 doi:10.20944/preprints202107.0254.v1

Article

An Enhancement Method of Rapidly-exploring Random Tree Robot Path Planning using Midpoint Interpolation

Jin-Gu Kang [1], Yong-Sik Choi [2] and Jin-Woo Jung [1,*]

[1] Department of Computer Science and Engineering, Dongguk University, Seoul 04620, Korea; kanggu12@dongguk.edu
[2] Department of Artificial Intelligence, Dongguk University, Seoul 04620, Korea; sik2230@dongguk.edu
* Correspondence: jwjung@dongguk.edu; Tel.: +82-2-2260-3812

Abstract: To solve the problem that sampling-based Rapidly-exploring Random Tree (RRT) method is difficult to guarantee optimality. This paper proposed the Post Triangular Processing of Midpoint Interpolation method minimized the planning time and shorter path length of the sampling-based algorithm. The proposed Post Triangular Processing of Midpoint Interpolation method makes a closer to the optimal path and somewhat solves the sharp path problem through the interpolation process. The experiments were conducted to verify the performance of the proposed method. Applying the method proposed in this paper to the RRT algorithm increases the efficiency of optimization compared to the planning time.

Keywords: robot path planning; RRT; midpoint interpolation; triangular rewiring; path smoothness

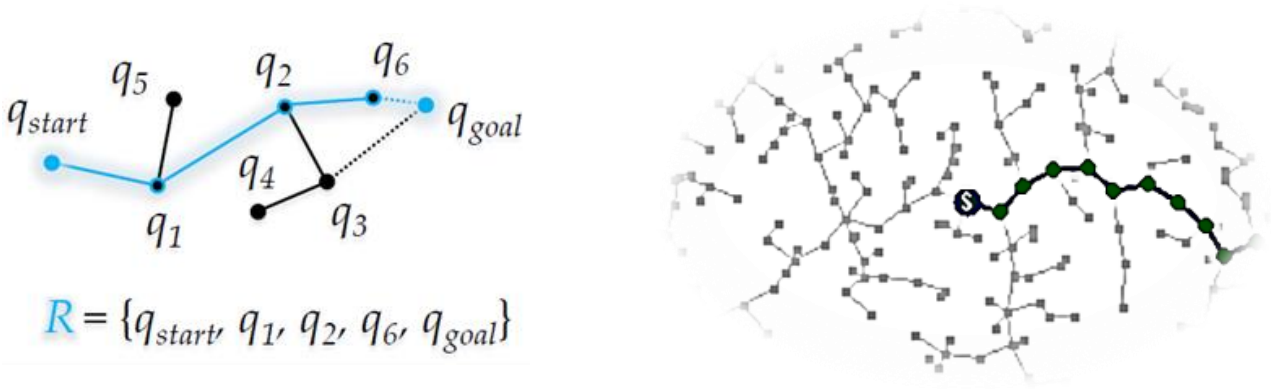
1. Introduction

The goal of path planning [1] is to plan a path for a mobile robot to proceed from a starting point to a destination point in Euclidean space as efficiently as possible while avoiding obstacles while maintaining optimality, clearance, and completeness. optimality refers to always planning a path with the ideal path length, clearance refers to how unlikely it is for obstacles and the mobile robot to collide. completeness is defined as the property to construct a path from a start point to a destination point without colliding with obstacles.

Sampling-based RRT (Rapidly-exploring Random Tree) algorithm [2], it is difficult to ensure optimality. As shown in Figure 1(a), the RRT algorithm is a path planning that involves repeatedly adding a randomly sampled position as a child node in a tree with the starting point as the root node until the destination point is reached. The tree extends out in the shape of a stochastic fractal as shown in Figure 1(b) and has an algorithm of locating the destination point with this algorithm.

The RRT algorithm and other sampling-based algorithms [3-4] offer the benefit of planning a path in a shorter time with less computing than traditional path planning algorithms like Visibility Graph-based [5], Cell Decomposition-based [6], and Potential Field-based [7]. On the other hand, it does not ensure optimality and has the drawback of being probabilistically assured completeness. The latter is also known as Probabilistic completeness [8], which implies that completeness is assured when the number of random samples is infinite, but not always when the number of random samples is limited. The goal of this research is to look at the RRT algorithm, which ensures completeness and performs better than the related works.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 12 July 2021 doi:10.20944/preprints202107.0254.v1



(a) **(b)**

Figure 1. Overview of RRT algorithm: **(a)** The planned path R from the starting point q_{start} through the waypoints q_1, q_2, q_6 to the destination q_{goal} (q_i is the point on the path); **(b)** The process of finding a destination point by a stochastic fractal shape from the root node(S) as starting point.

The proposed "Post Triangular Processing of Midpoint Interpolation" method is effective in path planning algorithms that do not guarantee optimality, such as the RRT algorithm that locally piecewise linear shape, and can be used as a post-processing method after a path has been planned using one of these algorithms.

The sampling-based path planning algorithm's primary strength is the fast-planning speed based on the small amount of computation compared to the traditional path planning algorithms [3]. Since this proposed method has this strength, should not be slow compared to the RRT algorithm.

Performance verification using simulation in various environments and mathematical modeling was used to validate the performance of the proposed method in this paper. The case in which the proposed algorithm to the sampling-based path planning method is applied and the case in which it is not applied are compared through simulation. Here, the planning time and path length of the first complete path that first reaches a destination point from a starting point are evaluated.

2. Rapidly-exploring Random Tree (RRT)

Steven M. LaValle proposed the RRT algorithm in 1998 [2], which is a representative algorithm of the sampling-based path planning algorithm. It is designed to have a lot of degrees of freedom and is useful for planning a path under non-holonomic constraints.

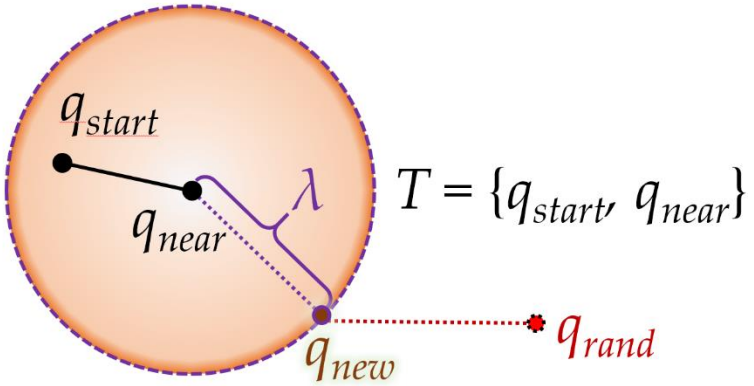


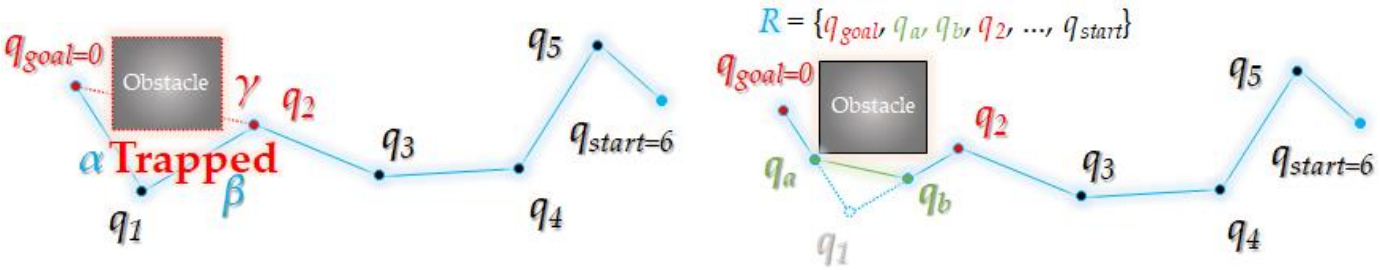
Figure 2. The process of the RRT algorithm: When creating a new node q_{new} at a position separated by a step length λ in the direction of the random sample position(q_{rand}) based on the q_{near} node (position) closest to the random sample position (q_{rand}) in the tree T with the starting point q_{start} as the root node.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 12 July 2021 doi:10.20944/preprints202107.0254.v1

As shown in Figure 2, when a random sample is generated in the Configuration Space, the nearest node to the position of the random sample is identified among the nodes constituting the tree with the starting point as the root node. A new node is generated at the random sample position and inserted into the tree if the random sample position is nearer than the step length. The process of tree extension is repeated till the destination point is achieved.

