Technische Universität Hamburg-Harburg

Institute for Software Systems

Area Coverage Algorithms for Multiagent Surveillance Tasks

Master Thesis

Author: Pavichaya Eaungpulswat
Supervisors: Prof. Dr. Ralf Möller
Prof. Dr. Herbert Werner

May 21, 2012

Declaration

I hereby assert that this thesis has been compiled by me with only the help of the auxiliary material listed in the text or in the bibliography.

Hamburg, May 21, 2012

Pavichaya Eaungpulswat

Contents

Abstract

This thesis presents algorithms for area coverage in non-communicative and communicative surveillance tasks using a team of unmanned aerial vehicles (UAVs). In non-communicative tasks, the UAV agents sweep back and forth over a region of interest to perform area coverage. The polygon area decomposition and individual areas coverage algorithms are presented. The algorithms are intended for a situation where the UAV agents have to start and end area coverage at the same base station. In communicative area coverage, the methods for static node positioning and dynamic node repositioning are presented. The algorithms for dynamic repositioning are applied to assist a situation where the nodes are at the predetermined locations from static positioning, but there is one or more missing nodes in the region of interest. The results and analysis of the algorithms on the given situation is presented.

Chapter 1

Introduction

The area coverage problem in this thesis is divided into area coverage without communication (non-communicative) and area coverage with communication (communicative). The purpose of both area coverage is to sense a region of interest with equipped camera or other type of sensor. The agent may take a video or pictures of an observed region to provide an efficient overview to a base station. The applications for area coverage are such as forest fire monitoring, military surveillance, aerial mapping, private security, and other remote surveillance tasks.

Unmanned aerial vehicles (UAVs) can be used in area coverage tasks. They are sometimes preferred over manned aircraft in a task such as a routine or risky surveillance mission. The UAVs are relatively inexpensive, have smaller size than manned aircraft, and can be used to minimize the cost of the mission.

This thesis concerns cooperative area coverage operation of multi-agent. Multiple UAVs are used to perform the tasks cooperatively. They enhance the effectiveness of the tasks in large area coverage and explore different aspect of locations at the same time.

When the number of UAV agents is limited and is not enough to entirely cover a region of interest, offline path planning is used for area coverage. The task for each agent is subdivided disjointedly. Each agent performs area coverage task by sweeping and sensing its assigned region. A team of agent cooperates to complete the task as quickly as possible.

The UAV agent sweeps back and forth to cover its region. The back and forth motion contains sharp turns and is not applicable in fixed wing UAVs, since the fixed wing aircrafts are restricted to constant forward movement. They cannot make sharp turns, or stop and maintain a specific position. Quadrotor, on the other hand, has the ability to hover in place, make tight turns, and move in any direction [10]. Nevertheless, when the

route contains sharp turns, turning would require the quadrotor to slow to a halt at sharp corners [16]. An example from [4] shows that moving a quadrotor along the shortest safe path with sharp corners requires more time than moving along the free path of the quadrotor. Hence, the number of sharp turns is related to the amount of time used in performing area coverage.

The area coverage route should not overlap and should lead to the complete area coverage with minimum time and energy. Overlapped area or area redundancy occurs when a location has been visited by one of the agents and is visited again by the same or different agent in the team. This wastes time and energy consumption in the mission. The estimated lifetime of the agents is determined from the amount of battery remaining in the agents.

Considering an environment where a group of agents assembles at a base station, area coverage algorithm should start and end at the same position. The problem then arises as in which way to divide an area and how to allocate the subareas to the agents. A polygon area decomposition problem presented in [15] is used to solve the problem. In addition, another area decomposition algorithm is proposed aiming to improve area coverage performance of the UAVs.

In this thesis, the region of interest in a disaster area or where there is no networking infrastructure is assumed. When the number of agents is not enough to spread over the region, the agents have to perform area coverage non-communicatively. However, when there are enough agents to entirely cover the region of interest, mobile ad hoc network (MANET) can be applied and communication is possible.

In area coverage with communication, the agents can communicate among themselves and to the base station. The agents are positioned that the base station and all the agents stay connected during the mission. Each agent senses its own region and sends the data back to the base station. This area coverage allows the surveillance to last longer than area coverage that sweeps along the region, since each agent observes only one location. The communicative area coverage requires more number of agents, however, it gives the base station near real-time surveillance data.

