Annals of Mathematics and Artificial Intelligence

Capturing Intruders in Cluttered Environments Assisted by Information Networks

--Manuscript Draft--

Manuscript Number:	AMAI-D-15-00162
Full Title:	Capturing Intruders in Cluttered Environments Assisted by Information Networks
Article Type:	Regular Submission
Abstract:	Consider a team of robots deployed to capture intruders in an environment cluttered with many obstacles. Here, we say that a robot captures an intruder in the case where the intruder is within the maximum range of a weapon on the robot. Suppose that an information network is built in the workspace to cover the entire workspace. Using the network, a robot can detect the position of any intruder in almost real-time. An intruder has full knowledge of the search environment and can maneuver at unbounded speed to avoid robots. Thus, we require a coordinated strategy to guarantee capture of all intruders in the workspace. Building blocks for our coordinated strategy are the sweep and the guard actions which correspond to the execution of distributed routines on the robot team which respectively: 1) guarantee capture of intruders while sweeping a passage; 2) guarantee that no intruder crosses an intersection without being captured. Each action has a cost which is the number of robots needed to execute it. Considering the number of robots required for the sweep and the guard actions, we simplify a cluttered environment as the weighted Voronoi diagram such that each vertex and edge has its weight. Our contributions are as follows. First, we introduce a method to extract the weighted Voronoi diagram from a cluttered environment. Then, our problem becomes capturing intruders in this weighted graph structure. Second, we introduce the weighted graph searching strategy assisted by the information network. Based on the searching strategy, we derive an upper bound for the number of robots required to clear the weighted Voronoi diagram. Finally, we present the switching strategy to improve the time efficiency of capturing intruders while not increasing the number of robots.

Annals of Mathematics and Artificial Intelligence manuscript No.
(will be inserted by the editor)

Capturing Intruders in Cluttered Environments Assisted by Information Networks

Jonghoek Kim

Received: date / Accepted: date

Abstract Consider a team of robots deployed to capture intruders in an environment cluttered with many obstacles. Here, we say that a robot captures an intruder in the case where the intruder is within the maximum range of a weapon on the robot. Suppose that an information network is built in the workspace to cover the entire workspace. Using the network, a robot can detect the position of any intruder in almost real-time. An intruder has full knowledge of the search environment and can maneuver at unbounded speed to avoid robots. Thus, we require a coordinated strategy to guarantee capture of all intruders in the workspace. Building blocks for our coordinated strategy are the sweep and the guard actions which correspond to the execution of distributed routines on the robot team which respectively: 1) guarantee capture of intruders while sweeping a passage; 2) guarantee that no intruder crosses an intersection without being captured. Each action has a cost which is the number of robots needed to execute it. Considering the number of robots required for the sweep and the guard actions, we simplify a cluttered environment as the weighted Voronoi diagram such that each vertex and edge has its weight. Our contributions are as follows. First, we introduce a method to extract the weighted Voronoi diagram from a cluttered environment. Then, our problem becomes capturing intruders in this weighted graph structure. Second, we introduce the weighted graph searching strategy assisted by the information network. Based on the searching strategy, we derive an upper bound for the number of robots required to clear the weighted Voronoi diagram. Finally, we present the switching strategy to improve the time efficiency of capturing intruders while not increasing the number of robots.

Keywords Voronoi diagram · weighted graph search · graph clear · visible intruder · information network

Jonghoek Kim
Agency for Defense Development,
Changwon, South Korea
E-mail: jonghoek.kim@add.re.kr

1 Introduction

Monitoring and securing of large cluttered environments is an important task in military scenarios. We consider a scenario of deploying multiple robots with limited range sensors to capture all intruders in a cluttered workspace. Here, we say that a robot captures an intruder in the case where the intruder is within the maximum range of a weapon on the robot.

Suppose that an information (sensor) network is built in the workspace. Such an information network becomes feasible due to the research advances in ad-hoc networking [15,9,35,28,17,16]. In this workspace, each information node can detect a nearby intruder and let a robot access the position of the intruder in almost real-time.

Many papers on pursuit-evasion problems [7,3,16,15,19,30,23,26,13,14,20,21, 2,10,34,33] have designed strategies that maximize searcher performance against a worst-case intruder. In such settings, a worst-case intruder is characterized by unbounded speed, complete awareness of searcher location and intent, and full knowledge of the search environment.

[15] introduced a strategy to capture a worst-case intruder assisted by information networks. [15] assumed that a robot moves along a Voronoi edge of an workspace while detecting the position of any intruder using information networks. [1] Each corridor in an workspace is represented by a Voronoi edge. [15] assumed that as a robot moves along a Voronoi edge, the maximum range of the weapon on the robot covers the corridor associated to the edge. In this case, an intruder cannot move along the corridor without being captured by the robot.

In this paper, we assume that the maximum range of a weapon on the robot is not long enough to cover a corridor in an workspace. In this case, robots must move in formation to capture a worst-case intruder. Building blocks for our coordinated capturing strategy are the *sweep* and the *guard* actions which correspond to the execution of distributed routines on the robot team which respectively: 1) guarantee capture of intruders while sweeping a passage; 2) guarantee that no intruder crosses an intersection without being captured.