3. Proposed Post Triangular Processing Method of Midpoint Interpolation

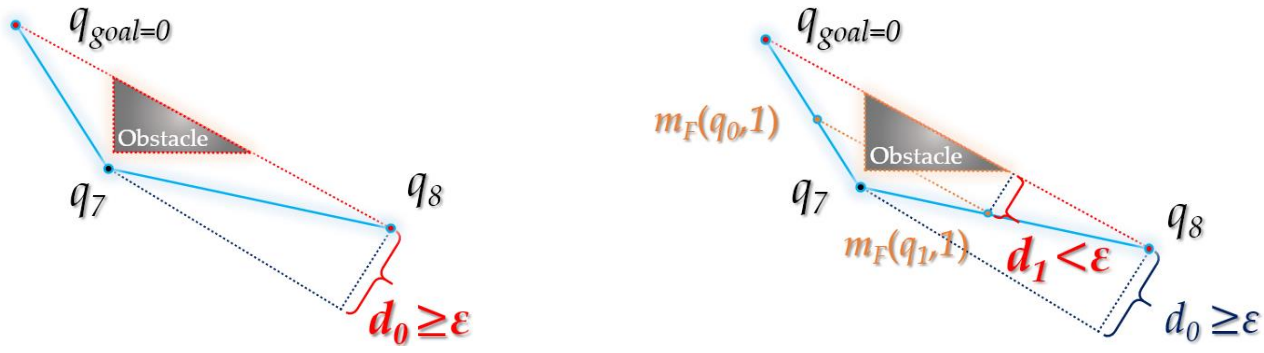
The proposed Post Triangular Processing of Midpoint Interpolation method can be applied to path planning algorithms that do not guarantee optimality such as RRT (Rapidly-exploring Random Tree) algorithm, rewire and midpoint interpolation based on the triangular inequality principle.



(a) **(b)**

Figure 3. Summary of Post Triangular Processing of Midpoint Interpolation method: **(a)** When the line segment γ with node q_0 and its grandparent node q_2 in tree R is not free from obstacle collision; **(b)** Grandparent node q_2 of node q_0 is connected as the parent node of node q_0 and the parent node q_1 is deleted from the tree.

The basic principle is that the node serving as a waypoint in the planned path checks whether an obstacle collides with its own grandparent node, and if it is free from obstacle collision, the grandparent node rewires to the parent node. If it is not free from obstacle collision, as shown in Figure 3, the locally piecewise linear shape path created between the node, its parent node and grandparent node is made a more optimal path through the interpolation process. In this process, a new node is interpolated into the path and deviates from the piecewise linear, so it has the advantage of being able to somewhat solve the sharp path problem (The problem that a mobile robot that has kinematic constraints because the slope is not smooth).



(a) **(b)**

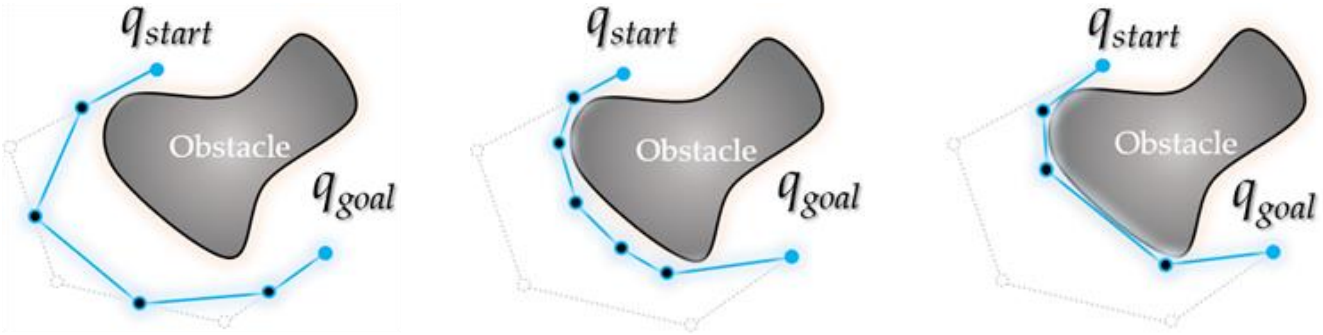
Figure 4. Interpolation function of Post Triangular Processing of Midpoint Interpolation method: **(a)** When the height d_0 of a triangle formed by waypoints q_0, q_1 and q_2 of the path is greater than ε (interpolation continuation); **(b)** When the height d_1 of the triangle formed by the midpoints $m_F(q_0,1)$ and q_1 of q_0 and q_1 and the midpoint $m_F(q_1,1)$ of q_1 and q_2 is less than ε (interpolation stop).

This Post Triangular Processing of Midpoint Interpolation method was designed based on the Polygon approximation algorithm [9-10], so as shown in Figure 4, the path is calculated through a constant value called ε (the threshold of minimum clearance) ($\varepsilon > 0$). It determines how closely the obstacle is approximated, or in other words, how close to the optimal path it is. Here in Figure 4, d follows Equation 1:

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 12 July 2021 doi:10.20944/preprints202107.0254.v1

$$d_n(q_i) = \begin{cases} \left(d_{n-1}(q_i)\right)/2, & n > 0 \\ \left(2\sqrt{s(s-\alpha)(s-\beta)(s-\gamma)}\right)/\gamma, & n = 0 \end{cases} \quad (s = (\alpha+\beta+\gamma)/2) \qquad (1)$$

For the waypoint q_i, the value of d decreases by 1/2 as interpolation proceeds (n). The initial value d_0 is the height of the triangle formed by the three line segments α, β, γ ($\gamma<\alpha+\beta$) and the value of d_n become (d_n-1)/2, when let α be the line segment from q_i to the first and next waypoints, β to be the line segment from q_i to the first next waypoint and the second next waypoint, γ to be the line segment from q_i to the second next waypoint. This d values as a measure to confirm the clearance of the obstacle compared to ε. This is because the smaller the d, the closer the path to the obstacle.



(a) **(b)** **(c)**

Figure 5. Difference according to ε value in the Post Triangular Processing of Midpoint Interpolation method: **(a)** Result at the ε value; **(b)** Result at a value set smaller than ε (smooth path); **(c)** Results at values set smaller than that ε (closer to optimal path).

However, since optimality and clearance are opposite attributes, the closer ε is to 0 as shown in Figure 5, the more similar the path or path length to the visibility graph, but it is not smooth path [11], The farther away from 0 (within a significant value where the path is modified), the farther from the optimum, but a smooth (kinetic) path is made, so ε should be set to a suitable value according to the environment.

The following Algorithm 1 shows the pseudocode of the proposed Post Triangular Processing of Midpoint Interpolation method. It is mainly composed of Post Triangular function (Algorithm 2) and Interpolation function (Algorithm 3).

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 12 July 2021 doi:10.20944/preprints202107.0254.v1

Algorithm 1 Pseudocode of the Proposed "Post Triangular Processing of Midpoint Interpolation" Method.

Input:

$R \leftarrow$ path from {*RRT*/ …}

$C \leftarrow$ position set of all (measured) boundary points in all (known) obstacles

$\varepsilon \leftarrow$ threshold value of minimum clearance

Output:

$R \leftarrow$ modified path R

Initialize:

$f_{modify} \leftarrow$ ***true***

Procedure *postTriProcOfMidInterpolation*
Begin

```
While f_modify Do
    f_modify ← false      // is the path modified
    t ← 0      // index of the currently focused point
    q_child ← first point in R
    q_parent ← next point of q_child in R
    While not [q_parent is last point in R] Do
        q_ancestor ← next point of q_parent in R
        If not isTrapped(q_child, q_ancestor, C) Then
            R ← postTriangular(R, ε, t, f_modify)
        Else
            R ← interpolation(R, C, ε, t, f_modify)
            q_child ← t-th point in R
        q_parent ← next point of q_child in R
End
```

The input value of the Post Triangular Processing of Midpoint Interpolation method consists of the path R planned through the path planning algorithm such as the RRT algorithm, the obstacle area information C, and the threshold value ε of the minimum clearance.

