The 11th International Conference on Ubiquitous Robots and Ambient Intelligence (URAI 2014)
Nov. 12 – 15, 2014 at Double Tree Hotel by Hilton, Kuala Lumpur, Malaysia

Robust path planning for autonomous vehicle in position uncertainty

Jinkyu Yim[1], Minsung Kim[1], Seho Shin[1], and Jaeheung Park[1,2]

[1]Department of Transdisciplinary Studies, Seoul National University, Seoul 443-270, Korea
[2]Advanced Institutes of Convergence Technology, Suwon, Korea
(Tel : +82-31-888-9146; E-mail: minsungkim|shinsh|park73@snu.ac.kr)

Abstract - The abrupt and frequent change of a path for autonomous vehicle might occur in the presence of position uncertainty due to sensor noises or errors in localization. While path planning of an autonomous vehicle utilize multiple sensors to generate the path, it is necessary to not only reduce such errors but also to plan a consistent path. Therefore, we propose a framework that can generate a stable path by considering the history of previous paths. The result of experimental simulations indicates the decrease in rate of switching path directions in three different scenarios in position uncertainty. This framework demonstrates the effectiveness of path generation by applying a history parameter to modify cost function of the A* algorithm.

Keywords - Path planning, Path re-planning, Position uncertainty, Path history, A-star algorithm

1. Introduction

Path planning algorithms have been widely developed as a fundamental aspect of autonomous vehicle navigation. Path planning allows a vehicle to move from starting location to a goal location via collision-free motion. In order to plan an optimal path, different sensors of autonomous vehicle are used to detect the changing environment around the vehicle. Additionally, the construction of local map is necessary to the converged data of sensors into a more useful form. The path is generated based on the local map, which is constructed by sensing data in case of varying environment.

There are two conventional methods used for path planning, i.e. the potential field and Voronoi diagram [**?**],[**?**]. Since the potential field is sensitive to local minima, thus, a number of advanced algorithms and extensions of this method have been introduced. Typically, the recovery methods have been used to resolve the local minima. However, the generated path of recovery methods is not optimal [**?**]. The other traditional method for path planning is applying the Voronoi diagram. However, the optimal path cannot be generated if narrow ranges of sensors are utilized in this method. The limit of Voronoi diagram is revealed under conditions with sensor noise or utilizing short range sensors, which maximizes the distance between an autonomous vehicle and an obstacle.

Those methods for path planning mentioned above do not take into account the changes that might occur during successive queries for re-planning paths. Besides, uncertainty of surrounding environment due to errors in GPS and obstacle detection occur in autonomous vehicle driving condition. These errors in localization and sensor noise are directly related to the changes in continuous path planning.

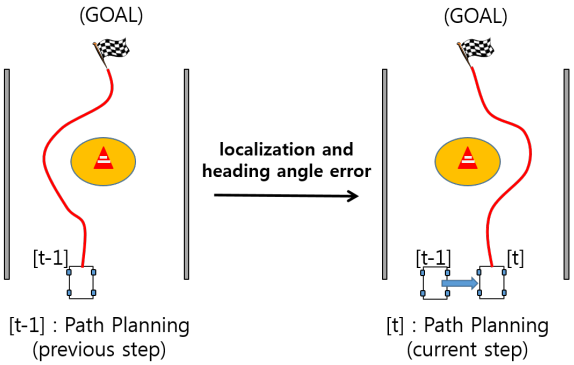

Fig. 1 An example of the problems: If there are some errors with respect to the position, the planned path will be changed in directions.

In order to solve these problems, various methods were proposed for compensation of localization errors, such as utilizing high-precision instruments or sensor filtering techniques. Nevertheless, the problems are still not solved in urban environments e.g. multipath problems of GPS, sensing errors due to road conditions, and low cost GPS devices.

We propose a method that can generate robust paths accounting for past history. Based on A* algorithm, the proposed method decreases the frequency of directional changes in path in the situation such as localization errors or sensor noises.

This paper is organized as follows: In Section II, the backgrounds of this paper are described. In Section III, we provides our framework and the experimental results are discussed in Section IV. The paper is concluded in Section V.

