Hindawi Publishing Corporation
Mathematical Problems in Engineering
Volume 2016, Article ID 3973467, 19 pages
http://dx.doi.org/10.1155/2016/3973467

Research Article

Adaptation Algorithm of Geometric Graphs for Robot Motion Planning in Dynamic Environments

Jae-Han Park, Ji-Hun Bae, and Moon-Hong Baeg

Robotics R&D Group, Korea Institute of Industrial Technology (KITECH), Ansan, Republic of Korea

Correspondence should be addressed to Jae-Han Park; hans1024@kitech.re.kr

Received 22 September 2015; Accepted 6 December 2015

Academic Editor: Hou-Sheng Su

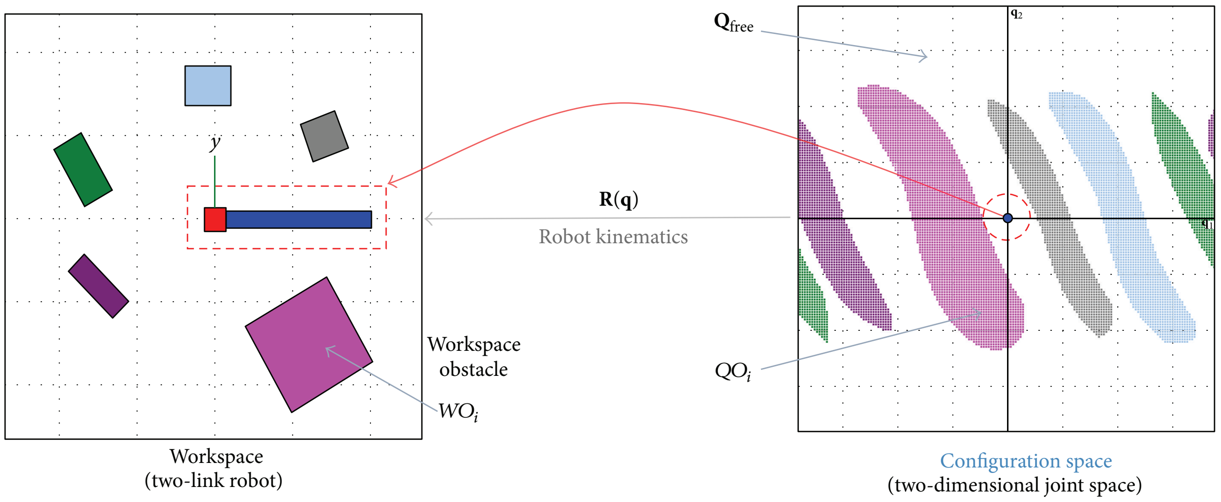
This study proposes an adaptive graph algorithm for collision-free motion planning of articulated robots in dynamic environments. For this purpose, deformations of the configuration space were analyzed according to the changes of the workspace using various simulations. Subsequently, we adopted the principles of gas motion dynamics in our adaptation algorithm to address the issue of the deformation of the configuration space. The proposed algorithm has an adaptation mechanism based on expansive repulsion and sensory repulsion, and it can be performed to provide the entire adaptation using distributed processing. The simulation results confirmed that the proposed method allows the adaptation of the roadmap graph to changes of the configuration space.

1. Introduction

Robot motion planning in dynamic environments has been regarded by researchers as a challenging problem. The motion planning problem, particularly for articulated robots, is known to be difficult because of the geometrical and algebraic complexities resulting from the increase in the degrees of freedom (DOFs) of the robot. The representative statement of this motion planning problem is known as the configuration space formulation. The key function of a configuration space is to represent the robot as a point in an appropriate space and map the obstacles in this space. This mapping transforms the problem of planning the motion of a dimensioned object into a planning problem for the motion of a point. It also makes the motion constraints of the robot more explicit [1–3].

For collision-free manipulation, especially in dynamic environments, planning algorithms must overcome additional challenging problems [4, 5]. The first problem is the unpredictable radical deformation of the configuration space owing to the change of the workspace. Various simulations have shown that the extent of the changes in the shape of the obstacles in the configuration space is closely related to the distance between the robot and the obstacles in the workspace. As a result, small changes in the workspace could cause extreme deformation of the configuration space when the body of the robot approaches the obstacles in the workspace. Thus, it is essential to manage these radical deformations of the configuration space. The second problem is the large processing time required for reconstructing the configuration space, because articulated robots generally have high-DOF kinematic structures.

Although extensive work has been performed on robot motion planning, most of the widely used algorithms utilize sampling-based methodologies. These sampling-based methods construct geometric graphs in collision-free configuration spaces to obtain solutions to motion planning problems; moreover, they employ a variety of strategies for generating samples and connecting the samples with paths. Sampling-based algorithms, such as *probabilistic roadmaps* (*PRM*) [6] and *rapidly exploring random trees* (*RRT*) [7], have been shown to perform well in practice [8–11] and have theoretical guarantees, including probabilistic completeness [12, 13]. Recently, with the increasing requirements for the use of robots in dynamic or time-varying environments, the extension of the motion planning problem to dynamic environments has emerged as an important issue. The application of sampling-based planning algorithms is mostly limited to static environments because they cannot guarantee fast responses to the processing time constraints and do not have the computing abilities to manage the deformation of the



y
Workspace obstacle
WO_i
Workspace (two-link robot)
$\mathbf{R}(\mathbf{q})$
Robot kinematics
$\mathbf{Q}_{\text{free}}$
QO_i
q_2
q_1
Configuration space (two-dimensional joint space)

FIGURE 1: Mapping relations between the workspace and the configuration space.

configuration space. Some studies have attempted to extend sampling-based methods to dynamic environments by incorporating the notion of time as an additional dimension in the configuration space [14, 15]. Other approaches are based on the replanning method, which interleaves planning with execution, in order to study unpredictable environments [5, 16, 17]. Nevertheless, it may be difficult for these algorithms to handle dynamic constraints in real applications because the amount of computation can increase exponentially with the extension of the configuration space dimensions.

In this paper, we propose an algorithm for geometric graphs that can adapt to a changeable configuration space topology. We contrived this algorithm via inspiration from recent investigations, such as multiagent systems [18–20], consensus control [21–24], and distributed computing [25–27]. In the proposed adaptation algorithm, each node moves in a free configuration space independently, similar to the intelligent agent of multiagent systems, by detecting the deformation of the configuration space; then, the graph updates its topology with respect to the changed node information. It was confirmed that the graph can perform the entire adaptation to the change of configuration space using the adaptation algorithm. Moreover, the algorithm is based on the distributed computation algorithm; hence, it could be executed in parallel by many-core systems.

2. Dynamic Environments

2.1. Configuration Space Formulation. The *configuration space* $\mathbf{Q}$ is the space of all possible configurations of the system; a point $\mathbf{q}$ in this space fully describes the volume of the robot $\mathbf{R}(\mathbf{q})$ in the workspace $\mathbf{W}$ [1, 2]. Figure 1 shows the mapping relations between the workspace and the configuration space for a two-link manipulator. In articulated robots, the configuration space $\mathbf{Q}$ generally represents the joint space, which is a compact subset of the n-dimensional Euclidean space $\mathbf{R}^n$, where n is the DOF of the robot. The configuration space can be partitioned into two regions, one representing the collision-free space $\mathbf{Q}_{\text{free}}$ and the other representing the region $\mathbf{Q}_{\text{collide}}$, in which the robot collides with the workspace obstacles. Every obstacle WO_i in the workspace is mapped as a *configuration space obstacle* QO_i:

$$QO_i = \{\mathbf{q} \in \mathbf{Q} \mid \mathbf{R}(\mathbf{q}) \cap WO_i \neq \phi\}. \tag{1}$$

The union of all the configuration space obstacles

$$\mathbf{Q}_{\text{collide}} = \bigcup_{i=1}^{m} QO_i \tag{2}$$

is called the configuration obstacle region, and the set

$$\begin{aligned} \mathbf{Q}_{\text{free}} &= \mathbf{Q} \setminus \mathbf{Q}_{\text{collide}} = \mathbf{Q} \setminus \bigcup_{i=1}^{m} QO_i \\ &= \left\{\mathbf{q} \in \mathbf{Q} \mid \mathbf{R}(\mathbf{q}) \cap \left(\bigcup_{i=1}^{m} WO_i\right) = \phi\right\} \end{aligned} \tag{3}$$

is called the *free space* or *free configuration space* $\mathbf{Q}_{\text{free}}$. Any configuration $\mathbf{q}$ in $\mathbf{Q}_{\text{free}}$ is called a free configuration, indicating that the robot does not collide with any obstacles in the workspace.

2.2. Deformation of Configuration Space. Dynamic environments are workspaces in which various changes can occur. Changes in the workspace lead to deformation of the configuration space; this relationship between the two spaces is determined by the robot kinematics equation $\mathbf{R}(\mathbf{q})$. In general, the relationship between the workspace and the configuration space is nonlinear because almost all robot kinematics equations are described by nonlinear functions. Thus, the relations between the two spaces are difficult to describe and analyze mathematically. By observing various deformations of the configuration space, we can categorize them into *scaling*, *translation*, and *merging deformations*.

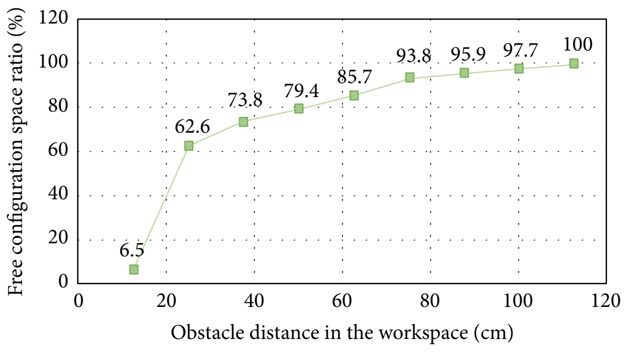


Figure 2: Free configuration space ratio with respect to the distance of the robot base from an obstacle in the workspace.

2.2.1. Scaling Deformation. Obstacles in the configuration space change in size according to the change of the workspace. The size of an obstacle is determined by the distance between the coordinate of the robot base and that of the obstacle in the workspace. Figure 2 shows the ratio of free configuration space in the two-dimensional workspace and in the configuration space as a function of the distance between the obstacle and the robot base in the workspace. Obstacles placed outside the working area of the robot do not exist in the configuration space. However, if an obstacle moves toward the robot base within the robot working area, the ratio of free configuration space is gradually decreased. In particular, the ratio of configuration space rapidly decreases when the obstacle and the robot base are close. In Figure 2, there is a radical decrease of 70% to 10% in the ratio of free configuration space when the distance decreases from 30 cm to 10 cm. These results confirm that the change of distance between the obstacle and the robot base changes the scale of the obstacle in the configuration space; in particular, when the distance is small, the change of scale could be radical.

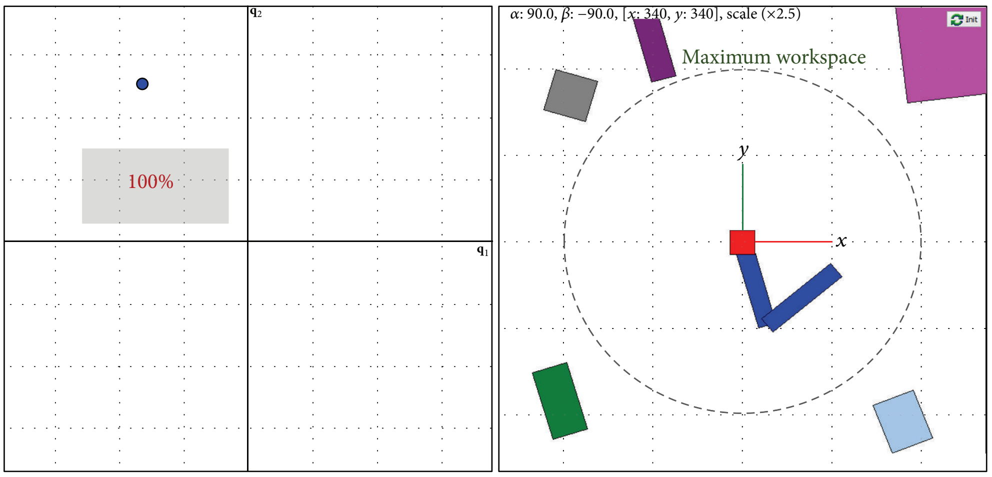
Figure 3 shows the deformation of the configuration space for varying obstacle distances in the workspace. The left and right images show the configuration space and the corresponding workspace, respectively. In Figure 3(a), the entire space is the free configuration space because all objects are placed outside the working area of the robot. Figure 3(b) shows the introduction of an obstacle into the configuration space caused by the movement of the obstacle in the workspace. As a result, the ratio of free configuration space decreases to 93.3%. Furthermore, Figure 3(c) shows the more radical deformation of the configuration space by the approach of the obstacle to the robot base. Note that while the movement distances from Figures 3(a) to 3(b) and from Figures 3(b) to 3(c) are similar, the changes of the obstacle size and the free configuration space ratio are larger in Figure 3(c).