The f_{modify} is a variable that determines whether the input path R has been modified by this method, and if the path is modified even once, the entire process is repeated. If the path modification does not occur in the process of repeating again, the algorithm is terminated. t refers to the index of the waypoint of R that is currently focused. That is, if t is 0, it means the starting point, which is the first point of R.

In R, when the first focusing point is q_{child}, the next point of that point q_{child} is q_{parent}, and the next point of that point q_{parent} is mentioned $q_{ancestor}$, it is determined whether the distance between q_{child} and $q_{ancestor}$ is free from obstacle collision (*isTrapped*() function). If it is free from collision, it calls *postTriangular*(), otherwise it calls *interpolation*(). *postTriangular*() connects q_{child} and $q_{ancestor}$ like the Triangular Rewiring method [4], and the q_{parent} between them is deleted from the path. *interpolation*() finds (interpolates) the point between q_{child} and q_{parent}, and between q_{parent} and $q_{ancestor}$ that are free from obstacle collision when connected and rewire q_{child}, $q_{ancestor}$ and those two points found. If R and t are updated due to *postTriangular*() or *interpolation*(), update q_{child} (the t-th waypoint of R), q_{parent} and $q_{ancestor}$ accordingly. If q_{parent} is the last point in R, check f_{modify}. Otherwise, repeat the above process again for the updated q_{child} and $q_{ancestor}$.

Here, path modification by *postTriangular*() deletes the waypoints and makes a more optimal path, but has the effect of sharpening the path shape, and path modification by

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 12 July 2021 doi:10.20944/preprints202107.0254.v1

interpolation() creates new waypoint between the waypoints. Adding/inserting has the effect of making a more optimal path while also smoothing the path shape. Of course, in the effect of making a more optimal path, path modification by *postTriangular*() is more efficient than path modification by *interpolation*().

Algorithm 2 Pseudocode of the Proposed "Post Triangular" Function.

Input:

R ← path R from *postTriProcOfMidInterpolation*

t ← point index t from *postTriProcOfMidInterpolation*

f_{modify} ← boolean f_{modify} from *postTriProcOfMidInterpolation*

Output:

R ← modified path R

f_{modify} ← result of boolean f_{modify} //**return by reference**

Procedure *postTriangular* **From** *postTriProcOfMidInterpolation*

Begin

1 q_{child} ← t-th point in R

2 q_{parent} ← next point of q_{child} in R

3 $q_{ancestor}$ ← next point of q_{parent} in R

4 R ← **Delete** path<q_{child}, q_{parent}> and path<q_{parent}, $q_{ancestor}$> from R

5 R ← **Insert** path<q_{child}, $q_{ancestor}$> to R

6 f_{modify} ← ***true***

End

The input value of *postTriangular*() of the Post Triangular Processing of Midpoint Interpolation method consists of the path R, the focusing point index t, and the path modification f_{modify} from the Post Triangular Processing of Midpoint Interpolation method.

Rewiring is performed on the t-th waypoint q_{child} of R, the next point q_{parent}, and the next point $q_{ancestor}$ of that point again. First, delete the path between q_{child} and q_{parent}, and the path between q_{parent} and $q_{ancestor}$. Then insert the path between q_{child} and $q_{ancestor}$. Finally, f_{modify} returns 'true' because the path has been modified. Here, Path<A, B> means a partial path from the waypoint A to the waypoint B in the complete path.

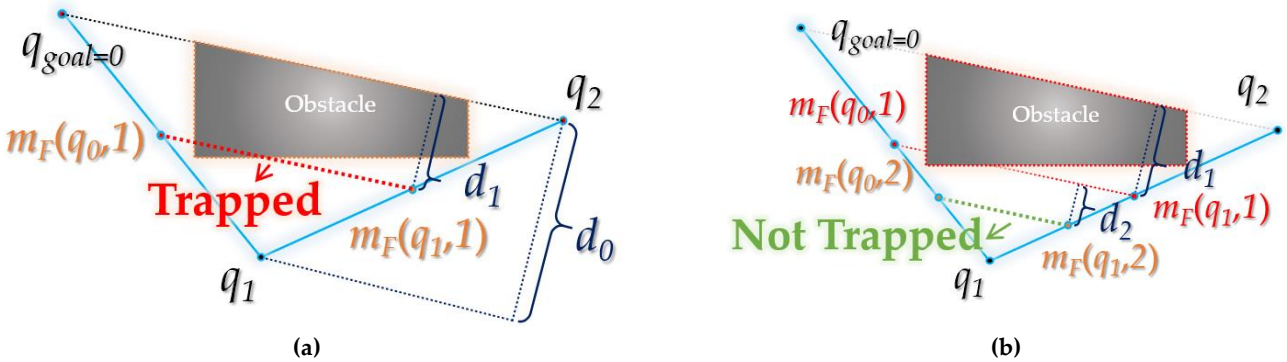


Figure 6. Details of Post Triangular Processing of Midpoint Interpolation method: **(a)** When the midpoint $m_F(q_0,1)$ of the waypoint q_0, q_1 of the path and the midpoint $m_F(q_1,1)$ of q_1, q_2 are not free from obstacle collision; **(b)** When the midpoint $m_F(q_0,2)$ of the interpolation point $m_F(q_0,1)$, q_1 and the midpoint $m_F(q_1,2)$ between q_1, the interpolation point $m_F(q_1,1)$ are free from obstacle collision.

The proposed Post Triangular Processing of Midpoint Interpolation method is to find a free interpolation point ($m_F(q_0)$, $m_F(q_1)$) from obstacle collisions between waypoints (q_0~q_1, q_1~q_2) while descending in the direction of the midpoint (q_1) as shown in Figure 6 in the interpolation process.

However, the interpolation point m_F follows Equation 2:

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 12 July 2021 doi:10.20944/preprints202107.0254.v1

$$m_F(q_i,k)=\begin{cases}\left(\frac{m_F(q_i,k-1).x+\zeta(q_i).x}{2},\frac{m_F(q_i,k-1).y+\zeta(q_i).y}{2}\right), & k>0\\ q_i, & k=0\end{cases}\quad(k\in\mathbb{N})\qquad(2)$$

When the k-th interpolation point of the interpolation point q_i is $m_F(q_i,k)$, the 0-th interpolation point becomes itself q_i, and the 1-st interpolation point is the midpoint of q_i and the point $\xi(q_i)$ next to q_i. , and the 2-nd interpolation point becomes the midpoint of $m_F(q_i,1)$ and $\xi(q_i)$. Here, $\xi()$ is a function that receives the node as a variable and returns the parent node of that node. The n-squared ($n\geq0$) of the $\xi()$ function can be expressed as $\xi^n(q_i):=(\overbrace{\xi\circ\xi\circ\ldots\circ\xi}^{n})(q_i)$, and if n is 0, $\xi^0(q_i):=q_i$holds. That is, $m_F(q_i,k)$ ($k>0$) becomes the midpoint between $m_F(q_i,k\text{-}1)$ and $\xi(q_i)$. At this time, d becomes (d_k-1)/2.

The following Algorithm 3 shows the pseudocode of *interpolation*() of the proposed Post Triangular Processing of Midpoint Interpolation method.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 12 July 2021 doi:10.20944/preprints202107.0254.v1

Algorithm 3 Pseudocode of the Proposed 'Interpolation' Function.

