International Journal of Automation and Computing 16(4), August 2019, 475-490
DOI: 10.1007/s11633-018-1146-7

Time-space Viewpoint Planning for Guard Robot with Chance Constraint

Igi Ardiyanto[1] Jun Miura[2]

[1]Department of Electrical Engineering and Information Technology, Universitas Gadjah Mada, No. 2 Jl. Grafika, Yogyakarta 55281, Indonesia

[2]Department of Computer Science and Engineering, Toyohashi University of Technology, 1-1 Hibarigaoka, Tenpaku-cho, Toyohashi, Aichi 441-8580, Japan

Abstract: This paper presents a novel movement planning algorithm for a guard robot in an indoor environment, imitating the job of human security. A movement planner is employed by the guard robot to continuously observe a certain person. This problem can be distinguished from the person following problem which continuously follows the object. Instead, the movement planner aims to reduce the movement and the energy while keeping the target person under its visibility. The proposed algorithm exploits the topological features of the environment to obtain a set of viewpoint candidates, and it is then optimized by a cost-based set covering problem. Both the robot and the target person are modeled using geodesic motion model which considers the environment shape. Subsequently, a particle model-based planner is employed, considering the chance constraints over the robot visibility, to choose an optimal action for the robot. Simulation results using 3D simulator and experiments on a real environment are provided to show the feasibility and effectiveness of our algorithm.

Keywords: Guard robot, viewpoint planning, state-time space, uncertainty, topology, chance-constraint.

1 Introduction

Recently, robotic technologies have taken a forward leap and currently they are greatly demanded in many applications. Its utilization is broad, ranging from home appliances to industrial sectors. There are couples of rationale behind the expansion of the robots; to alleviate the human's jobs, to substitute the human on hazardous works, or to exploit its precision for some specific tasks. When a robot is required to replace or to support the human in a certain task, it implies that the robot needs to imitate, fully or partially, what the human workers do. This imitation can be in the form of actions, procedures, or even the human thinking perspective to finish the job. With regards to such problem, here we present one example of the task which a robotic framework is expected to perform, as described by the following problem setting.

Let us imagine an indoor environment such as museum, office, gallery, or exhibition room. Suppose there is a very important person (VIP, e.g., minister, chief, or officer) visiting the museum and is to be guarded. Ordinarily, a group of guardians (or so-called "securities") are assigned to do the guarding job. The most important task of these guardians is to watch over the VIP for the entire time, yet they should not disturb and restrict his/her mobility. Additional duties may be added as well, such as documenting the overall activities of the VIP.

Now we aim to substitute the human guardians with the robotic platform. The task imitation for the above case is straightforward, the robot should mimic the behavior of the human guardians. Nevertheless, several considerations need to be contemplated:

1) The number of robots used for guarding the VIP eventually becomes one important factor, if the cost is restricted.

2) The robot has limitation on the battery capacity, affecting its working time duration.

3) If documenting the activity of the VIP is included as the robot task, it is preferable to highly reduce the robot's instability (e.g., due to its excessive movement), to acquire a less-noise video or image.

4) The robot should be non-intrusive, i.e., it must not disrupt and alter the current activity of the VIP.

Considering the above restrictions, we propose a novel planning algorithm for a single guard robot to imitate the job of the human guardians. Since the robot price tends to be very expensive, here we emphasize on the use of single robot for guarding purpose as indicated by point 1) in the above consideration. The robot task is then minimally rephrased as follows:

"The guard robot should maintain the visibility towards the target person while minimizing its movement."

The term "minimizing its movement" is raised, basic-

Research Article
Manuscript received January 20, 2018; accepted June 29, 2018; published online September 22, 2018
Recommended by Associate Editor Qing-Long Han

ally for tackling the problems specified by points 2) and 3). By minimizing the robot movement, we expect to reduce the energy which may lead to a longer duration of the robot working time. It also means the robot tends to be in an idle condition. As the result, a less-noise video of the target person (when documentation is intended) can be obtained due to the stability improvement.

Based on the above expectations, our main idea is to make the robot move only when it is needed, i.e., whenever the target is predicted to leave the robot's field-of-view, and mostly stay at a certain location which holds a wide view. We then introduce viewpoint terminology in our proposed guard robot planner. The viewpoint is described as a point from which the idle robot can safely and steadily watch over the target person for a long time.

The viewpoints are utilized to assist the action planning processes. Our strategy is to move the robot from one viewpoint to another. By exploiting the viewpoints, we expect to reduce the search spaces of the robot. We also introduce a concept of escaping gaps to lessen the target person prediction space, which makes the guard robot problem become more tractable. Using this approach, the robot will not disturb the target person since it attempts to "see from a distance", granting the consideration described by point 4) above.

1.1 Related works

1.1.1 Art gallery problem

For the matter of guarding an indoor environment, the art gallery problem (AGP) is related to our proposed guard robot. AGP is defined as the problem of discovering a minimum number of guards, typically in the form of sensors or cameras, such that they cover all interiors of the environment. The AGP has been widely studied, especially by computational geometry communities (e.g., [1–4]).

The use of viewpoints in our proposed approach can be perceived as another form of AGP. In conjunction with the planner, our proposed algorithm creates a dynamic version of AGP, with a single robot be in charge of guarding the entire environment by visiting the viewpoints as needed. Contrarily, the original AGP statically puts a set of guards for the same purposes.

1.1.2 Pursuit-evasion

The classical pursuit-evasion problem aims to make the pursuer(s) capture the escaping evader(s). This problem is also highly addressed by a vast number of researches, e.g., [5–9]. Naturally, one may guess that our guard robot is a variant of pursuit-evasion problem where the evader (target person) does not try to escape. One notable thing is that in our guard robot problem, the game does not necessarily end when the target is already "captured". Instead, the robot will continuously and optimally act to cover the target until it leaves the environment.

1.1.3 Watchman route problem

This problem aims to plan the shortest route on which a robot or watchman can inspect every point inside a polygon. The watchman route problem (WRP) is also popular among the robotic and computational geometry researchers, e.g., [10–13]. At a glance, our proposed problem resembles the classical WRP, except that the guard robot has a target to be "followed" and its additional objective is to minimize the movement.

1.1.4 Person following robot

The most closely related works to our guard robot problem is the person following robot, which has a long history in the robotic researches (e.g., [14–17]). This algorithm is also suitable for solving the proposed problem. The main difference is that the person following algorithm continuously makes the robot move and attempts to be within a certain distance towards the target. In contrast, our approach versatilely tries to understand the environmental topology and the robot only moves when it is necessary to keep the target person under its vicinity. By this strategy, it is expected to reduce the energy used by the robot.

1.2 Our contributions

Our contributions mainly concern with the introduction of a novel framework and the cooperation between the topological viewpoints and an on-line planning strategy for solving the guard robot problem. It is also worth to note that we are raising a distinctively new variant of robotic problem, compared with the previously mentioned related works. This new task includes the problem of keeping a target under the robot visibility while reducing its movement.

The guard robot task poses a distinctive challenge. Unlike the camera placement problem where the switching between cameras can be done instantly, the guard robot has a "time delay" for navigating between two points. It enforces the guard robot to have a bound from which it guarantees that the target is always under the robot sight. It should be noted that the robot is non-holonomic which imposes another difficulty as explained in [18, 19], and the cooperation is not an option since we use only a single robot, unlike the one in [20]. Nevertheless, applying the plannning strategy for the multi robot cases is indeed a promising research direction. Interested readers are directed to see the works in [21–23] for such problems.

It is highlighted that this paper extends our previous work[24]. In this paper, we provide two significant enhancements; we improve the viewpoint selection method via cost-based set covering problem, and introduce a particle-based stochastic predictive method to establish the global planner for the guard robot.

1.3 Text organization

We organize the rest of this paper as follows. A math-

ematical definition of our guard robot is presented in Section 2. Section 3 describes the concept of viewpoints and escaping gaps utilized in our approach, as well as the strategy for simplifying the guard robot problem. Section 4 explains the extraction of viewpoints by utilizing the topological features. Section 5 describes the geodesic motion model for both the robot and the target person. Subsequently, the planning method applied to the guard robot problem is elaborated in Section 6. The proposed approach is then verified on various experiments in Section 7. In the end, the conclusion and future directions of this work are provided.

2 Problem definition

Consider a typical indoor workspace $\mathcal{Q} = \{\mathcal{Q}_f, \mathcal{Q}_n\} \subseteq \mathbf{R}^m$, $m \geq 2$, constrained by walls and possibly obstacles. $\mathcal{Q}_f$ and $\mathcal{Q}_n$ are respectively perceived as "passable" and "non-passable" regions for the robot. Let $\{q_t^r, q_t^h\} \in \mathcal{Q}$ denote the robot and the target person state at a certain time t. The robot is equipped with sensors to sweep the free space, creating a continuous boundary which depicts the "visible" area for the robot, denoted by $\oint B(q_t^r)$. The guard robot problem is subsequently described by the following mathematical expression:

$$\text{Min} \qquad \int_0^\infty \left| \frac{\partial(q_t^r)}{\partial t} \right| \mathrm{d}t \tag{1}$$

$$\text{s.t.} \qquad \oint B(q_t^r) \bigcap \overline{q_t^h q_t^r} = \emptyset \tag{2}$$

$$\{q_t^r, q_t^h\} \in \mathcal{Q}_f \tag{3}$$

where $\overline{q_t^h q_t^r}$ denotes a straight line connecting q_t^h and q_t^r.

The above expression implies we attempt to reduce the total movement of the robot, indicated by summation of the first order derivative of the robot state over time in (1), while the main constraint is to keep the target q_t^h within the visibility boundary of the robot (see (2)). Obviously, the robot state's derivative can be approximated by discrete-time dynamic model

$$\frac{\partial(q_t^r)}{\partial t} \approx f(u_t, \epsilon_t) : q_t^r \mapsto q_{t+1}^r, \quad t \in [0, \infty] \tag{4}$$

where $u_t \in \mathcal{U}$ is the control input and ϵ_t denotes the uncertainty. Hence, (1) gives an implication that we are minimizing controls applied to the robot.

The analytical solution of the above optimization problem is challenging due to the following reasons:

1) The future information of the target state q_t^h is not available at $t > 0$, assuming $t = 0$ is the current state, unless it is carefully modeled according to the environment (still, the uncertainty will be large).

2) The robot states, dynamics, and sensors may incorporate the uncertainty too.

3) Since the visibility boundary of the robot is unique for each state q_t^r, it means the optimization should examine the constraint over a huge space of all probable future states of both the robot and the target person.