Based on the sweep and the guard actions, we simplify a cluttered environment as the weighted Voronoi diagram such that each vertex and edge has its weight. Each weight $w : V \bigcup E \to Z^{+}$, is a mapping describing the clearing requirement of edges and vertices. Clearing an edge e requires at least $w(e)$ robots to "sweep" along the passage associated to e. Also, clearing a vertex v requires at least $w(v)$ robots to "guard" the intersection associated to v.

Once we extract the weighted Voronoi diagram from a cluttered environment, our problem becomes capturing intruders in this weighted graph structure. In this paper, we introduce the weighted graph searching strategy assisted by the information network.

Both the authors of [1,11] and the authors of [23,24,22,25] considered capturing intruders on a weighted graph. The fundamental distinction between [1,11] and [23,24,22,25] is as follows: [1,11] is on weighted edge-searching, and [23,24,22, 25] is on graph-clear. In weighted edge-searching, searchers are deployed to guard vertices. On the other hand, in graph-clear, edges are blocked instead.

[1] Voronoi diagrams have been widely used for topological maps in robotics, c.f. [18,5,31,32, 4,27,6,25], as well as for studying coverage problems in sensor networks [8,29].

Our paper proposes a weighted edge-searching strategy, since robots are deployed to guard vertices. However, as far as we know, no paper has considered an intruder capturing strategy on a weighted graph assisted by information networks.

As we use information networks, it is inevitable that time delay in data transfer occurs. This paper presents an intruder capturing strategy which is robust to time delay in data transfer using networks.

Based on the proposed searching strategy, we derive an upper bound for the number of robots required to clear the weighted Voronoi diagram. Finally, we present the switching strategy to improve the time efficiency of capturing intruders while not increasing the number of robots.

In Section 2, we introduce background information and previous works related to this paper. In Section 3, we introduce a method to extract the weighted Voronoi diagram from a cluttered environment. In Section 4, we present the weighted graph searching strategy assisted by the information network. Section 5 provides Conclusions.

2 Preliminaries and Background

This section provides background information and previous works that are used to provide the new contributions.

2.1 Graph Theory

We review some general notions in graph theory, e.g., [12]. An undirected graph G is defined by a set $G = (V(G), E(G))$, where $V(G)$ denotes the vertex set and $E(G)$ is a set of unordered pairs of vertices where multiple edges between vertex pairs are allowed. A vertex v is said to be *adjacent* to another vertex w if the graph contains an edge $\{v, w\}$. In this case, v is a *neighbor* to w. A *walk* is an alternating sequence of vertices and edges in a graph such that each vertex belongs to the edge immediately before and after it in the sequence. A graph G is *connected* if there is a walk between every pair of distinct vertices. A *cycle* is a graph that consists of certain number of vertices connected to form a closed chain. The *subgraph* of G induced by a set of vertices $S \subset V(G)$ is the graph (S, E_S) where $E_S = \{\{x, y\} \in E(G) : x, y \in S\}$.

Given G, *splitting* of a vertex v is an operation $\Phi(G, v) = G' = (V', E')$. Here, $V' = (V \backslash \{v\}) \bigcup \{v^1, ...v^{deg(v)}\}$ and $E' = (E \backslash (\bigcup_{u \in N_v} \{u, v\}) \bigcup_{u \in N_v} \{u, \nu(u)\}$ where N_v denotes a neighboring vertex set of v and $\nu : N_v \to \{v^1, ...v^{deg(v)}\}$ is a bijection. [2] Figure 1 illustrates splitting of v. In this figure, v is split into two vertices v^1 and v^2.

[2] This operation is not well-defined in the case where self-loop is attached to the split vertex v. When there is a self-loop at v, we replace the self-looping edge $\{v, v\}$ by two edges $\{v, n\}$ and $\{n, v\}$, where n is a new degree-2 vertex. Thereafter, we implement a vertex splitting at v.

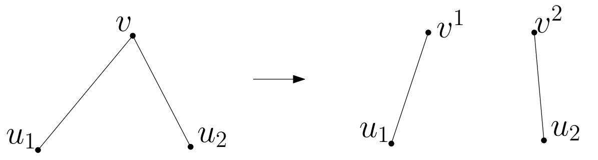


Fig. 1 Splitting of a vertex v.

2.2 Generalized Voronoi Graph(GVG)

This section reviews the generalized Voronoi graph(GVG) in [5,31,6,25]. Consider a connected and compact workspace $W \subset \mathcal{R}^2$ whose boundary, ∂W, is a regular curve. Let O_1, O_2,..., O_n be n closed and convex obstacles in W. Non-convex obstacles are modeled as the union of convex obstacles. The GVG is based on the following distance function:

$$d_i(x) = min_{c_0 \in O_i} \|x - c_0\|, \tag{1}$$

$$\nabla d_i(x) = \frac{x - c_0}{\|x - c_0\|}, \tag{2}$$

where $d_i(x)$ is the distance from a point x to an obstacle O_i, and the vector $\nabla d_i(x)$ is a unit vector in the direction from c_0 to x, where c_0 is the closest point to x in O_i.

