A Computational Interactive Approach to Multi-agent Motion Planning

Sang-Hoon Ji, Jeong-Sik Choi, and Beom-Hee Lee

Abstract: It is well known that mathematical solutions for multi-agent planning problems are very difficult to obtain due to the complexity of mutual interactions among multi-agents. Most of the past research results are thus based on the probabilistic completeness. However, the practicality and effectiveness of the solution from the probabilistic completeness is significantly reduced by heavy computational burden. In this paper, we propose a practically applicable solution technique for multi-agent planning problems, which assures a reasonable computation time and a real world application for more than 3 multi-agents, for the case of general shaped paths in agent movement. First, to reduce the computation time, an extended collision map is developed and utilized for detecting potential collisions and obtaining collision-free solutions for multi-agents. Second, a priority for multi-agents is considered for successive and interactive modifications of the agent movements with lower priority. Various solutions using speed reduction and time delay of the relevant agents are investigated and compared in terms of the computation time. A practical implementation is finally provided for three different types of agents to emphasize the effectiveness of the proposed interactive approach to multi-agent planning problems.

Keywords: Computational approach, extended collision map, motion planning, multi-agents.

1. INTRODUCTION

Multi-agent motion planning is one of the most interesting and essential research fields in robotics. The demand for various specialized robots has been increasing rapidly with the advancement of robot technology. For example, guidance, security, and fire detection robots are being used in buildings. Whether robots can safely execute their given missions in a common workspace is a key technique in multi-agent control.

Multi-agent motion planning has been studied for the last several decades. The traditional approach to multi-agent control may be classified into two approaches-centralized and distributed, or centralized and decoupled [1,2]. In the centralized approach, all agents are treated as a single system with many degrees of freedom. Because a central system can act as a planner, it can plan optimally. On the contrary, in the distributed approach [3], each agent is treated as a single, basically independent system. These two approaches form a traditional foundation for multi-agent motion planning.

Multi-agent motion planning, however, is still a challenging field of research, having some technical difficulties in resolving conflict among agents. The centralized approaches have been faced with problems such as the curse of dimensionality, complexity, computational difficulty, and NP-hard problems [4-6]. These problems are due to the fact that one system alone takes up the whole burden for planning motions of all agents interactively. In addition, an essential assumption that information from all agents can be transmitted to a supervisory system is also seen as unrealistic [3].

The decentralized approach has its own inherent problems such as mutual interference; local minima, parallel run phenomenon, and negotiation fail [7-9]. Moreover, this approach only gives a sub-optimal solution because of the limitation of computational capacity in architecture. Above all, the dispute among agents cannot be settled down by arbitration. To overcome these problems in the two approaches, we propose a new and compromising method in this paper.

The proposed computational approach to multi-agent motion planning relies upon a useful tool – the extended collision map. The basic concept of the centralized approach is adopted because we can find a

Manuscript received May 11, 2006; revised December 27, 2006; accepted March 8, 2007. Recommended by Editorial Board member Sooyong Lee under the direction of Editor Jae-Bok Song. This work was supported in part by the Seoul R&D Program, Social Security Robot Program funded by the Ministry of Commerce, Industry and Energy, Automation and Systems Research Institute (ASRI) in Seoul National University, and the Brain Korea 21 Project.

Sang-Hoon Ji, Jeong-Sik Choi, and Beom-Hee Lee are with the School of Electrical Engineering and Computer Sciences, Seoul National University, San 56-1, Shillim-dong, Kwanak-gu, Seoul 151-742, Korea (e-mails: {robot91, jsforce, bhlee}@snu.ac.kr).

complete solution of multi-agent control even though there is a computational drawback. To overcome the drawbacks of the centralized approach, we applied several concepts in our approach: 1) intelligent space, which can provide a central planner with essential and necessary information for motion planning; 2) extended collision map method, which has been developed for multi-agent conflict resolution; 3) agent task priority for settling disputes among agents; and, 4) trajectory planning, which can simplify the solution of this approach.

The remainder of the paper is organized as follows: Section 2 briefly describes previous research work. Section 3 defines our research and shows the detailed approach conceptually. Section 4 presents the concept of the key technique of this paper-extended collision map method. Section 5 shows the implementation method in detail, and specifies the application method of the computational approach. Section 6 provides an implementation for 3 heterogeneous agents, and finally this paper is concluded in Section 7.

2. PREVIOUS WORK

A good survey of basic motion planning and its approaches can be found in [1,2,10]. In the early centralized approach, a central planner applied the methods that were originally developed for a single robot motion to multi-agent motion planning. The number of increased agents, however, made the planner unfeasible, because of high dimensionality. To overcome this problem, the randomized planning technique has been proposed and has shown several successful demonstrations [11]. Some notable concepts-artificial potential field, RPP (Randomized Path Planner) and PRM (Probabilistic Roadmap Method)-have been applied in the randomized planning [12,13]. On the other hand, the research based on the distributed approach also proposed different schemes for multi-agent motion planning. For example, the concepts such as CT-space, dialogue model, behavior inference, direct communication, voting and cooperation are proposed.

Li and Chou developed a planner, based on RPP, for many robots [14]. It has reduced inter-robot collisions and controlled the motions of crowd robots with potential field and grouping. Barber, Liu and Ramaswamy proposed another concept of E-PERT (extended PERT), which can maintain traceable temporal relations among parallel activities [15]. By project management, the planner can detect collisions among agents and schedule the movements of the agents. Also, the timed automata method, based on a scheduling technique, was introduced recently [16]. Azarm and Schmidt presented another concept including negotiation and dynamic priority assignment for conflict-free motion planning [17]. When an agent predicts a collision with another agent, the agent compares the cost function to that of other agents and determines the agent to which the higher priority should be assigned. Dias and Stentz provided the market-based multi robots coordination approach and showed the performance of their method compared with those of the centralized and behavioral approaches [18].

Most of these methods emphasize a strategy for path planning or a demonstration of well-designed rules for collision avoidance. In this paper, we focus on the development of an effective and feasible tool incorporating the concepts of trajectory planning and agent task priority. This approach has not yet been found in the previous research.

3. PROBLEM FORMULATION

A multi-agent system (MAS) consists of several components with specific roles within an organizational structure. Agents, communication systems, and motion planning systems are essential components in a multi-agent system. Particularly, the agent detection system is also needed in the centralized system. Our research is confined only within the motion planning in the centralized approach. For this reason, some assumptions are needed and other components have to be modeled to solve the motion planning problem. In addition, we present a collision-free strategy, and apply this strategy for coping with the previous problems in the centralized approach. On the basis of these assumptions and strategies, a key technique for multi-agent motion planning is presented in this section.