2.2.2. Translation Deformation. The position of an obstacle in the configuration space is determined by the direction of the obstacle in the workspace with respect to the robot base coordinate. Thus, if the obstacle position changes while preserving the distance from the robot base in the workspace, a translation motion of the obstacle occurs in the configuration space. Figure 4 shows a simplified example of the result of the translation motion in the configuration space. In Figure 4(a), the obstacle is placed at 0° with respect to the robot base coordinate in the workspace; in the configuration space, the obstacle is located at the origin. If the obstacle moves to the direction of 90° while maintaining the same distance, there is a translation motion to the right direction of the obstacle in the configuration space without a notable change of size, as in Figure 4(b). In contrast, as shown in Figure 4(c), if the obstacle moves to the direction of -90°, there is a translation motion to the left of the obstacle in the configuration space. From these results, it is evident that the change of direction of the object with respect to the robot base coordinate in the workspace produces translation motions of the obstacle position in the configuration space.

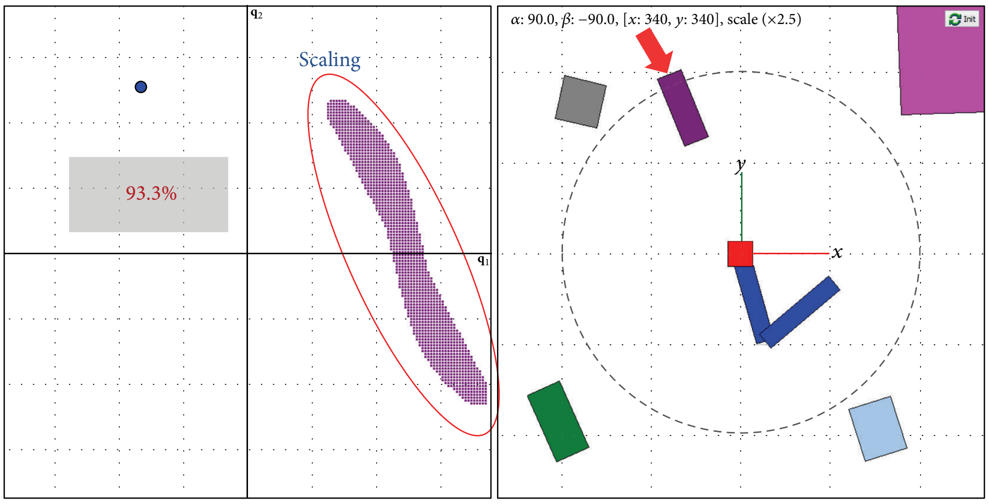
2.2.3. Merging Deformation. Obstacles in the configuration space consist of the union set of all obstacles converted from the workspace. In general, there is no merging phenomenon in the workspace because changes in the workspace are rigid body motions without a scale change. However, in the configuration space, there can be a merging phenomenon with respect to the change of scale and position of the obstacles. Figure 5 shows a merging process in the configuration space according to the movements of multiple objects in the workspace. The merging phenomenon can be more prominent if the objects and the robot are close; this complicates the prediction of the deformation of the configuration space.

In the mobile manipulation problem, which is a challenging task in robot motion planning, the robot position and direction change by the movement of the mobile robot. If the robot moves by turning, the obstacles near the robot move by turning in the opposite direction in the workspace; thus, it can be predicted that all these obstacles move in the same direction. In addition, if the robot base coordinate moves by the motion of the mobile base, some obstacles will be closer, and some will be farther from the robot base. In this case, some obstacles in the configuration space will be expanded, and some will be contracted, according to their distances. Therefore, in mobile manipulation problems involving the movement of the mobile base, the environments become dynamic even when the workspace is a fixed environment. Figure 6 shows the deformation of the configuration space when the robot base moves in different directions. As shown in Figures 6(b) and 6(c), closer obstacles in the workspace undergo expansion, and farther obstacles undergo contraction; furthermore, some merging occurs by the expansions of an obstacle.

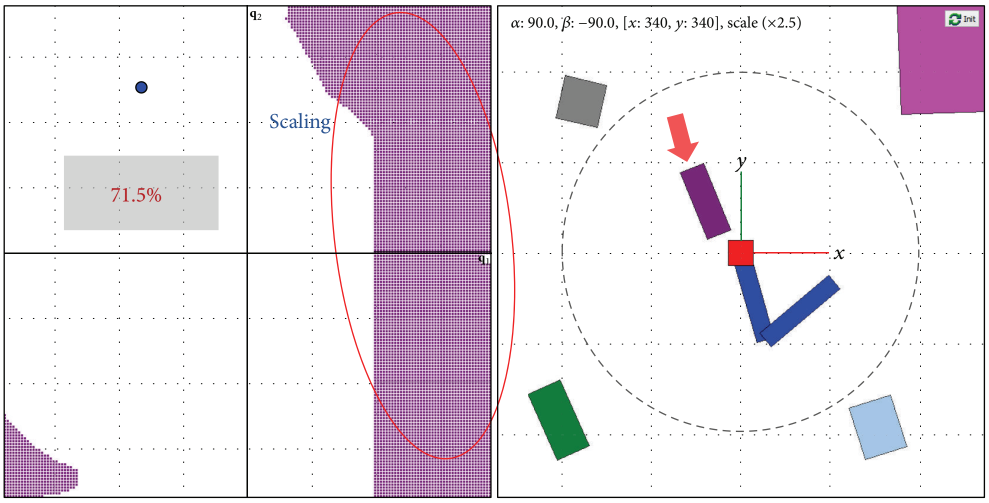
In dynamic environments, the deformations of the configuration space are combinations of the three nonlinear deformations of *scaling*, *translation*, and *merging*. Sometimes, small changes in the workspace can generate considerable deformations in the configuration space; the combined changes from multiple obstacle movements in the workspace appear as a complex deformation in the configuration space. These radical deformations of the configuration space further



(a)

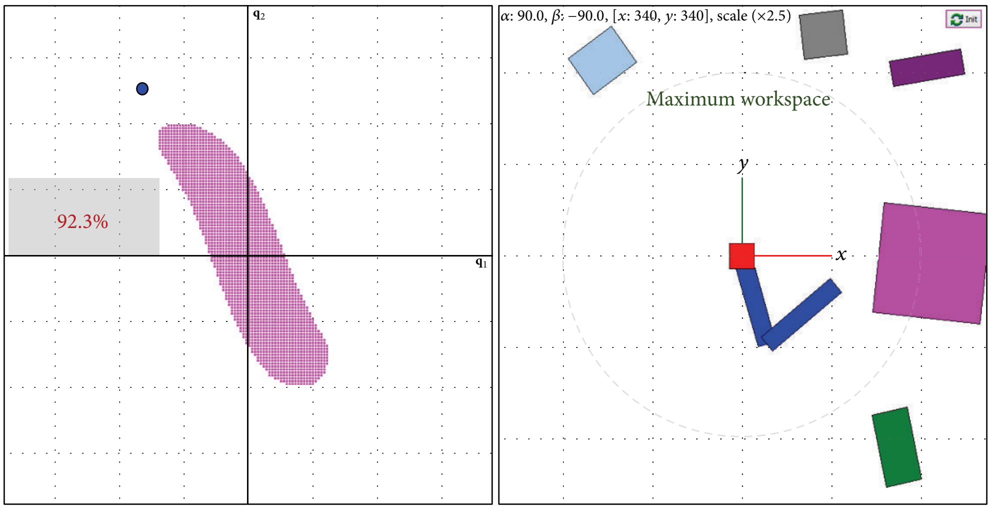


(b)

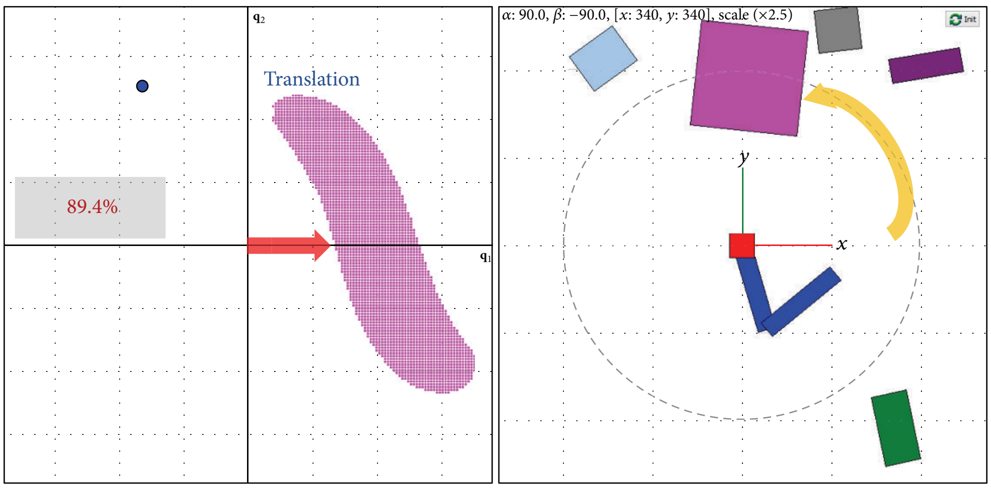


(c)

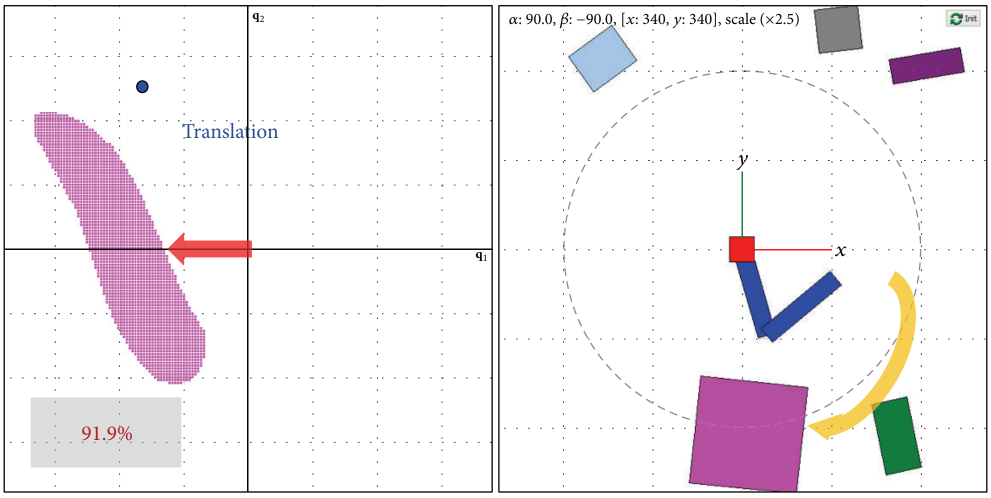
FIGURE 3: Scaling deformation of the configuration space caused by the movement of a workspace object. (a) No object in the configuration space; (b) appearance of the configuration space object; (c) expansion of the configuration space object by movement toward the robot in the workspace.