2. Backgrounds

2.1 Related works and Problems

The uncertainties of sensing position or sensor noises cause the changes in path planning of autonomous vehicle (Fig.1). The direction of the planned path can be changed by the position uncertainties, and desultory changes in path planning may place the vehicle in unstable condition. In addition, the possibility of a crash between a vehicles path and any obstacles increases. Cases with position uncertainties due to limitations of the autonomous vehicle sensing systems are given below.

• Position error of the vehicle using low cost GPSs

- Multipath problems in urban environments
- The position sensing error of the obstacle caused by road conditions(i.e. occlusion or sensor noise)

while using cameras or lidars. The path history constructs path planning with the previous data and experiences. This experience-based path planning methods have been applied in robotics and autonomous vehicle. One of experience-based path planning methods, the local path-planning method introduced a selective approach by using a path library to generate the shortest path with given start and goal positions [**?**], [**?**]. The other method also has the advantage of reducing computational time to generate complex motions of animate characters [**?**],[**?**].

Similar to the method described above, high-quality motion libraries have been used offline to generate transitions between the humanoid hands and feet on varied terrain [**?**]. Hauser et al. presented the computational efficiency using pre-stored datasets which can be used in path-planning methods. The path-planning algorithm using a randomized path planner was developed to plan a path online and save a sampling of experiential data that could be used for future planning tasks via the potential function in the same workspace [**?**].

Berenson et al. presented the Lighting framework, which is to reduce the computational time in path planning of a mobile manipulator. If new queries are called, the path data are extracted and modified with respect to both the time constraints required for re-planning and the correlations that consider the predefined constraints using a pre-stored path library [**?**].

We suggest a method of storing the fixed numbers of previous paths. This approach is similar to the path-planning method using previous paths and predefined information. However, our method focuses on planning a robust path using previous data, while other methods were mainly focused on computational efficiency in the path-planning process.

2.2 A* algorithm

This paper is based on the A* algorithm, which is a type of graph search algorithm. The main idea of the A* algorithm is to avoid expanding paths that are already expensive. The premise of the algorithm is that it uses a heuristic function to choose the shortest path from the current position to the goal position. The A* algorithm uses an evaluation function given by

$$f(n) = g(n) + h(n). \tag{1}$$

The estimated total cost of a path through n nodes to the goal position is denoted as $f(n)$, and $g(n)$ is the cumulative cost for n nodes. $h(n)$ is the estimated cost from n node to goal node. The best first search is realized when the value of $f(n)$ is the same as the value of $h(n)$. Though (1) always guarantees the optimal path, the vehicle can be unstable when errors related to external obstacles occur. Similar results could result even while using other graph search algorithms. Because, procedures of the calculation of the path cost in graph search algorithms are independent with the errors mentioned above. In this paper, we address this problem using a graph search algorithm for local path planning.

Fig. 2 illustrates the procedure to generate a path in

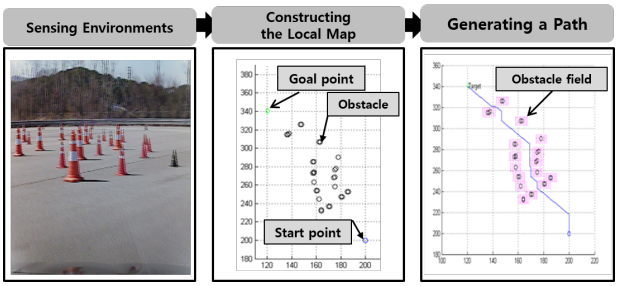

Fig. 2 The procedures to generate a path

this paper. The path-planning procedure for a local path generation begins with the configuration of a local map after sensing the surrounding environment(left). The local map is composed by the standard of the current vehicle coordinate system and has an octa-directional lattice structure. The positions of the obstacles were recognized by a lidar sensor installed on the vehicle. The vehicle is defined as a point. To avoid conflict with an obstacle, we increased the size of the obstacle relative to the vehicle dimension size. This is called an obstacle field. The goal position for the local path planning is a point amongst the global path, with respect to the current position of the vehicle. To determine the goal position in the local map, we converted the global goal position into the vehicles coordinate system based on its current position and heading angle. The A* algorithm generates the shortest route given the previously defined inputs to the local map. To use the A* algorithm to generate movement paths, we set the anterior direction limit to three($\pm$45,0 degree angle on the front) because of the nonholonomic condition of the vehicle.