In centralized multi-agent control, a supervisory system should identify the position of all agents and have the ability for information transmission to the agents. To do this, we adopt a concept of intelligent space, which is the 3D space equipped with ubiquitous sensors for the agent detection [19-21]. In this space, the central planner always addresses the position information of all agents at each sampling time and transmits collision-free solutions to the agents.

Agents in robotics can have either car-like or human-like shapes. Computation load can be reduced by using a simple model of an agent, so we model an agent as a circle. It is expressed by radius r and a center point ($p(t)=[p_x(t), p_y(t)]$). An agent has physical constraints-velocity and acceleration limitation denoted by (1). An agent was assumed to always move in full speed within physical constraints and have a trapezoid model of the velocity profile. This assumption simplifies motion planning, because the central planner only has to consider the speed down for collision avoidance. All the collision-free strategies in our approach are based on this

assumption.

$$| \dot{p}(t) | \leq v_{\max} , | \ddot{p}(t) | \leq a_{\max} \quad (1)$$

Motion planning consists of path and trajectory planning. Path planning may be an effective method for a few agents, but it would become complex and be faced with an NP-hard problem for congested agents. This is because the path modification of an agent makes the previous planning useless, so planning safe paths for all agents simultaneously may become a very difficult problem. Regarding this problem, we take collision-avoidance into account at the stage of trajectory planning. In this paper, we assume that initial paths of agents are assigned in collision-free states. Many notable studies – distance based planning and bush algorithm-have shown the way to find the feasible paths in collision-free states [23]. By using these algorithms, the safe path in a stationary obstacle environment can be found easily. Now, we focus on the trajectory planning to resolve conflicts among moving obstacles or agents.

The increased number of agents would bring high dimensionality in the centralized approach. To tackle these critical issues, we use a strategy of the prioritized planning, where all agents are assigned with their own priority. This concept has been considered useful in solving the problems, since it can reduce the problem from single planning with high dimensionality to multi planning with low dimensionality [22]. Therefore, NP problems can be overcome using this strategy. In the real world, this concept is acceptable because all agents have their own specialized job and the importance of the jobs may be different. For example, saving endangered people is more important than vacuuming rugs. Thus, it is reasonable that an agent with the rescue work has higher priority and that it has the right to pass the intersection first if there are congested intersection areas in multi-agent motion.

4. EXTENDED COLLISION MAP

Based on the concepts specified in Section 3, we now address the key problems of collision detection and resolution in multi-agent motion planning. The extended collision map developed for multi-agent motion planning is based on the concept of the original collision map [24].

4.1. Original collision map for two agents

The concept of the original collision map was presented in the previous study [24]. It was developed for motion planning of two agents and provides the information of a potential collision between two agents. The original concept is as follows: An agent with a higher priority is called ‘agent 1’, and an agent

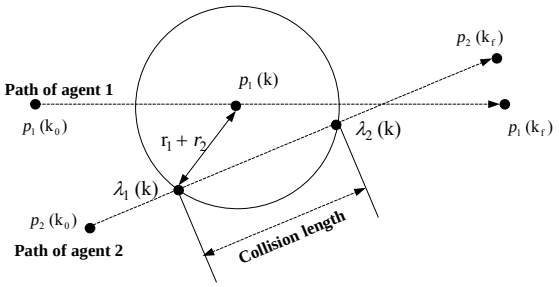


Fig. 1. Path of two agents and collision length

with a lower priority is called ‘agent 2’. The radii of the two agents are r_1 and r_2 respectively. Using the obstacle space scheme, agent 1 can be represented as the agent with a radius of r_1+r_2, and agent 2 can be considered as a point agent. Since agent 1 has higher priority, the original trajectory of agent 1 is assumed not to be changed. On the contrary, agent 2 must modify its trajectory if a collision is anticipated. Two agents are assumed to move along linear paths, as shown in Fig. 1. In the previous study, the path was set to a straight line to simplify the mathematical calculation.

If the path of agent 2 meets agent 1, with radius of r_1+r_2, the two agents will collide with each other. At this instant, the part of agent 2’s path that overlaps with agent 1’s path, is called the ‘collision length’, which is denoted by the portion between $\lambda_1(k)$ and $\lambda_2(k)$ in Fig. 1. These overlapped parts are examined at every instant of the sampling time k to construct a ‘collision region.’ If the TLVSTC (normalized traveled length versus servo time curve, simply, trajectory where normalized traveled length at time t is obtained by dividing the traveled length along the traveling path from start to the servo time t by total traveled length) of agent 2 arrives at the region, the two agents will collide with each other under the original trajectories.

This colliding case is shown in Fig. 2. In this figure, the vertical axis represents the traveled length of agent 2 and the horizontal axis represents the elapsed time. The collision between agent 1 and agent 2 can be analyzed algebraically by using Fig. 2. In Fig. 1, $p_1(k)$

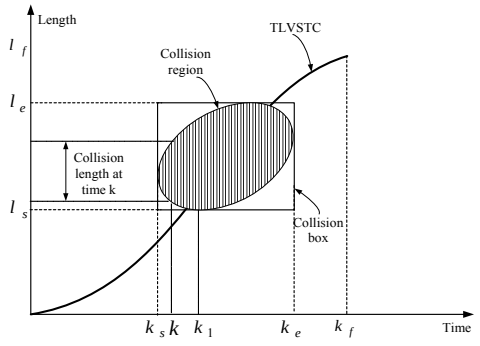


Fig. 2. TLVSTC and collision region.

is the center point of agent 1 at time k. If we represent the position of agent 2 at time k as $p_2(k)$, the original trajectory of agent 2 is:

$$p_2(k) = p_2(k_0) + \lambda(k)(p_2(k_f) - p_2(k_0)), \quad (2)$$

where $0 \leq \lambda \leq 1$, and $p_2(k_0)$ and $p_2(k_f)$ are the initial and final position of agent 2, respectively. The collision between two agents occurs at time k when the distance from $p_1(k)$ to the path of agent 2 in (2) is less than or equal to the radius of agent 1, (r_1+r_2). Thus, we first solve the following equation.

$$(r_1 + r_2)^2 = \| p_1(k) - p_2(k) \|^2 \quad (3)$$

If we replace $p_2(k)$ in (3) with (2), then we have:

$$(r_1 + r_2)^2 = [p_1(k) - p_2(k_0) - \lambda(p_2(k_f) - p_2(k_0))]^T [p_1(k) - p_2(k_0) - \lambda(p_2(k_f) - p_2(k_0))]. \quad (4)$$

Equation (4) is a quadratic equation in p. Thus, it has three possible solution cases. First, it may not have any real solutions: there is no collision between two agents. Second, it has one double real solution; agent 1 comes in contact with the agent 2 path. Finally, two real solutions exist in (4); agent 1 encroaches on the agent 2 path and the two agents may collide. To resolve this problem, the TLVSTC of agent 2 should not meet the collision region in Fig. 2.