(a)

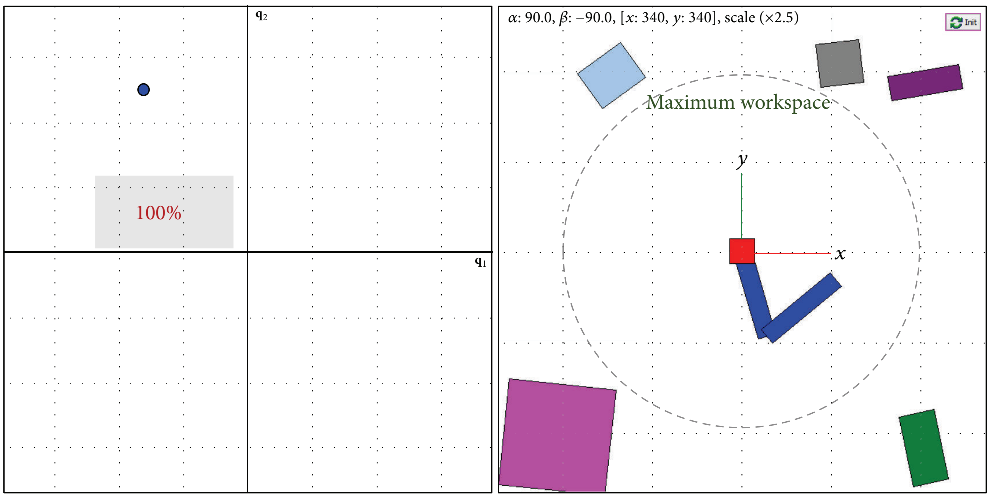


(b)

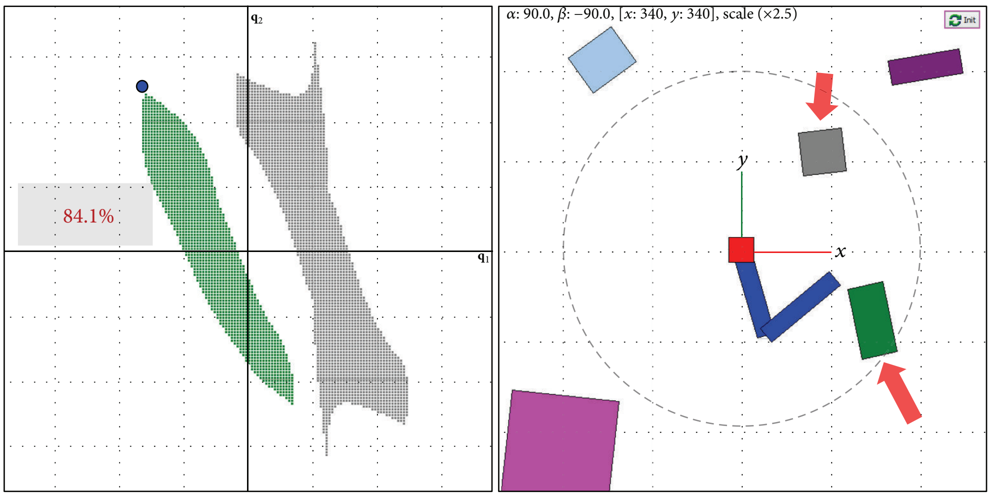


(c)

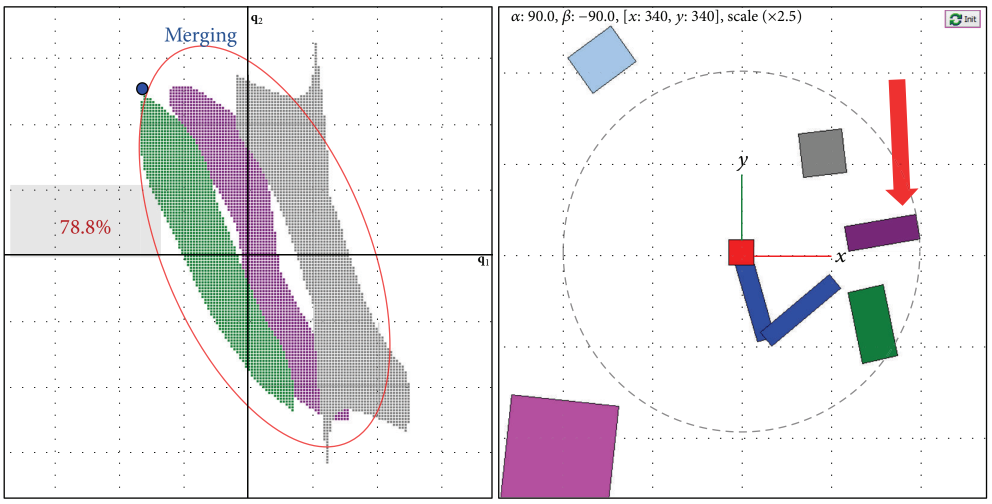
FIGURE 4: Translation of a configuration space object generated by a moving workspace object. (a) Object at 0°; (b) positive translation of the configuration space object (object at 90°); (c) negative translation of the configuration space object (object at −90°).



(a)



(b)



(c)

Figure 5: Merging of configuration space objects caused by moving workspace objects. (a) No configuration space object; (b) appearance of configuration space objects; (c) merging of configuration space objects by movement in the workspace.

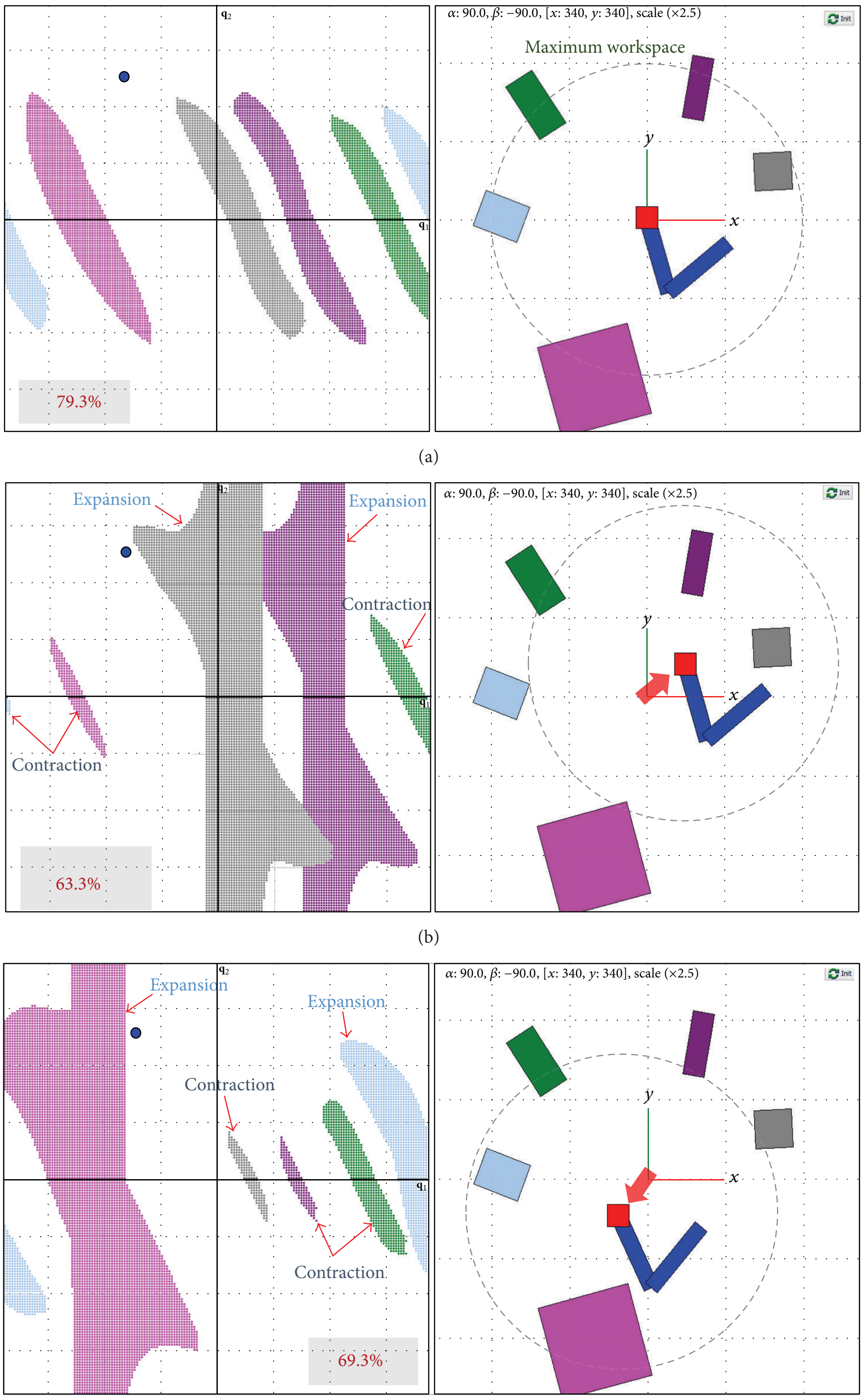

q2
q1
79.3%
α: 90.0, β: −90.0, [x: 340, y: 340], scale (×2.5)
Init
Maximum workspace
y
x
(a)
Expansion
Expansion
Contraction
Contraction
63.3%
α: 90.0, β: −90.0, [x: 340, y: 340], scale (×2.5)
(b)
Expansion
Expansion
Contraction
Contraction
69.3%
α: 90.0, β: −90.0, [x: 340, y: 340], scale (×2.5)
(c)

Figure 6: Expansion and contraction of configuration space objects by the movement of the robot base. (a) Robot placed at (0, 0); (b) robot moves to (24, 23); (c) robot moves to (−21, −23).

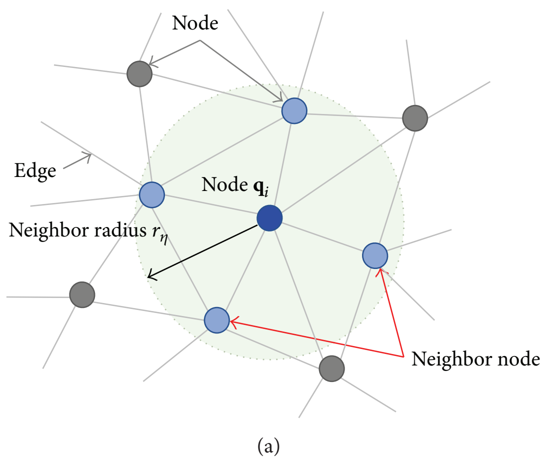


Figure 7: Neighbor node and sensing node. (a) Neighbor node for neighbor radius r_η; (b) sensing nodes around node $\mathbf{q}_i$ for sensing radius r_s.

complicate the motion planning problem in dynamic environments.

2.3. Processing Time for Modified Configuration Space. Another important problem that complicates motion planning in dynamic environments is the processing time required for the reconstruction of a modified configuration space [28, 29]. Three-dimensional sensors in the workspace can detect free ranges that are not occupied by obstacles. However, in the configuration space of articulated robots, there are no methods of detecting free ranges that are a set of collision-free motions. A motion of the robot in the workspace is expressed as a point in the configuration space; a collision test result for a robot motion in the workspace provides only the binary information of whether collision occurs or not for a point of the configuration space. Thus, the entire free configuration space can be constructed by the mapping of the binary collision test results for all configuration space points. However, this mapping is impossible in actual problems; hence, the free configuration space is represented with abstracted graphs that consist of nodes and edges.