The node positioning in this area coverage is divided into static positioning where the locations are predetermined from the base station and dynamic repositioning where the agents determine their locations autonomously. Dynamic repositioning can be used to assist area coverage when the nodes are randomly dispersed or when a region of interest is not entirely covered.

The outline of this thesis is as follows: section 2 presents non-communicative area coverage with relative and sensing capabilities of the UAVs and decomposition algorithm. Section 3 introduces a proposed area decomposition algorithm based on geometry of polygon and required subareas. Section 4 presents communicative area coverage where the agents and base station are connected. Conclusion and future work is presented in section 5.

Chapter 2

Non-Communicative Area Coverage

This section presents area coverage for multi-agent surveillance tasks where each agent performs its task disjointedly. Each agent is responsible only for its divided area and all the information from each agent are combined for an image of the entire area. The data such as assigned area and a route for each agent is calculated by the base station before the task start. During the task, an agent simply follows the predetermined route of its assigned region with the use of built-in Global Positioning System (GPS).

Maza and Ollero [22] have presented the problem of cooperatively searching a given area to detect objects of interest, using a team of heterogenous UAVs. The problem they have presented can be applied to an area coverage problem for multiagent surveillance tasks, as the function of the two tasks are related. Their algorithms start with assigning a region for each UAV, and then covering a region using a zigzag pattern.

An assigned region is derived according to an area decomposition algorithm. The region is partitioned from an entire area considering factors such as initial coordinates, camera angles, sensing width and relative capabilities of the UAV agent. The area partition used in this section is from area decomposition algorithm by Hert and Lumelsky [15].

After the subregion for an agent is assigned, a route within the sub-region is determined. The route is obtained regarding to the shape of the assigned region by finding a direction that yields minimum diameter function. Then the agent sweeps towards that direction using back and forth motion to entirely cover its assigned region. The sweep direction used for the route finding in this part is introduced by Huang [17].

An area decomposition including relative capabilities of the agents and a decomposition algorithm is presented in section 2.1. Section 2.2 presents individual area coverage including sensing capabilities and optimal line sweep decompositions.

2.1 Area Decomposition

The area decomposition problem is the problem of dividing a given area into a set of smaller subregions. There are various algorithms for area or task decomposition. These algorithms vary upon the shape of the environment and the divided subregions. For example, there is an algorithm for multiple UAVs area coverage in an arbitrary rectilinear workspace [1]. This partitioning algorithm is therefore divides a workspace into n contiguous rectilinear parts whose areas are in a pre-specified ratio. For general polygon environments, the most common and widely applicable form of polygon decomposition is triangulation. These decompositions create triangulations of polygons with various characteristics as can be seen in [6] [8] [9]. Apart from triangulation, the polygons can be decomposed into trapezoids [2], convex polygons [5] [11] [18] [25], star-shaped or monotone polygons [18], and rectangles [21].

In this thesis, the interested area partition problem is the one that divides a given arbitrary convex polygon into a number of convex pieces, each with a given area. The number of convex pieces varies according to the number of agents available for area coverage. The size of the divided area varies according to relative capabilities of each agent. The capabilities considered in area partition are such as flight speed, battery lifetime, and width of camera's view of the UAV agent.

Section 2.1.1 presents the relative capabilities of the UAVs which describes the difference between each agent. Section 2.1.2 presents a polygon area decomposition algorithm. The algorithm partitions a polygon according to required areas and starting points of the UAV agents.

2.1.1 Relative Capabilities of the UAVs

As the multiagents considered in this thesis are a set of small quadrotors, the capability of each agent is confined by its specifications such as weight, battery lifetime, payload, and sensors specifications. These factors strongly influence the agent's maximum flying range which is one of the fundamental measurements for the agent's capability. As the UAVs can be heterogeneous, the other factors namely flight speed, sensitivity to wind conditions, altitude required for the mission, and sensing width due to different camera's fields of view for each UAVs should also be taken into account.

The sensing width depends on the UAV's specification and also varies according to the flying altitude of the agents. The camera sensor has a bigger sensing width as the

UAV flies at higher altitude above the ground. Nevertheless, it gives poorer image quality for further processing. The flying altitude of a UAV depends on the purpose of the multiagent mission, and situation of that moment e.g. the wind condition. One have to choose whether to have a finer image quality or a larger sensor range.