4.2. Collision avoidance strategies using collision map

We know that it is difficult to mathematically represent the boundary line of the collision region since it is the set of the contour of the collision length for each sampling time. Thus, the concept of 'collision box' is introduced for simplifying the contour of the collision region and collision avoidance can be mathematically solved.

This concept can be also explained in Fig. 2. In this figure, k_s is the time when agent 1 starts overlapping agent 2's path. Also k_e is the time when agent 1 leaves agent 2's path. l_s and l_e are the minimum and maximum values of the collision length in the collision region, respectively. The coordinates of the edges of the collision box can be found from the above parameters, which can be used to control agent 2 to avoid collision.

There are two strategies that can be used to avoid collision, namely, time delay and speed reduction. Time delay is a strategy that can delay the start time of agent 2 for a period of time corresponding to the difference between k_e and k_1 to avoid the collision. Consequently, agent 2 reaches its goal at time:

$$k_f^1 = k_f + (k_e - k_1), \quad (5)$$

where k_e-k_1 is the delay time illustrated in Fig. 3. The idea of the minimum time delay was introduced in [25].

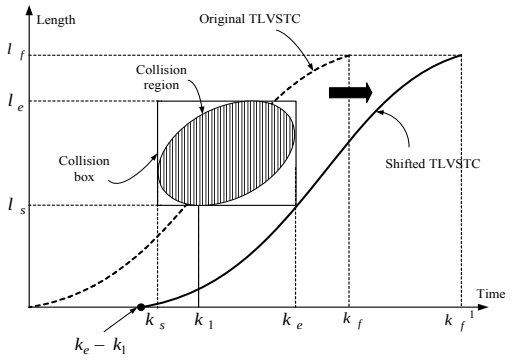


Fig. 3. Collision avoidance through time delay.

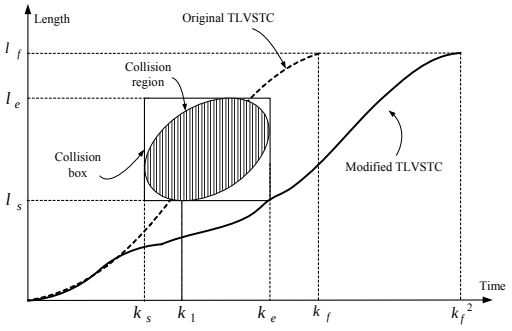


Fig. 4. Collision avoidance through speed reduction.

In contrast, in the strategy of speed reduction, all agents start simultaneously and the moving speed of agent 2 is changed to avoid collision. The velocity profile of agent 2 is modified so that agent 2 does not reach the collision region. Note that the lower priority agent will move at the maximum speed when the time delay strategy is used. This suggests that, in general, the speed reduction strategy has lower performance in arrival time than the time delay strategy. And only when the speed reduction strategy make TLVSTC pass through the lower-right point of the collision box in Fig. 4, can the strategy produce the same performance obtained by the time delay strategy.

4.3. Extended collision map

The collision map method would be regarded as an effective method for detecting potential collisions between agents, and we basically adopted this method for detecting potential collisions among multi-agents. However, this idea may not be directly applied to multi-agents because of some assumptions. First, the method considers the motions of only two agents. Second, the assumption – agent path is limited to a straight line-is unrealistic in a multi-agent environment. Thus, the two assumptions in the original concept are dismissed and changed for multi-agent motion planning as follows:

① The number of agents is n.

② The path of an agent may take any shape.

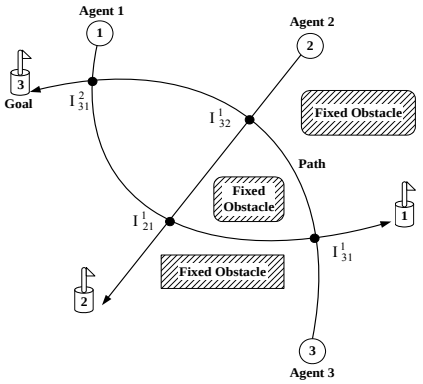


Fig. 5. Three agents with path in workspace.

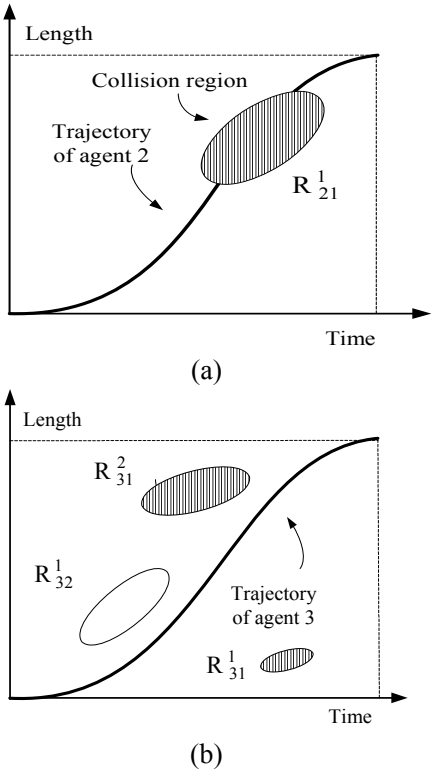


Fig. 6. Extended collision maps of agent 2 (a) and 3 (b) corresponding to Fig. 5.

Moreover, multi-agent motion planning demands more considerations, such as dimensionality, complexity, and interference among agents. On the basis of the modified assumptions, we developed a new tool, which is an extension of the original collision map concept. The tool also incorporates the strategy-agent task priority-previously stated in Section 3.