3. The proposed Framework

3.1 Overview of the proposed framework

The flow of the proposed framework is shown in Fig. 3. At the initial state without any history paths, the A* algorithm is used to generate a search path. And then, the generated path is stored in path library. Once at least one path is obtained, the path-planning algorithm produces additional paths by considering the history paths and consistency factors. Whenever the goal position changes, the library manager performs a updating process according to the distance between the new goal position and the current goal position stored in the library. Using this update process, we can select paths whether to be stored in the library or not. Among history paths, N candidate paths are chosen by the library manager as an input to the A* algorithm. N is the fixed number. After that, we generate the smooth paths using Bezier curves. And then, paths are saved in the path library. In this section, we focus on how to consider the history paths and consistency factors. As the consideration of the history paths and consistency factors, calculation of the parameters, the difference cost c_c and the cost gain w_c will be described in the section

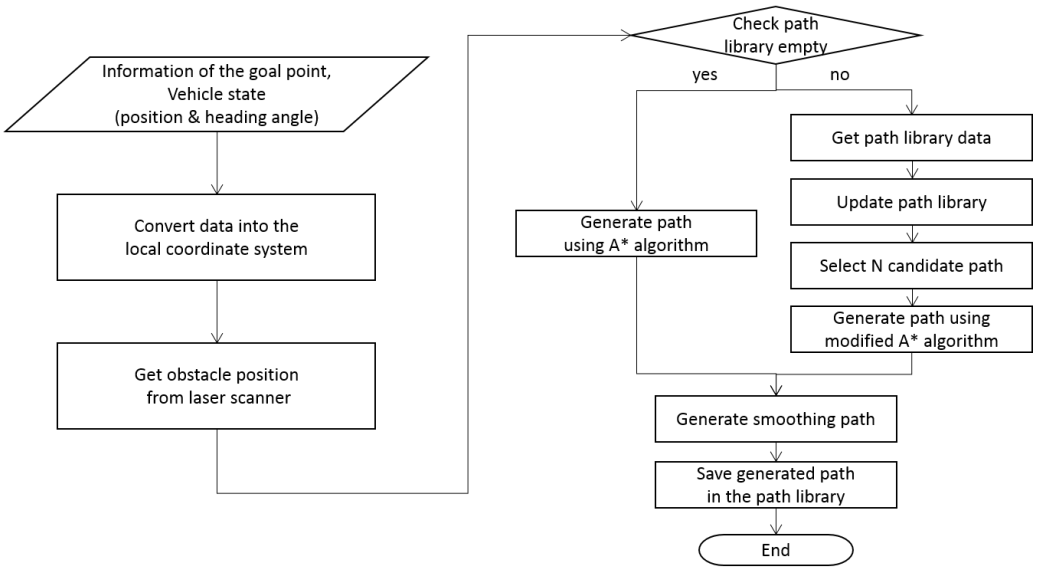

Fig. 3 The proposed framework

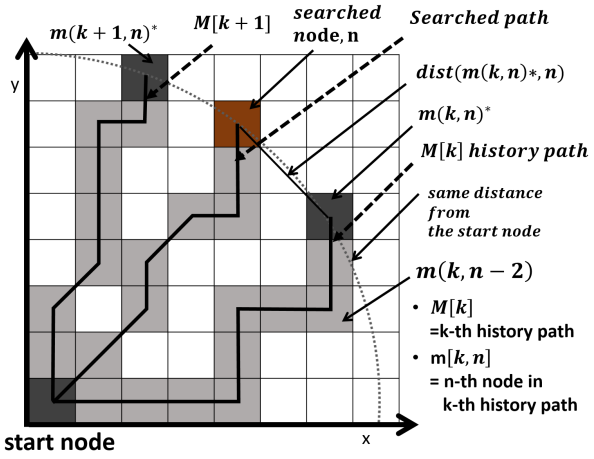

Fig. 4 Choosing $m(k,n)^*$ node with a search node.