Most robot motion planning algorithms have been derived from sampling-based algorithms, such as PRM and RRT; these methods describe the complex configuration space with geometric graphs through the construction of nodes and edges. In the PRM method, after a learning phase in which a graph for the entire configuration space is constructed, the optimal path between a given starting point and a goal point can be obtained from the learned graph. In contrast, the RRT method identifies a reachable path by expanding the graph using a tree-type approach. In the PRM method, the obtained path is the optimal path based on the previously constructed graph; the PRM method cannot be applied in dynamic environments because the graph must be reconstructed if the configuration space changes. Because the RRT method constructs a new graph for every path-finding query, it can be applied in changeable environments. However, the path obtained by the RRT method is not optimal

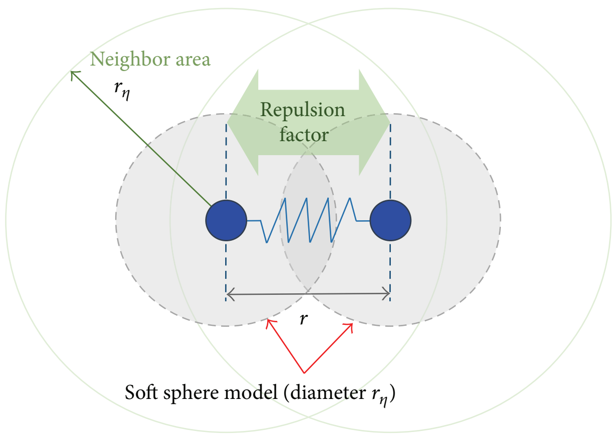


Figure 8: Expansive repulsion factor between two nodes in a neighbor area.

and requires considerable time for constructing the new graph. In addition, sampling-based algorithms perform tests for the construction of nodes and edges; particularly, tests for edge construction require significantly more time because collisions for line segments are determined by tests performed for numerous points on the edge. With the purpose of resolving these problems, a faster edge-test algorithm that uses the upper bound of robot motion has been proposed [30, 31]. Recently, with the development of parallel computing technologies [32–34], some approaches use GPU technology to improve processing time of robot collision checking [35–37]. Nevertheless, the reconstruction of updated graphs in real time to represent changes in dynamic environments remains a difficult problem.

3. Adaptive Roadmap Algorithm

In this study, we attempt to solve the problem of motion planning in dynamic environments. For this purpose, we propose an adaptation algorithm of geometric graphs, called

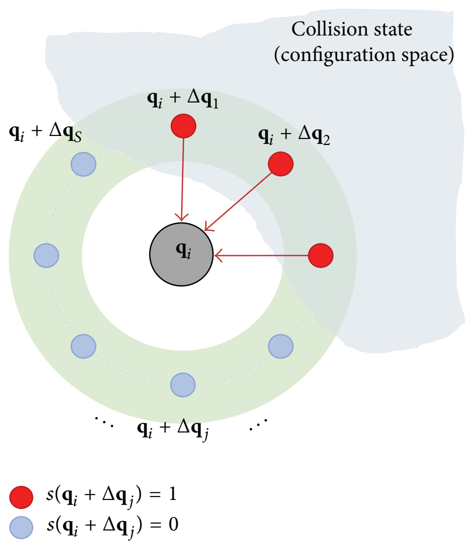

Collision state
(configuration space)
$\mathbf{q}_i + \Delta\mathbf{q}_1$
$\mathbf{q}_i + \Delta\mathbf{q}_2$
$\mathbf{q}_i + \Delta\mathbf{q}_S$
$\mathbf{q}_i$
$\mathbf{q}_i + \Delta\mathbf{q}_j$
● $s(\mathbf{q}_i + \Delta\mathbf{q}_j) = 1$
● $s(\mathbf{q}_i + \Delta\mathbf{q}_j) = 0$

Figure 9: Sensory repulsion factors generated by sensing nodes.

the *adaptive roadmap algorithm*, for collision-free motion planning in a deformable configuration space. The motion planning algorithm uses graphical information previously constructed by methods, such as PRM, to rapidly determine the optimal path from the learned roadmap. However, the learning or constructing process of the roadmap consumes much time; thus, it has been impossible to apply PRM algorithms to the motion planning problem in dynamic environments.

The proposed adaptive roadmap algorithm performs the adaptation process by changing the structure of the graph according to the deformation of the configuration space. To adjust the roadmap to changeable environments, each node in the graph moves automatically to the free configuration space. Then, the node updates its edge information with respect to the changed node information. These processes are distributed and performed in parallel for all nodes. Hence, this algorithm can adapt the roadmap to the change of the configuration space by changing the graph structure. In the proposed method, some new concepts, such as the neighbor node, neighbor radius, and sensing node, are introduced.

3.1. Definitions.

Definitions used are as follows:

(i) *Node* ($\mathbf{q}_i$): a sampled point in the n-dimensional free configuration space $\mathbf{Q}_{\text{free}}$ as a vertex of the graph;

(ii) *Edge* ($\mathbf{e}_{ij}$): a line segment that connects nodes $\mathbf{q}_i$ and $\mathbf{q}_j$ in the graph; the connection between nodes $\mathbf{q}_i$ and $\mathbf{q}_j$ is established when a route from $\mathbf{q}_i$ to $\mathbf{q}_j$ exists through a local planner or a navigation control function in the configuration space;

(iii) *Roadmap* ($\mathbf{G}$): an abstract geometric graph that represents the complex shape and structure of the free configuration space; a roadmap consists of a set of nodes $\mathbf{V}$ and a set of edges $\mathbf{E}$, where N is the number of nodes of the graph:

$$\mathbf{G} = \{\mathbf{V}, \mathbf{E} \mid \mathbf{V} = \{\mathbf{q}_1, \mathbf{q}_2, \ldots, \mathbf{q}_N\},\ \mathbf{E} = \{\mathbf{e}_{12}, \mathbf{e}_{15}, \ldots, \mathbf{e}_{Nk}\}\}; \tag{4}$$

(iv) *Neighbor node*: a node located near node $\mathbf{q}_i$, within a distance r_η; neighbor nodes can interact with node $\mathbf{q}_i$, and the set of neighbor nodes $\boldsymbol{\eta}_i$ can be represented as follows:

$$\boldsymbol{\eta}_i = \{\mathbf{q}_j \in \mathbf{V} \mid \|\mathbf{q}_j - \mathbf{q}_i\|_2 \le r_\eta,\ j \ne i\}; \tag{5}$$

(v) *Sensing node* ($\Delta\mathbf{q}_j \in \mathbf{R}^n$): a node that is sampled by Poisson-disk sampling [38] on $\partial\mathbf{B}_{r_s} = \{\mathbf{x} \in \mathbf{R}^n \mid \|\mathbf{x}\|_2 = r_s\}$, which is the surface of an n-dimensional hypersphere of radius r_s; sensing nodes detect the deformation of the configuration space around each node; the set of sensing nodes $\mathbf{S}$ can be represented as follows:

$$\mathbf{S} = \{\Delta\mathbf{x}_l \in \partial\mathbf{B}_{r_s} \mid l = 1, \ldots, n_s, \|\Delta\mathbf{x}_j - \Delta\mathbf{x}_k\|_2 > r_p,\ \forall j, k \in \{1, \ldots, n_s\}\}, \tag{6}$$

where r_p and n_s denote the Poisson-disk sampling radius and the number of sensing nodes, respectively.

Figure 7 illustrates the concepts of neighbor node and sensing node for a node $\mathbf{q}_i$.

3.2. Strategies for Adaptation of Roadmap.

The roadmap method is an abstract representation of the complex topology of the configuration space, with geometric graphs that consist of nodes and edges. In the previous section, we confirmed that a radical deformation of the configuration space can appear in dynamic environments. When changes occur in the workspace, they create various deformations of the configuration space; then, the constructed roadmap becomes useless. Thus, in this research, we propose an adaptation algorithm for roadmaps by adjusting the graph topology to the changed environments.

In the proposed adaptation algorithm, we adopted the motion dynamics of gases [39]. Gases are distributed evenly in the entire space of a container, regardless of its shape, by physical phenomena, such as diffusion and internal interactions, and collisions with the surface of the space boundary. The adaptation mechanism of the proposed algorithm is based on two repulsion factors: the expansive repulsion factor and the sensory repulsion factor. The expansive repulsion factor is a repulsive interaction between nodes in the roadmap. This factor can diffuse the distribution of nodes throughout the configuration space. The sensory repulsion factor represents the repulsion from sensing nodes around a node; these sensing nodes can detect the deformation of the configuration space. Because of the repulsion from the sensing nodes, nodes can move naturally toward the free area of the configuration space.

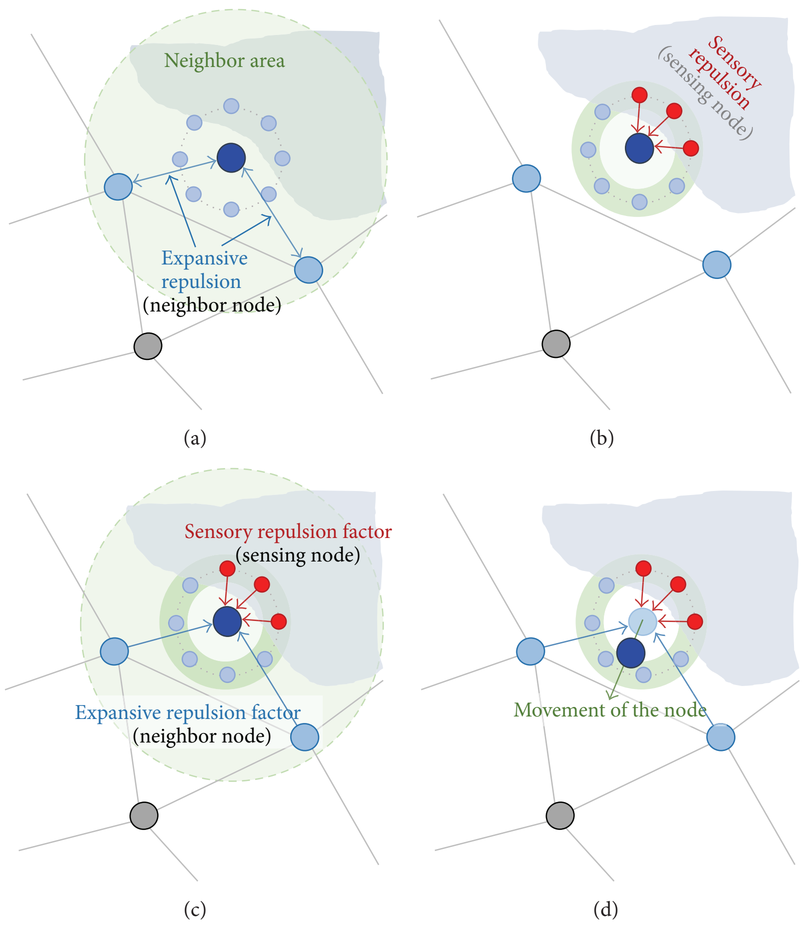


Figure 10: Interaction factors in a node. (a) Expansive repulsion from neighbor nodes; (b) sensory repulsion from sensing nodes that detect collisions; (c) total internal interaction factors; (d) movement of a node.

3.2.1. Expansive Repulsion Factor. The expansive repulsion factor expresses an interaction between nodes that can repel other nodes within the effective close area. Nodes in the effective close area are defined as neighbor nodes, and the threshold value that determines neighbor nodes is the neighbor radius r_η. Figure 8 shows the interactions between neighbor nodes in the neighbor area, which can be represented by the soft sphere model with diameter r_η.

If the distance r between two nodes is smaller than the neighbor radius r_η, expansive repulsion occurs. The strength of the interaction between the two nodes is determined by the repulsive potential function, which increases with decreasing node distance r. If the distance r is larger than r_η, the repulsive potential function has a very small value that cannot affect other nodes. In this study, we adopted the potential function proposed by Khatib [40]:

$$f_{\text{rep}}\left(r, r_\eta\right) = \begin{cases} \dfrac{1}{2}\left(\dfrac{1}{r} - \dfrac{1}{r_\eta}\right)^2, & \left(0 < r \le r_\eta\right), \\ 0, & (\text{otherwise}). \end{cases} \tag{7}$$

The expansive repulsion factor results from all such interactions between neighbor nodes. Thus, using the repulsive potential function $f_{\text{rep}}(r, r_\eta)$, the expansive repulsion factor for node $\mathbf{q}_i$ can be represented as follows:

$$\mathbf{F}^i_{\text{expansive}} = \sum_j f_{\text{rep}}\left(r_{ij}, r_\eta\right) \cdot \frac{\mathbf{q}_i - \mathbf{q}_j}{r_{ij}}, \qquad \left(r_{ij} = \left\|\mathbf{q}_i - \mathbf{q}_j\right\|_2, \ \mathbf{q}_j \in \boldsymbol{\eta}_i\right), \tag{8}$$

where r_{ij} and $\boldsymbol{\eta}_i$ are the distance of nodes i and j and the set of neighbor nodes, respectively.