4.3.1. Notation

In multi-agent environment, many intersections would exist in workspace and correspond to collision regions on the extended collision map. Thus, the intersection and its corresponding collision region should be described. An intersection in workspace is denoted by the symbol

$$I_{ij}^{k}\ ; i > j, \tag{6}$$

where i and j represent the priority number of the agent. Thus, the small number j indicates agent j is a higher priority agent than agent i, and k is the ordering number denoting intersections along the path of the agent i from the starting point. For example, the region I^2_{31} in Fig. 5 represents the second intersection between agents 3 and 1 from the starting point of agent 3. In this example, agent 3 has two intersections with agent 1: I^1_{31}, I^2_{31}, and one intersection with agent 2: I^1_{32}. The corresponding collision region of the intersection is expressed as:

$$\boldsymbol{R}_{\boldsymbol{ij}}^{\boldsymbol{k}}\ ; \text{a collision region corresponding to the intersection } \boldsymbol{I}_{\boldsymbol{ij}}^{\boldsymbol{k}}. \tag{7}$$

4.3.2. Formation of extended collision map

The extended collision map is basically formed by multiple collision regions. The method of collision region generation among agents is the same as the original concept of collision region in two agents. Thus, the multiple collision regions, which represent the potential collisions among agents and their higher priority agents, can be generated in the same manner. For example, the extended collision map of agent 3 in Fig. 6 is formed by collision regions with agents 1 (R^1_{31}, R^2_{31}) and 2 (R^1_{32}).

Consequently, the extended collision map can show all potential collisions related to an agent on a single 2D map. However, the extended collision maps formed in this process are not complete for collision-free motion planning because the trajectory planning of higher priority agents can affect the extended collision maps of lower priority agents. This effect has not been applied to the extended collision map yet. It is specified later in this section.

4.3.3. Formulation of collision-free state

Our objective is to make trajectories of all agents in a collision-free state while moving to their goals. To achieve this objective, we formulated the problem as follows: two steps are needed to specify the complete collision-free state of multi-agents.

① Collision-free state with any higher priority agent

② Collision-free state among all higher priority agents

In the original concept, collision-free planning was able to be accomplished in the first step. However, to consider interactive motions among agents, the second step is essential. Here, we define an existence of collision with the collision region R^k_{ij} as:

$$C_{ij}^{k} \equiv \begin{cases} 0\ ; \textit{no collision exist} \\ 1\ ; \textit{collision exist}. \end{cases} \tag{8}$$

It can be simply determined by checking the crossing, or touching, between R^k_{ij} and the trajectory of agent i. For example, the trajectory of agent 2 in Fig. 6 crosses collision region R^1_{21}, thus the value of C^1_{21} is one.

If no collision exists between agents i and j, it is expressed as:

$$\sum_{k=1}^{K} C_{ij}^{k} = 0, \tag{9}$$

where K is the total number of multiple collision regions between two agents. Since the shape of a path is allowed to be arbitrary, multiple collision regions may exist between two agents. This is important for the speed reduction strategy. Multiple collision regions R^1_{31}, R^2_{31} between agents 3 and 1 are shown in Fig. 6. Thus, the equation (9) represents the first step of a collision-free state between two agents.

Next, if agent i is under collision-free state with all higher priority agents, we can express it in the following equation as:

$$\sum_{j=1}^{i-1} \sum_{k=1}^{K} C_{ij}^{k} = 0, \tag{10}$$

where j denotes the higher priority agent to agent i. This equation of the second, includes that of the first step.

Finally, if all agents are under collision-free state, multi-agent motion planning is accomplished and can be described as the following final equation:

$$\sum_{i=2}^{N} \sum_{j=1}^{i-1} \sum_{k=1}^{K} C_{ij}^{k} = 0, \tag{11}$$

where N is the total number of agents. We do not need to consider the collision-free state of the highest priority agent (i=1) in the concept of prioritized planning.

In the case of Figs. 5 and 6, C^1_{21} is 1, which indicates that agent 2 will collide with agent 1 in R^1_{21}. And both C^k_{31} (k=1,2) and C^1_{32} are zero, so the collision-free state of agent 3 is also expressed as :

$$\sum_{j=1}^{2} \sum_{k=1}^{K} C_{3j}^{k} = 0. \tag{12}$$

Finally, the collision-free state of Fig. 5 is expressed as :

$$\sum_{i=2}^{3} \sum_{j=1}^{i-1} \sum_{k=1}^{K} C_{ij}^{k} = 1. \tag{13}$$

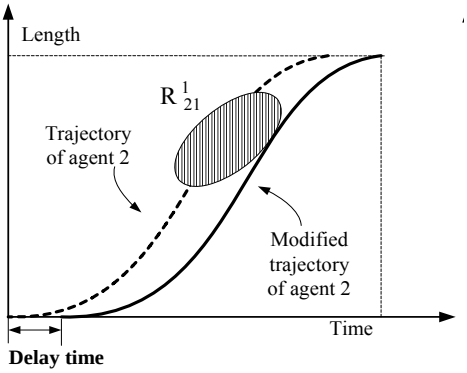
This equation suggests that one potential collision exists among three agents in the example.

4.3.4. Collision resolution using extended collision map

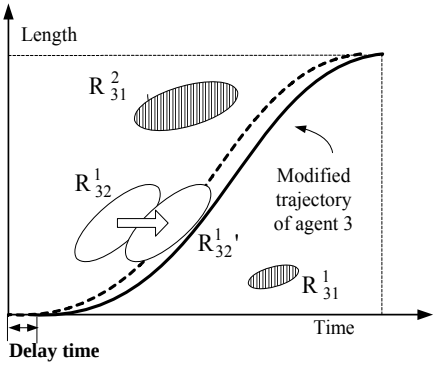
Applying the extended collision maps to each agent iteratively, we can avoid the potential collisions among agents. Fig. 7 shows the resolution process of the previous example. First, agent 1 can execute its motion, regardless of any agent motion, due to the highest priority. Next, the trajectory of agent 2 is modified to avoid the potential collision and becomes a collision-free state as shown in Fig. 7(a). Then the collision-state equation has the zero value:

$$\sum_{j=1}^{1} \sum_{k=1}^{K} C_{2j}^{k} = \sum_{k=1}^{1} C_{21}^{k} = C_{21}^{1} = 0. \tag{14}$$

Next, the central planner tries to resolve the potential collisions for agent 3. Note that information on the extended collision map of agent 3, however, was changed in the previous process. The collision region, which is related to agent 2 on the extended collision map of agent 3, is generated with the trajectory information of agent 2 and the modification of agent 2's motion changes the existence of potential collisions of agent 2 with agent 3. For this reason, the collision regions related to agent 2 on the extended collision map of agent 3 are no longer valid, which is a significant feature of the extended collision map. We



(a) Case of agent 2.