The basic building block of the GVG is the set of points equidistant from two obstacles O_i and O_j such that each point in this set is closer to O_i and O_j than to any other obstacles. This structure is termed the *generalized Voronoi edge*,

$$E_{ij} = \{x \in \mathcal{R}^2 : 0 \le d_i(x) = d_j(x) \le d_h(x) \ ,\forall h \ne i, j \text{ and } \nabla d_i(x) \ne \nabla d_j(x)\}. \tag{3}$$

Observe that E_{ij}, E_{ik}, and E_{jk} intersect to build a structure that is equidistant from three obstacles. Such a structure is termed a *meet point* and is denoted as V_{ijk}:

$$V_{ijk} = E_{ij} \bigcap E_{ik} \bigcap E_{jk}. \tag{4}$$

Also, when a GVG edge intersects an obstacle boundary at a point, this point is termed a *boundary point.* A meet point and a boundary point are depicted in Figure 2.

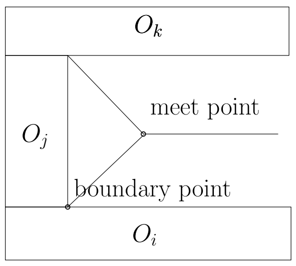


Fig. 2 A meet point and a boundary point.

3 The Weighted Voronoi Diagram

Recall that building blocks for our coordinated capturing strategy are the sweep and the guard actions. Based on the sweep and the guard actions, we simplify a cluttered environment as the weighted Voronoi diagram such that each vertex and edge has its weight. Each weight $w : V \bigcup E \rightarrow Z^{+}$, is a mapping describing the clearing requirement of edges and vertices. Clearing an edge e requires at least $w(e)$ robots to "sweep" along the passage associated to e. Also, clearing a vertex v requires at least $w(v)$ robots to "guard" the intersection associated to v.

In this section, we introduce the weighted Voronoi diagram D_w to represent a cluttered environment. Beforehand, we need to introduce several concepts. Let R_s denote the maximum range of a weapon on a robot. The *weapon footprint* of one robot is the circle with radius R_s centered at the robot. An intruder is *captured* in the case where it is inside the weapon footprint of a robot.

$q_i(x) = argmin_{c_0 \in O_i} \|x - c_0\|$ is the function to return the closest point to x from the obstacle O_i. Let $L(x, q_i(x))$ denote the line segment whose two endpoints correspond to x and $q_i(x)$ respectively. Let $\|L(x, q_i(x))\|$ denote the length of $L(x, q_i(x))$.

A meet point is equidistant from more than two points on obstacle boundaries. Let *closest points* denote these points on obstacle boundaries, which are equidistant from a meet point. Since a meet point is equidistant from closest points on obstacle boundaries, the following condition is satisfied:

- there exists a circle centered at a meet point intersecting obstacle boundaries at closest points. The interior of the circle does not intersect any obstacles.

The circle centered at a meet point satisfying the above condition is called an *intersection circle*.

Each vertex in D_w corresponds to a meet point or a boundary point in the GVG. One distinction from the GVG in Section 2.2 is that each vertex and edge in D_w has its weight. Each weight $w : V(D_w) \bigcup E(D_w) \rightarrow Z^{+}$, is a mapping describing the clearing requirement of edges and vertices.

From now on, we introduce a method to derive $w(v)$ and $w(E_{ij})$. Here, v is one vertex in $V(D_w)$, and E_{ij} is the Voronoi edge equidistant from two obstacles O_i and O_j. Deriving $w(v)$ and $w(E_{ij})$ depends on a strategy to clear the structure associated to v or E_{ij}.

In the case where v corresponds to a boundary point, $w(v)$ is zero, since no structure is associated to a boundary point.

Consider the case where v corresponds to a meet point. The structure associated to v is the set of line segments such that two endpoints of each line segment are v and a closest point on the intersection circle centered at v. To guard v indicates that at least $w(v)$ robots cover these line segments.

We provide a method to calculate $w(v)$, which is not zero. Since there are more than two closest points on an intersection circle, the structure associated to v is composed of more than two line segments. Suppose that the structure associated to v is composed of $L_s \geq 3$ line segments. Let $R(v)$ denote the radius of the intersection circle centered at v. To occupy the structure associated to v, we distribute $w(v) = \frac{L_s R(v)}{2R_s}$ robots so that the weapon footprints of the robots cover the structure.

Figure 3 illustrates one group of robots covering the structure associated to v. The intersection circle centered at v is depicted as a dotted circle. Also, other small circles represent the weapon footprints of all robots.

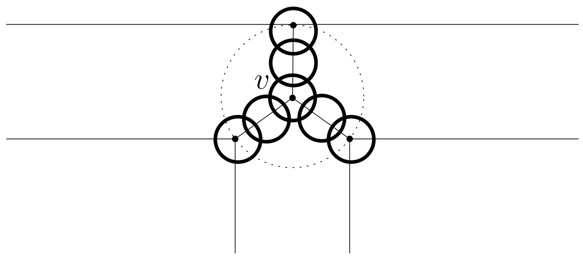


Fig. 3 One group of robots covering the structure associated to v.