3.2.2. Sensory Repulsion Factor. Sensory repulsion represents the repulsive interactions from sensing node $\Delta\mathbf{q}$, which are distributed evenly around a node and can detect the deformation of the configuration space. Because of the repulsion from sensing nodes, nodes can move naturally toward the free area of the configuration space. Three-dimensional sensors in the workspace can detect free ranges that are not occupied by obstacles. However, in the configuration space for articulated robots, there are no methods of detecting the free range. A motion of the robot in the workspace is expressed as a point in the configuration space; a collision test result for a robot motion in the workspace gives only the binary information of whether collision occurs or not

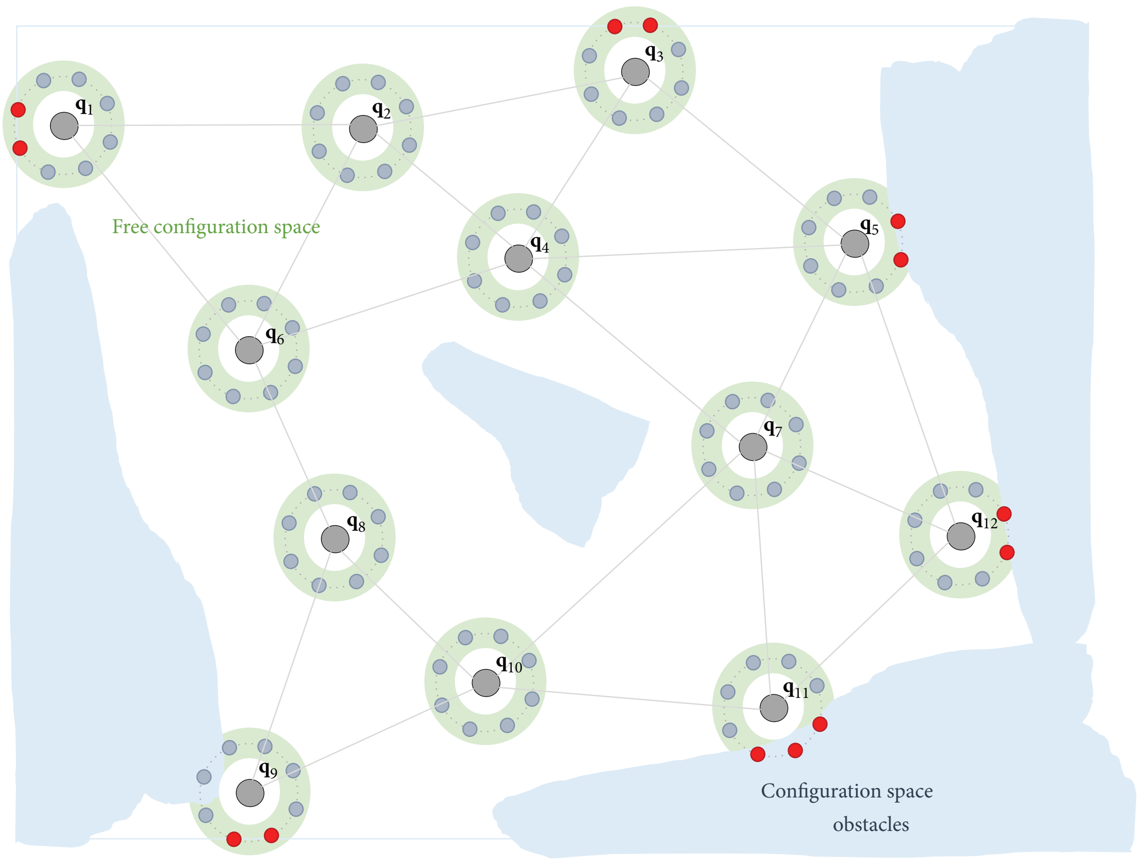


Figure 11: An example of the adaptive roadmap.

for a point in the configuration space. Thus, the data from the sensing nodes produce a binary-type random vector. Therefore, the sensory repulsive factor is similar to the bang-bang controller. Figure 9 shows the sensory repulsion factor generated by the sensing nodes, where $s(\mathbf{q})$ is the collision detection function in the workspace. The result of $s(\mathbf{q})$ is simple binary information; however, complex procedures are required to obtain this result, such as kinematic analysis in the workspace, generation of a collision check model, and collision testing with the environment model:

$$s(\mathbf{q}) = \begin{cases} 1, & (\text{collision detected}), \\ 0, & (\text{otherwise}). \end{cases} \tag{9}$$

The sensing nodes generate repulsion factors in the opposition direction for node $\mathbf{q}_i$. The sensory repulsion factor results from all such repulsive interactions of the sensing nodes. Thus, it can be represented as follows:

$$\mathbf{F}^i_{\text{sensory}} = -\sum_{j=1}^{n_S} \left(s\left(\mathbf{q}_i + \Delta\mathbf{q}_j\right) \cdot \Delta\mathbf{q}_j \right) \quad \left(\Delta\mathbf{q}_j \in \mathbf{S}\right), \tag{10}$$

where $\mathbf{S}$ is the set of sensing nodes.

3.2.3. Difference Equation for Adaptive Roadmap Algorithm. The adaptation mechanism of the proposed adaptive roadmap algorithm is based on two repulsion factors: the expansive repulsion factor $\mathbf{F}_{\text{expansive}}$ and the sensory repulsion factor $\mathbf{F}_{\text{sensory}}$. Figure 10 shows the overall interactions of the proposed algorithm for a node. In Figure 10(a), the expansive repulsion is illustrated as interactions with neighbor nodes. Figure 10(b) shows the sensory repulsion by detecting collision states in the configuration space. The red color of the sensing node indicates detection of a collision in the workspace. Figure 10(c) displays the total repulsion factors applied to the node, and Figure 10(d) shows the movement of the node caused by various interactions. Thus, based on this adaptation mechanism, the difference equation of node $\mathbf{q}_i$ can be expressed as follows:

$$\mathbf{q}_i[k+1] = \mathbf{q}_i[k] + G_1 \cdot \mathbf{F}^i_{\text{expansive}}[k] + G_2 \cdot \mathbf{F}^i_{\text{sensory}}[k], \tag{11}$$

where G_1 and G_2 are the gain of the expansive and sensory repulsion factor, respectively, and

$$\mathbf{F}^i_{\text{expansive}}[k] = \sum_{\mathbf{q}_j \in \boldsymbol{\eta}_i} \frac{f_{\text{rep}}\left(r_{ij}[k], r_\eta\right)}{r_{ij}[k]} \cdot \left(\mathbf{q}_i[k] - \mathbf{q}_j[k]\right), \quad \left(r_{ij}[k] = \left\|\mathbf{q}_i[k] - \mathbf{q}_j[k]\right\|_2\right),$$

$$f_{\text{rep}}\left(r, r_\eta\right) = \begin{cases} \dfrac{1}{2}\left(\dfrac{1}{r} - \dfrac{1}{r_\eta}\right)^2, & \left(0 < r \le r_\eta\right), \\ 0, & (\text{otherwise}), \end{cases}$$

$$\mathbf{F}^i_{\text{sensory}}[k] = -\sum_{\Delta\mathbf{q}_j \in \mathbf{S}} \left(s\left(\mathbf{q}_i[k] + \Delta\mathbf{q}_j\right) \cdot \Delta\mathbf{q}_j\right),$$

$$s(\mathbf{q}) = \begin{cases} 1, & (\text{collision detected}), \\ 0, & (\text{otherwise}). \end{cases} \tag{12}$$

Figure 11 shows an example of the total adaptive roadmap, which consists of 12 nodes, and each node has eight sensing nodes. The adaptive roadmap algorithm performs the entire adaptation using distributed processing of these complex procedures in each node. Thus, the computation of (11) should be performed in parallel at each node. Figure 12 shows the flowchart for processing the adaptive roadmap algorithm. In the flowchart, four procedures in the blue boxes can be performed dispersively at each node by parallel computing techniques, such as multithreading or GPU processing.

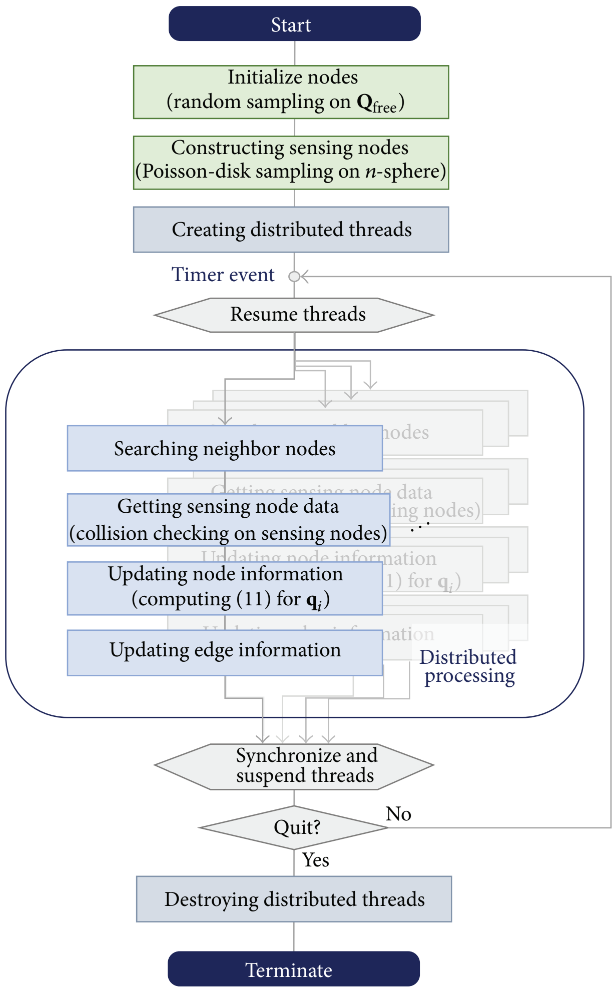

Start
Initialize nodes (random sampling on $\mathbf{Q}_{\text{free}}$)
Constructing sensing nodes (Poisson-disk sampling on n-sphere)
Creating distributed threads
Timer event
Resume threads
Searching neighbor nodes
Getting sensing node data (collision checking on sensing nodes)
Updating node information (computing (11) for $\mathbf{q}_i$)
Updating edge information
Distributed processing
Synchronize and suspend threads
Quit?
No
Yes
Destroying distributed threads
Terminate

Figure 12: Flowchart for processing the adaptive roadmap algorithm.

4. Simulation Results

4.1. Adaptation Results for Workspace Change. To verify the feasibility of the proposed adaptive roadmap method, simulations were performed in a simplified test environment. If the robot has three or more DOFs, the free configuration space becomes a complex high-dimensional space, which is difficult to visualize and identify. Thus, we considered a test environment with a two-link robot with a two-dimensional configuration space. In the test simulations, we set the number of nodes N to 100 for constructing the roadmap, and the gain of the expansive and sensory repulsion factor were set to 40 and 1.0, respectively. To perform the adaptation at each node, distributed computing with the multithreading technique was implemented, and 100 threads were allocated for the processing of each node.

Each joint of the robot was set to have an operating limit range of -170°–170°; thus, the configuration space had the same topology as the Euclidean space. If the two-link robot has no operating limit, the configuration space is a torus-type space; in the case of higher dimensions, the configuration space becomes a more complex space that is difficult to handle. Thus, we assumed that the robot had operating limits to simplify the configuration space, because most actual robots have joint limits.

Figure 13 shows the test environments for the simulations. Figure 13(a) shows the workspace conditions, a two-link robot, and five movable obstacles. Figure 13(b) shows the configuration space with respect to the workspace displayed in Figure 13(a). Obstacles in the configuration space are illustrated with the same colors as in the workspace.