(b) Case of agent 3.

Fig. 7. Collision resolution of agent 2 and 3 in case of Figs. 5 and 6.

call it the indirect effect from higher priority agents. It means that the valid extended collision map should incorporate the result of the trajectory modification of the higher priority agents.

Fig. 7(b) indicates this dynamic effect on the extended collision map of agent 3 and the collision resolution process. From the result of the modification in Fig. 7(a), the collision region R^1_{32} is changed into R^1_{32}', which leads to a new potential collision denoted by

$$\sum_{k=1}^{1} C_{32}^{k} = C_{32}^{1} = 1. \tag{15}$$

For the resolution of the changed collision region, the central planner modifies the trajectory of agent 3 as shown in Fig. 7(b), and it is expressed as:

$$\sum_{k=1}^{1} C_{32}^{k} = C_{32}^{1} = 0. \tag{16}$$

Since no collision exists between agents 3 and 1, agent 3 is under collision-free state as:

$$\sum_{j=1}^{2}\sum_{k=1}^{K} C_{3j}^{k} = \sum_{k=1}^{2} C_{31}^{k} + \sum_{j=1}^{1} C_{32}^{k} = 0 + 0 = 0. \tag{17}$$

Finally, we can achieve the collision-free motion planning of multi-agents in Fig. 5 according to the priority order, where agent 1 has the highest priority, agent 2 next, and agent 3 has the lowest priority.

$$\sum_{i=2}^{3}\sum_{j=1}^{i-1}\sum_{k=1}^{K} C_{ij}^{k} = \sum_{j=1}^{1}\sum_{k=1}^{K} C_{2j}^{k} + \sum_{j=1}^{2}\sum_{k=1}^{K} C_{3j}^{k} = 0 + 0 = 0. \tag{18}$$

5. COMPUTATIONAL APPROACH

The general schematic idea for multi-agent motion planning, the extended collision map, was presented in Section 4. We now show the detailed method to implement the idea in this section. In the previous studies, the concept of detection and resolution for two collision-free agents was presented, however, the detailed method to implement the concept was not considered for multi-agents because of the computational burden [24,25]. Recent computer technology now has the ability to implement for multi-agents, thus, the computational algorithm is proposed for implementation. The computational algorithm deals with two geometric shapes-the path of the agent is arbitrary and the contour of collision region is also arbitrary-in motion planning for multi-agents. Other approaches, such as mathematical or analytic approach, may be faced with difficulty to be formulated for this arbitrary shape.

Table 1. Brief algorithm of motion planning for multi-agents.

Input	① Paths of all agents ② Velocity profiles of all agents
Output	Collision-free velocity profiles of N-1 agents
Loop	*For* all agents (i=2 to N) *Do*
Step 1	**<Discretizing path of agent>**
1.1	Sample points on path of agent i
Step 2	**<Detecting potential collisions>**
For all higher priority agents (j=1 to i-1) *Do*	
2.1	Sample points on path of agent j
2.2	*For* all sampling points of agent i *Do*
	Calculate distance between two points of agents i,j
	If distance < r_1+r_2 *Then* a potential collision exist
	Generate a collision point on the map
	Else continue
	End of Loop-sampling points :Generate a collision region between two agents i and j
End of Loop - j : Generate collision regions related to agent i	
2.3	Incorporate trajectory of agent on map
Step 3	**<Resolving potential collisions>**
Repeat Do	
3.1	*If* trajectory passes through any region *Then*
3.2	Modify the trajectory to avoid the region
	Else continue
Until No collision exists on the map	
End of Loop i : *completion of motion planning*	

5.1. Computational algorithm for motion planning

The proposed computational algorithm consists of three parts (See Table 1 for details). First, the continuous paths of agents are discretized. Detection and resolution of potential collisions follow the discretization for the conflict resolution. The algorithm is based on the concept of agent task priority. Two pieces of information are given as input data: path and velocity profiles. The output of the algorithm is the collision-free motion information on all agents, except the highest priority agent.

5.2. Detection of potential collisions

Fig. 8 describes the process of detecting the potential collisions. By discretizing the continuous path of agent *i*, we can obtain sampling points, p_m and p_n, defined in Table 2. Potential collisions are detected in the computational approach by comparison of $r = r_i+r_j$ and distance d_{mn} between the two points p_m and p_n: If $d_{mn} \le r$, then *a potential collision exists.*

We solve the quadratic equation for collision detection, while the distance comparison is used in

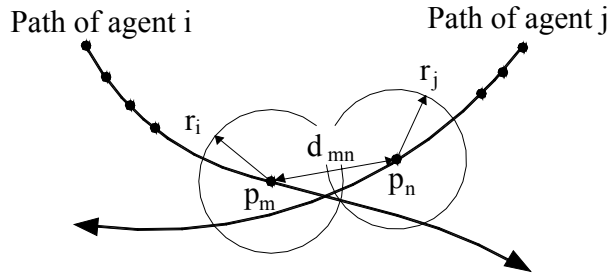


Fig. 8. Paths of two agents i and j and detection of potential collision from the paths.

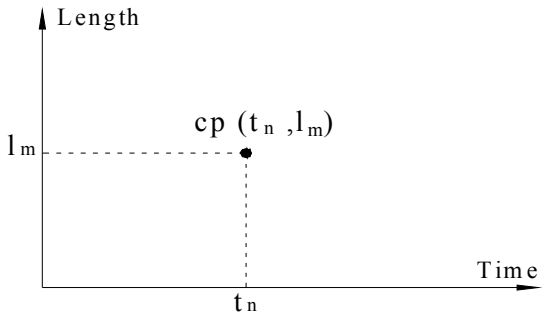


Fig. 9. A collision point corresponding to p_m and p_n in Fig. 8.

computational algorithm. If we find a sampling point with a potential collision, the point can be marked on the extended collision map, which is called the 'collision point' as shown in Fig. 9. In Section 3, we have shown that the collision information in the extended collision map is fundamentally generated with the elapsed time of the higher priority agent and traveled length of the target agent. Thus, the collision point $cp(t_n, l_m)$ in Fig. 9 is made from the elapsed time of the point p_n and traveled length of the point p_m. Collision regions are generated by merging the adjacent collision points on which there is a potential collision, with respect to the sampling points of two agents' paths, which is shown in Step 2.2 of Table 1.