3.2 and 3.3, respectively.

3.2 A* algorithm considering history paths: Difference Cost, c_c

$$g(n) = g(n-1) + d(n, n-1) + w_c c_c \tag{2}$$

Eq. (2) represents a modified cost function with respect to Eq. (1). c_c is the cost for the difference between the search node and the history paths, and w_c is the difference cost gain value. w_c will be addressed in the next subsection. By the criteria of A* algorithm, $g(n-1)$ means the cumulative distance for $n-1$ nodes, and $d(n, n-1)$ is the Euclidean distance between nodes n and $n-1$. Eq.(2) considers the distance cost of a particular search node and the difference cost between the cost of N history paths and a particular search node.

$$m(k,n)^* \tag{3}$$
$$= argmin_{m(k,n)\in M[k]} |dist(start, m) - dist(start, n)|$$
$$where \;\; k \in \{1, N\}$$

The method for determining a $m(k,n)^*$ node is illustrated in Fig. 4. As shown in the figure, the vehicle is positioned at a start node which is the start point of path planning. $M[k]$ is a k-th node set that generates a path from the start node to the goal node. Using (3), we determine the set of nodes, using $m(k,n)^*$, which minimizes the difference relative to start node between a current search node and the node of k-th history path, $M[k]$. $dist(m(k,n)^*, n)$ means the Euclidean distance between two points, the selected node, $m(k,n)^*$, and the current search node, n. The distance between $m(k,n)^*$ and the current search node is defined as a difference cost. We calculated the difference cost for the whole path stored in path library. And then, the smaller difference cost of path are chosen. Thus, the number of these $m(k,n)^*$s is N. Using this method, all of the difference costs for N history paths are calculated.

$$Reliability = \frac{N - N_{min}}{N_{max} - N_{min}} \tag{4}$$

Then, C_{avg_c} is obtained; it is the mean value of the entire difference costs between N history paths and the corresponding number of search nodes. However, C_{avg_c} need to be considered with the number of history paths because the greater the number of input paths given to the A* algorithm is, the greater the reliability of the calculation.

$$c_c(n) = C_{avg_c} \times (1 + \frac{N - N_{min}}{N_{max} - N_{min}}) \tag{5}$$

N, N_{min}, and N_{max} are the number of the history path, and the minimum and maximum number of the selected

history paths, respectively. The normalized reliability for the difference value that uses the minimum and maximum number of history paths is calculated by (4). Finally, the difference cost for the history paths named c_c is obtained by both C_{avg_c} and the normalized reliability in (5).

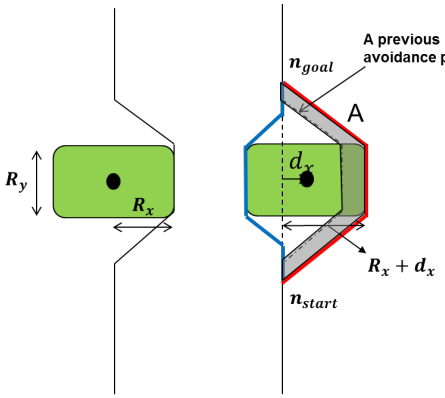

Fig. 5 An example of the different direction for avoiding the obstacle with the position uncertainty.

3.3 Difference Cost Gain w_c

There are three paths (dotted line, red and blue path) shown in Fig. 5. Dotted path is a previous avoidance path without any errors. Red and blue path are also avoidance paths with the error, d_x which means the error mentioned before. Parameter d_x is the horizontal error. We considered only lateral error. The direction of the red path is the same as the previous avoidance path and the direction of the blue path is different from the previous path. The vehicle was considered by a point named n_{start} and goal position was named as n_{goal}. The black circle in Fig. 5 means an obstacle which is located in the center of the path. The size of the obstacle needs to increase by vehicles size for stable obstacle avoidance. The green area represents the obstacle zone which is composed of R_x and R_y. R_x and R_y mean the half of length and width of the obstacle zone respectively.