4) The robot space of interest is very large too, whereas the entire workspace could become a destination for the robot to move, as long as it satisfies the visibility constraint.

3 Simplifying the guard robot problem

As mentioned above, the guard robot problem encounters some challenges. In a nutshell, those problems are a very large robot goal space and the future information uncertainty of the robot and the target person. Here we contemplate to ease those challenges using the following approaches.

3.1 Polygonal representation of the environment

We aim to relax the workspace $\mathcal{Q} = \{\mathcal{Q}_f, \mathcal{Q}_n\}$ into a simpler 2D planar polygonal form, as the environment map tends to have a complex shape. A set of procedures for simplifying $\mathcal{Q}$ is then implemented (similar steps can be found in [24]), as follows:

1) **Binarization**. A binary map $\mathcal{I}(q)$ is obtained by mapping each state $q \in \mathcal{Q}$ as

$$\mathcal{I}(q) = \begin{cases} 1, & \text{for } \forall q \in \mathcal{Q}_f \\ 0, & \text{otherwise.} \end{cases} \tag{5}$$

2) **Smoothing**. For reducing noises in the map, morphological operations are performed, using opening and closing operators.

3) **Contour extraction**. An algorithm introduced by Suzuki and Abe[25] is subsequently engaged to extract the contour from the binary map $\mathcal{I}(q)$, yielding an outer contour $\oint B^{outer}$ and (possibly) k-inner contours $\oint B_k^{hole}$. Both are a set of closed segment chains.

4) **Line segments simplification**. Finally, Douglas-Pecker algorithm[26] is used for simplifying the contours $\oint B^{outer}$ and $\oint B_k^{hole}$. It produces a closed, connected polygon $\mathcal{P}$ with the outer boundary $\delta\mathcal{P}$ and k-inner boundaries $\delta\mathcal{H}_k$ where k denotes the number of holes[1] inside $\mathcal{P}$. A point p satisfying

$$\{p \in \{\mathcal{P} \cap \neg(\bigcup^k \mathcal{H}_k)\}\} \tag{6}$$

is called the interior point of $\mathcal{P}$. From now, (6) becomes $p \in \mathcal{P}$ to describe the interior point p, for the sake of simplicity. The state $q \in \mathcal{Q}$ is also interchangeable with p such that $q \in \mathcal{P}$ has the same meaning as $p \in \mathcal{P}$.

[1]A geometrical term to define a closed polygon which is not connected to the exterior boundary.

Fig. 1 shows the instance of simplified map created from the original environment. The white area in Fig. 1 (c) represents the free space for the robot and target person to move.

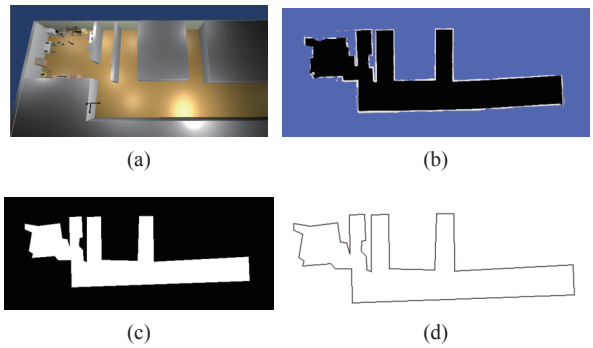


Fig. 1 Simplifying environment map: (a) Original environment; (b) Map from simultaneous localization and mapping (SLAM) algorithm; (c) Binarized map; (d) Extracted polygon.

3.2 Visibility polygon

Visibility polygon becomes one of fundamental problems in our guard robot, since we deal with the visibility constraint of the target person towards the robot. The visibility polygon of a point is given by Definition 1.

Definition 1. Let q be a point inside $\mathcal{P}$, another point $s \in \mathcal{P}$ is considered visible from q if $\overline{qs} \subset \mathcal{P}$, where $\overline{qs}$ is a line segment.

The visibility polygon $\mathbb{V}(q)$ is then defined as

$$\mathbb{V}(q) = \{\forall s \in \mathcal{P} \mid \overline{qs} \subset \mathcal{P}\}. \tag{7}$$

3.3 Concept of viewpoints for reducing the robot goal space

Let us take a look of Fig. 2. When static guards are used (e.g., cameras), the art gallery problem (AGP) can be adopted for solving the target person visibility problem. By a finite number of cameras, the AGP holds a guarantee to cover the entire building. Thereafter, to watch over a person inside it, one can easily switch his attention to the camera on which the person is visible. Fig. 2 portrays the camera switching process when a person moves from A to B. This also applies to all other cameras.

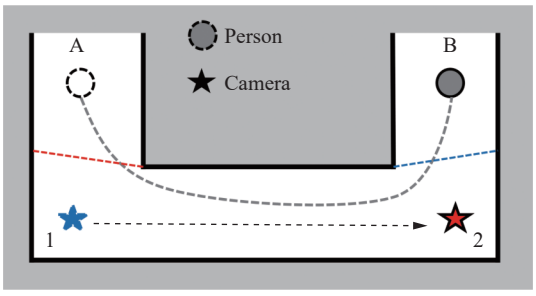


Fig. 2 Illustration of camera switching to keep the target under visibility. When the person moves from A until the blue line (camera 1 visibility), camera 1 does the "watching" task over the person. The "watching" task is then taken over by camera 2 from the blue line until the person reaches B. Color versions of the figures in this paper are available online.

By substituting each camera with a virtual point, let us imagine a robot is used to "see" the same person. The camera switching process now becomes the robot movement from the virtual point 1 to 2, to hold the same visibility towards the person. These virtual points are named viewpoints. The guard robot problem subsequently becomes the problem of determining the feasible viewpoint for the robot to watch over the target person.

The main point of the above illustration is, we basically expect to reduce the large goal space of the guard robot into a small set of viewpoints $\mathcal{V} = \{v_1, v_2, \cdots, v_n\}$, which hold the same visibility guarantee towards the target person. This property becomes the basis of our proposed approach for the guard robot problem.

3.4 Concept of escaping gaps

As we restrict our problem to an indoor environment, we can exploit the fact that the person movement should not violate the environment restrictions, e.g., walls. Accordingly, the possible scenarios for the robot to lose its visibility towards the target are when the target passes escapinggaps.

Escaping gaps are described as a set of locations at which the target may vanish from the robot's view. This idea is similar to the popular term frontiers for exploration of an unknown space (e.g., [5]).

Definition 2. Recall $\delta\mathcal{P}$ as boundary of the workspace polygon $\mathcal{P}$ and q is a point inside $\mathcal{P}$. Let $\delta\mathcal{V}(q_t^r)$ denote boundary of the visibility polygon of the robot at current time. The escaping gaps $\mathcal{G}$ are then defined as

$$\mathcal{G} = \left\{\forall q | q \in \left(\delta\mathcal{V}(q_t^r) \bigcap \neg\delta\mathcal{P}\right)\right\}. \tag{8}$$

Fig. 3 visually describes (8). The blue line in Fig. 3 represents escaping gaps which lie on the visibility polygon but do not lie on the workspace boundary. Typically, the escaping gaps $\mathcal{G}$ are in the form of a set of lines, as shown in Fig. 3. These lines are subsequently discretized into a set of points, such that every point in $\mathcal{G}$ is separated by a minimum distance d_g (currently, $d_g = 0.25\,\text{m}$).

While the viewpoints are important for simplifying the goal space for the robot, the escaping gaps are indispensable for relaxing the visibility constraint and the prediction space of the target person. Let $\forall\psi \in \Psi$ be all possible future paths which can be taken by the target person bounded by $\delta\mathcal{P}$. The path space is possibly infinite

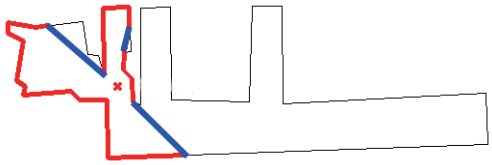


Fig. 3 Escaping gaps representation, denoted by blue lines. Together with the red lines, it forms the visibility polygon of the robot (marked by the red cross)

unless it is well modeled, denoted by $\Psi_{model} \subset \Psi$. Let $\Psi_g \subset \Psi, \forall g \in \mathcal{G}$ be the target person paths towards escaping gaps. Since Ψ is bounded, the only possible paths breaking the visibility constraint are Ψ_g, or, $\Psi_g \cap \Psi_{model}$ when the model is used.

There are two possible outcomes when the model is used:

1) $(\Psi_g \cap \Psi_{model}) \subseteq \Psi_g$, which is basically the path inside the model leading to escaping gaps.

2) $(\Psi_g \cap \Psi_{model}) = \emptyset$, which means the target person path never leaves $\mathbb{V}(q_0^r)$.

Subsequently, it is enough to generalize Ψ_g as the possible paths breaking the visibility constraint. We are then able to raise Definition 3.

Definition 3. Under the assumption that the target person q_t^h cannot pass through $\delta\mathcal{P}$ and it is initially inside the robot visibility $(q_0^h \in \mathbb{V}(q_0^r))$, any future action taken by the target person is guaranteed under visibility of the stationary guard robot $\mathbb{V}(q_0^r)$, except for the worst-case scenarios, i.e., it is leaving through escaping gaps $\mathcal{G}$.

Definition 3 gives the following consequences:

1) It implies we can set our focus only on predicting the worst-case scenarios instead of making a very accurate model for the target movement, to keep the target visible to the robot. Hence, we only need to have a reliable prediction of those worst-case conditions.

2) The robot does not need to take any action unless the target is predicted to leave the visibility scope via escaping gaps. This behavior directly fulfills our purpose, i.e., to reduce the robot movement.

Now, we have already reduced the prediction space of the target person, and the rest is to predict its movement towards escaping gaps. However, carrying out a long prediction of the human future movement is error-prone. Moreover, there is no guarantee that the target person always has an intention to go through escaping gaps. Hence, we update the target movement in an iterative fashion to obtain a reliable prediction.

The robot can taken action accordingly based on the above prediction of the worst-case conditions. In connection with the concept of viewpoints, the robot actions are either to go to a viewpoint which covers the possible escaping gap efficiently, or, to stay in the current robot states (e.g., by making $u_t = 0$, which is preferable) if the target is predicted to never pass through the escaping gaps.