Figure 14 shows the processing procedures of the adaptive roadmap algorithm. Figure 14(a) displays the initial state of the roadmap, in which 100 nodes were scattered randomly in the configuration space. Figures 14(b) and 14(c) illustrate the states of the adaptive roadmap after the 5th and 10th iterations, respectively, in which the nodes disperse in the free configuration space. Figure 14(d) shows the state of the adaptive roadmap at equilibrium, after 100 iterations; at this state, the nodes are distributed evenly in most of the free configuration space. Thus, the coverage, designated by the yellow area, of the distribution of the nodes is almost complete, and the edges are well connected throughout the free configuration space. These processing procedures confirmed that the adaptive roadmap can change the structure of a random initial state to adapt to the changed configuration space.

Figure 15 shows the motion planning results for two test cases using the roadmap information after performing the adaptation. Figures 15(a) and 15(c) represent the roadmap graphs and the planned paths of the robot motion in the configuration space. Figures 15(b) and 15(d) show the operational trajectories of robot motion in the workspace according to the planned paths. From the motion planning results of Figure 15, it can be confirmed that the collision-free robot motions were

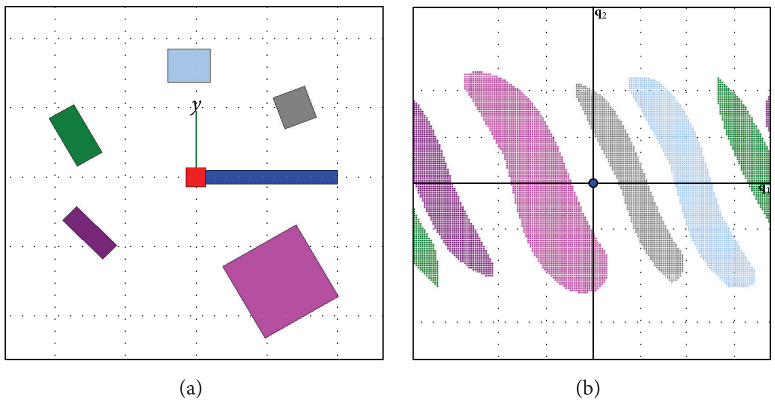


FIGURE 13: Test workspace and the corresponding configuration space. (a) Two-link robot and obstacles in workspace; (b) configuration space for the workspace in (a).

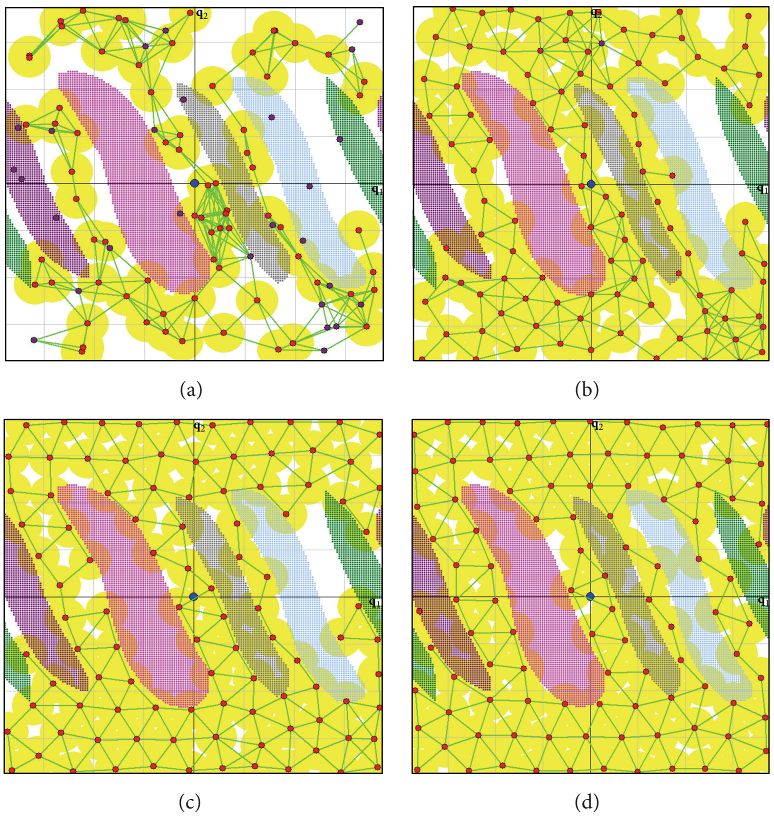


FIGURE 14: Running procedures of the adaptive roadmap algorithm. (a) Initial state; (b) state of the roadmap after the 5th iteration; (c) state of the roadmap after the 10th iteration; (d) state of the roadmap after the 100th iteration.

obtained easily using the adapted roadmap graph, and the planned motion paths were similar to the optimal path.

Next, we tested the behavior of the adaptive roadmap algorithm for changes of the workspace. Figure 16 shows the modified test workspace and the corresponding configuration space. In the changed workspace of Figure 16(a), the positions of five obstacles were changed, and the configuration space of Figure 16(b) was largely deformed, accordingly.

Figure 17 shows the processing procedures of the adaptive roadmap algorithm for the changes of the workspace. Figure 17(a) displays the initial state of the roadmap. Figures 17(b) and 17(c) show the states of the adaptive roadmap after the 5th and 10th iterations, respectively, in which the nodes disperse in the changed configuration space. Figure 17(d) shows the state of the adaptive roadmap at equilibrium, after 100 iterations. In this state, the coverage with respect to the

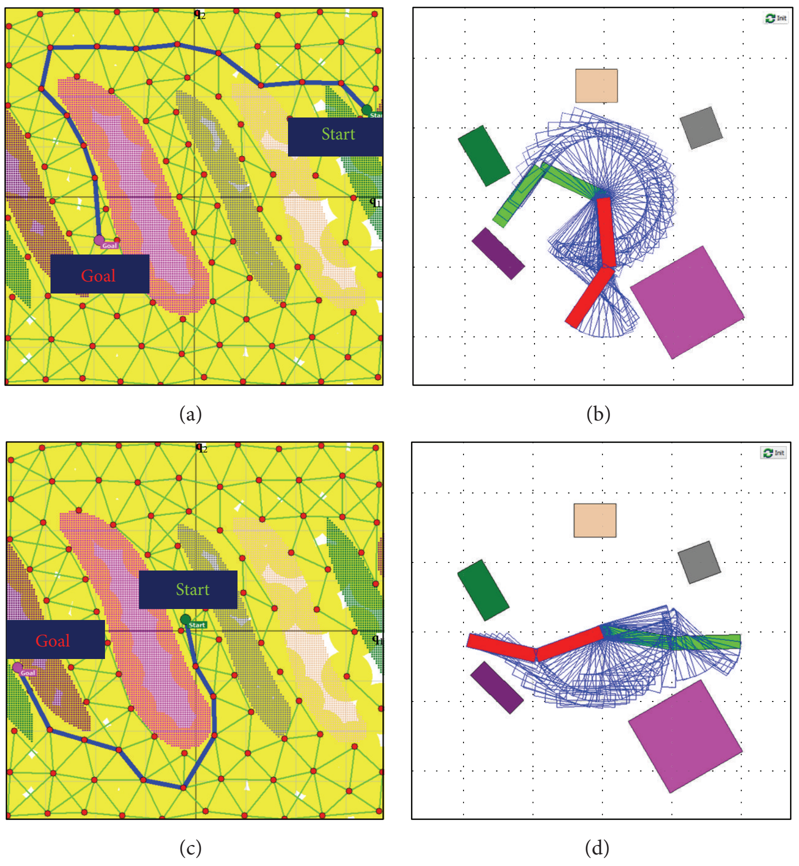


FIGURE 15: Motion planning results and trajectories of robot motion for two test cases (start motion (green); goal motion (red)). (a) Roadmap graph and the planned motion (blue) in the configuration space for the collision-free path from [155°, 78°] to [−85°, −39°]; (b) motion trajectory in the workspace for (a); (c) roadmap graph and the planned motion (blue) in the configuration space for the collision-free path from [−9°, 10°] to [−160°, −33°]; (d) motion trajectory in the workspace for (c).

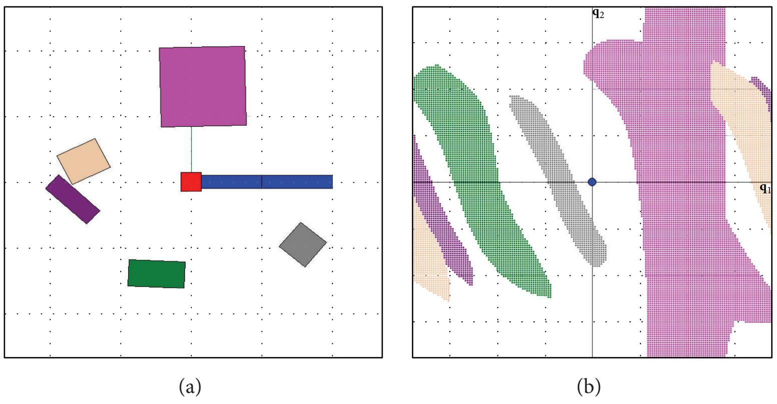


FIGURE 16: Changed workspace and the corresponding configuration space. (a) Two-link robot and obstacles in the workspace; (b) configuration space of the workspace shown in (a).

distribution of nodes is almost complete, and the edges are well connected throughout the changed configuration space. The results of this simulation confirmed that the adaptive roadmap can perform a successful adaptation in the case of a radical deformation of the configuration space.

Figure 18 shows the motion planning results in the changed workspace, shown in Figure 16, for two test cases using the adapted roadmap information. Figures 18(a) and 18(c) represent the roadmap graphs and the planned paths of robot motion in the configuration space. Figures 18(b) and

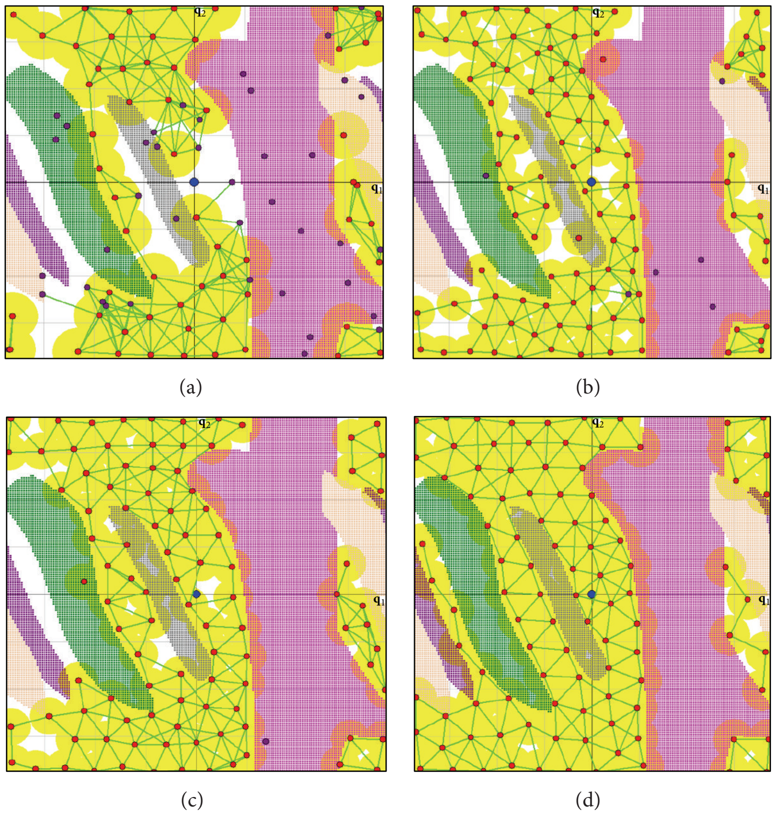


Figure 17: Adaptation procedures for the changed configuration space. (a) Initial state; (b) state of the roadmap after the 5th iteration; (c) state of the roadmap after the 10th iteration; (d) state of the roadmap after the 100th iteration.

18(d) show the operational trajectories of robot motion in the workspace according to the planned paths. Similar to the results of Figure 15, the paths of robot motion for the two test cases were obtained for the changed workspace, and the acquired motion paths seem to be close to the optimal path.