We define the gain as the difference gain, w_c. Parameter w_c is calculated by the case that two total cost of avoidance paths are the same when d_x is the value of RMS error ($\bar{d}_x$) of the obstacle position. The horizontal RMS error is obtained from sensor specifications and localization method. We used w_c as a gain parameter of the difference cost in this paper.

$$\begin{aligned} g_{path_A}(n_{goal}) &= d(path_A) + w_c c_c(path_A) \\ &= 2G_d(R_x + \bar{d}_x) + R_y \\ &+ w_c(2R_x d_x + \bar{d}_x^2 + d_x R_y) \end{aligned} \tag{6}$$

where

$$\begin{aligned} d(path_A) &= 2 \times G_d(R_x + \bar{d}_x) + R_y \\ c_c(path_A) &= 2R_x\bar{d}_x + \bar{d}_x^2 + d_x R_y. \end{aligned} \tag{7}$$

$$\begin{aligned} g_{path_B}(n_{goal}) &= d(path_B) + w_c c_c(path_B) \\ &= 2 \times G_d(R_x - \bar{d}_x) + R_y + 4\bar{d}_x \\ &+ w_c(2R_x^2 + 2R_x\bar{d}_x + \bar{d}_x^2 + 2R_x R_y - \bar{d}_x R_y) \end{aligned} \tag{8}$$

where

$$\begin{aligned} d(path_B) &= 2 \times G_2(R_x - \bar{d}_x) + R_y + 4\bar{d}_x \\ c_c(path_B) &= 2R_x^2 - 2R_x\bar{d}_x + \bar{d}_x^2 + 2R_x R_y - \bar{d}_x R_y. \end{aligned} \tag{9}$$

$$w_c = \frac{2 \times (G_d - 1) \times \bar{d}_x}{R_x^2 + R_x R_y - 2R_x\bar{d}_x - \bar{d}_x R_y} \tag{10}$$

As shown in Fig. 5, the cost for selecting path A, the same direction of the previous path, is obtained by (6). $g_{path_A}(n_{goal})$ is the total cost of selecting path A, $d(path_A)$ is the distance cost of selecting path A, and $w_c c_c(path_A)$ is the difference cost between path A and the previous path. Eq. (6) to (10) were derived by the geometric shapes illustrated in Fig. 5. $d(path_A)$ is the geometric distance from the start node, n_{start}, to the goal node, n_{goal}. G_d is the diagonal distance of the lattice using the A* algorithm. The right and left distance for moving path is one. $c_c(path_A)$ is the total difference cost of selecting path A, which is represented by the gray region in Fig. 5. Because the A* algorithm moves through the lattices, the total difference cost between path A and the previous path can be illustrated by the gray region. In the same way, $g_{path_B}(n_{goal})$ is obtained using (9). Then, the path with a smaller cost is selected to avoid the obstacle based on criterion of the A* algorithm as shown in Fig. 6. Based on criterions of the A* algorithm, the path

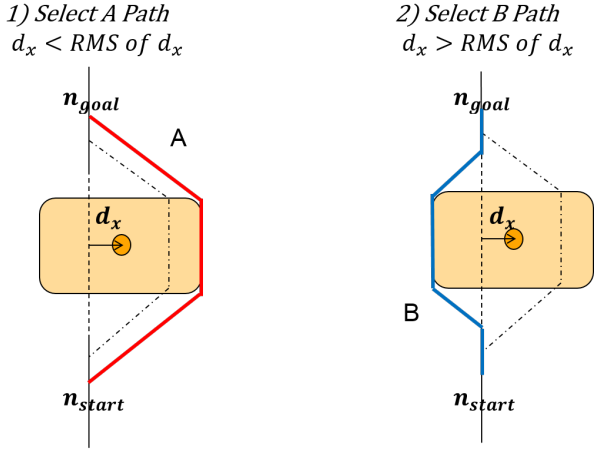

Fig. 6 Selecting the path

with more smaller cost is selected to avoid the obstacle. When the total difference cost of the two paths have a equal value, the gain parameter, w_c is calculated by (10).