3.5 Proposed framework for the guard robot

Using the above concepts, we re-establish the guard robot problem as follows:

$$\min \quad \sum u_t, \qquad t \in [0, \infty] \tag{9}$$

$$\text{s.t.} \quad \{q_t^h \in \psi_g\} \in \mathbb{V}(q_t^r \in \psi_r) \tag{10}$$

$$\psi_r \simeq f(u, \epsilon) : [q_0^r, v] \mapsto \mathbf{R}^m \tag{11}$$

$$\forall \psi_g \in \Psi_g \tag{12}$$

$$\forall v \in \{q_0^r, \mathcal{V}\} \tag{13}$$

$$\{q_t^r, q_t^h\} \in \mathcal{P}. \tag{14}$$

The above formulation principally tells the robot to choose the viewpoint destination which minimizes the future control u_t (see (9)) and always holds the target visibility towards the robot (see (10)). The constraint in (10) indicates that each state of the person staying on its path towards the escaping gap ψ_g (see (12)) must be visible from the robot at anytime t. Function $f(\cdot)$ in (11) represents the robot kinematic model for moving to viewpoint v, parameterizing the robot path ψ_r, in order to keep track and encounter the targeted person as suggested by (10). Equation (13) gives the robot a choice to stay at the current state q_0^r (i.e., $u_t = 0$) when it is deemed the target will never leave the robot visibility (e.g., the environment is completely convex or the target does not move at all).

We then propose a framework for solving the guard robot problem by dividing it into two stages: off-line and on-line stages. In the off-line stage, we examine the environment to get viewpoints. We then use those viewpoints to make action plans for the robot real-time, according to the current prediction of the robot and the target states. Fig. 4 shows our proposed framework. Section 4 will describe the detail of viewpoint extraction and how the action plans are executed in response to the predicted target movement.

4 Determining viewpoints for the guard robot

We basically exploit topology of the environment to obtain the viewpoints. Here, human reasoning and intuitions are imitated. Intuitively, topological features like the intersection is deemed as a place where one can stay for a long time to take the video since it covers a wide area (i.e., the intersection connects several corridors or hall way). Contrarily, the corner of a room does not have such benefit like the intersection, but it may encounter

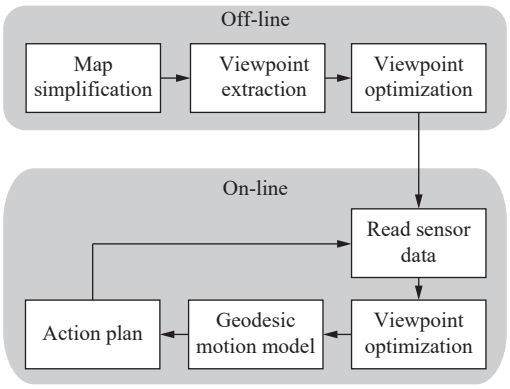


Fig. 4 Block diagram of proposed movement planning algorithm

some details of the room. By considering such features, we expect to get a set of viewpoints which is suitable for our guard robot problem.

4.1 Viewpoint candidates from topological features

For obtaining viewpoints, we first extract the possible location of the viewpoints using the human intuition as mentioned above. Here the topological features of the environment are adopted and selected as the candidates, as follows:

1) Vertex viewpoints. The polygon vertices are among the important features in a polygon, which are also employed by the Art Gallery Problem, such as [2] and [4]. In reality, a vertex of a polygon represents a corner of a room on which an observer (e.g., guard robot) may be placed. To obtain the vertex viewpoints, all polygon vertices are simply taken out, including its holes (if any). It should be noted that the actual placement of the guard robot on these vertex viewpoints should consider the distance towards the wall, as will be later explained in the viewpoints optimization section.

2) Skeleton viewpoints. As described above, an intersection of corridors in a building may grant wider view, compared with the wall or the corner of the room. It leads us to deliberate the topological shape of the environment for the considered guard robot problem. Therefore, a skeletonization technique for capturing topology of the polygon is utilized.

For acquiring the skeleton vertices, a skeleton map is initially constructed using the Laplacian of distance transform technique[24]. We build a distance transform map $\mathcal{D}$ (Fig. 5(a)) given by

$$\mathcal{D}(q) = \begin{cases} \|q - q'\|, & \text{for } q \in \mathcal{P}, q' \in \{\delta\mathcal{P}, \delta\mathcal{H}_k\} \\ 0, & \text{otherwise} \end{cases} \tag{15}$$

where q' is the nearest non-passable point (e.g., walls) to q.

A Laplacian operator is subsequently applied to $\mathcal{D}$, to obtain the skeleton map $\mathcal{K}$, denoted by

$$\mathcal{K}(q) = \sum_{i=1}^{m} \frac{\partial^2 \mathcal{D}}{\partial q_i^2} \tag{16}$$

where q_i denotes the i-axis of the point q, and m is the dimension of $\mathbf{R}^m$ (hence, $m = 2$). $\mathcal{K}$ is then binarized by a threshold. The skeleton viewpoints are then derived by extracting all junctions and endpoints of the skeleton (Fig. 5(b)). Both types of viewpoints (vertices and skeleton) are then coalesced, yielding a set of viewpoints candidates $\mathcal{V}$.

(a) (b)

Fig. 5 Skeleton viewpoints extraction: (a) Distance transform of the map; (b) Acquiring skeleton vertices. The circles denote the skeleton viewpoints.

4.2 Cost-based set cover problem for viewpoints optimization

Optimizing the viewpoint candidates to cover the entire environment becomes our next task. Since the optimization of the covering task, e.g., the original art gallery problem, is generally considered as non-deterministic polynomial (NP)-hard[27], here a hybrid optimization is proposed. First, the coverage problem is transformed into a set covering problem, a family of integer linear programming. A heuristically probabilistic approach is then performed for further optimization.

4.2.1 Arrangement of the viewpoint's visibility

We first introduce a prerequisite transformation to bring the viewpoint optimization into a set covering problem, that is the polygon arrangement. By borrowing the definition of the arrangement from [28], given a finite set of viewpoint candidates $\mathcal{V}$, we acquire a set of visibility polygon $\mathbb{V}(\mathcal{V})$. The arrangement $\mathcal{A}(\mathcal{V})$ is then defined as the subdivision of the plane created by the intersection of all vertices of $\mathbb{V}(\mathcal{V})$, such that $\mathcal{A}(\mathcal{V}) : \mathbb{V}(\mathcal{V}) \mapsto \mathcal{F}_c$. Each subdivision area of $\mathcal{A}(\mathcal{V})$ is called a face, denoted by $f_c \in \mathcal{F}_c$. Fig. 6 shows the definition of the arrangement and face.

Inversely, the visibility polygon of a viewpoint $\mathbb{V}(v)$ can be constructed as a set of faces f_c "seen" by the

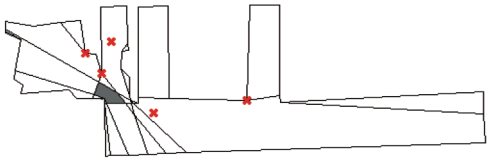


Fig. 6 Arrangement of a set of viewpoints. The red crosses represent the viewpoint candidates. All edges are the result of combining the visibility polygon of all viewpoints. The gray area is one of the faces created by the arrangement.

viewpoint $v \in \mathbb{V}$, such that

$$\mathbb{V}(v) \approx \{ \bigcup_{\forall f_c \in \mathcal{F}_{c1}} f_c | \mathcal{F}_{c1} \subset \mathcal{F}_c \} \tag{17}$$

where

$$\mathcal{F}_{c1} = \{ \forall f_c \in \mathbb{V}(v) \}. \tag{18}$$

Subsequently, a definition of the polygonal coverage is raised as follows.

Definition 4. The coverage of a polygon $\mathcal{P}$ by a finite set of viewpoints $\mathcal{V}$ is guaranteed under circumstances

$$P = \bigcup_{\forall f_c \in \mathcal{F}_c} f_c \tag{19}$$

where $\mathcal{F}_c$ are composed by the arrangement of $\mathbb{V}(\mathcal{V})$.

Strictly speaking, all faces will always cover the polygon as long as its composing set of viewpoints $\mathcal{V}$ also has a full coverage. We accordingly search for the optimal number of $\mathcal{V}$ which satisfies Definition 4.

4.2.2 Viewpoint optimization as the set covering problem

There are two consequences given by (17) and (19) :

1) There may exist a face "seen" by more than one viewpoint, or geometrically

$$\mathbb{V}(v_1) \bigcap \mathbb{V}(v_2) \approx \mathcal{F}_{c1} \bigcap \mathcal{F}_{c2} \neq \emptyset, \quad \{v_1, v_2\} \in \mathcal{V}. \tag{20}$$

It simply means some viewpoints may cover the same area.

2) Summation of all faces in $\mathcal{F}_c$ should cover the entire polygon $\mathcal{P}$, in order to comply with Definition 4.

It leads to the problem of assigning the smallest set of viewpoints $\mathcal{V}$ such that summation of their faces in the $\mathcal{A}(\mathcal{V})$ satisfies (19). Accordingly, the viewpoint coverage can be formulated into integer linear programming, more precisely, a set covering problem.

Consider an $M \times N$ matrix $\boldsymbol{A}$, the set covering problem (SCP) is described as a problem of finding a subset of the columns of $\boldsymbol{A}$ which covers all rows at a minimum cost[29]. The viewpoint coverage over a polygon is then defined using the SCP formulation as

$$\min \quad \sum_{j=1}^{J} v_j c(v_j), \quad \text{for } v_j \in \mathcal{V} \tag{21}$$

$$\text{s.t.} \quad \sum_{j=1}^{J} a_{ij} v_j \geq 1, \quad \{i = 1, \cdots, I\} \tag{22}$$

$$v_j \in \{0, 1\}, \quad \{j = 1, \cdots, J\} \tag{23}$$

$$a_{ij} \in \{0, 1\}, \quad a_{ij} \in \boldsymbol{A} \tag{24}$$

where I and J respectively denote the number of faces of the arrangement and that of the viewpoint candidates.

The above minimization implies the solution of the viewpoint coverages is obtained by making the viewpoints to be the sets used for covering and enforcing the faces of the arrangement as the elements to be covered. The viewpoint placement inside the polygon is modeled by testing it using a binary condition ($v_j = 1$ if the viewpoint is inserted into the set). Furthermore, a face $row(a_{ij})$ is set to 1 if it is seen by the viewpoint v_j (see (24)). Inequality of (22) ensures that a certain row must be covered by at least one column (i.e., a face should be covered by at least one viewpoint). It will guarantee that $\mathcal{P}$ will be fully covered.

Function $c(v_j)$ in (21) indicates the use of a non-unicost variant of SCP. $c(v_j)$ can be any arbitrary function which exhibits the importance of each viewpoint. Consider a robot is used as the viewpoint. It is supposed to be located not too close to the walls, to avoid a collision. This problem is then accommodated using a distance function as

$$c(v_j) = \mathrm{e}^{-\|v_j - \delta\mathcal{P}\|}. \tag{25}$$

Equation (25) implies the robot is preferred to be located far from the walls.