The extended collision map of an agent is also formed simply with this algorithm. This simple process, only distance comparison, can reduce the computational load. Additionally, distance comparison can be performed regardless of the shape of the path, thus an agent is able to take any shape of path. Consequently, potential collisions can be easily detected with the concept of collision point in the computational approach.

Table 2. Notation and meaning in Figs. 8 and 9.

Notation	Meaning
r_i, r_j	radius of agent i, j
p_m, p_n	a sampling point on paths of agent i, j, $1 \le m \le M$, $1 \le n \le N$ M, N : total number of sampling points of path i and j
d_{mn}	distance between p_m and p_n
$cp(t_n,l_m)$	a collision point corresponding to p_m and p_n

5.3. Resolution of potential collisions

Potential collisions are resolved when a trajectory avoids all collision regions in the extended collision map. Thus, a velocity profile has to be modified with a computational collision-free strategy if any touching or crossing exists between a trajectory and a collision region. The time delay and speed reduction strategy with collision box were presented in the previous study [24,25]. The advantage of this study is in simple calculation, and the disadvantage is in the loss of traveling time. In this paper we adopt the concept of minimum time delay strategy and use the concept of the collision point instead of the concept of the collision box for planning the collision-free motions of multi-agents with the arbitrary shaped paths.

Checking for a crossing or touching between a trajectory and a collision region is the first procedure in the collision resolution process in computational approach. Fig. 10 simply describes the method that checks for crossing or touching. The initial trajectory is denoted with *Traj.a.* We calculate the distance between sampling points on the trajectory and collision points inside the collision region. If the distance is less than the preset tolerance number (TOL), we consider a collision to be predicted at the sampling point (black point on Fig. 10); therefore, the central planner dismisses the trajectory and delays the starting time of the trajectory by a predetermined unit delay time (**dk**) and obtains *Traj.b*. The planner repeats this procedure until it finds a collision-free trajectory (*Traj*.c).

Next, we find a complete collision-free trajectory. Since trajectory modification may cause another crossing or touching, we should always recheck the collision-free state in the extended collision map after

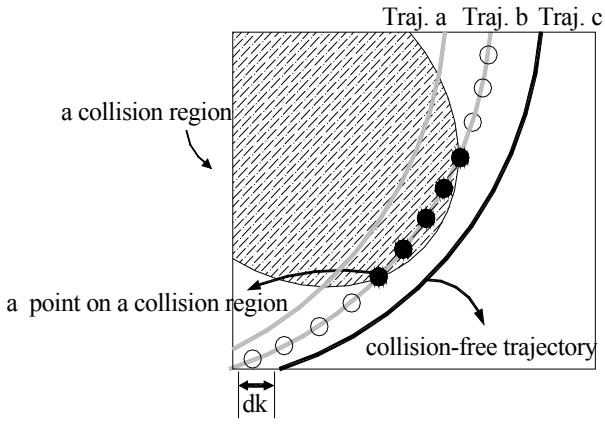


Fig. 10. Computational method to find a collision-free trajectory on a collision region.

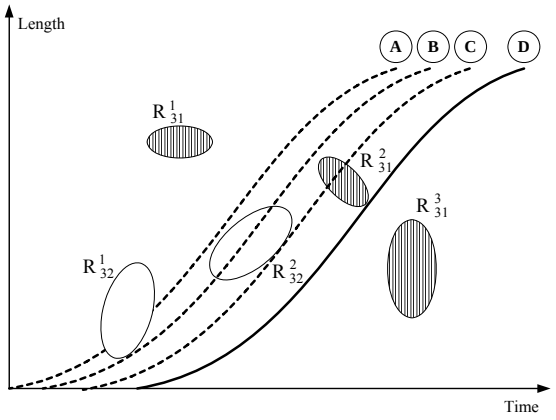


Fig. 11. An example of computational method to obtain a collision-free trajectory on the extended collision map.

any trajectory modification. Note that this rechecking process is not always performed in the priority order. Fig. 11 shows an example of this non-priority process. Trajectory Ⓐ is given initially, which crosses collision region R^1_{32}. To avoid the crossing, the central planner modifies the trajectory Ⓐ into Ⓑ. But the new trajectory Ⓑ causes a new crossing with R^2_{32}, where the previous trajectory Ⓐ did not pass. Since these collision regions are not located in priority order, we should check for a crossing or touching around all collision regions. In this way, a collision-free trajectory Ⓓ can be obtained from this process.

6. IMPLEMENTATION

The proposed method has been implemented for three heterogeneous multi-agents in Fig. 12. Fig. 13 shows the initial locations and their paths for the three

Fig. 12. Three heterogeneous multi-agents.

Table 3. Physical constraints of three agents.

Agent	Priority	Radius	Max. Vel.	Max. Accel.
A1	1	35cm	1.5 m/s	0.4 m/s^2
A2	2	35cm	1 m/s	0.3 m/s^2
A3	3	35cm	0.8 m/s	0.2 m/s^2

agents. Table 3 shows a priority and physical constraints of each agent in the implementation. Three agents are assumed to have different characteristics, including maximum velocities and accelerations in navigation. The trapezoidal velocity profiles of the agents are determined under the given dynamic constraints. The sampling time for discretizing the continuous path is 0.1sec and the tolerance number for checking crossing or touching between the trajectory and the collision region is 0.1sec in the extended collision map. The workspace for multi-agents is 7m x 5m, and the Pentium IV 2.4GHz processor is used for this implementation.

There are four intersections in the workspace as shown in Fig. 13. These are collision regions in the extended collision maps of agents 2 and 3 as shown in Fig. 14 and 15. There are two potential collisions: R^1_{21}, R^1_{32}; therefore, the collision-free state equation is denoted as:

$$\sum_{i=2}^{3}\sum_{j=1}^{i-1}\sum_{k=1}^{K} C_{ij}^{k} = 2. \tag{19}$$

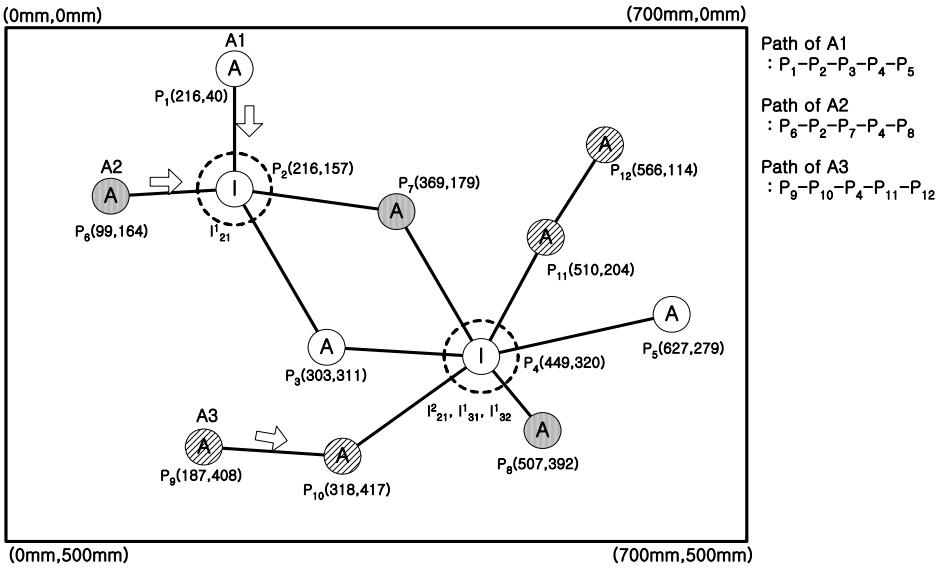
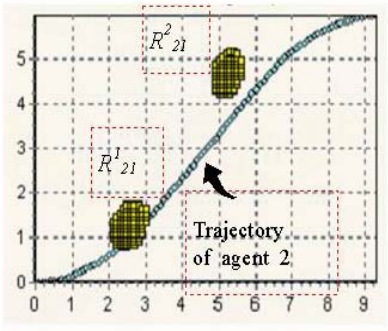
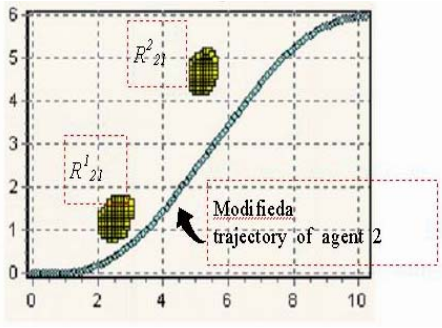


Fig. 13. Path information for three agents.

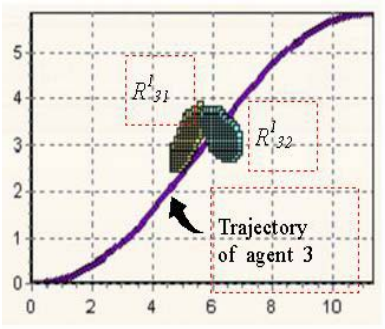


(a) Initial state.

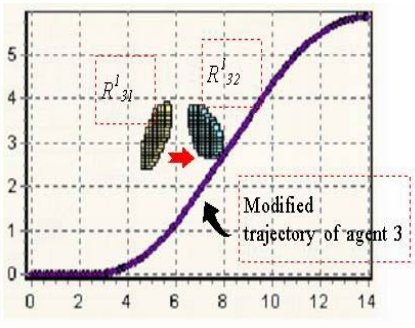


(b) Collision-free state.

Fig. 14. Resolution of potential collisions of agent 2 using the extended collision map.



(a) Initial state.



(b) Collision-free state.

Fig. 15. Resolution of potential collisions of agent 3 using the extended collision map.

For the resolution of potential collisions, the central planner modifies the trajectory of agent 2 with the minimum time delay (See Fig. 14(b)). This modification of agent 2 results in the new collision regions in the extended collision map for the lower priority agent as stated in Section 4. In Fig. 15(b), the collision region R^1_{32} is related to agent 2 and thus it is shifted to the right. With this modified extended collision map, the central planner is able to accomplish the collision-free motion planning completely as:

$$\sum_{i=2}^{3}\sum_{j=1}^{i-1}\sum_{k=1}^{K} C_{ij}^{k} = 0. \tag{20}$$

Table 4 shows the result of motion planning of this example. In fact, this result is obtained easily from the implementation. We identify the feasibility of motion planning from this result. The total time consumed in traveling the full path for all agents is presented in Table 4, for three collision-free strategies. Agent 1 has the right to move without any interference, and thus, its traveling time is always the same, while agents 2 and 3 have different traveling times depending on the collision-free strategy. Minimum time delay strategy is found to be the most effective in this motion planning. Compared to 13.15sec as the planned traveling time, 63ms as the computation time is very small. So, it can be applicable to controlling multi-agents in the real world because it can plan collision-free motions of multi-agents within a feasible time.

Table 4. Traveling and computation time of agents in Fig. 15.

Agent	Traveling distance	Traveling time by algorithms		
		Time Delay	Speed Reduction	Minimum Time Delay
A1	6.84m	8.33sec	8.33sec	8.33sec
A2	5.96m	10.21sec	11.77sec	9.31sec
A3	5.86m	13.95sec	17.30sec	13.15sec
Computation time		under 1ms	under 1ms	63ms

The computation time shown in Table 4 is an important parameter for real-time implementation. Time delay and speed reduction algorithm require, at most, 1 ms in the example. Minimum time delay algorithm, however, needs 63ms for computation time. Considering that the motion, with the traveling time 13.15sec is planned, the computation time is acceptable. Nevertheless, we need to focus on the traveling time difference between the minimum time delay and others, because the approximation of a collision region as a collision box reduces the computation burden greatly.

7. CONCLUSIONS

The purpose of this study is to present a new

approach to multi-agent motion planning. The extended collision map method has been developed for this purpose. It shows the significant advantage that all information on potential collisions in multi-agents is provided in the map; therefore, a supervisory system can easily plan complete collision-free motions for multi-agents. To realize this tool practically, we use the interactive computational approach. Thanks to simple algorithms and acceptable computation time, we can use the extended collision map method for congested multi-agents. In conclusion, the result of this study indicates that collision-free motion planning of multi-agents can be accomplished easily as shown in the implementation.

The following questions are left for future studies. The first one is to obtain a global optimal solution in multi-agent motion planning. Several concepts such as agent priority reassignment can be taken into account in this topic. The next, is the case of the emergence of an unexpected object, such as a human, in the navigation. In this fortuitous case, motions of agents should be recalculated immediately according to the movement of an unexpected object.