Input:

R ← path R from *postTriProcOfMidInterpolation*

C ← position set C from *postTriProcOfMidInterpolation*

ε ← threshold value ε from *postTriProcOfMidInterpolation*

t ← point index t from *postTriProcOfMidInterpolation*

f_{modify} ← boolean f_{modify} from *postTriProcOfMidInterpolation*

Output:

R ← modified path R

t ← updated point Index t **// return by reference**

f_{modify} ← result of boolean f_{modify} **// return by reference**

Initialize:

q_{child} ← t-th point in R

q_{parent} ← next point of q_{child} in R

$q_{ancestor}$ ← next point of q_{parent} in R

Procedure *interpolation* **From** *postTriProcOfMidInterpolation*

Begin

d ← altitude of triangle consisting of q_{child}, q_{parent} and $q_{ancestor}$ with base<q_{child}, $q_{ancestor}$>

m_a ← midpoint between q_{child} and q_{parent}

m_b ← midpoint between q_{parent} and $q_{ancestor}$

While ***true*** **Do**

  If $d >= \varepsilon$ **Then**

    If not *isTrapped*(m_a, m_b, C) **Then**

      R ← **Delete** path<q_{child}, q_{parent}> and path<q_{parent}, $q_{ancestor}$> from R

      R ← **Insert** path<q_{child}, m_a>, path<m_a, m_b> and path<m_b, $q_{ancestor}$> to R

      f_{modify} ← ***true***

      Break

    Else

      d ← d / 2

      m_a ← midpoint between m_a and q_{parent}

      m_b ← midpoint between m_b and q_{parent}

  Else

    t ← t + 1

    Break

End

The input value of *interpolation*() of the Post Triangular Processing of Midpoint Interpolation method consists of the path R, the obstacle area information C, the focusing point index t, and the path modification f_{modify} from the Post Triangular Processing of Midpoint Interpolation method.

From the t-th waypoint q_{child} of R, the next point q_{parent}, and the next point $q_{ancestor}$ of that point, the height d of the triangle is obtained, m_a is the midpoint of q_{child} and q_{parent}, and m_b is the midpoint of q_{parent} and $q_{ancestor}$. If the path between m_a and m_b is free from obstacle collision (*isTrapped*()), delete the path between q_{child} and q_{parent}, and insert the path between q_{child} and m_a, the path between m_a and m_b, and the path between m_b and $q_{ancestor}$. Also, since there is a modified the path, f_{modify} becomes 'true', returns it, and the method terminates. If the line segment between m_a and m_b is not free from obstacles, the value of d is decreases by 1/2, m_a is updated to the midpoint of m_a and q_{parent}, and m_b is updated to the midpoint of m_b and q_{parent}, determine whether m_a and m_b are free from obstacles again. This repeat process proceeds until a case is found in which m_a and m_b are free from obstacles or d becomes

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 12 July 2021 doi:10.20944/preprints202107.0254.v1

smaller than ε. If d becomes smaller than ε, the value of t is increased by 1 and the method is terminated.

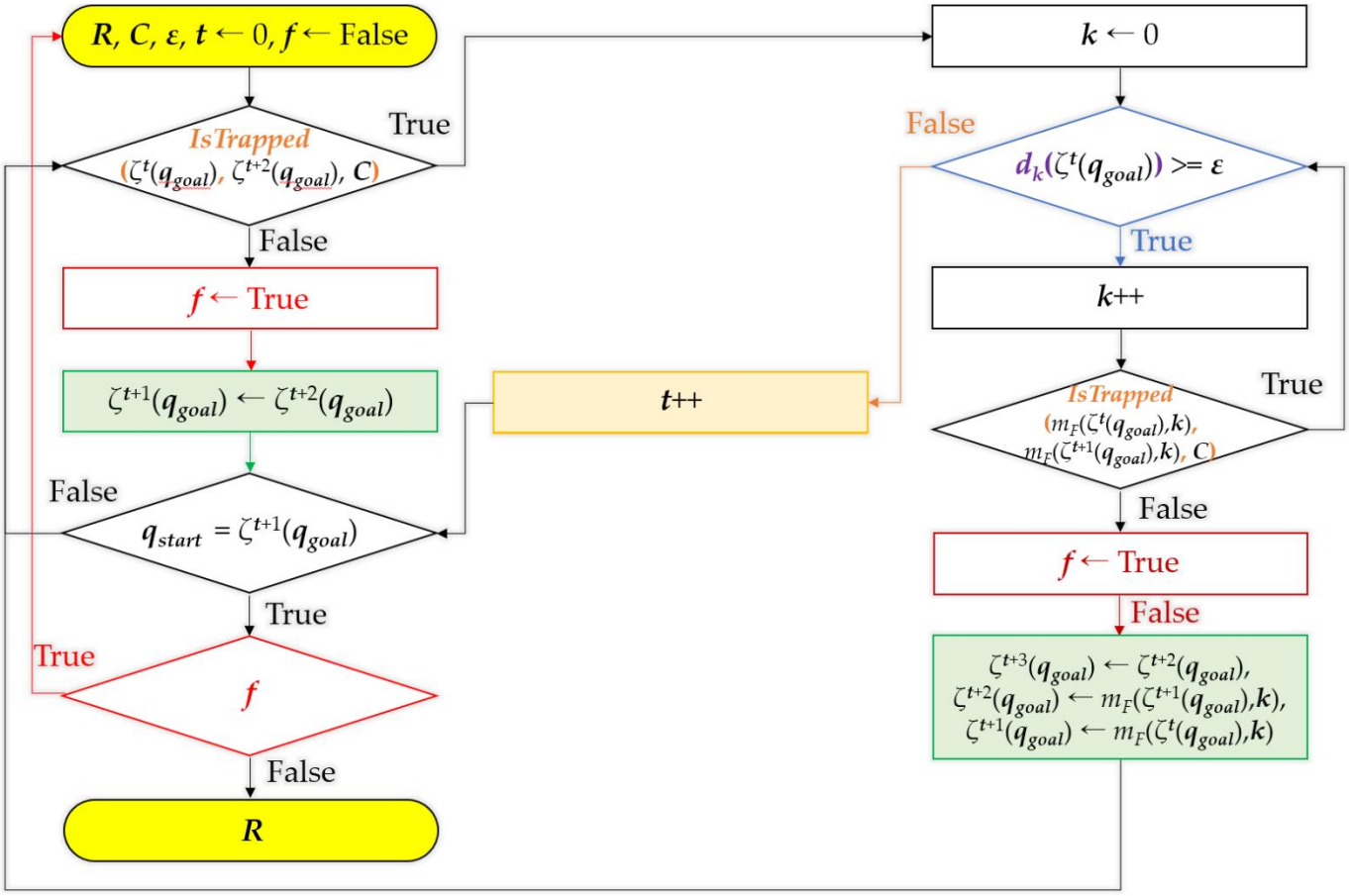


Figure 7. Flow chart of Post Triangular Processing of Midpoint Interpolation method.

The following Figure 7 shows the overall flow chart of the proposed Post Triangular Processing of Midpoint Interpolation method. Here, $\xi^t(q_{goal})$ means the t-th next waypoint from the starting point q_{goal} of the path R, and $\xi^{t+n}(q_{goal})$ means the n-th next waypoint in the $\xi^t(q_{goal})$. That is, there are n waypoints between $\xi^t(q_{goal})$ and $\xi^{t+n}(q_{goal})$.

4. Experimental Results

The path between RRT in various environment maps through simulation and the RRT algorithm to which the proposed Post Triangular Processing of Midpoint Interpolation method is applied were used to validate the performance of the method proposed in this paper and the results of path planning were compared.

The average value after repeated trial 100 times (sampling position is changed for each trial) of the path length(px) and the planning time(ms) of the first complete path is the performance measures compared (The first complete path that first reaches a destination point from a starting point are evaluated).