The structure associated to E_{ij} is the passage between two obstacles O_i and O_j. Clearing E_{ij} requires at least $w(E_{ij})$ robots to sweep the passage between O_i and O_j.

We provide a method to calculate $w(E_{ij})$. Our strategy to clear the structure associated to E_{ij} is using a moving hinge which is composed of two line segments. Let x denote the center of the moving hinge. In the moving hinge, two line segments are $L(x, q_i(x))$ and $L(x, q_j(x))$ respectively. Initially, x is located at one endpoint of E_{ij}. At this moment, $q_i(x)$ and $q_j(x)$ are closest points on the intersection circle centered at x. Then, x moves along E_{ij} until it meets another endpoint of E_{ij}. While x moves, $\|L(x, q_i(x))\|$ is equal to $\|L(x, q_j(x))\|$, since x is on E_{ij}. As the hinge moves, robots continue to cover it with their weapon footprints. $2max_{x \in E_{ij}} \|L(x, q_i(x))\|$ determines the number of robots required to clear the passage associated to E_{ij}. Considering this sweeping strategy, we set $w(E_{ij}) = \frac{max_{x \in E_{ij}} \|L(x,q_i(x))\|}{R_s}$.

Figure 4 illustrates one group of robots covering the hinge composed of $L(x, q_i(x))$ and $L(x, q_j(x))$. E_{ij} is depicted as the dotted line. The weapon footprints of all robots are depicted as circles.

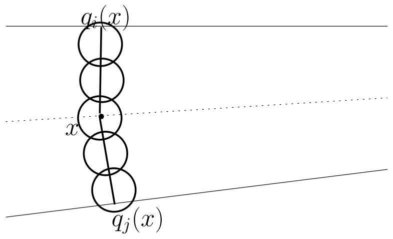
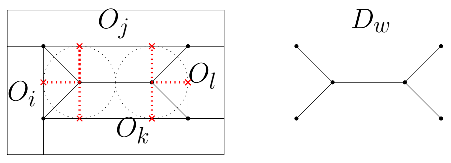


Fig. 4 One group of robots covering the hinge composed of $L(x, q_i(x))$ and $L(x, q_j(x))$.

Figure 5 shows an illustration of D_w corresponding to an workspace surrounded by four convex obstacles. In this figure, R_s is 0.25. Two intersection circles are depicted with two dotted circles in this figure. The radius of each intersection circle is 1. Also, closest points on obstacle boundaries are depicted with red crosses. Red dotted lines indicate line segments associated to a meet point. For every edge e in $E(D_w)$, $w(e) = \frac{1}{0.25} = 4$. In this figure, v_{ijk} is the Voronoi vertex equidistant from three obstacles O_i, O_j, and O_k. Since $L_s = 3$, $w(v_{ijk}) = \frac{3}{0.5} = 6$. This implies that the number of robots required to occupy the structure associated to v_{ijk} is 6.



$$w(E_{ij}) = w(E_{ik}) = w(E_{lk}) = w(E_{lj}) = 4.$$
$$w(E_{kj}) = 4.$$
$$w(v_{ikj}) = w(v_{lkj}) = 6.$$

Fig. 5 D_w corresponding to an workspace surrounded by four convex obstacles.

4 Clearing the Weighted Voronoi Diagram Assisted by Information Networks

Until now, we introduced a method to extract the weighted Voronoi diagram D_w from a cluttered environment. Now, our problem becomes capturing intruders in this weighted graph structure.

In this section, we introduce the weighted graph searching strategy assisted by the information network. Based on the searching strategy, we derive an upper bound for the number of robots required to clear the weighted Voronoi diagram.

We introduce a *guard*, whose role is to guard a vertex in D_w. Let $H = max_{\forall p \in V(D_w) \bigcup E(D_w)} w(p)$. H *free robots* form one group while moving along D_w to capture intruders. In the case where they are on an edge in D_w, they form a moving hinge to sweep along the passage associated to the edge. Since $H \geq max_{\forall e \in E(D_w)} w(e)$, no intruder can cross the hinge without being captured. In the case where the free robots meet a vertex in D_w, they guard the intersection associated to the vertex. Since $H \geq max_{\forall v \in V(D_w)} w(v)$, no intruder can cross the intersection without being captured.

Denote the minimum number of robots to capture all intruders on D_w as $s_I(D_w)$ where I indicates the information network. We derive an upper bound for $s_I(D_w)$.

We first introduce a coordinated capturing strategy to capture one intruder in the case where D_w is a tree graph, say T. Suppose n is a vertex of T. Then, we define a *branch* of T at n as the maximal subtree of T, denoted by T', if n has degree one in T'. Lemma 1 provides the searching strategy using H free robots.