4.2.3 Iterative probabilistic viewpoint optimization

Algorithm 1. Iterative viewpoint optimization

Require:

1) $\mathcal{V}_0$: initial set of guard candidates
2) $|\mathcal{V}_0|$: cardinality of $\mathcal{V}_0$

Ensure:

3) $\mathcal{V}_{opt}$: optimized guards
4) **procedure** IterGuardOpt ($\mathcal{V}_0$)
// Initial optimization
5) $\mathcal{V}_{opt} \leftarrow$ **solveSCP**($\mathcal{V}_0$)
6) $U \leftarrow |\mathcal{V}_{opt}|$ /* initial upper bound */
// The real loop starts here
7) **faceCostMap**($\mathcal{V}_{opt}$) /* see (26) */
8) $\mathcal{V}_{sampling} \leftarrow$ **sample**($\mathcal{V}_{opt}, 2|\mathcal{V}_{opt}|$) /* see text */
9) $\mathcal{V}_+ \leftarrow$ **solveSCP**($\mathcal{V}_{opt} \bigcup \mathcal{V}_{sampling}$) /* see (21) */
10) **If** $|\mathcal{V}_+| > U$ **then**
11) $\mathcal{V}_{opt} = \mathcal{V}_{opt} \bigcup \mathcal{V}_+$
12) **go to** 8
13) **else if** $|\mathcal{V}_+| < U$ **then**
14) $\mathcal{V}_{opt} = \mathcal{V}_+$
15) $U \leftarrow |\mathcal{V}_{opt}|$
16) **go to** 7
17) **else**
18) **if** $D_{hd}(\mathcal{V}_+, \mathcal{V}_{opt}) >$ threshold **then** /* see (27) */
19) $\mathcal{V}_{opt} = \mathcal{V}_{opt} \bigcup \mathcal{V}_+$
20) **go to** 7
21) **else**
22) $\mathcal{V}_{opt} = \mathcal{V}_+$
23) **break**
24) **end if**

25) **end if**
26) **return** $\mathcal{V}_{opt}$
27) **end procedure**

Equation (21) does not mean the SCP optimization produces the most optimal viewpoints for covering a polygon. It only ensures the coverage and attains the optimal combination among $\mathcal{V}$, by means of the SCP. This matter appears since there is no guarantee whether the optimal viewpoints are already in $\mathcal{V}$.

We then come up with a strategy to cope with it. Essentially, we make attempt to reduce the viewpoints by using an iterative optimization. Here we evaluate the area which has mutual viewpoints coverage (as pointed out by (20)) using a probabilistic randomized search, alternated by the non-unicost SCP. We aim to cut down the mutual viewpoints coverage by an alternative viewpoint.

Let $\mathcal{V}_{opt}$ denote the optimal set of viewpoints from the previous iteration and $z(f_c)$ be the face cost map function describing the number of viewpoints which "see" the face f_c and is defined by

$$z(f_c) = |\forall v \in \mathcal{V}|, \quad f_c \subset \mathbb{V}(v). \tag{26}$$

$z(f_c)$ is then used as a distribution function for sampling a set of auxiliary viewpoint candidates. We currently sample the auxiliary viewpoint candidates $\mathcal{V}_{sampling}$ from $\mathcal{F}_c$ using $z(f_c)$, as much as twice of $|\mathcal{V}_{opt}|$. It is expected that we will obtain more samples for the face covered by more guards. The SCP algorithm is accordingly performed on the concatenated $\mathcal{V}_{sampling}$ and $\mathcal{V}_{opt}$ for obtaining a new optimal set of viewpoints, and the process is repeated as shown by Algorithm 1.

A Hausdorff metric is adopted to ascertain the stopping condition of the iterative optimization process, defined as

$$\begin{aligned} D_{hd}(\mathcal{V}_+, \mathcal{V}_{opt}) &= \max\{d(\mathcal{V}_+, \mathcal{V}_{opt}), d(\mathcal{V}_{opt}, \mathcal{V}_+)\} \\ d(\mathcal{V}_+, \mathcal{V}_{opt}) &= \max_{v_+ \in \mathcal{V}_+} \min_{v_{opt} \in \mathcal{V}_{opt}} \|v_+ - v_{opt}\| \end{aligned} \tag{27}$$

where $\mathcal{V}_+$ represent results of the current SCP optimization. The above metric has a physical meaning that is when the algorithm converges, the viewpoint locations are stabilized.

5 Geodesic motion model for robot and target person movement

Since an indoor environment is used, a simple linear motion model is not a favorable option. Under a long prediction time, the linear model most likely violates the obstacle constraints in a closed workspace. A natural way to predict the movement on an indoor setting is to follow the shape of the environment. Here, we propose a geodesic motion model for achieving such requirement.

Consider a monotonic wavefront $\Phi(q)$ commenced from a source point q_0 which moves across the configuration, the travel time T of the wavefront in every point $q \in \mathcal{P}$ can be calculated by

$$\begin{aligned} \Phi(q) &\simeq |\nabla T(q)| = \frac{1}{\zeta(q)} \\ \Phi(q_0) &= 0 \end{aligned} \tag{28}$$

where $\zeta(q)$ denotes a velocity function for specifying the speed of the wavefront. This problem is widely known as Eikonal equation problem. According to [30], (28) can be approximated by the first order finite difference scheme

$$\left(\frac{T(q) - T_1}{\Delta x}\right)^2 + \left(\frac{T(q) - T_2}{\Delta y}\right)^2 = \frac{1}{\zeta(q)^2} \tag{29}$$

where

$$\begin{aligned} T_1 &= \min(T(q_{x+1,y}), T(q_{x-1,y})) \\ T_2 &= \min(T(q_{x,y+1}), T(q_{x,y-1})) \end{aligned} \tag{30}$$

where Δx and Δy respectively represent the difference of q along x-axis and y-axis, while $q_{x+1,y}$ is the right side neighbor of q along x-axis in Cartesian coordinate.

Numerical solution of (29) is given by

$$\Phi(q) \simeq T(q) = \begin{cases} T_1 + \dfrac{1}{\zeta(q)}, & \text{for } T_2 \geq T \geq T_1 \\ T_2 + \dfrac{1}{\zeta(q)}, & \text{for } T_1 \geq T \geq T_2 \\ \Omega(T_1, T_2, \zeta(q)), & \text{for } T \geq \max(T_1, T_2) \end{cases} \tag{31}$$

where $\Omega(T_1, T_2, \zeta(q))$ denotes a quadratic solution[2] of (29).

We define the velocity model for both the robot and the target person, as follows:

1) The robot is expected to safely move within the environment. To achieve it, the distance map $\mathcal{D}$ in (15) is utilized as the velocity function and normalized to the maximum speed of the robot φ_r,

$$\zeta_r(q) = \varphi_r \frac{\mathcal{D}(q)}{\|\mathcal{D}(q)\|}. \tag{32}$$

It implies the robot is given a higher velocity at the location farther from obstacles (see Fig. 7 (a)).

2) For the target, the current target velocity φ_h is taken into account by using velocity cone model combined with the distance map $\mathcal{D}$. Let $\mathcal{C}$ be the cone area which consists of all points $q \in \mathcal{P}$ satisfying

$$\mathcal{C} = \left\{\forall q | q \in \mathcal{P} \wedge \left(\angle q \leq \angle \vec{\varphi}_h \pm \frac{\pi}{3}\right)\right\} \tag{33}$$

where $\angle \vec{\varphi}_h$ denotes the orientation of φ_h. Subsequently,

[2]The quadratic solution means a quadratic equation $ax^2 + bx + c = 0$ has the solution $\dfrac{-b \pm \sqrt{b^2 - 4ac}}{2a}$.

(a) (b)

Fig. 7 The travel time map of the robot (a) and the target (b). The black circle represents the robot's current position. The blue circle with line denotes the target position and its predicted movement.

the velocity function of the cone area ζ_C is given by

$$\zeta_C(q) = \begin{cases} \varphi_h, & \text{for } q \in \mathcal{C} \\ \varepsilon, & \text{for } q \notin \mathcal{C} \wedge q \in \mathcal{P} \\ 0, & \text{otherwise} \end{cases} \tag{34}$$

where ε is a small constant, ζ_C and $\mathcal{D}$ are then merged using Hadamard product to obtain the velocity model for the target person.

$$\zeta_h(q) = \zeta_C(q) \circ \left(\frac{\mathcal{D}(q)}{\|\mathcal{D}(q)\|} \right). \tag{35}$$

Fig. 7(b) shows that the travel time of the target person follows the shape of the environment.

By substituting (32) and (35) into $\zeta(q)$ in (29), the geodesic model of the travel time for the robot $\Phi_r(q)$ and the target person $\Phi_h(q)$ are then acquired.

6 Planning the robot action over viewpoints

In this section, we present the planning method for solving the minimization problem in (9)–(14). By Definition 3, the planning action can also be interpreted as the problem of finding the optimal time and location for the robot to move, while intercepting the possibility of losing the target visibility. Either way, the robot is preferred to be idle.

One may wonder whether using the travel time fields $\Phi_r(q)$ and $\Phi_h(q)$ is sufficient to directly discover the solution of the above optimal time problem. Let τ_i^h be the estimated time for the target person to reach the escaping gap $g_i \in \mathcal{G}$, and τ_j^r be the time for the robot to reach a viewpoint $v_j \in \mathcal{V}$ (precisely, $\mathcal{V}_{opt}$) covering g_i. As long as $\tau_i^h > \tau_j^r$, the target person does not seem to leave the robot visibility while the robot is staying idle. Yet, the time values offered by $\Phi(q)$ are deterministic and does not consider the uncertainty of both the robot and the target person future states. Here a particle-based motion model over the travel time field $\Phi(q)$ is proposed to solve the problem.

6.1 Representing future states as particles

As previously described, instead of using a linear motion model for predicting the future states, we employ a geodesic motion model which is more representative for an indoor problem setting. Let $\psi_i^h : [q_0^h, g_i] \mapsto \mathbf{R}^m$ be a continuous curve describing the path of the target person towards each escaping gap $g_i \in \mathcal{G}$. These paths are obtained by performing gradient descent search over $\Phi_h(q)$. The same definition is applied to $\psi_j^r : [q_0^r, v_j] \mapsto \mathbf{R}^m$, which represents the path of the robot towards each viewpoint $v_j \in \mathcal{V}$, using $\Phi_r(q)$.