Various environmental maps were examined and used to validate the performance of the proposed path planning algorithm in related works. Since the efficiency of the performance measures expected during the experiment somewhat varies based on the composition, such as the number, location, or shape of obstacles, it is consequential to choose which environment map to utilize.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 12 July 2021 doi:10.20944/preprints202107.0254.v1

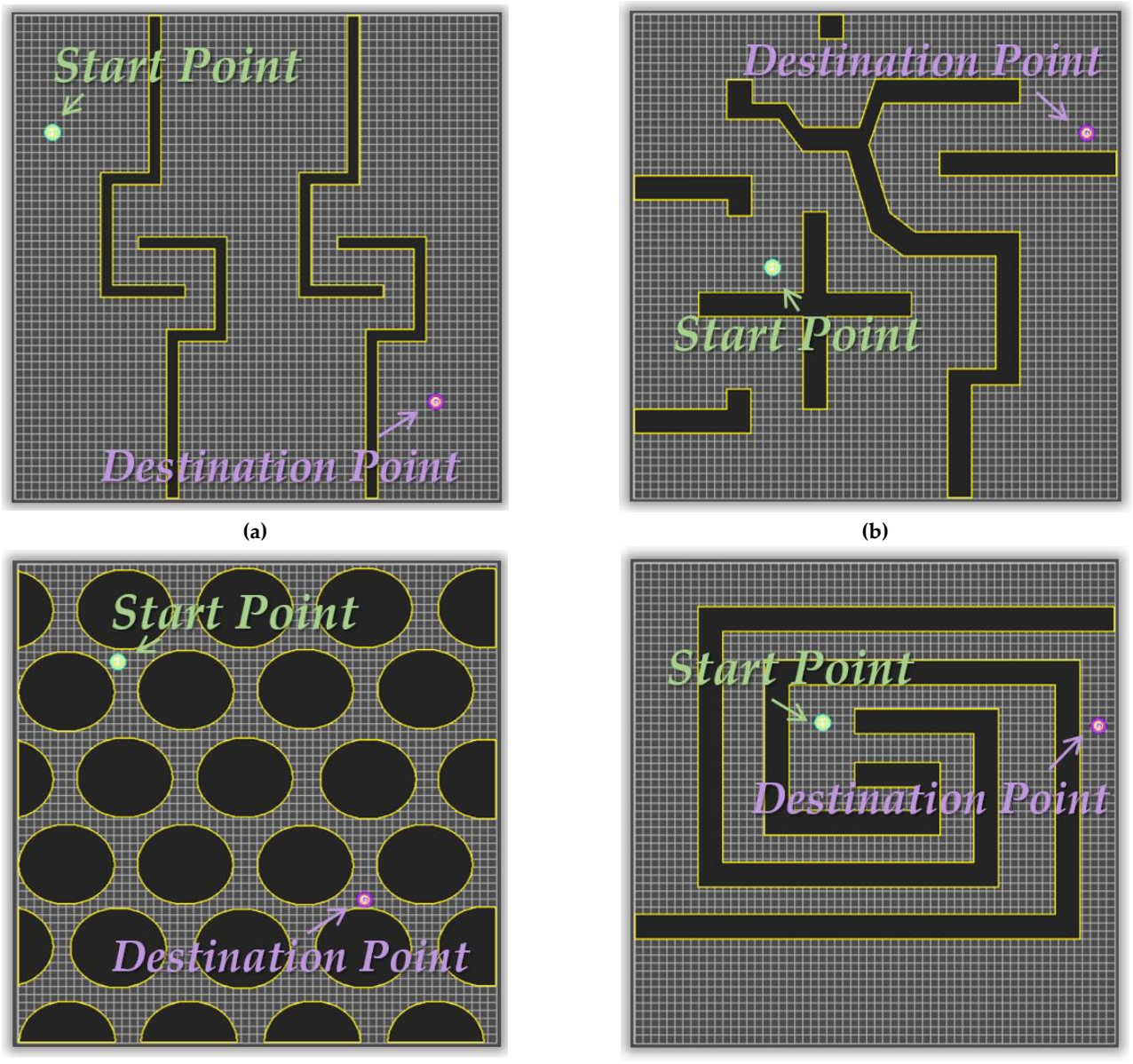


Figure 8. Environment maps for experiments: **(a)** Map 1; **(b)** Map 2; **(c)** Map 3; **(d)** Map 4.

The four environment maps used in the experiment are shown in Figure 8. The Four environment maps are partially referred to the experimental environment proposed by Jihee Han in 2017 [12]. All environment maps are 600*600px in size, with a 30px step length. The start point(S) is represented by the green circle, while the destination point(G) is represented by the purple circle. An obstacle is a black polygon with a yellow contour (blue in the experimental results).

Map 1 of Figure 8(a) appears to be an efficient environment for validating optimality and completeness, it is weakly environment to sampling-based path planning algorithms like the RRT algorithm. Lots of samplings are required since the probability of finding a solution is low. Map 2 of Figure 8(b) appears to be an efficient environment for validating optimality and completeness of the path planning algorithm. Map 3 of Figure 8(c) is an environment that is efficient for validating the optimality and the planning time of the path planning algorithm as it is consists of obstacles (50 vertices) that approximate circle. Map 4 of Figure 8(d) is an environment that is efficient for validating the optimality and the planning time of the path planning algorithm and is a weak environment to sampling-based path planning algorithms such as the RRT algorithm.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 12 July 2021 doi:10.20944/preprints202107.0254.v1

The number of samples and planning time required shoots up in case the path to the destination point is narrow or few entrances, since the sampling-based path planning algorithm depends on probabilistic completeness.

Table 1. Computer performance for simulation.

H/W	**Specification**
CPU	Intel Core i7-6700k 4.00GHz (8 CPUs)
RAM	32768MB (32GB DDR4)

The specification of the computer used in the simulation is shown in Table 1. The simulator used for the simulation [4] was developed based on C# WPF (Microsoft Visual Studio Community 2019 Version 16.1.6 Microsoft .NET Framework Version 4.8.03752), and only a single thread was used for calculations except for the visual part. generally, depending on the specification of the computer, the result of performance Measurement such as planning time may be variations during the simulation.

Validate the experimental results (path length, planning time) in the four environment maps that the Post Triangular Processing of Midpoint Interpolation method (ε: 50, 30, 10px) is applied to the RRT algorithm and its path planning results. Since ε requires a higher amount of computation as it gets smaller, it was set to a suitable value nearby according to the 30px step length of the experimental environment.

The experimental results for each map are consist of a figure and table. The figure is a path planning result for each algorithm is shown in the case of a single trial (the figure for each algorithm is not the result of repeated trials). The table is shown an average value that results of planning time and path length from path planning repeated trials. There may be a significant difference between the performance observed visually and the numerical results in the table for one of the repeated trials. The form of the planned path for each algorithm is shown in the figure for visual refer. furthermore, the proposed Post Triangular Processing of Midpoint Interpolation method is used to see whether there is a region where the piecewise linear form path is smoothed.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 12 July 2021 doi:10.20944/preprints202107.0254.v1