Lemma 1 *To capture an intruder, H free robots move along T as one group. Whenever they meet a vertex in $V(T)$, they guard the vertex and obtain the branch containing the intruder. Then, the free robots choose the branch containing the* *intruder and sweep the edge contained in the branch until they meet another vertex. Iterate this until the intruder is captured.*

To make Lemma 1 feasible, H free robots must obtain the branch containing the intruder whenever they meet a vertex in $V(T)$. This is feasible, since information nodes are deployed to cover the entire workspace. Lemma 2 shows that the capturing strategy in Lemma 1 is robust to time delay in data transfer using the information network.

Lemma 2 *Using the capturing strategy in Lemma 1, the intruder is captured in finite time regardless of time delay in data transfer using the information network.*

Proof Suppose that H free robots meet a vertex n. Then, they guard the intersection associated to n.

Suppose that an intruder at a branch, say b, is detected at time step t_1 and that the detection event is broadcasted to the free robots at time step t_2. According to the definition of a branch, n is the only vertex connecting b with $T \setminus b$. Blocked by the free robots at n, the intruder cannot escape from b between two time steps t_1 and t_2. Thus, as the robots choose b and sweep the edge contained in b, the intruder cannot escape from b. As we iterate this, the intruder is captured in finite time.

Recall that we consider a worst-case intruder. Hence, an intruder moves along edges of D_w at unbounded speed to avoid both the free robots and guards. Since an intruder can move at unbounded speed, an intruder can escape from the free robots using a cycle in D_w. To block escape of an intruder, we deploy guards at vertices in D_w. If guards are deployed at a vertex, then the vertex becomes unavailable to intruders. Hence, once guards are deployed at a vertex, we mark the vertex as *guarded*. We say that a cycle C is *blocked* if any vertex in $V(C)$ is guarded.

We define N_g as the minimum number of guards to block all cycles contained in D_w. Computational method to search for N_g corresponds to a weighted set cover optimization problem which is known to be NP-hard in computer science. In our problem of searching for N_g, the universal set U corresponds to the set of all cycles that are contained in D_w. Let a vertex in D_w be identified by v_i where $i \leq |V(D_w)|$. We build a family of subsets $F = \{s_1, s_2, ..., s_{|V(D_w)|}\}$ of U. Here, s_i is the set of cycles satisfying that every cycle in s_i has the vertex v_i in it. This implies that every cycle in s_i is blocked by guarding $v_i \in V(D_w)$. The cost of each s_i, say $c(s_i)$, is $w(v_i)$. Our goal is to find $P \subset F$ so that it contains all cycles in U while minimizing $\sum_{s_i \in P} c(s_i)$.

For this weighted set cover optimization problem, approximation algorithms returning near-optimal solutions exist. For example, the greedy algorithm for weighted set cover problem builds a cover by repeatedly choosing a set s_i that minimizes the weight $c(s_i)$ divided by the number of elements in s_i not yet covered by chosen sets[36].

Suppose N_g guards are deployed to block all cycles contained in D_w. Since all cycles in D_w are blocked, no cycle is available to an intruder, i.e., available set of points for an intruder is a tree. Thus, an intruder cannot escape from the free robots using Lemma 1. Furthermore, the free robots can capture all intruders by chasing one intruder at a time. Since we need H free robots and N_g guards to capture all intruders on D_w, $s_I(D_w) \leq H + N_g$ follows.

Theorem 1 $s_I(D_w) \leq H + N_g$.

Algorithm 1 describes our weighted graph searching strategy. Let r_f denote one group of free robots. In practice, r_f cannot move with unbounded speed to capture an intruder. To minimize the time spent to capture an intruder, r_f finds the shortest (hop distance) path from the current position of r_f to every intruder using the breadth-first search algorithm. Among these intruders, r_f selects the one which can be caught within the minimum time interval and set it as $TARGET$. Then, r_f chases $TARGET$ using the information network.

Algorithm 1 Capturing all intruders in D_w

```
1: $N_g$ guards are deployed to block all cycles in $D_w$;
2: $r_f \leftarrow H$ free robots;
3: $r_f$.CLEAR($D_w$) using Algorithm 2;
```

Algorithm 2 r_f.CLEAR(D_w)

```
1: repeat
2:     $r_f$ searches for an intruder in $D_w$, which can be caught within the minimum time interval
       and set it as $TARGET$;
3:     $r_f$.CAPTURE($TARGET$) using Algorithm 3;
4: until  all intruders on $D_w$ are captured;
```

Algorithm 3 r_f.CAPTURE($TARGET$)

```
1: $T \subset D_w \leftarrow$ the tree which is available to $TARGET$;
2: if $r_f$ is not on $T$ then
3:     $r_f$ moves along $D_w$ to reach a vertex in $T$;
4: end if
5: repeat
6:     $r_f$ meets a vertex in $T$ and guards the vertex;
7:     $r_f$ chooses the branch of $T$ containing $TARGET$ and sweeps the edge contained in the
       branch until meeting another vertex;
8: until $r_f$ captures $TARGET$, and $TARGET$ is removed from the list of intruders;
```

4.0.1 The Switching Strategy

In Algorithm 1, only free robots move while other robots guard their designated locations. Considering the efficiency of capturing intruders, it is desirable to make a guard move if necessary. We present a switching strategy to make a guard move to capture intruders.