Distributions of the future state trajectories of both the robot and the target person are then approximated using particles, respectively corresponding to each path ψ_i^h and ψ_j^r. Consider $\boldsymbol{q}_t^{h,1:N} = \{q_t^{h,1}, q_t^{h,2}, \cdots, q_t^{h,N}\}$ or simply write $\boldsymbol{q}_t^h$ as particles, which probabilistically describe the target person state at time t, where N is the number of particles. In addition, let $\boldsymbol{q}_t^r$ be the particles of the robot. The path ψ_i^h is subsequently parameterized by a set of particles $\{\boldsymbol{q}_k^h, \boldsymbol{q}_{k+1}^h, \cdots\}$, where $k \in \{0, \Delta t, 2\Delta t, \cdots, \tau_i^h\}$ and Δt is time step. Accordingly, the path for the robot ψ_j^r is described by $\{\boldsymbol{q}_k^r, \boldsymbol{q}_{k+1}^r, \cdots\}$ with $k \in \{0, \Delta t, \cdots, \tau_j^r\}$.

For each path, we then propagate the particles as follows:

$$\begin{aligned} \boldsymbol{q}_{k+1}^h &= \boldsymbol{q}_k^h + \text{diag}((\hat{q}_t^h - \overline{\boldsymbol{q}}_k^h)\boldsymbol{I}) + \boldsymbol{w}_h \\ \boldsymbol{q}_{k+1}^r &= \boldsymbol{q}_k^r + \text{diag}((\hat{q}_t^r - \overline{\boldsymbol{q}}_k^r)\boldsymbol{I}) + \boldsymbol{w}_r \\ \hat{q}_t^h &\sim q_{(k+1)\Delta t}^h \in \psi_i^h \\ \hat{q}_t^r &\sim q_{(k+1)\Delta t}^r \in \psi_j^r \end{aligned} \tag{36}$$

where $\overline{\boldsymbol{q}}_k^h$ represents the mean of each particle $\boldsymbol{q}_k^h$, $\hat{q}_t^h$ represents the point in the path ψ_i^h which has Δt difference with $\overline{\boldsymbol{q}}_k^h$, $\boldsymbol{w}_h$ describes the uncertainty, function diag$(\cdot)$ returns a vector of matrix's diagonal, and $\boldsymbol{I}$ is an identity matrix. Therefore, $\{\overline{\boldsymbol{q}}_k^r, \hat{q}_t^r, \boldsymbol{w}_r\}$ notations follow the similar descriptions, except they are now meant for the robot.

The physical meaning of (36) is that the particles for the target person try to follow the shape of the environment as pointed out by (35). While for the robot, the particle sequences mean the robot is controlled towards the viewpoint using the speed suggested by the velocity function in (32). It is basically the approximation of the robot control u described in (4). Fig. 8 shows the particle representations of the future states.

6.2 Action plan based on chance constraint bound

In a greedy fashion, estimating the optimal time for the robot to start moving can be done by evaluating all possible combinations of the target and robot particles, the escaping gaps, and the viewpoints over time. It should be done while ensuring the visibility towards the target person during the movement for each state of the robot and target person.

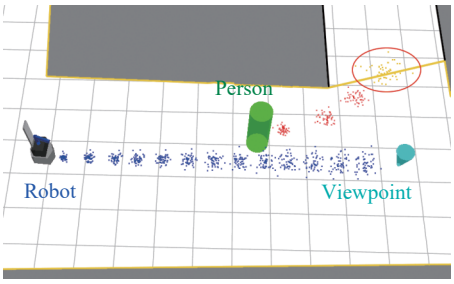


Fig. 8 Particle illustration for the future states of the robot (blue) and the target person (red). The yellow line forms the robot visibility polygon. The red ellipse denotes the predicted states of the target which violates the robot visibility.

Let λ be time delay for the robot to start moving from the current state. We expect to maximize λ for reducing the robot movement,

$$\max \quad \lambda, \quad \lambda = \{0, \cdots, \tau_i^h - \tau_j^r\} \tag{37}$$

$$\text{s.t.} \quad \boldsymbol{q}_k^h \in \mathbb{V}(\boldsymbol{q}_k^r) \tag{38}$$

$$\forall \boldsymbol{q}_{k+\lambda}^h \in \psi_i^h, \quad \forall \boldsymbol{q}_k^r \in \psi_j^r \tag{39}$$

$$\psi_i^h : [q_0^h, g_i] \mapsto \mathbf{R}^m, \quad \psi_j^r : [q_0^r, v_j] \mapsto \mathbf{R}^m \tag{40}$$

$$\forall g_i \in \mathcal{G}, \quad \forall v_j \in \mathcal{V}. \tag{41}$$

Equation (39) means the robot movement is delayed until time $k+\lambda$ (i.e., the target person has already moved for λ unit time when the robot just starts to move, $k=0$) for all possible λ. The maximum value of λ is $\tau_i^h - \tau_j^r$ since for $\tau_i^h < \tau_j^r$, the target person will obviously leave the robot visibility. Calculating the visibility of the target person for each possible robot state in (38) is costly and becomes the burden for running the optimization in real time.

Instead of relying on the above greedy search to determine the optimal time to move, here we propose a technique to reduce the computational efforts, based on Theorem 1.

Theorem 1. The visibility of all states of the target person in a trajectory $\boldsymbol{q}_{1:\tau}^h$ is guaranteed, as long as $\boldsymbol{q}_\tau^h \in \mathbb{V}(\boldsymbol{q}_0^r)$.

Proof. The proof is straightforward. By Definition 3, $\boldsymbol{q}_{1:\tau}^h$ is bounded by $\mathcal{P}$. Since the trajectory is monotonically increasing by means of the geodesic model, visibility of the last state $\boldsymbol{q}_\tau^h$ also implies the visibility of the rest states. □

By the fact that our proposed algorithm runs iteratively, instead of finding the optimal time for the robot to move in the future, we are more interested in examining whether it grants the visibility guarantee by stopping at the current state. By Theorem 1, such guarantee can be acquired by evaluating the visibility of $\boldsymbol{q}_{\tau i,j}^h$ (the target person state when the robot reaches the viewpoint v_j, while the target is predicted to go to g_i). It implies the target should be still visible by the time the robot arrives at the viewpoint.

To correspond with the nature of the particle usage (i.e., uncertainty), we accordingly propose utilization of chance constraint, which is widely used in optimal control and obstacle collision assessment (e.g., [31, 32]), for verifying bound of the visibility.

Theorem 2. Probability of keeping the visibility of the target person is designated by chance constraint bound

$$Pr(\boldsymbol{q}_{\tau_{i,j}}^h \notin \mathbb{V}(\boldsymbol{q}_0^r)) \leq \gamma \tag{42}$$

and it can be approximated using particles

$$\left(\frac{1}{N}\sum_{n=1}^{N} q_{\tau_{i,j}}^{h,n} \notin \mathbb{V}(q_0^r)\right) \leq \gamma. \tag{43}$$

Proof. We can write the expectation of event $h(\boldsymbol{q}_{\tau_{i,j}}^h, \boldsymbol{q}_0^r) = \left(\mathbf{q}_{\tau_{i,j}}^h \notin \mathbb{V}(\boldsymbol{q}_0^r)\right)$, which is the left-hand side of (42), as

$$\mathbf{E}[h(\boldsymbol{q}_{\tau_{i,j}}^h, \boldsymbol{q}_0^r)] = \iint h(q_{\tau_{i,j}}^h, q_0^r) f(q_{\tau_{i,j}}^h) f(q_0^r) \mathrm{d}q_{\tau_{i,j}}^h \mathrm{d}q_0^r \tag{44}$$

where $f(q_{\tau_{i,j}}^h)$ and $f(q_0^r)$ respectively represent the probability density function of $\boldsymbol{q}_{\tau_{i,j}}^h$ and $\boldsymbol{q}_0^r$. Since this integral is difficult to be evaluated in a closed form, it is approximated as

$$\mathbf{E}[h(\boldsymbol{q}_{\tau_{i,j}}^h, \boldsymbol{q}_0^r)] \approx \frac{1}{N^2}\sum_{n=1}^{N}\sum_{n=1}^{N} h(q_{\tau_{i,j}}^{h,n}, q_0^{r,n}). \tag{45}$$

By assuming a small uncertainty for the current robot state $\boldsymbol{q}_0^r$, we can approximate its mean as

$$\boldsymbol{q}_0^r \approx q_0^r = \frac{1}{N}\sum_{n=1}^{N} q_0^{r,n}. \tag{46}$$

Hence, the event $h(\boldsymbol{q}_{\tau_{i,j}}^h, \boldsymbol{q}_0^r) \approx h(\boldsymbol{q}_{\tau_{i,j}}^h, q_0^r)$. Subsequently, the left-hand side of (43) is proved. □

Equation (42) means the target person state at time $\tau_{i,j}$ is permissible to be outside the robot visibility with probability at most γ, to be called "visible target". Using Theorem 2, the optimal action u_t^* for the robot is then selected as

$$u_t^* = \begin{cases} 0, & \text{for } \chi = 1 \\ \arg\min_u \phi(u, v, g), & \text{otherwise} \end{cases} \tag{47}$$

where

$$\chi = \bigwedge_{\forall g_i \in \mathcal{G}, \forall v_j \in \mathcal{V}} Pr(\boldsymbol{q}^h_{\tau_{i,j}} \notin \mathbb{V}(\boldsymbol{q}^r_0)) \leq \gamma \tag{48}$$

and

$$\begin{aligned} \phi(u, v, g) \simeq & f(u, \epsilon) : [\boldsymbol{q}^r_0, v] \mapsto \mathbf{R}^m \\ & \{\forall v \in \mathcal{V} | g \in \mathbb{V}(v)\} \\ & \{\forall g \in \mathcal{G} | Pr(\boldsymbol{q}^h_{\tau_{i,j}} \notin \mathbb{V}(\boldsymbol{q}^r_0)) > \gamma\}. \end{aligned} \tag{49}$$

Equation (45) means the robot is "safe" to stop at the current position when all visibility bounds towards the escaping gaps are not violated by the predicted movement of the target person. Either way, the robot should move to the viewpoint which covers the violated escaping gap. In case the robot faces several violated escaping gaps, the argument minimum over (46) suggests the robot to choose the viewpoint which covers the most critical escaping gap and gives the minimum effort $f(u, \epsilon)$ to execute the trajectory towards it.