Based on the relative capabilities of the agents, the proportions of the area of the region P are determined in order to assign them to each agent. These proportions are represented by a set of values c_i where $i = 1, ..., n$ and n is the number of UAV agents, with

$$0 < c_i < 1 \quad \text{and} \quad \sum_{i=1}^{n} c_i = 1.$$

Therefore, the problem considered is as follows: Given a polygon P and n starting points (sites) $S_1, \cdots, S_n$ on the polygon, divide the polygon into n nonoverlapping polygons $P_1, \cdots, P_n$ such that

$$Area(P_i) = c_i \times Area(P) \quad \text{and} \quad S_i \text{ is on } P_i.$$

2.1.2 Algorithm

This area decomposition algorithm is presented by Hert and Lumelski [15]. It uses sweep-line and divide-and-conquer techniques to construct a polygon partition. The area partition problem is constrained by area requirements of each subregion and starting position of each agent.

In an area partition of a polygon P, a set of nonoverlapping sub-polygons $P_1, ..., P_n$ with a specified area and a set of start positions of the agent for performing area coverage $S_1, ..., S_n$ are divided. Each polygon P_i consists of a particular point S_i on the border of P_i. For each start position S_i, an area requirement or desired area of each polygon P_i is denoted by $AreaRequired(S_i)$.

In the problem, $AreaRequired(S_i) = c_i \times Area(P)$ where $0 < c_i < 1$ and $\sum_{i=1}^{n} c_i = 1$ is considered. Furthermore, for any set of start position S,

$$AreaRequired(S) = \sum AreaRequired(S_i).$$

For the area partition of n agents, the desired subregions can be achieved using $n-1$

line segments or the area partition algorithm has to perform $n-1$ times.

Let $L = (L_s, L_e)$ denotes a line segment oriented from L_s to L_e dividing the polygon P into two convex pieces. P_L^r is a convex polygon to the right of L and P_L^l is the complement of P_L^r; $P - P_L^r$. The algorithm for dividing a simply connected, convex polygon into two smaller regions is as follow:

Algorithm 1 Divide a convex polygon into two smaller polygons [15]

Input: convex polygon P;
the list of vertices and start positions $W(P) = w_k$, $k = 1, ..., m$ in CCW order;
the set of start positions $S(P) = S_1, ..., S_n$ numbered according to their order in the list $W(P)$

Output: two polygons P_L^l and P_L^r

1. Assume $S_1 = w_k$; $L \leftarrow (w_1, w_k)$
2. $S(P_L^r) \leftarrow \{S_1\}$
3. **while** $Area(P_L^r) < AreaRequired(S(P_L^r))$ and $L_e \neq S_n$ **do**
   $k \leftarrow k+1$
   $L_e \leftarrow w_k$
   end while
4. **if** $L_e = S_1$ and $Area(P_L^r) > AreaRequired(S(P_L^r))$ **then**
   Move L_s CCW along P_L^r until $Area(P_L^r) = AreaRequired(S(P_L^r))$
   else if $L_e = S_n$ and $Area(P_L^r) < AreaRequired(S(P_L^r))$ **then**
   Move L_s CW along P until $Area(P_L^r) = AreaRequired(S(P_L^r))$
   else
   Interpolate L_e CW until $Area(P_L^r) = AreaRequired(S(P_L^r))$
   end if
5. $P_L^l = P - P_L^r$
6. $L = (L_s, L_e)$