4. Experimental results and Analysis

The simulated experiments were performed using a processor with eight 3.4 GHz cores and 4 GB RAM, and a Windows XP 32 bit operating system. To simulate the experiment, we used a Random Gaussian distribution for

the error of the vehicles position and heading angle. The values of variation were acquired via experimental results using a Sick 511 lidar on a real vehicle driving at 20 km per hour. We repeated the experiment one thousand times.

The first scenario is the case where the obstacle is positioned along the central direction of the vehicle. The path direction switched 507 times out of 1000 on the path toward the goal position, using the A* algorithm. However, the number of times the direction changed using the proposed algorithm decreased, as shown in Fig. 7. In the case where the number of history paths is greater than nine, the number of directional changes needed dramatically decreased. As shown in Fig. 8, the computational

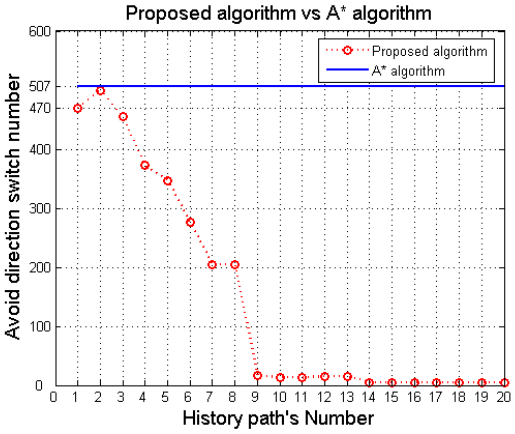

Fig. 7 Comparison of the number of the switched direction.

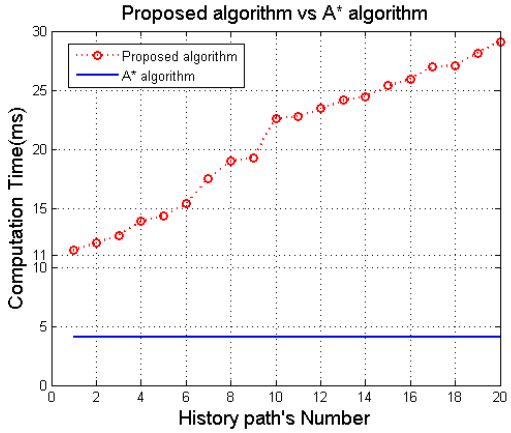

Fig. 8 Comparison of the computational time.

time needed to resolve a path increases with an increasing number of history paths stored. For the case using the A* algorithm, only the operation with the optimal path is considered; our proposed method increases the computational time because the calculation regarding difference with the history path is considered.

The second scenario is the case in which there is an obstacle occluded obscured by another obstacle. Fig. 9 shows the second scenario. The white regions of Fig. 9 represent the obstacle zone. The top of the figure represents a situation with the occluded obstacle. The left figure displays a sudden change while avoiding obstacles

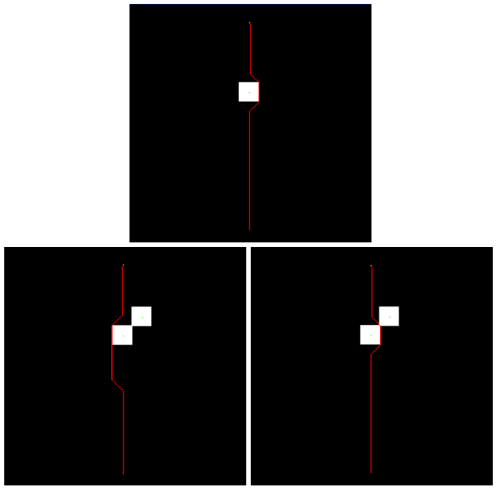

Fig. 9 The second scenario: The proposed method can generate the path considering the previous paths with the occluded obstalce.

using the conventional A* algorithm. Because the conventional A* algorithm uses only the current information to generate the path, it changes the path direction even though avoiding the obstacle can be realized with only minor deviations from the previous path. In addition, these changes make the vehicle unsafe. Fig. 9 (right) shows that the proposed algorithm can plan the modified path within a modest changes range of path deviation. As shown in the right figure, our method maintains the consistency of the path direction by considering the difference cost based on the path history and it can searches a new avoidance path. Therefore, stable and consistent path planning is possible for the case wherein an obstacle is occluded by another one.