Before presenting our switching strategy (see Algorithm 4), we need to introduce a new graph G_T which represents the available set of points for an intruder.

Suppose we deploy N_g guards to block all cycles in D_w. Since all cycles in D_w are blocked, no cycle is available to an intruder, i.e., available set of points for an intruder is a tree. By splitting guarded vertices in D_w, we obtain G_T. G_T is composed of several disjoint tree graphs, since all cycles in D_w are blocked. In other words, $G_T = \{T_1, ...T_L\}$ where T_i denotes the ith tree graph. We say that a tree graph T_i is *cleared* when there is no intruder in T_i. We say that T_i is *contaminated* when there is an intruder in T_i. A robot is *adjacent* to a tree graph T_i if the robot guards a vertex of T_i.

Figure 6 illustrates D_w with corresponding G_T. In this figure, v_1 and v_2 in D_w are guarded to block all cycles in D_w. By splitting v_1 and v_2 in D_w, we obtain G_T. G_T is composed of two disjoint tree graphs T_1 and T_2, since all cycles in D_w are blocked.

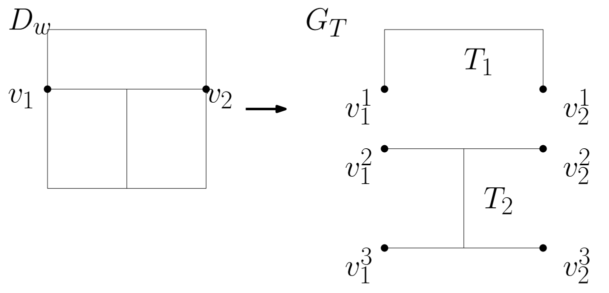


Fig. 6 An illustration of D_w with corresponding G_T.

The idea of our switching strategy is as follows. As time goes on, the number of cleared trees increases. The guards which are not adjacent to any contaminated tree do not have to guard their designated locations anymore. In this case, we can let guards maneuver to improve the time efficiency of graph clear operation considerably.

Lemma 3 *Consider the intruder capturing strategy using Algorithm 4. Suppose we choose n groups of robots $(r_1, r_2, ...r_n)$ to clear every tree graph $(T_1, T_2, ...T_n)$ in $TreeSet$. Using Algorithm 5, r_i, $\forall i \leq n$, clears T_i in finite time.*

Proof Suppose we choose n groups of robots $(r_1, r_2, ...r_n)$ to clear every tree graph $(T_1, T_2, ...T_n)$ in $TreeSet$. The guarding position of each robot in r_i, $\forall i \leq n$, is not adjacent to T_i or to any tree in $TreeSet$. Thus, even if every robot in r_i leaves its guarding position to clear T_i using Algorithm 5, no intruder can sneak into T_i or into any tree in $TreeSet$. According to Lemma 1, r_i captures every intruder in T_i one by one. Thus, T_i is cleared in finite time.

Lemma 4 *Consider the intruder capturing strategy using Algorithm 4. Once a tree graph is cleared, no intruder can sneak into the tree graph again.*

Proof Suppose we choose one group of robots, say r_u, to capture all intruders in one contaminated tree graph $T_u \in G_T$. In Algorithm 4, S is initialized as the set of guards, which are not adjacent to any contaminated tree. Thus, the guarding position of each robot in r_u is not adjacent to any contaminated tree. This implies that guards adjacent to a contaminated tree do not move at all. Thus, as every robot in r_u leaves its guarding position to clear T_u, no intruder can sneak into any cleared tree graph.

Algorithm 4 Clearing D_w Using $H + N_g$ Robots

```
$N_g$ robots guard vertices of $D_w$ to block all cycles in $D_w$;
Let $G_T = \{T_1, ...T_L\}$;
There are additional $H$ free robots which are not adjacent to any tree graph in $G_T$;
repeat
  Let $S$ denote the set of guards, which are not adjacent to any contaminated tree;
  $TreeSet = \emptyset$;
  $i \leftarrow 1$;
  repeat
    if $T_i \in G_T$ is contaminated then
      if the number of robots in $S$, which are not adjacent to any tree graph in $G_T \setminus (TreeSet \bigcup \{T_i\})$, is bigger than or equal to $H$ then
        $TreeSet = TreeSet \bigcup \{T_i\}$;
        $r_i \leftarrow H$ robots in $S$, which are not adjacent to any tree graph in $G_T \setminus TreeSet$;
        $S = S \setminus r_i$;
        $i = i + 1$;
      else
        $i = L + 1$;
      end if
    end if
  until $i == L + 1$;
  Simultaneous Clearing of Every Tree in $TreeSet$(Algorithm 5)
until all tree graphs are cleared
```

Algorithm 5 Simultaneous Clearing of Every Tree in $TreeSet$

```
$T_j \leftarrow$ every tree in $TreeSet$;
$P_j \leftarrow$ the position at which each robot in $r_j$ is located;
Each robot in $r_j$ moves along $D_w$ to gather at one vertex in $T_j$;
$r_j$.CLEAR($T_j$) using Algorithm 2;
repeat
  $r_j$ searches for a contaminated tree, say $T_k$, in $TreeSet$;
  $r_j$.CLEAR($T_k$) using Algorithm 2;
until all trees in $TreeSet$ are cleared
Each robot in $r_j$ gets back to the previous position $P_j$;
```

Thus, once a tree graph is cleared, no intruder can sneak into the tree graph again.