The proposed method in this paper can be used in most multi-agent environments such as airports, harbors, and big buildings using multiple robots. For example, there is time enough for identifying the routes and the velocity profile of airplanes and planning collision-free motions of all the planes through some wide area communication technologies including GPS and satellite communication. All airplanes can track their given paths with the bounded cross track error, which can be estimated by numerical methods. Therefore, a supervisory system in an airport is able to control all the airplanes automatically with this method.

REFERENCES

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

[2] K. Fujimura, *Motion Planning in Dynamic Environment*, Springer-Verlag, NewYork, 1991.

[3] M. B. Dias and A. Stentz, "Opportunistic optimization for market-based multirobot control," *Proc. of IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, 2002.

[4] J. H. Reif, "Complexity of the mover's problem and generalizations," *Proc. of the 20th IEEE symp. on Foundations of Computer Science*, pp. 412-427, 1979.

[5] J. F. Canny, *The Complexity of Robot Motion Planning*, MIT Press, 1988.

[6] S. Akella and S. Hutchinson, "Coordinating the motions of multiple robots with specified trajectories," *Proc. of the IEEE Int. Conf. on Robotics & Automation*, Washington DC, May, 2002.

[7] R. R. Murphy, *Introduction to AI Robotics*, The MIT Press, 2000.

[8] J. Hasegawa, K. Kurihara, and N. Nishiuchi, "Collision-free path planning method for mobile robot," *Proc of IEEE Int. Conf. on Systems, Man and Cybernetics*, 2002.

[9] C. Mudgal and J. Vassileva, "An influence diagram model for multi-agent negotiation," *Proc. of IEEE Int. Conf. on MultiAgent System*, pp. 451-452, July, 2000.

[10] Y. K. Hwang and N. Ahuja, "Gross motion planning - a survey," *ACM Computing Surveys*, vol. 24, no. 3, pp. 219-291, September 1992.

[11] J. Barraquand, L. Kavraki, J. C. Latombe, T. Y. Li, and P. Raghavan, "A random sampling scheme for path planning," *Int. Journal of Robotic Research*, vol. 16, no. 6, pp. 759-774, Dec, 1997.

[12] J. Barraquand and J. Latombe, "Robot motion planning: A distributed representation approach," *Int. Journel of Robotics Research*, vol. 10, pp. 628-649, 1991.

[13] L. Kavraki, P. Svestka, J. Latombe, and M. Overmars, "Probabilistic roadmap for fast path planning in high-dimensional configuration spaces," *IEEE Trans. on Robotics and Automation*, vol. 12, pp. 566-580, 1996.

[14] T. Y. Li and H. C. Chou, "Motion planning for a crowd of robots," *Proc. of IEEE Int. Conf. on Robotics & Automations*, September 2003.

[15] K. S. Barber, T. H. Liu, and S. Ramaswamy, "Conflict detection during plan integration for multi-agent systems," *IEEE Trans. on Systems, Man, and Cybernetics*, vol. 31, no. 4, pp. 616-627, August 2001.

[16] M. M. Quottrup, T. Bak, and R. Izadi-Zamanabadi, "Multi-robot planning: A timed automata approach," *Proc. of IEEE Int. Conf. on Robotics and Automation*, pp. 4417-4422, April, 2004.

[17] K. Azarm and G. Schmit, "Conflict-free motion of multiple mobile robots based on decentralized motion planning and negotiation," *Proc. of IEEE Int. Conf. on Robotics and Automation*, pp. 3526-3533, April 1997.

[18] K. S. Barber, T. H. Liu, and D. C. Han, "Agent-oriented design," *Proc. of the Ninth European Workshop on Modeling Autonomous Agents in a Multi-Agent World, Valencia, Spain, Lecture Notes in Computer Science: Lecture Notes in Artificial Intelligence,* F. J. Garijo and M.Boman, Eds., Spring-Verlag, NewYork, 1999.

[19] J. H. Lee and H. Hashimoto, "Intelligent space," *Proc. of IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, vol. 2, pp. 1358-1363, Nov. 2000.

[20] O. Kubitz, M. O. Berger, and R. Dumoulin,

"Application of radio frequency identification devices to support navigation of autonomous mobile robots," *Vehicle Technology Conf.*, vol. 1, pp. 126-130, 1997.

[21] H. Norihiro, K. Kiyoshi, M. Kehji, and S. Yasuyuki, "Collaborative capturing of experiences with ubiquitous sensors and communication robots," *Proc. of IEEE Int. Conf. on Robotics & Automation*, pp. 4166-4171, September 2003.

[22] M. Erdmann and T. Lozano-Perez, "On multiple moving object," *Proc. of IEEE Int. Conf. on Robotics and Automation*, vol. 3, pp. 1419-1424, 1986.

[23] R. Siewart and I. R. Nourbakhsh, *Introduction to Autonomous Mobile Robots*, The MIT Press, 2003.

[24] B. H. Lee and C. S. G. Lee, "Collision-free motion planning of two robots," *IEEE Trans. on Systems, Main, and Cybernetics*, vol. 17, no 1, pp. 21-31, January/February 1987.

[25] C. Chang, M. J. Chung, and B. H. Lee, "Collision avoidance of two general robot manipulators by minimum delay time," *IEEE Trans. on Systems, Main, and Cybernetics*, vol. 24, no. 3, pp. 517-522, March 1994.

Sang-Hoon Ji received the B.S. and M.S. degrees in Control and Instrumentation Engineering and the Ph.D. degree in Electrical Engineering and Computer Sciences from Seoul National University, Seoul, Korea in 1995, 1997, and 2007, respectively. From 1997 to 2002, he was a Research Engineer at IAE, Yongin, Korea. Currently, he is a Senior Engineer of ASRI at Seoul National University.

Jeong-Sik Choi received the B.S degree in Agricultural and Machinery Engineering and the M.S. degree in Electrical Engineering from Seoul National University in 1998 and 2004, respectively. Since 2004, he is a Ph.D. candidate at the School of Electrical Engineering and Computer Sciences at Seoul National University, Seoul, Korea.

Beom-Hee Lee received the B.S. and M.S. degrees in Electronics Engineering from Seoul National University, Seoul, Korea in 1978 and 1980, respectively, and the Ph.D. degree in Computer, Information and Control Engineering from the University of Michigan, Ann Arbor, in 1985. From 1985 to 1987, he was with the School of Electrical Engineering at Purdue University, West Lafayette, IN, as an Assistant Professor. He joined Seoul National University in 1987, where he is currently a Professor at the School of Electrical Engineering and Computer Sciences. Since 2004, he has been a Fellow of the Robotics and Automation Society.