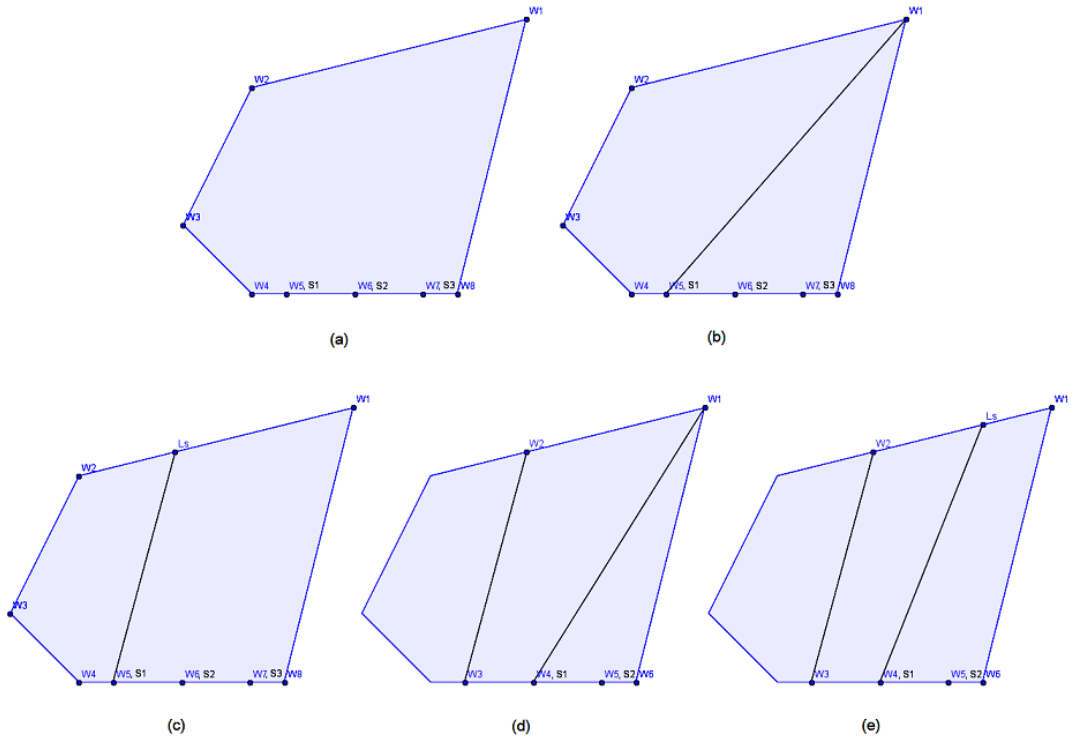

Figure 2.1: Steps for polygon area decomposition. In this figure, the polygon is divided into three polygons with equal areas

Algorithm 1 summarizes the procedure for dividing a given convex polygon P into two smaller polygons. The steps are visually presented in figure 2.1. In the figure, a given convex polygon is considered as the interested region for area coverage and there are three UAV agents performing the task. Figure (a) shows a convex polygon P with five vertices w_1, w_2, w_3, w_4, w_8 and start positions of the three UAVs S_1, S_2, S_3 or $w5, w6, w7$. Figure (b) shows the polygon P with line segment $L = (L_s, L_e)$ initialized as the segment (w_1, w_5). Then it checks whether the area of polygon $P_L^r(w_1, w_2, w_3, w_4, w_5)$ is smaller or bigger than the required area. If the area of P_L^r is bigger, moves L_s from w_1 counter clockwise along the polygon P until the area P_L^r is equal to the required area. If the area of P_L^r is smaller, then moves L_s clockwise. Figure (c) shows the line segment $L = (L_s, L_e)$ that makes area of P_L^r equal to the required area. According to Algorithm 1, the polygon area decomposition has completed its task in dividing a polygon P into two smaller polygons. Yet, three polygons are needed for area coverage task. The algorithm has to be repeated for a convex polygon $P_L^l(w_1, w_2, w_3, w_6)$. Figure (d) shows a decomposition of convex polygon P_L^l. Line segment L is initialized as the segment (w_1, w_4). Figure (e) shows the complete polygon area decomposition where the polygon P is divided equally into three polygons.

2.2 Individual Areas Coverage

Once the desired subregion is divided for each UAV, the path for covering each subregion is needed for the agent to perform area coverage. The path for individual area coverage is based on the shape of the subregion and sensing capability of the agent. Section 2.2.1 presents the sensing capabilities of the agents and their resulting sensing widths. Section 2.2.2 presents the path for covering a subregion using back and forth motion.

2.2.1 Sensing Capabilities

To examine a region of interest, a team of UAVs have to perform a cooperative search operation over the region. Each UAV is equipped with sensors or cameras. The sensors allow the vehicles to determine their own coordinates relative to a Base Coordinate System (BCS) and the coordinates of points detected in their sensing regions. The BCS is fixed in the environment defining x-axis towards north, y-axis towards west, and z-axis points upward.

Each UAV has associated an UAV Coordinate System (UCS); x-axis forward, y-axis

left, z-axis up. When a vehicle moves, the starting point and orientation of the UAV change. A fixed on board camera in each UAV can orient in $x-z$ plane of the UCS. The orientation is defined by an angle $-\alpha$ with respect to the x-axis.

