Manipulator control with superquadric artificial potential functions: theory and experiments", *IEEE Transactions on Systems, Man and Cybernetics*, Vol. 20 No. 6, pp. 1423-36.

Yamamoto, Y. and Yun, X. (1994), "Coordinating locomotion and manipulation of a mobile manipulator", *IEEE Transactions on Automatic Control*, Vol. 39 No. 6, pp. 1326-32.

Yamamoto, Y. and Yun, X. (1997), "A modular approach to dynamic modeling of a class of mobile manipulators", *International Journal of Robotics & Automation*, Vol. 12 No. 2, pp. 41-8.

Corresponding author

Yoshio Yamamoto can be contacted at: yoshio@keyaki.cc.u-tokai.ac.jp