Theorem 2 *Using Algorithm 4, all intruders are captured in finite time.*

Proof Suppose we choose one group of robots, say r_i, to capture all intruders in one contaminated tree graph $T_i \in G_T$. According to Lemma 3, r_i clears T_i in finite time. Moreover, according to Lemma 4, once T_i is cleared, no intruder can sneak into T_i again.

Suppose that there is at least one contaminated tree as i in Algorithm 4 is set to 1. We prove that as i in Algorithm 4 changes from 1 to $L + 1$, $TreeSet$ contains at least one contaminated tree. This further indicates that the number of cleared trees increases, since every tree in $TreeSet$ is cleared simultaneously using Algorithm 5.

We prove that as i in Algorithm 4 increases from 1 to $L+1$, $TreeSet$ contains at least one contaminated tree. Suppose that i in Algorithm 4 has been increasing from 1 to l. Suppose that T_l is contaminated and that $TreeSet$ has been empty until i reaches l. Whenever $TreeSet$ is empty, S contains additional H free robots which are not adjacent to any tree graph in G_T. These robots are not adjacent to any tree graph in $G_T \setminus (\{T_l\})$. Hence, T_l is added to $TreeSet$. We proved that as i in Algorithm 4 increases to $L+1$, $TreeSet$ contains T_l which is contaminated.

Since Algorithm 4 requires $H + N_g$ robots to capture all intruders in a graph D_w, we obtain the following theorem.

Theorem 3 *The number of robots required to clear D_w is equal to or less than $H + N_g$.*

Algorithm 4 is time-efficient, since every tree graph in $TreeSet$ can be cleared simultaneously using Algorithm 5. For example, assume that there are n tree graphs in $TreeSet$. Correspondingly, n groups of robots are selected to clear n tree graphs respectively. n groups can run Algorithm 5 simultaneously to clear n tree graphs. In this way, time efficiency of Algorithm 4 increases significantly compared to the case where only H free robots maneuver.

Algorithm 4 is designed not to make a robot idle. Suppose that T_i is in $TreeSet$ and that r_i is selected to clear T_i. Suppose that there are only a few intruders in T_i and that T_i is cleared before all trees in $TreeSet$ are cleared. In this case, r_i must not be idle. Therefore, once r_i finishes clearing T_i, it begins to clear another contaminated tree in $TreeSet$. This process is presented in Algorithm 5.

5 Conclusions

Consider a team of robots deployed to capture intruders in an environment cluttered with many obstacles. We assume that a robot can detect the position of any intruder using the information network. This paper introduces a method to extract the weighted Voronoi diagram from a cluttered environment. Then, our problem becomes capturing intruders in this weighted graph structure. We introduce the weighted graph searching strategy assisted by the information network. Based on the searching strategy, we derive an upper bound for the number of robots required to clear the weighted Voronoi diagram. Finally, we present the switching strategy to improve the time efficiency of capturing intruders while not increasing the number of robots.

References

1. Barriére, L., Ere, L.B., Flocchini, P., Fraigniaud, P., Santoro, N.: Capture of an intruder by mobile agents. In: Proc. of the fourteenth annual ACM symposium on Parallel algorithms and architectures, pp. 200–209. Canada (2002)
2. Bienstock, D., Seymour, P.: Monotonicity in graph searching. Journal of Algorithms **12**(2), 239–245 (1991)
3. Brandenburg, F.J., Herrmann, S.: Graph searching and search time. SOFSEM 2006: Theory and Practice of Computer Science **3831**, 197–206 (2006)
4. Choset, H., Konukseven, I., Burdick, J.: Mobile robot navigation: issues in implementating the generalized Voronoi graph in the plane. In: Proc. of IEEE/SICE/RSJ International Conference on Multisensor Fusion and Integration for Intelligent Systems, pp. 241–248. Washington DC, USA (1996)
5. Choset, H., Nagatani, K.: Topological simultaneous localization and mapping (SLAM): toward exact localization without explicit localization. IEEE Transactions on Robotics and Automation **17**, 125–137 (2001)