6.3 Further consideration for the chosen action

Up to now, we have determined the bound for the robot to safely keep the target person visibility by staying at the current state. Let us consider another case, illustrated by Fig. 9. Fig. 9 depicts that according to the current calculation (remember that our algorithm was done iteratively), the robot is suggested "safe" to stay at the current position. However, moving to the viewpoint may have a benefit to make the robot hold the visibility for a longer time, since the future escaping gap will disappear (or, shifted to a farther place). Here, an additional rule is incorporated into the optimization to let the robot execute the optimal action.

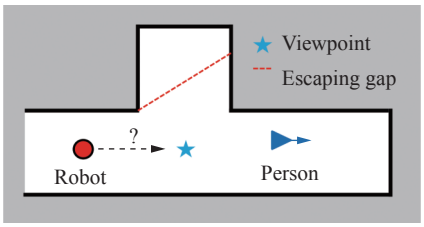


Fig. 9 A case when moving the robot may be better than keeping it idle

The rule basically examines distance of the nearest escaping gaps $\|g_{ftr}\|$ created by the future states of the robot $\boldsymbol{q}^r_{\tau^r_j}$ (when the robot reaches the viewpoint v_j) towards the predicted target state at the same time step (i.e., $\boldsymbol{q}^h_{\tau^r_j}$). Equation (44) is then slightly modified as

$$u^*_t = \begin{cases} 0, & \text{for } \chi = 1 \quad \&\& \\ & \|g_{ftr}\| - \|g_{now}\| \leq \eta \\ f(u, \epsilon) : [\boldsymbol{q}^r_0, v^*] \mapsto \mathbf{R}^m, & \text{for } \chi = 1 \quad \&\& \\ & \|g_{ftr}\| - \|g_{now}\| > \eta \\ \arg\min_u \phi(u, v, g), & \text{otherwise} \end{cases} \tag{50}$$

where $\|g_{now}\|$ denotes the distance between the target person and the nearest escaping gap for the current time, v^* represents the viewpoint leading to the condition described in Fig. 9, and η is a distance threshold.

The selected action drawn from (47) is deemed as the planning result. It is then sent as a set of waypoints to a local motion planner[16] to be executed.

7 Experiments and results

The implementation of our movement planning algorithm for the guard robot is done on a Windows PC (Core i7 2.4 GHz, 16 GB RAM) using C++ programming language. The movement planner as well as the supporting algorithms (e.g., robot controller, localization, local planner, etc.) are implemented in a distributed manner as RT-components, which are software modules running on robot technology (RT)-middleware[33] environment. Mobile robot programming toolkit (MRPT) library[34] is also used mainly for visualization purposes. The proposed algorithm's feasibility is then evaluated by both simulations and the real experiments. We also provide a comparison with the person following algorithm to clearly certify the benefit of our movement planner.

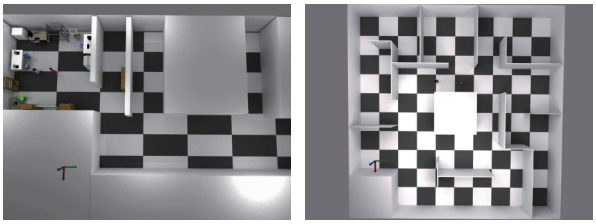
For both simulations and the real experiments, our movement planner runs in two stages: off-line and on-line stages. In the off-line stage, the map data is acquired from a simultaneous localization and mapping (SLAM) algorithm[35]. We then retrieve viewpoints from the map using the algorithm mentioned in Section 4. These viewpoints are subsequently utilized to make a global plan for the robot in the on-line stage. The action yielded by the global plan is accordingly executed by a local motion planner ([16, 36]).

7.1 Simulations using a realistic 3D simulator

A Linux PC (Core i5, 2.1 GHz) is utilized for running the 3D simulator[37] and interconnected to the movement planner through a socket communication[38]. All sensor data such as laser range measurement and the robot odometry are simulated and taken from the simulator, resembling the real condition. For the target person, we simulate and administer a predefined path for a human object, such that it will move continuously and independently around the simulated environment.

We arrange two different environments for conducting simulations. The first map imitates the real first floor of information and communication technology (ICT)

building at our university (see Fig. 10(a)) with a slight modification. The second map represents a more challenging environment where it has several rooms and holes (see Fig. 10(b)). Table 1 displays parameter settings used in the simulations. Those parameters are chosen empirically by considering limitation of the robot speed and maneuverability, and the computing power of the processor.



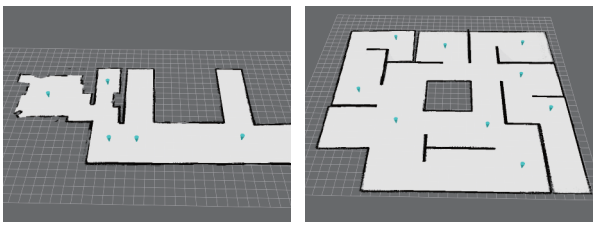
(a) (b)

Fig. 10 Environment used for simulations: (a) The first floor of ICT (information and communication technology) building (environment A); (b) Generic complex map (environment B).

Table 1 Parameter settings

Parameters	Value
Person velocity	0.8 m/s
Robot max velocity	0.7 m/s
Simulation length	
Environment A	1504 steps
Environment B	473 steps
Time step	0.25 s
N (number of particles)	50
γ (see (42))	0.05
η (see (47))	5.0

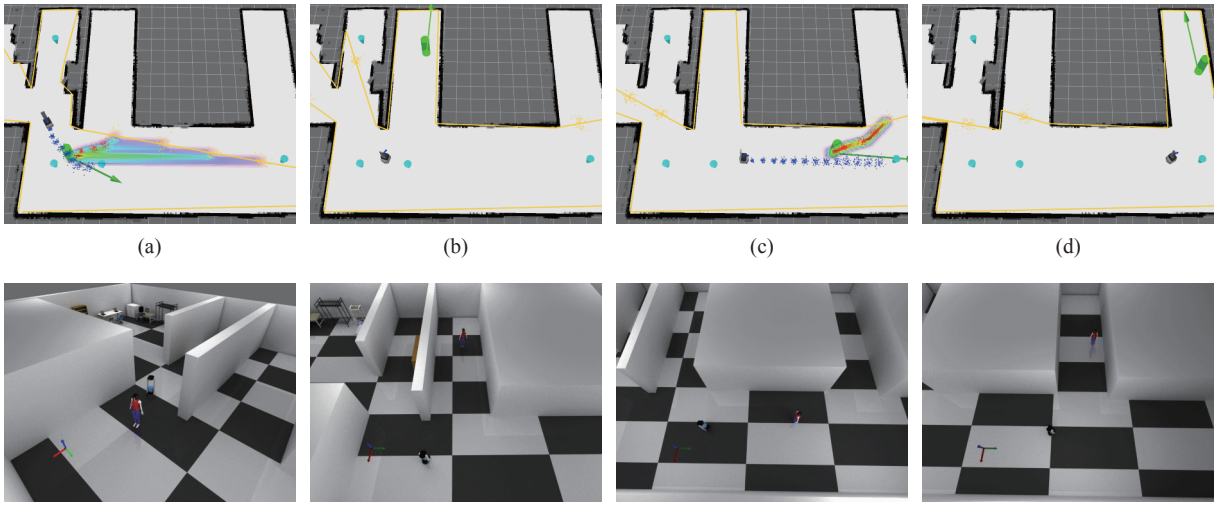
Fig. 11 shows the viewpoints obtained by our viewpoint extractor. The viewpoints mainly reside at the middle of a room, or at the intersections connecting several corridors. It justifies our assumptions in Section 4, declaring such places are suitable for the robot to have a wide visibility for a longer time.



(a) (b)

Fig. 11 Viewpoints obtained by the viewpoint extractor, shown by cyan cones, in: (a) environment A; (b) environment B.

The simulation results are then exhibited by Figs. 12 and 13, respectively for the environments A and B. For each figure, the top row represents the movement planning results and the bottom row shows its associated states in the 3D simulator. When the target person is predicted to go through the escaping gaps (e.g., Figs. 12(a), 12(c) and 13(a)–13(c)), our algorithm makes a plan for the robot to move towards the viewpoint which covers those escaping gaps. In Figs. 12(b), 12(d) and 13(d), when the target person enters a "dead end" area, instead of following closely behind the target person, the robot is staying near the viewpoint to observe it. It implies our movement planner is able to predict the situ-



(a) (b) (c) (d)

(e) (f) (g) (h)

Fig. 12 Executing movement planner on environment A. Top row shows the movement planning results. Bottom row shows its corresponding states on the 3D simulator. For the top row figures, the green cylinder with arrow represents the target person and its moving direction. The jetmap (colored cost map) represents the predicted future movement of the person. Cyan cones denote the viewpoints. The black object with a sequence of blue particles represents the robot and its movement controls. Yellow lines show the visibility polygon of the robot.

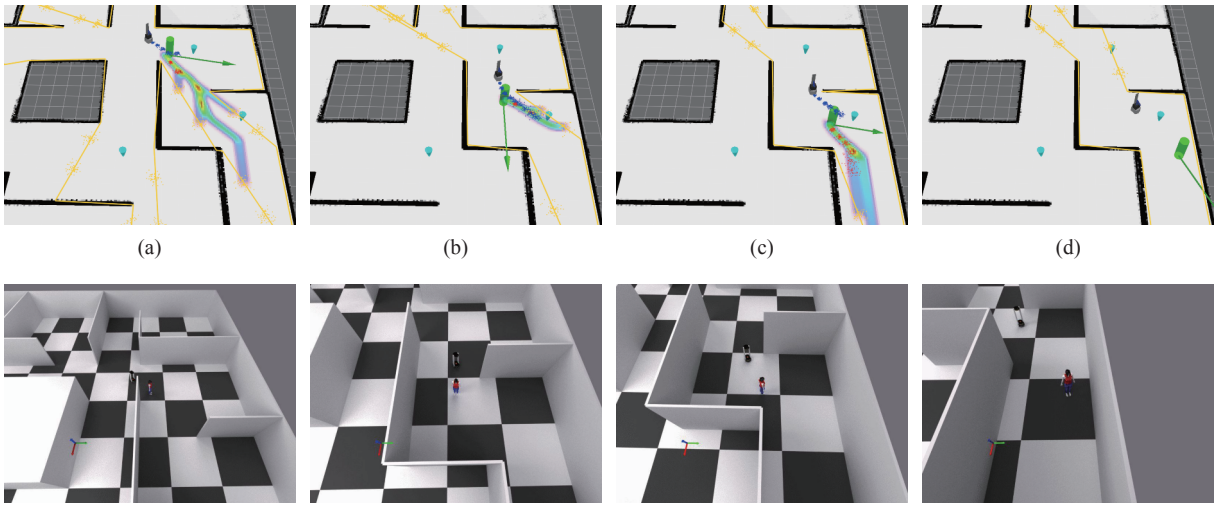


Fig. 13 Executing movement planner on environment B. Here, the same figure information as in Fig. 12 is applied too