4.2. Adaptation Results for Various Deformations. Subsequently, the behavior of the adaptive roadmap was observed in dynamic environments. To generate continuous changes of the workspace, the positions of all obstacles were modified slightly and randomly for every simulation step, and considerable obstacle position changes were applied at every 50th step.

Figure 19 shows the results of the adaptive roadmap under dynamic environment conditions. Figure 19(a) presents the results for the total expansive repulsion. The red line shows the free configuration ratio, and the green line shows the total expansive repulsion. When a radical change occurs, the total expansive repulsion exhibits a spiky transient pattern; however, it returns to a stable state after some iterations. The gray line in Figure 19(b) represents the coverage for the configuration space. When a radical change occurs, the coverage briefly decreases; however, it stays near 100% throughout the running of the adaptive roadmap algorithm.

Figure 20 shows the steady state of the adaptive roadmap for various configuration space conditions. The free configuration space ratio ranged from 89.6% to 38.7%, which confirmed that the adaptive roadmap converged to equilibrium for various environments. Further, we have published a video of the continuous behavior of the adaptive roadmap algorithm. The detailed adaptation process of the roadmap graph can be seen at [41].

5. Conclusions

In this paper, we propose an algorithm that can adapt to a radically changeable configuration space. For this purpose, we first analyzed the deformation of the configuration space with respect to the change of the workspace. To address the issue of the deformation of the configuration space, we presented an adaptation algorithm for the roadmap graph based on distributed processing. In this algorithm, each node in the graph moves in the free configuration space by detecting the deformation of the configuration space; then, the node updates its edge information with respect to the changed node information. The adaptation mechanism of the proposed algorithm is based on two basic repulsion factors: the expansive repulsion factor and the sensory repulsion factor. Moreover, the adaptive roadmap can perform the entire adaptation by distributed processing of the complex procedure in each node. To verify the effectiveness of the proposed method, simulations of a continuously changing

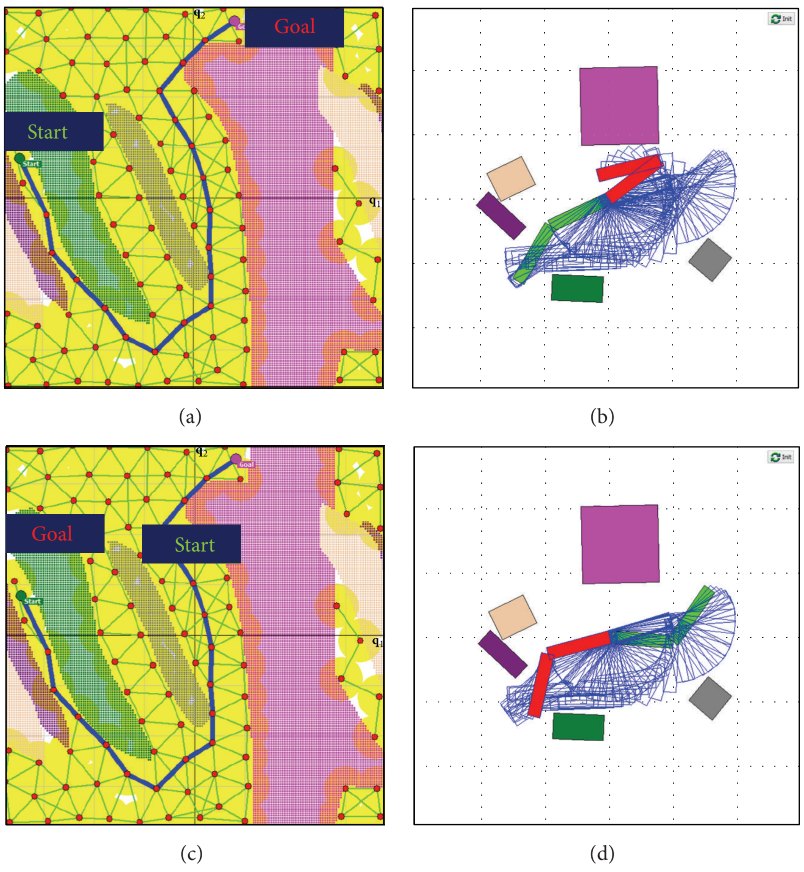


FIGURE 18: Motion planning results and trajectories of robot motion for two test cases (start motion (green); goal motion (red)). (a) Roadmap graph and the planned motion (blue) in the configuration space for the collision-free path from [−155°, 35°] to [37°, 157°]; (b) motion trajectory in the workspace for (a); (c) roadmap graph and the planned motion (blue) in the configuration space for the collision-free path from [−4°, 59°] to [−165°, 62°]; (d) motion trajectory in the workspace for (c).

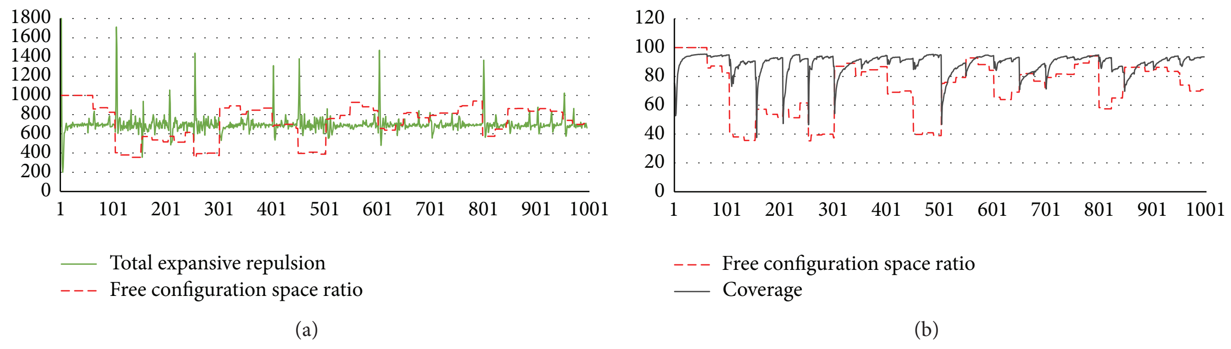


FIGURE 19: Simulation results of the adaptive roadmap for various conditions of the configuration space volume (red). (a) Total expansive repulsion (green); (b) coverage of the roadmap (gray).

workspace were performed in a test environment of a two-link manipulator case. The simulation results confirmed that the graph could perform the entire adaptation with respect to the change of the configuration space using the proposed algorithm.

As future work, high-dimensional configuration space and complex problems, such as redundancy and mobile manipulation for actual robots, will be considered. Thus, to overcome these problems, we should deliberately utilize massively parallel processing technology using GPUs.

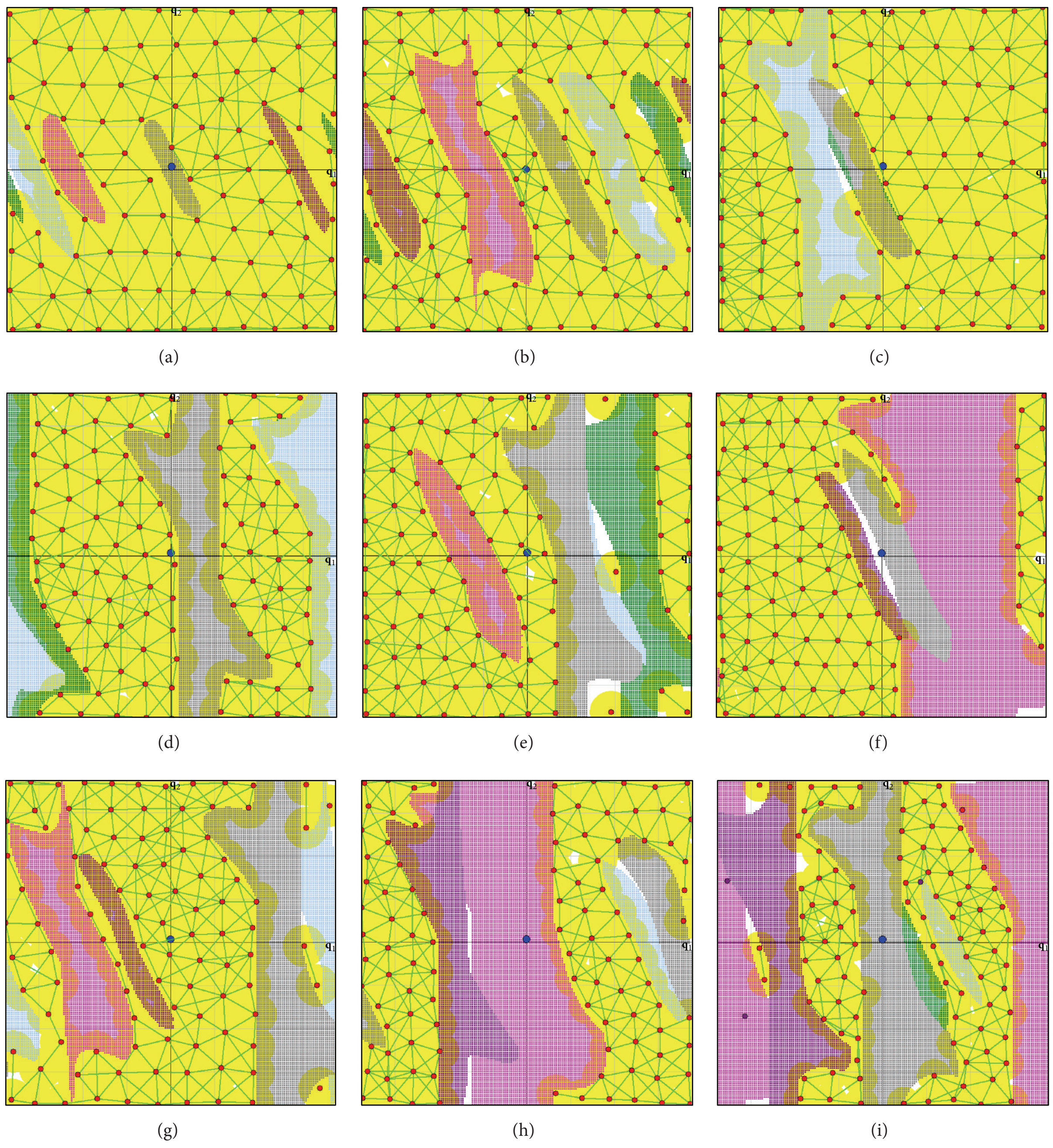


Figure 20: Topologies of the adaptive roadmap at equilibrium for various conditions of the free configuration space ratio. (a) 89.6%; (b) 69.3%; (c) 79.5%; (d) 59.9%; (e) 59.3%; (f) 54.1%; (g) 57.8%; (h) 48.68%; (i) 38.7%.

Moreover, the application of the current results to the study of complex systems will require further investigation. The adaptive roadmap algorithm is strongly related to the complex system that interacts with many independent agents internally. Therefore, the further development of the adaptive algorithm by combining it with complex systems science should be considered as important future work.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

Acknowledgments

This research was financially supported by the Ministry of Trade, Industry and Energy (MOTIE) and Korea Institute for

Advancement of Technology (KIAT) through the Promoting Regional Specialized Industry.

References

[1] J. C. Latombe, *Robot Motion Planning*, Kluwer Academic, 1991.

[2] H. Choset, K. M. Lynch, S. Hutchison et al., *Principles of Robot Motion: Theory, Algorithms and Implementations*, The MIT Press, 2005.

[3] J. T. Schwartz and M. Sharir, "On the piano movers' problem. III. Coordinating the motion of several independent bodies: the special case of circular bodies moving amidst polygonal barriers," *International Journal of Robotics Research*, vol. 2, no. 3, pp. 46–75, 1983.

[4] C. Park, J. Pan, and D. Manocha, "Real-time optimization-based planning in dynamic environments using GPUs," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA '13)*, pp. 4090–4097, IEEE, Karlsruhe, Germany, May 2013.

[5] T. Kunz, U. Reiser, M. Stilman, and A. Verl, "Real-time path planning for a robot arm in changing environments," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS '10)*, pp. 5906–5911, Taipei, Taiwan, October 2010.