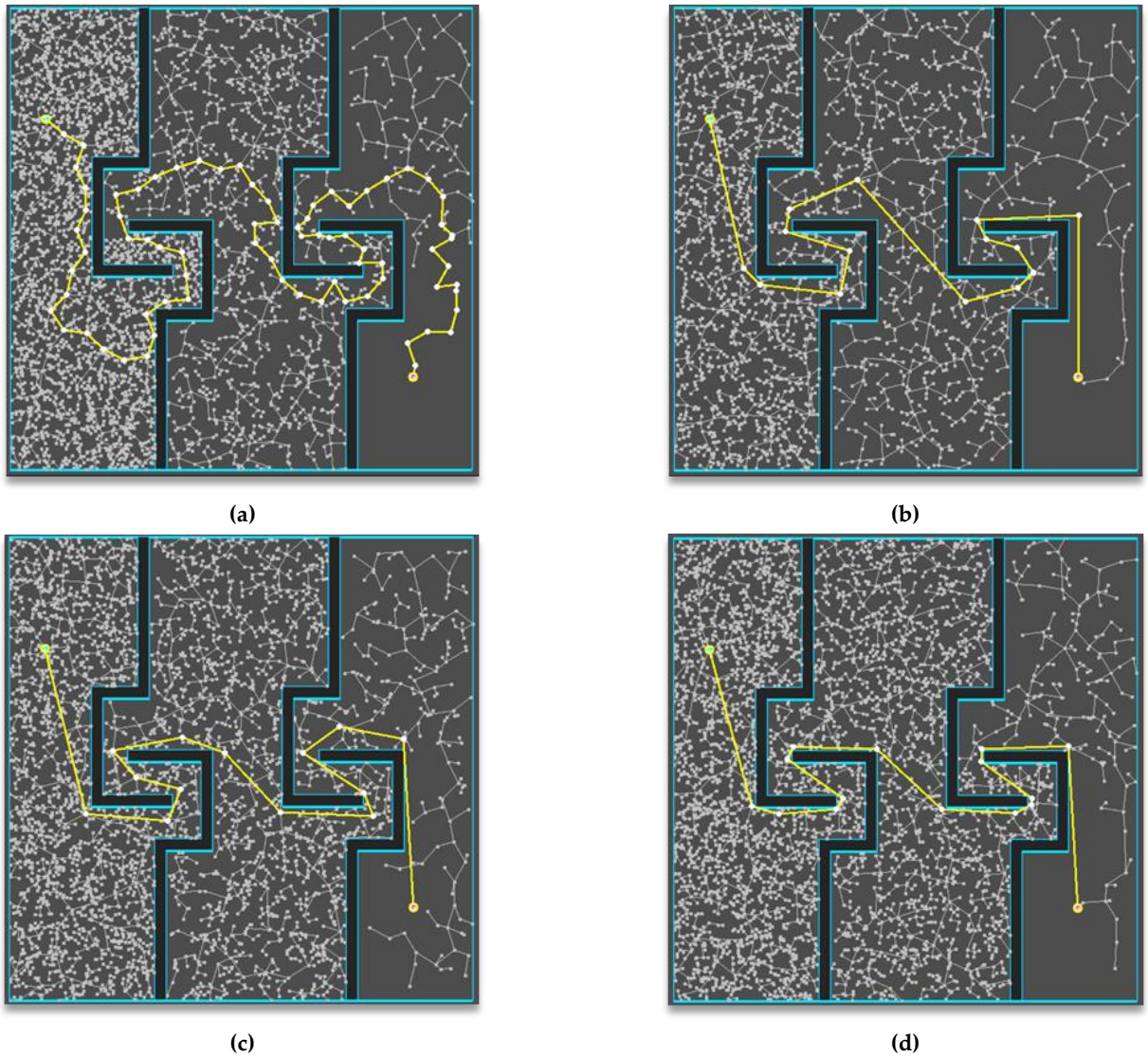


Figure 9. Experimental results of map 1: (a) RRT; (b) Proposed method (ε: 50px) applied; (c) Proposed method (ε: 30px) applied; (d) Proposed method (ε: 10px) applied.

The path planning results as shown in Figure 9 for map 1 among the specified environment maps for each algorithm. In terms of appearance, the one applied with the Post Triangular Processing of Midpoint Interpolation method (ε: 10px) seems to have a smooth slope compared to other algorithms, and the path length with the Post Triangular Processing of Midpoint Interpolation method (ε: 10px) seems to be the shortest.

Table 2. Experimental results of map 1 (The parentheses on the right of each number (average of repeated 100 times) are relative ratios to RRT results (values less than 1 are counted as 1)).

Performance	**RRT**	**Proposed method (ε: 50px)**	**Proposed method (ε: 30px)**	**Proposed method (ε: 10px)**
Path length (px)	1944 *(100%)*	1403 *(72%)*	1325 *(68%)*	1224 *(62%)*
Planning time (ms)	697 *(100%)*	698 *(100%)*	698 *(100%)*	698 *(100%)*

The path planning results(average value after repeated trial 100 times) as shown in Table 2 for map 1 among the specified environment maps for each algorithm. The path length applying the Post Triangular Processing of Midpoint Interpolation method (ε: 10px) becomes 62% (1224/1994(%)) compared to the RRT, which is the shortest compared to other algorithms, and the planning time is all similar.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 12 July 2021 doi:10.20944/preprints202107.0254.v1

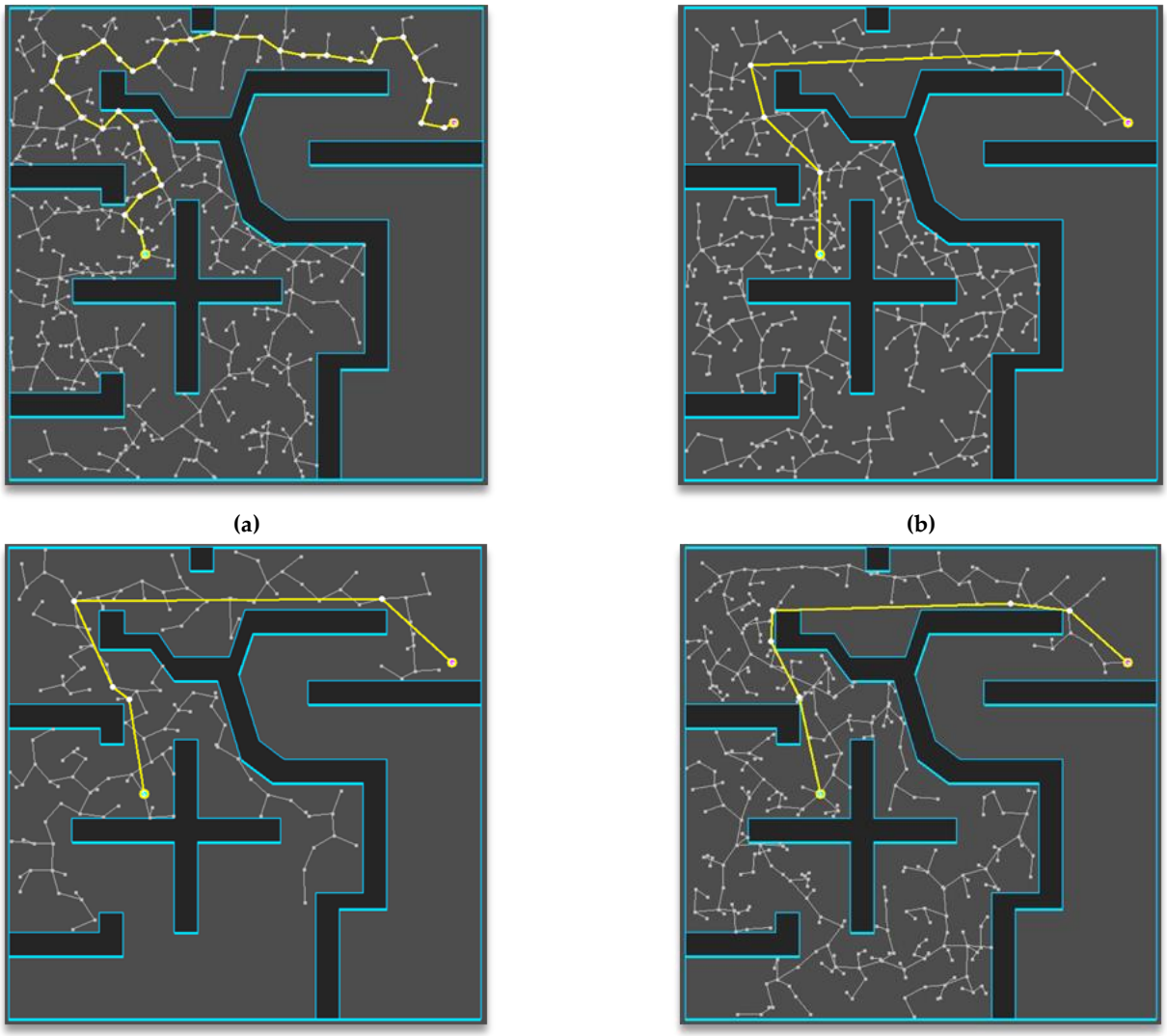


Figure 10. Experimental results of map 2: (a) RRT; (b) Proposed method (ε: 50px) applied; (c) Proposed method (ε: 30px) applied; (d) Proposed method (ε: 10px) applied.

The path planning results as shown in Figure 10 for map 2 among the specified environment maps for each algorithm. In terms of appearance, the one applied with the Post Triangular Processing of Midpoint Interpolation method (ε: 10px) seems to have a smooth slope compared to other algorithms, and the path length with the Post Triangular Processing of Midpoint Interpolation method (ε: 10px) seems to be the shortest.

Table 3. Experimental results of map 2 (The parentheses on the right of each number (average of repeated 100 times) are relative ratios to RRT results (values less than 1 are counted as 1)).