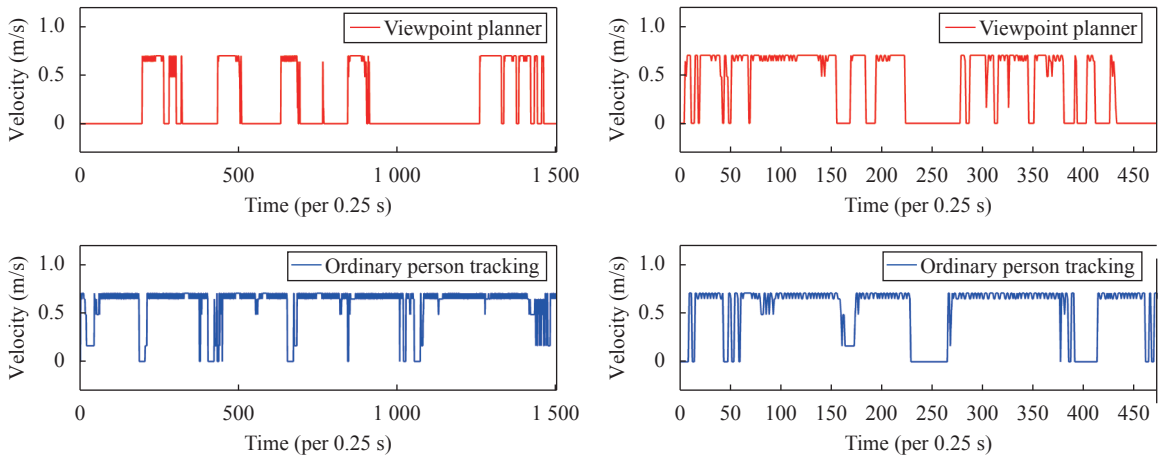
ation that the target person will unlikely escape from the robot visibility. It also clearly distinguishes our method from the ordinary person tracking algorithm.

One may wonder that the robot is not staying at the viewpoint when it is in an idle condition (see Figs. 12(b), 12(d) and 13(d)). It actually justifies the main purpose of our movement planner, i.e., to reduce the movement. The robot does not have to exactly "reach" the designated viewpoint as long as it is safe enough to keep the target person under its visibility. Thanks to the iterative visibility bound checking given in (44) and (47), our proposed method can evaluate and predict the target person states and the eligibility of the robot to move or to be idle time-by-time, ensuring the optimality of the planner.

7.2 Comparison with the person following algorithm

To get a picture about the effectiveness of the robot movement, we compare performance of our movement planner with the ordinary person tracking algorithm as described in [16], under the condition that exactly the same path and behavior of the target person are used. Fig. 14 exhibits the recorded velocity profiles of the robot during runtime for both algorithms. It can be noted that our movement planner produces longer idle condition compared with the person tracking algorithm, indicated by longer zero velocity. It means the robot has an effective movement for observing the target person, without always closely following him. In the environment B, the condition is not much idle as the one in the environment A, since the target person mainly moves along corridors rather than any dead end or rooms.

We then quantitatively compare the used energy for both movement planning and the ordinary person tracking algorithms, as well as our previous method[24], using following metric:



(a) Velocity profile for environment A

(b) Velocity profile for environment B

Fig. 14 Comparison of controls yielded by movement planner and ordinary person tracking algorithm for: (a) environment A, (b) environment B.

$$Energy = \sum_{t=0}^{T} (u_t)^2 \tag{51}$$

where u_t denotes the control applied to the robot, and T is total time for one simulation.

From the above metric, we expect to capture the total movement of the robot. Table 2 displays the average energy for each algorithm after five simulations. It clearly shows the benefit of our proposed algorithm which is able to reduce the movement and leads to the energy saving. Our proposed algorithm also has a slightly better performance compared with our previous approach.

Table 2 Comparison of energy used by the robot (lower is better)

	Proposed viewpoint planner	Person tracking algorithm	Previous approach[24]
Environment A	205.53	592.91	220.73
Environment B	130.06	166.96	135.15

7.3 Experiments on a real environment

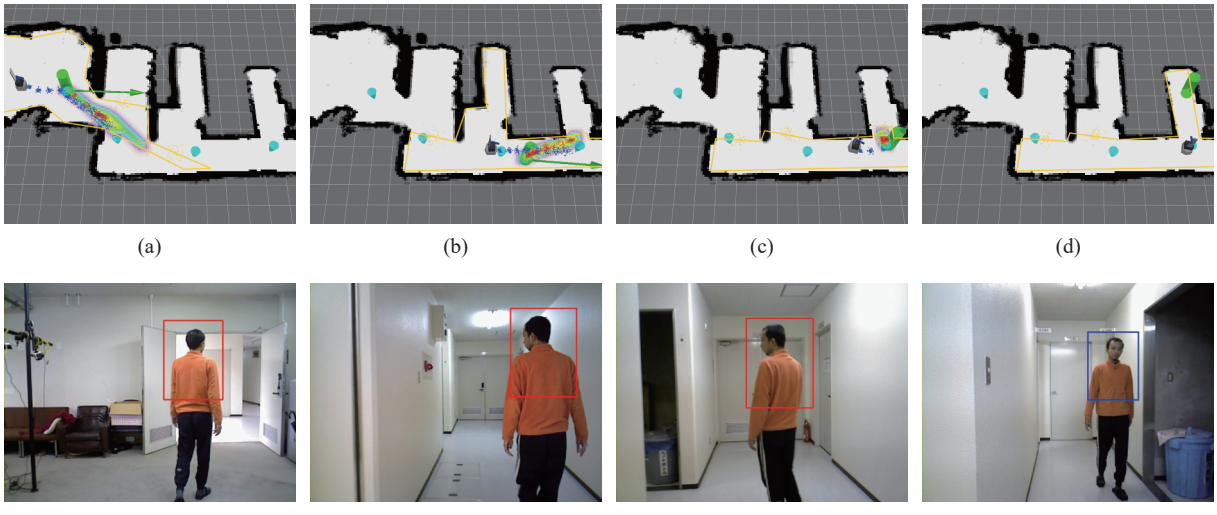
Finally, we evaluate performance of the proposed guard robot algorithm using the real robot in a real environment. A Pioneer 3DX robot equipped with laser range sensors, a camera, and a laptop PC are used. The laptop PC is basically utilized for acquiring the sensor data, sending commands to the robot, and distributing the workload. It is then connected to the main PC running the movement planning algorithm through the same communication manner as the one in simulations. The parameter setting is also the same as the simulation, except for the person velocity which depends on the tracker. For obtaining the target person data, an image-based[39] and laser-based person tracker were used.

We carry out the experiment at ICT building of our university, which is exactly the same as the simulation with environment A described in Section 7.1, but now with the real robot. Fig. 15(a) exhibits the obtained map of the building. As shown in Fig. 15(d), the robot only needs to "stay" while observing the target person, since it is predicted not to leave the robot visibility. This behavior of the robot is also confirmed by the velocity profile given in Fig. 16, where the robot does not always move to closely follow the target. The ability to keep the view towards the target person is then proved by the bottom row of Fig. 15. This experiment result also demonstrates the feasibility of our algorithm to be used in real-time.

8 Conclusions

We have described a novel movement planning algorithm for the guard robot using particle model-based planning approach. We utilize the topology of the environment to make an effective movement of the robot. A geodesic motion model is also used for predicting both the robot and the target person movement. Simulation and experiment results show that the proposed algorithm can reduce the movement of the robot, thereby saving energy used by the robot.

The current implementation does not consider any partial occlusion by a smaller object. It may happen that the target person is actually visible from the robot, but the planner assumes it is occluded due to these objects. It then may deviate the results of the movement planner. Such problem should be tackled in the future. Other possible interesting future researches will be the implementation for outdoor problem cases, and the use of unmanned aerial vehicle which has more degrees of freedom to freely observe the target person.



(a) (b) (c) (d)

(e) (f) (g) (h)

Fig. 15 Executing movement planner on the real environment. Top row shows the planning results on the map. Bottom row shows the target person view from the robot, associated with the plan on the top row.

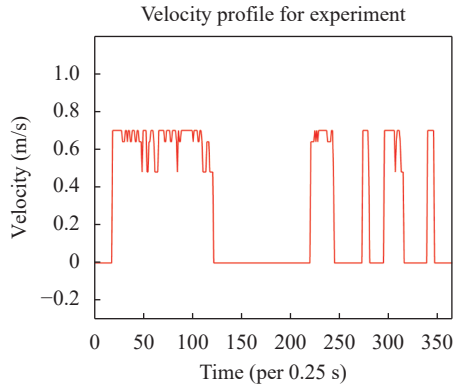


Fig. 16 Velocity profile of the real experiment

References

[1] J. O'Rourke. Art Gallery Theorems and Algorithms. Oxford, UK: Oxford University Press, 1987.

[2] V. Pinciu. A coloring algorithm for finding connected guards in art galleries. *Discrete Mathematics and Theoretical Computer Science*, C. S. Calude, M. J. Dinneen, V. Vajnovszki, Eds., Berlin Heidelberg, Germany: Springer, pp. 257–264, 2003. DOI: 10.1007/3-540-45066-1˙20

[3] L. Erickson, S. LaValle. An art gallery approach to ensuring that landmarks are distinguishable. In *Proceedings of International Conference on Robotics: Science and Systems*, RSS, Los Angeles, USA, 2011.

[4] A. Bottino, A. Laurentini. A nearly optimal algorithm for covering the interior of an art gallery. *Pattern Recognition*, vol. 44, no. 5, pp. 1048–1056, 2011. DOI: 10.1016/j.patcog.2010.11.010.

[5] J. W. Durham, A. Franchi, F. Bullo. Distributed pursuit-evasion with limited-visibility sensors via frontier-based exploration. In *Proceedings of IEEE International Conference on Robotics and Automation*, IEEE, Anchorage, USA, pp. 3562–3568, 2010. DOI: 10.1109/ROBOT.2010.5509347.

[6] N. Noori, A. Beveridge, V. Isler. Pursuit-evasion: A toolkit to make applications more accessible. *IEEE Robotics & Automation Magazine*, vol. 23, no. 4, pp. 138–149, 2016. DOI: 10.1109/MRA.2016.2540138.