The third scenario is the case in which the avoidance

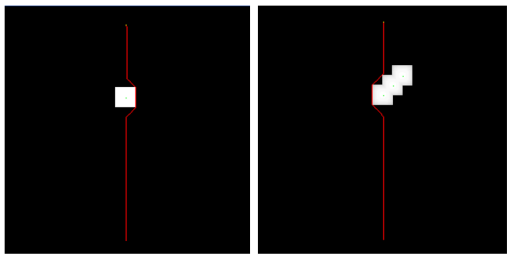

Fig. 10 The third scenario: The proposed method can be adapted with the new direction for avoiding obstacles.

path must change owing to a narrow, inaccessible section behind the obstacles. This problem is different from the previous cases because the direction of the path related to the previous path is not feasible. However, the proposed framework was able to successfully alter the direction. After all, our method can be adapted in the new environment without following only the previous path.

5. Conclusion

In this paper, we propose a framework that can realize the stable path planning of an autonomous vehicle

in the presence of localization errors and sensor noise errors. The proposed framework creates a robust path using the gain parameter and difference cost parameter that consider the consistency of the previous path based on data from a path library. We found that the number of path directional changes decreased when the proposed framework was used for path planning. Simulation results showed that this method was able to generate a path for avoiding obstacles, even when such obstacles are discovered suddenly. We will seek to apply the proposed framework to real-time scenarios in future investigations.

Acknowledgment

This research was supported by the promotion for development of unmanned autonomous ground vehicle technology program (No. 10043052) funded by the Ministry of Knowledge Economy (MKE, South Korea).

References

[1] O.Khatib, "Real-time obstacle avoidance for manipulators and mobile robots," *International Journal of Robotics Research*, no. 5, pp. 90–98,1986.

[2] S. Garrido, L. Moreno, and D. Blanco, "Voronoi diagram and fast marching applied to path planning," *Robotics and Automation,IEEE International Conference on*, pp. 3049–3054,2006.

[3] R. Mahkovic, and T. Slivnik,"Generalized local Voronoi diagram of visible region," *Robotics and Automation,IEEE International Conference on*, pp. 349–355, 1998.

[4] Knepper, A. Ross, and T. Mason, "Empirical sampling of path sets for local area motion planning," *Experimental Robotics, International Symposium on*, pp. 451– 462, 2009.

[5] Green, J. Colin, and A. Kelly, "Toward optimal sampling in the space of paths," *Robotics Research, International Symposium of*, pp. 171–180, 2007.

[6] J. Lee, J. Chai, P. Reitsma, J. Hodgins, and N. Pollard, "Interactive control of avatars animated with human motion data," *Association for Computing Machinery, Transactions on Graphics*, vol. 21, no. 3, pp. 491–500, 2002.

[7] M. Lau, and J. Kuffner, "Behavior planning for character animation," *Eurographics, ACM SIGGARAPH Symposium on Computer Animation*, pp. 271–280, 2005.

[8] K. Hauser, T. Bretl, K. Harada, and J.C. Latombe, "Using motion primitives in probabilistic sample-based planning for humanoid robots," *Algorithmic Foundation of RoboticsVII, Springer*, pp. 507–522,2008.

[9] S. Caselli, and M. Reggiani, "ERPP: an Experience-based randomized path planner," *Robotics and Automation,IEEE International Conference on Robotics and Automation*, vol. 2,pp. 1002–1008, 2000.

[10] D. Berenson, P. Abbeel, and K. Goldberg, "A Robot Path Planning Framework that Learns from Experience," *Robotics and Automation,IEEE International Conference on Robotics and Automation*, pp.3671-3678, 2003.