Performance	**RRT**	**Proposed method (ε: 50px)**	**Proposed method (ε: 30px)**	**Proposed method (ε: 10px)**
Path length (px)	986 *(100%)*	780 *(79%)*	752 *(76%)*	730 *(74%)*
Planning time (ms)	10 *(100%)*	10 *(100%)*	11 *(110%)*	11 *(110%)*

The path planning results(average value after repeated trial 100 times) as shown in Table 3 for map 2 among the specified environment maps for each algorithm. The path length applied with the Post Triangular Processing of Midpoint Interpolation method (ε: 10px) becomes 74% (730/986(%)) compared to the RRT, which is the shortest compared to other algorithms, and the planning time is similar to the RRT algorithm when the Post

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 12 July 2021 doi:10.20944/preprints202107.0254.v1

Triangular Processing of Midpoint Interpolation method (ε: 50px) is applied. However, the absolute time difference is 1ms, and the difference seems to be large when the method is applied because it is an environment that requires less planning time.

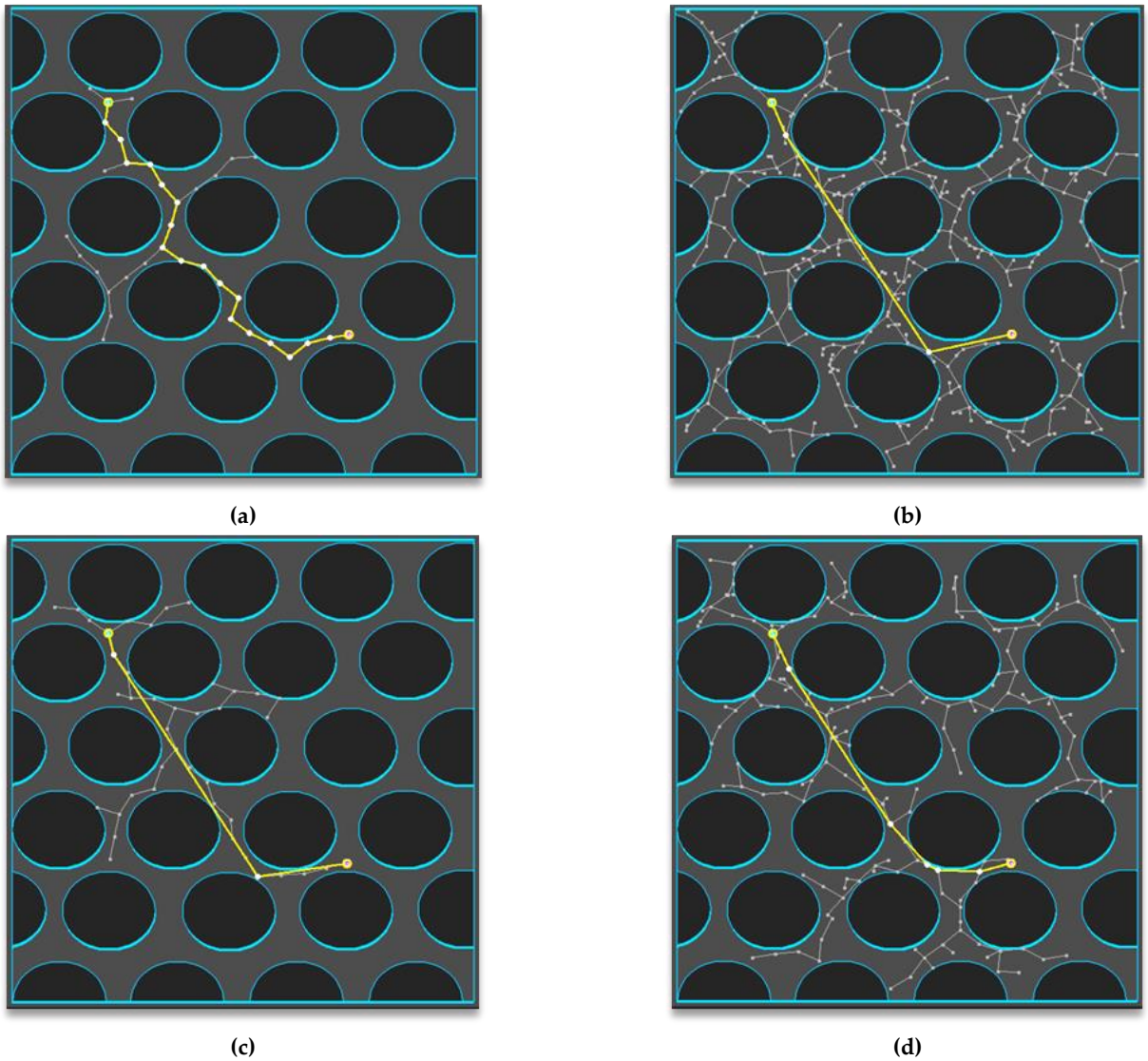


Figure 11. Experimental results of map 3: (a) RRT; (b) Proposed method (ε: 50px) applied; (c) Proposed method (ε: 30px) applied; (d) Proposed method (ε: 10px) applied.

The path planning results as shown in Figure 11 for map 3 among the specified environment maps for each algorithm. In terms of appearance, the one applied with the Post Triangular Processing of Midpoint Interpolation method (ε: 10px) seems to have a smooth slope compared to other algorithms, and the path length with the Post Triangular Processing of Midpoint Interpolation method (ε: 10px) seems to be the shortest.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 12 July 2021 doi:10.20944/preprints202107.0254.v1

Table 4. Experimental results of map 3 (The parentheses on the right of each number (average of repeated 100 times) are relative ratios to RRT results (values less than 1 are counted as 1)).

Performance	RRT	Proposed method (ε: 50px)	Proposed method (ε: 30px)	Proposed method (ε: 10px)
Path length (px)	613 *(100%)*	535 *(87%)*	512 *(83%)*	505 *(82%)*
Planning time (ms)	6 *(100%)*	7 *(116%)*	8 *(133%)*	8 *(133%)*

The path planning results(average value after repeated trial 100 times) as shown in Table 4 for map 3 among the specified environment maps for each algorithm. The path length applying the Post Triangular Processing of Midpoint Interpolation method (ε: 10px) becomes 82% (505/613(%)) compared to the RRT, which is the shortest compared to other algorithms, and the planning time is the most similar to that of the Post Triangular Processing of Midpoint Interpolation method (ε: 10px) as it takes 16% more than the RRT algorithm. However, the absolute time difference is 2ms, and since it is an environment that requires less planning time, the difference appears to be large when the method is applied.