[7] K. Klein, S. Suri. Capture bounds for visibility-based pursuit evasion. *Computational Geometry*, vol. 48, no. 3, pp. 205–220, 2015. DOI: 10.1016/j.comgeo.2014.10.002.

[8] M. V. Ramana, M. Kothari. Pursuit-evasion games of high speed evader. *Journal of Intelligent & Robotic Systems*, vol. 85, no. 2, pp. 293–306, 2017. DOI: 10.1007/s10846-016-0379-3.

[9] N. M. Stiffler, J. M. O'Kane. Complete and optimal visibility-based pursuit-evasion. *The International Journal of Robotics Research*, vol. 36, no. 8, pp. 923–946, 2017. DOI: 10.1177/0278364917711535.

[10] J. S. B. Mitchell. Approximating watchman routes. In *Proceedings of the Twenty-fourth Annual ACM-SIAM Symposium on Discrete Algorithms*, Society for Industrial and Applied Mathematics, Philadelphia, USA, pp. 844–855, 2013.

[11] A. A. Ismail, S. Herdjunanto, P. Priyatmadi. Ant system algorithm for finding optimal path on travelling salesman problem with path restriction. *Jurnal Nasional Teknik Elektro dan Teknologi Informasi*, vol. 1, no. 3, pp. 43–48, 2012.

[12] H. Emadi, T. S. Gao, S. Bhattacharya. Visibility-based target-tracking game: Bounds and tracking strategies. *IEEE Robotics and Automation Letters*, vol. 2, no. 4, pp. 1917–1924, 2017. DOI: 10.1109/LRA.2017.2714980.

[13] X. H. Tan. Approximation algorithms for the watchman route and zookeeper's problems. *Discrete Applied Mathematics*, vol. 136, no. 2–3, pp. 363–376, 2004. DOI: 10.1016/S0166-218X(03)00451-7.

[14] N. Bellotto, H. S. Hu. People tracking with a mobile robot: a comparison of Kalman and particle filters. In *Proceedings of the 13th IASTED International Conference on Robotics and Applications*, ACTA Press, Anaheim, USA, 2007.

[15] T. Linder, S. Breuers, B. Leibe, K. O. Arras. On multi-modal people tracking from mobile platforms in very crowded and dynamic environments. In *Proceedings of IEEE International Conference on Robotics and Automation*, Stockholm, Sweden, pp. 5512–5519, 2016. DOI: 10.1109/ICRA.2016.7487766.

[16] I. Ardiyanto, J. Miura. Real-time navigation using randomized kinodynamic planning with arrival time field. *Robotics and Autonomous Systems*, vol. 60, no. 12, pp. 1579–1591, 2012. DOI: 10.1016/j.robot.2012.09.011.

[17] R. Liu, G. Huskić, A. Zell. Dynamic objects tracking with a mobile robot using passive UHF RFID tags. In *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems*, Chicago, USA, pp. 4247–4252, 2014. DOI: 10.1109/IROS.2014.6943161.

[18] Q. J. Peng, X. M. Kang, T. T. Zhao. Effective virtual reality based building navigation using dynamic loading and path optimization. *International Journal of Automation and Computing*, vol. 6, no. 4, pp. 335–343, 2009. DOI: 10.1007/s11633-009-0335-9.

[19] A. M. Rao, K. Ramji, B. S. K. Sundara Siva Rao, V. Vasu, C. Puneeth. Navigation of non-holonomic mobile robot using neuro-fuzzy logic with integrated safe boundary algorithm. *International Journal of Automation and Computing*, vol. 14, no. 3, pp. 285–294, 2017. DOI: 10.1007/s11633-016-1042-y.

[20] B. Das, B. Subudhi, B. B. Pati. Cooperative formation control of autonomous underwater vehicles: an overview. *International Journal of Automation and Computing*, vol. 13, no. 3, pp. 199–225, 2016. DOI: 10.1007/s11633-016-1004-4.

[21] B. D. Ning, Q. L. Han, Z. Y. Zuo. Distributed optimization for multiagent systems: An edge-based fixed-time consensus approach. *IEEE Transactions on Cybernetics*, Online-first. DOI: 10.1109/TCYB.2017.2766762.

[22] B. D. Ning, Q. L. Han, Z. Y. Zuo, J. Jin, J. C. Zheng. Collective behaviors of mobile robots beyond the nearest neighbor rules with switching topology. *IEEE Transactions on Cybernetics*, vol. 48, no. 5, pp. 1577–1590, 2018. DOI: 10.1109/TCYB.2017.2708321.

[23] Z. Y. Zuo, Q. L. Han, B. D. Ning, X. H. Ge, X. M. Zhang. An overview of recent advances in fixed-time cooperative control of multiagent systems. *IEEE Transactions on Industrial Informatics*, vol. 14, no. 6, pp. 2322–2334, 2018. DOI: 10.1109/TII.2018.2817248.

[24] I. Ardiyanto, J. Miura. Visibility-based viewpoint planning for guard robot using skeletonization and geodesic motion model. In *Proceedings of IEEE International Con-*

ference on Robotics and Automation, Karlsruhe, Germany, pp. 652–658, 2013. DOI: 10.1109/ICRA.2013.6630643.

[25] S. Suzuki, K. Abe. Topological structural analysis of digitized binary images by border following. *Computer Vision, Graphics, and Image Processing*, vol. 30, no. 1, pp. 32–46, 1985. DOI: 10.1016/0734-189X(85)90016-7.

[26] D. H. Douglas, T. K. Peucker. Algorithms for the reduction of the number of points required to represent a digitized line or its caricature. *Cartographica: The International Journal for Geographic Information and Geovisualization*, vol. 10, no. 2, pp. 112–122, 1973. DOI: 10.3138/FM57-6770-U75U-7727.

[27] D. Lee, A. Lin. Computational complexity of art gallery problems. *IEEE Transactions on Information Theory*, vol. 32, no. 2, pp. 276–282, 1986. DOI: 10.1109/TIT.1986.1057165.

[28] E. Fogel, D. Halperin, R. Wein. *CGAL Arrangements and Their Applications – A Step-by-Step Guide*, Berlin Heidelberg, Germany: Springer, 2012.

[29] D. P. Williamson, D. B. Shmoys. *The Design of Approximation Algorithms*, New York, USA: Cambridge University Press, 2011.

[30] J. A. Sethian. A fast marching level set method for monotonically advancing fronts. *Proceedings of the National Academy of Sciences of the United States of America*, vol. 93, no. 4, pp. 1591–1595, 1996. DOI: 10.1073/pnas.93.4.1591.

[31] L. Blackmore, M. Ono, B. C. Williams. Chance-constrained optimal path planning with obstacles. *IEEE Transactions on Robotics*, vol. 27, no. 6, pp. 1080–1094, 2011. DOI: 10.1109/TRO.2011.2161160.

[32] J. P. Calliess, D. Lyons, U. D. Hanebeck. Lazy auctions for multi-robot collision avoidance and motion control under uncertainty. *Advanced Agent Technology*, F. Dechesne, H. Hattori, A. ter Mors, J. M. Such, D. Weyns, F. Dignum, Eds., Berlin Heidelberg, Germany: Springer, pp. 295–312, 2012. DOI: 10.1007/978-3-642-27216-5˙21.

[33] N. Ando, T. Suehiro, K. Kitagaki, T. Kotoku, W. K. Yoon. RT-middleware: distributed component middleware for RT (robot technology). In *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems*, Edmonton, Canada, pp. 3933–3938, 2005. DOI: 10.1109/IROS.2005.1545521.

[34] J. L. B. Claraco. Development of scientific applications with the mobile robot programming toolkit. The MRPT reference book, in Machine Perception and Intelligent Robotics Laboratory, University of Malaga, Malaga, Spain, 2008.

[35] S. Thrun, W. Burgard, D. Fox. *Probabilistic Robotics*, Cambridge, USA: The MIT Press, 2005.

[36] I. Ardiyanto, J. Miura. Heuristically arrival time field-biased (HeAT) random tree: An online path planning algorithm for mobile robot considering kinodynamic constraints. In *Proceedings of IEEE International Conference on Robotics and Biomimetics*, Karon Beach, Thailand, pp. 360–365, 2011. DOI: 10.1109/ROBIO.2011.6181312.

[37] G. Echeverria, N. Lassabe, A. Degroote, S. Lemaignan. Modular open robots simulation engine: Morse. In *Proceedings of IEEE International Conference on Robotics and Automation*, Shanghai, China, pp. 46–51, 2011. DOI: 10.1109/ICRA.2011.5980252.

[38] I. Ardiyanto, Y. Okada, J. Miura. Rt components for using morse realistic simulator for robotics. In *Proceedings of the 13th SICE System Integration Division Annual Conference*, Fukuoka, Japan, pp. 535–538, 2012.

[39] I. Ardiyanto, J. Miura. Partial least squares-based human upper body orientation estimation with combined detection and tracking. *Image and Vision Computing*, vol. 32, no. 11, pp. 904–915, 2014. DOI: 10.1016/j.imavis.2014.08.002.

Igi Ardiyanto received the B. Eng. degree in electrical engineering from Universitas Gadjah Mada, Indonesia in 2009, the M. Eng. and Dr. Eng. degrees in computer science and engineering from Toyohashi University of Technology (TUT), Japan in 2012 and 2015, respectively. He joined the TUT-NEDO (New Energy and Industrial Technology Development Organization, Japan) research collaboration on service robots, in 2011. He is now an assistant professor at Universitas Gadjah Mada, Indonesia. He received several awards, including Finalist of the Best Service Robotics Paper Award at the *2013 IEEE International Conference on Robotics and Automation* and Panasonic Award for the 2012 RT-Middleware Contest.

His research interests include planning and control system for mobile robotics, deep learning, and computer vision.

E-mail: igi@ugm.ac.id (Corresponding author)

ORCID iD: 0000-0002-0006-1458

Jun Miura received the B. Eng. degree in mechanical engineering in 1984, the M. Eng. and Dr. Eng. degrees in information engineering in 1986 and 1989, respectively, all from the University of Tokyo, Japan. In 1989, he joined Department of Computer-controlled Mechanical Systems, Osaka University, Japan. Since April 2007, he has been a professor at Department of Information and Computer Sciences, Toyohashi University of Technology, Japan. From March 1994 to February 1995, he was a visiting scientist at the Computer Science Department, Carnegie Mellon University, USA. He received the Best Paper Award from the Robotics Society of Japan in 1997. He was also selected as one of the six finalists for the Best Paper Award at *1995 IEEE International Conference on Robotics and Automation*.

His research interests include mobile robotics, computer vision, intelligent transportation system, and pattern recognition.

E-mail: jun.miura@tut.jp