[6] L. E. Kavraki, P. Švestka, J.-C. Latombe, and M. H. Overmars, "Probabilistic roadmaps for path planning in high-dimensional configuration spaces," *IEEE Transactions on Robotics and Automation*, vol. 12, no. 4, pp. 566–580, 1996.

[7] S. M. LaValle and J. J. Kuffner Jr., "Randomized kinodynamic planning," *International Journal of Robotics Research*, vol. 20, no. 5, pp. 378–400, 2001.

[8] L. E. Kavraki, J. C. Latombe, R. Motwani, and P. Raghavan, "Randomized query processing in robot path planning," in *Proceedings of the 27th Annual ACM Symposium on Theory of Computing (STOC '95)*, pp. 353–362, Las Vegas, Nev, USA, June 1995.

[9] J. Pan, L. Zhang, and D. Manocha, "Collision-free and smooth trajectory computation in cluttered environments," *International Journal of Robotics Research*, vol. 31, no. 10, pp. 1155–1175, 2012.

[10] J. Vannoy and J. Xiao, "Real-time adaptive motion planning (RAMP) of mobile manipulators in dynamic environments with unforeseen changes," *IEEE Transactions on Robotics*, vol. 24, no. 5, pp. 1199–1212, 2008.

[11] M. S. Branicky, M. M. Curtiss, J. Levine, and S. Morgan, "Sampling-based planning, control and verification of hybrid systems," *IEE Proceedings: Control Theory and Applications*, vol. 153, no. 5, pp. 575–590, 2006.

[12] S. Karaman and E. Frazzoli, "Sampling-based algorithms for optimal motion planning," *International Journal of Robotics Research*, vol. 30, no. 7, pp. 846–894, 2011.

[13] A. M. Ladd and L. E. Kavraki, "Measure theoretic analysis of probabilistic path planning," *IEEE Transactions on Robotics and Automation*, vol. 20, no. 2, pp. 229–242, 2004.

[14] K. Hauser, "On responsiveness, safety, and completeness in real-time motion planning," *Autonomous Robots*, vol. 32, no. 1, pp. 35–48, 2012.

[15] F. Belkhouche, "Reactive path planning in a dynamic environment," *IEEE Transactions on Robotics*, vol. 25, no. 4, pp. 902–911, 2009.

[16] S. Petti and T. Fairchard, "Safe motion planning in dynamic environments," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 2210–2215, Edmonton, Canada, August 2005.

[17] W. Lu, G. Zhang, and S. Ferrari, "An information potential approach to integrated sensor path planning and control," *IEEE Transactions on Robotics*, vol. 30, no. 4, pp. 919–934, 2014.

[18] R. Olfati-Saber, "Flocking for multi-agent dynamic systems: algorithms and theory," *IEEE Transactions on Automatic Control*, vol. 51, no. 3, pp. 401–420, 2006.

[19] F. Cucker and S. Smale, "Emergent behavior in flocks," *IEEE Transactions on Automatic Control*, vol. 52, no. 5, pp. 852–862, 2007.

[20] W. Ren and Y. Cao, *Distributed Coordination of Multi-agent Networks: Communications and Control Engineering Series*, Springer, Berlin, Germany, 2011.

[21] R. Olfati-Saber, J. A. Fax, and R. M. Murray, "Consensus and cooperation in networked multi-agent systems," *Proceedings of the IEEE*, vol. 95, no. 1, pp. 215–233, 2007.

[22] J. Wang, Z. Liu, and X. Hu, "Consensus control design for multi-agent systems using relative output feedback," *Journal of Systems Science and Complexity*, vol. 27, no. 2, pp. 237–251, 2014.

[23] J. Hu and Y. S. Lin, "Consensus control for multi-agent systems with double-integrator dynamics and time delays," *IET Control Theory & Applications*, vol. 4, no. 1, pp. 109–118, 2010.

[24] G. Xie and L. Wang, "Consensus control for a class of networks of dynamic agents," *International Journal of Robust and Nonlinear Control*, vol. 17, no. 10, pp. 941–959, 2007.

[25] L. Xiao, S. Boyd, and S.-J. Kim, "Distributed average consensus with least-mean-square deviation," *Journal of Parallel and Distributed Computing*, vol. 67, no. 1, pp. 33–46, 2007.

[26] M. Chiang, S. H. Low, A. R. Calderbank, and J. C. Doyle, "Layering as optimization decomposition: a mathematical theory of network architectures," *Proceedings of the IEEE*, vol. 95, no. 1, pp. 255–312, 2007.

[27] F. Zhang, J. Jia, and L. Yang, "Adaptive synchronization in complex network with different order node dynamics," *Discrete Dynamics in Nature and Society*, vol. 2015, Article ID 919146, 7 pages, 2015.

[28] J. Schulman, Y. Duan, J. Ho et al., "Motion planning with sequential convex optimization and convex collision checking," *International Journal of Robotics Research*, vol. 33, no. 9, pp. 1251–1270, 2014.

[29] J. Pan, I. A. Sucan, S. Chitta, and D. Manocha, "Real-time collision detection and distance computation on point cloud sensor data," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA '13)*, pp. 3592–3599, IEEE, Karlsruhe, Germany, May 2013.

[30] F. Schwarzer, M. Saha, and J.-C. Latombe, "Adaptive dynamic collision checking for single and multiple articulated robots in complex environments," *IEEE Transactions on Robotics*, vol. 21, no. 3, pp. 338–353, 2005.

[31] F. Schwarzer, M. Saha, and J. C. Latombe, "Exact collision checking of robot paths," in *Algorithmic Foundations of Robotics V.*, Springer, Berlin, Germany, 2004.

[32] M. Damrudi and N. Ithnin, "An optimization of tree topology based parallel cryptography," *Mathematical Problems in Engineering*, vol. 2012, Article ID 871091, 10 pages, 2012.

[33] S. K. Samudrala, J. Zola, S. Aluru, and B. Ganapathysubramanian, "Parallel framework for dimensionality reduction of large-scale datasets," *Scientific Programming*, vol. 2015, Article ID 180214, 12 pages, 2015.

[34] B. He, L. Tang, J. Xie, X. Wang, and A. Song, "Parallel numerical simulations of three-dimensional electromagnetic radiation with MPI-CUDA paradigms," *Mathematical Problems in Engineering*, vol. 2015, Article ID 823426, 9 pages, 2015.

[35] C. Lauterbach, Q. Mo, and D. Manocha, "gProximity: hierarchical GPU-based operations for collision and distance queries," *Computer Graphics Forum*, vol. 29, no. 2, pp. 419–428, 2010.

[36] J. Pan and D. Manocha, "GPU-based parallel collision detection for fast motion planning," *International Journal of Robotics Research*, vol. 31, no. 2, pp. 187–200, 2012.

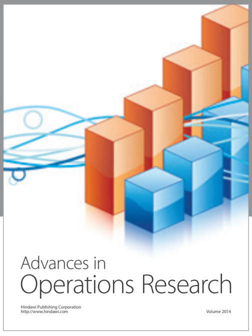
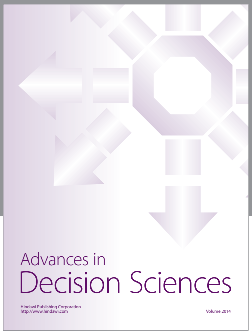
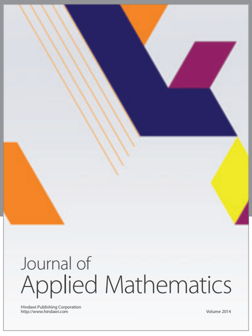
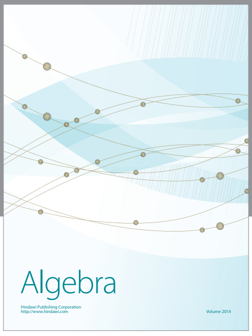
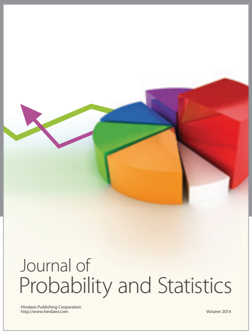
[37] D. Kim, J.-P. Heo, J. Huh, J. Kim, and S.-E. Yoon, "HPCCD: hybrid parallel continuous collision detection using CPUs and GPUs," *Computer Graphics Forum*, vol. 28, no. 7, pp. 1791–1800, 2009.

[38] M. S. Ebeida, S. A. Mitchell, A. Patney, A. A. Davidson, and J. D. Owens, "A simple algorithm for maximal poisson-disk sampling in high dimensions," *Computer Graphics Forum*, vol. 31, no. 2, pp. 785–794, 2012.

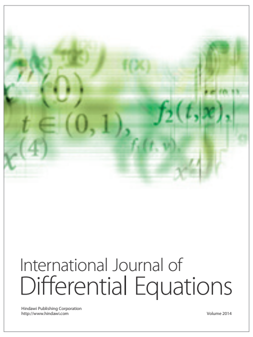
[39] E. A. Jackson, *Equilibrium Statistical Mechanics*, Prentice-Hall, Upper Saddle River, NJ, USA, 1968.

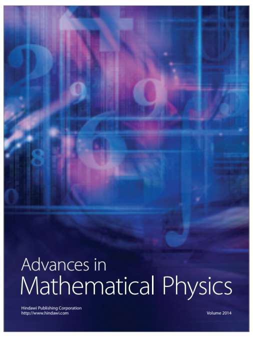
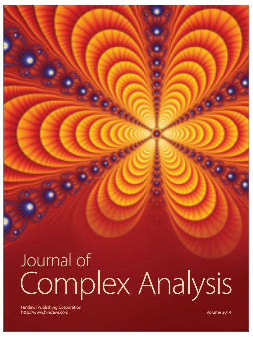
[40] O. Khatib, "Real-time obstacle avoidance for manipulators and mobile robots," *International Journal of Robotics Research*, vol. 5, no. 1, pp. 90–98, 1986.

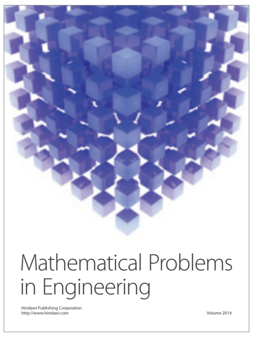
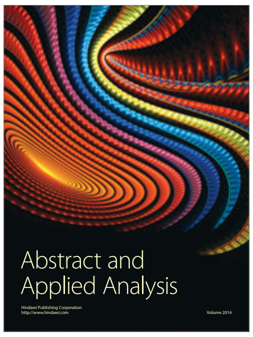
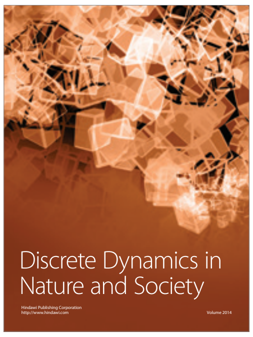
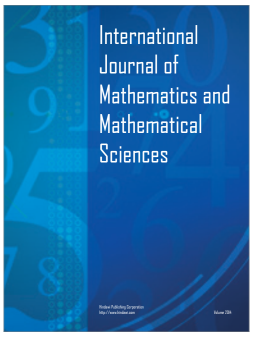
[41] YouTube, Adaptive roadmap algorithm, November 2015, https://www.youtube.com/watch?v=aanjEEkZ6n8.


Advances in Operations Research

Advances in Decision Sciences

Journal of Applied Mathematics

Algebra

Journal of Probability and Statistics

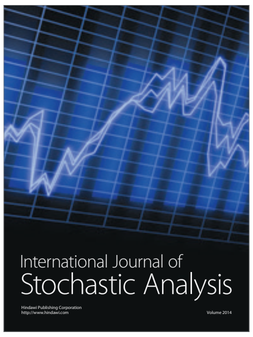
The Scientific World Journal


International Journal of Differential Equations

International Journal of Combinatorics

Advances in Mathematical Physics

Journal of Complex Analysis

Journal of Mathematics

Mathematical Problems in Engineering

Abstract and Applied Analysis

Discrete Dynamics in Nature and Society

International Journal of Mathematics and Mathematical Sciences

Journal of Discrete Mathematics

Journal of Function Spaces

International Journal of Stochastic Analysis

Journal of Optimization