6. Choset, H., Walker, S., Eiamsa-Ard, K., Burdick, J.: Sensor-based exploration: Incremental construction of the hierarchical generalized Voronoi graph. The International Journal of Robotics Research **19(2)**, 126–148 (2000)
7. Chung, T.H., Hollinger, G.A., Isler, V.: Search and pursuit-evasion in mobile robotics. Autonomous Robots **31**, 299–316 (2011)
8. Cortés, J., Martínez, S., Karatas, T., Bullo, F.: Coverage control for mobile sensing networks. IEEE Transactions on Robotics and Automation **20**(2), 243–255 (2004)
9. Culler, D., Estrin, D., Srivastava, M.: Overview of sensor networks. IEEE Computer Magazine **37**(8), 41–49 (2004)
10. Dendris, N.D., Kirousis, L.M., Thilikos, D.M.: Fugutive-search games on graphs and related parameters. Theoretical Computer Science **172**, 233–254 (1997)
11. Dereniowski, D.: Connected searching of weighted trees. Theoretical Computer Science **412**, 5700–5713 (2011)
12. Douglas, B.W.: Introduction to Graph Theory, 2 edn. Prentice Hall, Illinois, USA (2001)
13. Fomin, F.V., Fraigniaud, P., Nisse, N.: Nondeterministic graph searching: From pathwidth to treewidth. Algorithmica **53(3)**, 358–373 (2009)
14. Fomin, F.V., Thilikos, D.M.: An annotated bibliography on guaranteed graph searching. Theoretical Computer Science **399**, 236–245 (2008)
15. Kim, J.: Cooperative exploration and protection of a workspace assisted by information networks. Annals of Mathematics and Artificial Intelligence **70**, 203–220 (2014)
16. Kim, J., Maxon, S., Egerstedt, M., Zhang, F.: Intruder capturing game on a topological map assisted by information networks. In: Proc. of IEEE Conference on Decision and Control, pp. 6266–6271. Orlando, USA (2011)
17. Kim, J., Zhang, F., Egerstedt, M.: Simultaneous cooperative exploration and networking based on Voronoi diagrams. In: Proc. of IFAC Workshop on Networked Robotics, pp. 1–6. Colorado, USA (2009)
18. Kim, J., Zhang, F., Egerstedt, M.: A provably complete exploration strategy by constructing Voronoi diagrams. Autonomous Robots **29(3-4)**, 367–380 (2010)
19. Kim, J., Zhang, F., Egerstedt, M.: Simultaneous Cooperative Exploration and Networking. Scholars' Press (2013)
20. Kirousis, L.M., Papadimitriou, C.H.: Interval graphs and searching. Discrete Mathematics **55**(2), 181–184 (1985)
21. Kirousis, L.M., Papadimitriou, C.H.: Searching and pebbling. Theoretical Computer Science **42**(2), 205–218 (1986)
22. Kolling, A., Carpin, S.: Extracting surveillance graphs from robot maps. In: Proc. of Intelligent Robots and Systems, pp. 2323 – 2328. Nice (2008)
23. Kolling, A., Carpin, S.: The graph-clear problem: definition, theoretical properties and its connections to multirobot aided surveillance. In: Proc. of the 2007 IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 1003–1008. San Diego, USA (2009)
24. Kolling, A., Carpin, S.: Pursuit-evasion on trees by robot teams. IEEE Transactions on Robotics **26**, 32–47 (2009)
25. Kolling, A., Carpin, S.: Surveillance strategies for target detection with sweep lines. In: Proc. of Intelligent Robots and Systems, pp. 5821 – 5827. USA (2009)
26. Lapaugh, A.: Recontamination does not help to search a graph. Journal of the ACM **40(2)**, 224–245 (1993)
27. Lavalle, S.M.: Planning Algorithms. Cambridge University Press (2006)
28. Liu, H., Li, J., Xie, Z., Lin, S., Whitehouse, K., Stankovic, J.A., Siu, D.: Automatic and robust breadcrumb system deployment for indoor firefighter applications. In: Proceedings of the 8th international conference on Mobile systems, applications, and services (MobiSys 10), pp. 21–34. ACM, San Francisco, California, USA (2010)
29. Martínez, S., Cortés, J., Bullo, F.: Motion coordination with distributed information. IEEE Control Systems Magazine **27(4)**, 75–88 (2007)
30. Megiddo, N., Hakimi, S.L., Garey, M.R., Johnson, D.S., Papadimitriou, C.H.: The complexity of searching a graph. Journal of the ACM **35**, 18–44 (1988)
31. Nagatani, K., Choset, H.: Toward robust sensor based exploration by constructing reduced generalized Voronoi graph. In: Proc. of IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 1687–1692. Kyongju, Korea (1999)
32. Rao, N., Stoltzfus, N., Iyengar, S.: A retraction method for learned navigation in unknown terrains for a circular robot. IEEE Transactions on Robotics and Automation **7**, 699–707 (1991)

33. Richerby, D., Thilikos, D.M.: Graph searching in a crime wave. SIAM Journal on discrete mathematics **23(1)**, 349–368 (2009)
34. Seymour, P.D., Thomas, R.: Graph searching and a min-max theorem for tree-width. J. Combin. Theory Ser. B **58**, 22–33 (1993)
35. Souryal, M.R., Geissbuehler, J., Miller, L.E., Moayeri, N.: Real-time deployment of multihop relays for range extension. In: Proceedings of the 5th international conference on Mobile systems, applications and services (MobiSys 07), pp. 85–98. ACM, New York, NY, USA (2007)
36. Vazirani, V.V.: Approximation Algorithms. Springer (2010)