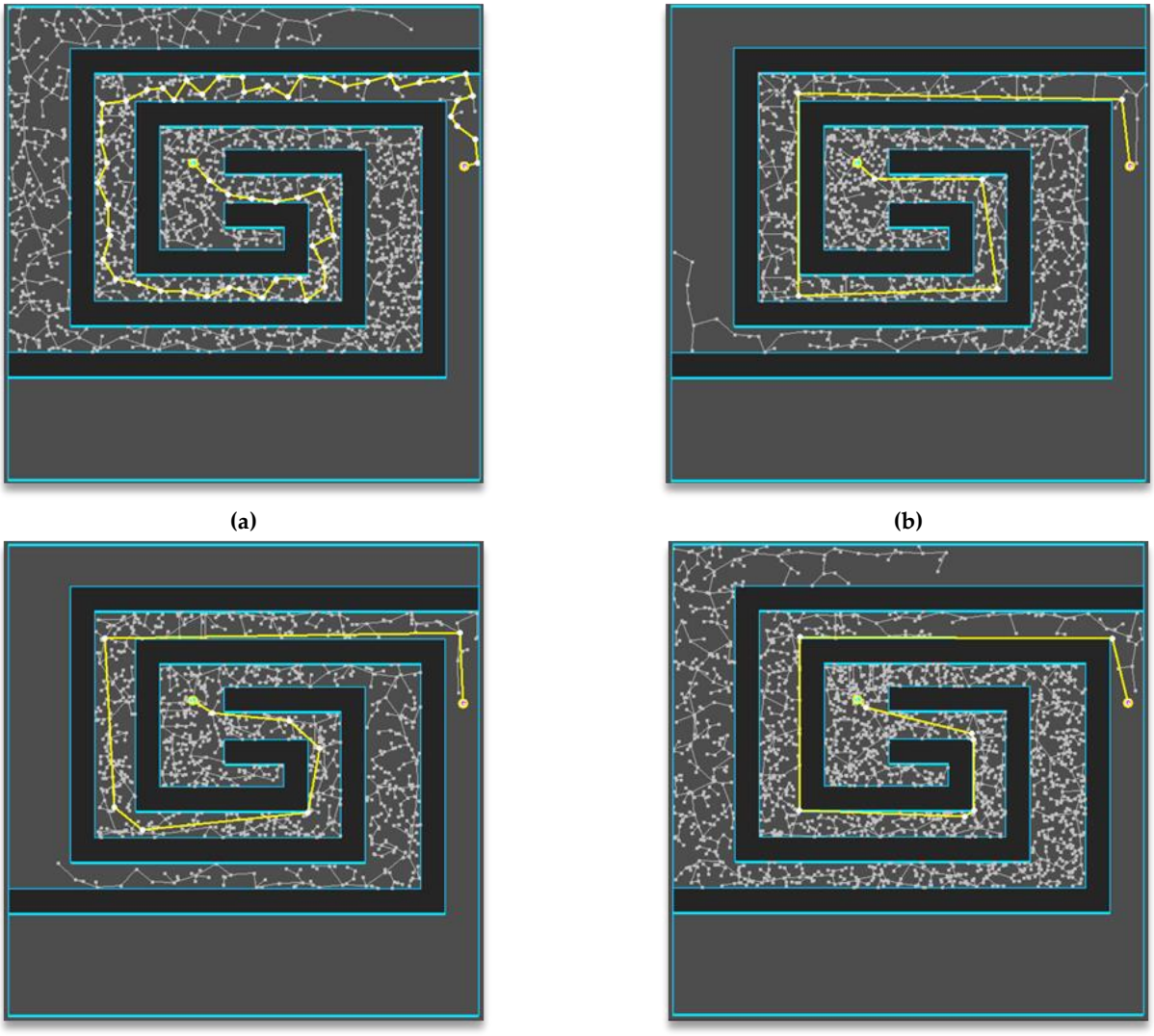


Figure 12. Experimental results of map 4: (a) RRT; (b) Proposed method (ε: 50px) applied; (c) Proposed method (ε: 30px) applied; (d) Proposed method (ε: 10px) applied.

The path planning results as shown in Figure 12 for map 4 among the specified environment maps for each algorithm. In terms of appearance, the one applied with the Post

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 12 July 2021 doi:10.20944/preprints202107.0254.v1

Triangular Processing of Midpoint Interpolation method (ε: 30px) seems to have a smooth slope compared to other algorithms, and the path length with the Post Triangular Processing of Midpoint Interpolation method (ε: 10px) seems to be the shortest.

Table 5. Experimental results of map 4 (The parentheses on the right of each number (average of repeated 100 times) are relative ratios to RRT results (values less than 1 are counted as 1)).

Performance	**RRT**	**Proposed method (ε: 50px)**	**Proposed method (ε: 30px)**	**Proposed method (ε: 10px)**
Path length (px)	1536 *(100%)*	1284 *(83%)*	1261 *(82%)*	1190 *(77%)*
Planning time (ms)	1752 *(100%)*	1753 *(100%)*	1753 *(100%)*	1753 *(100%)*

The path planning results(average value after repeated trial 100 times) as shown in Table 5 for map 4 among the specified environment maps for each algorithm. The path length applying the Post Triangular Processing of Midpoint Interpolation method (ε: 10px) becomes 77% (1190/1536(%)) compared to RRT, which is the shortest compared to other algorithms, and the planning time is similar for all.

Consequently, the Post Triangular Processing of Midpoint Interpolation method (ε: 10px) performed well in the path length aspect for all maps, demonstrating that the proposed method is efficient in terms of optimality.

5. Conclusion

In this research, the "Post Triangular Processing of Midpoint Interpolation" method minimized the planning time and shorter path length of the sampling-based algorithm.

The proposed Post Triangular Processing of Midpoint Interpolation method made a more to closer optimal path and somewhat solves the sharp path problem through the interpolation process. furthermore, all path planning algorithms that plan a locally piecewise linear path could apply the proposed algorithm. This strength has the significance that it can be applied not only to the algorithm presented in this paper but also to various path planning algorithms. The proposed method validated the performance after applied to the RRT algorithm and its path planning results through simulation. It was verified that the path length was shortened by 18-38% (average 26%) depending on the threshold ε when applied to the RRT algorithm in the four different environment maps. As a result, the RRT algorithm applying the proposed Post Triangular Processing of Midpoint Interpolation method showed a more optimal path.

Author Contributions: Idea and conceptualization: J.-G.K., and J.-W.J.; Methodology: J.-G.K., and J.-W.J.; Software: J.-G.K., and J.-W.J.; Experiment: J.-G.K., and J.-W.J.; Validation: J.-G.K., Y.-S.C., and J.-W.J.; Investigation: J.-G.K. and J.-W.J.; Resources: J.-G.K. and J.-W.J.; Writing: J.-G.K., Y.-S.C., and J.-W.J.; Visualization: J.-G.K. and J.-W.J.; Project administration: J.-W.J. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by the National Research Foundation of Korea(NRF) grant funded by the Korea government(MSIT) (No. 2020R1F1A1074974), the Ministry of Trade, Industry and Energy(MOTIE) and Korea Institute for Advancement of Technology(KIAT) through the International Cooperative R&D program. (Project No. P0016096), the AURI(Korea Association of University, Research institute and Industry) grant funded by the Korea Government(MSS : Ministry of SMEs and Startups). (No.S3041234, HRD program for Enterprise linkages R&D) and the MSIT(Ministry of Science and ICT), Korea, under the ITRC(Information Technology Research Center) support program(IITP-2021-2020-0-01789) supervised by the IITP(Institute for Information & Communications Technology Planning & Evaluation).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

 doi:10.20944/preprints202107.0254.v1

References

1. Schwab, K. *The Fourth Industrial Revolution*; Currency: 2017.
2. LaValle, S.M. Kuffner, J.J., Jr. Randomized kinodynamic planning. *International Journal of Robotics Research* **2001**, *20*. 378–400.
3. Kuffner, J.J., Jr. LaValle, S.M. RRT-connect: An Efficient Approach to Single-query Path Planning. In Proceedings of the IEEE International Conference on Robotics and Automation, San Francisco, CA, USA, 24–28 April 2000 Volume 2, pp. 995–1001.
4. Kang, J.-G. Lim, D.-W. Choi, Y.-S. Jang, W.-J. Jung, J.-W. Improved RRT-Connect Algorithm Based on Triangular Inequality for Robot Path Planning. *Sensors* **2021**, *21*, 333-364.
5. Roy, D. Visibility graph based spatial path planning of robots using configuration space algorithms. *Robotics and Autonomous System* **2009**, *24*, 1–9.
6. Katevas, N.I. Tzafestas, S.G. Pnevmatikatos, C.G. The approximate cell decomposition with local node refinement global path planning algorithm: Path nodes refinement and curve parametric interpolation. *Journal of Intelligent and Robotic Systems* **1998**, *22*, 289–314.
7. Warren, C.W. Global Path Planning using Artificial Potential Fields. In Proceedings of the International Conference on Robotics and Automation, Scottsdale, AZ, USA, 14–19 May 1989 Volume 1, pp. 316–321.
8. LaValle, S.M. *Rapidly-Exploring Random Trees: A New Tool for Path Planning* Springer: London, UK, 1998.
9. Jung, J.-W. So, B.-C. Kang, J.-G. Lim, D.-W. Son, Y. Expanded Douglas–Peucker polygonal approximation and opposite angle based exact cell decomposition for path planning with curvilinear obstacles. *Applied Sciences* **2019**, *9*, 638-654.
10. Jung, J.-W., So, B.-C., Kang, J.-G., Jang, W.-J. Circumscribed Douglas-Peucker Polygonal Approximation for Curvilinear Obstacle Representation. In Proceedings of the IEEE 2019 7th International Conference on Robot Intelligence Technology and Applications (RiTA), Daejeon, Korea, November 2019 pp. 237-241.
11. Kwon, J. Choi, K. Kinodynamic Model Identification: A Unified Geometric Approach. *IEEE Transactions on Robotics* **2021**. (Early Access)
12. Han, J. Mobile robot path planning with surrounding point set and path improvement. *Applied Soft Computing* **2017**, *57*, 35–47.