As the UAV moves along a defined path, it takes shots or video over an interested region. An *imaged area* on the terrain is defined from the *image pyramid* of the camera as shown in figure 2.2. Considering a plain terrain, it can be shown that the sensing width of an UAV moving in the $x-z$ plane of the UCS is given by [22]:

$$w = 2z_{BCS}\tan\gamma[\sin\alpha + \cos\alpha\tan(\frac{\pi}{2} - \alpha - \beta)] \tag{2.1}$$

where z_{BCS} is the altitude of the UAV, and angles β and γ determine the field of view of the camera.

To generalize the planar algorithm in equation 2.1, the non-planar surface in three-dimensional environment is considered. The area to be covered must be a *vertically projectively planar surface*; a vertical line passing through any point on the surface intersects it at only one point.

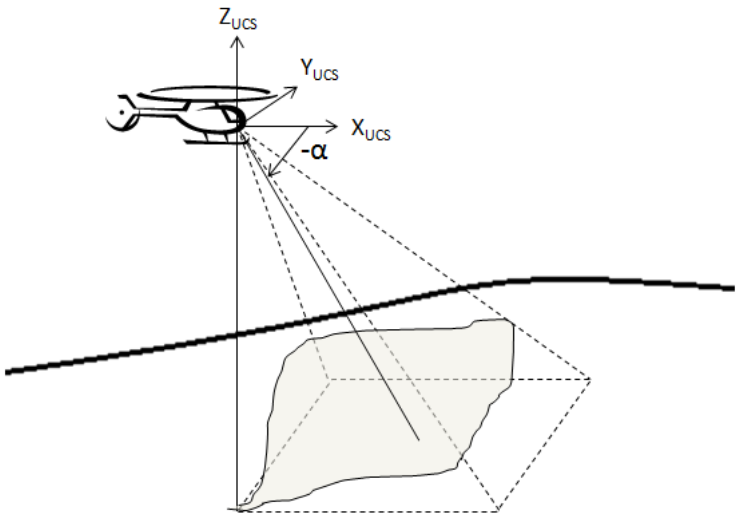

Figure 2.2: The imaged area is the intersection of the image pyramid and the terrain [22].

2.2.2 Optimal Line Sweep Decompositions

Subregions can be covered by sweeping the agents using back and forth motion. This coverage can be improved by determining the optimal sweep direction. The optimal line sweep decompositions presented by Huang [17] is used to optimize the amount of time used in individual area coverage. The method finds the route direction that minimizes the number of turns when performing area coverage.

Since this area coverage path contains sharp turns, the UAV requires more time when turning comparing to going straight. When turning, the UAV needs to slow down, stay hovering, make the turn, and then accelerate again for efficient area coverage. Accordingly, finding the path that minimizes the number of turns reduces the coverage time as illustrated in Figure 2.3. The figure shows that different sweep direction can result in large difference in number of turns, even though the total length is the same.

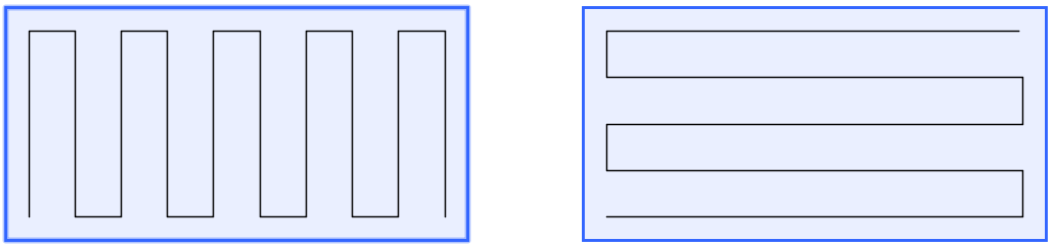

Figure 2.3: The number of turns is different as the agent moves along different sweep directions. This results in different amount of time needed for covering a region [17].

The optimal sweep direction can be determined from the minimum altitude of the polygon or the minimum diameter function. A diameter function $d(\theta)$ describes the height of the polygon along the sweep direction. For a given angle θ, the diameter of a polygon is determined by rotating the polygon by $-\theta$ and measuring the height difference between its highest and lowest point [17]. The length of a diameter function can be found by considering the height of the polygon as it rolls along a flat surface. Figure 2.4 shows an example of a diameter function of a rectangle generated using [19].

Figure 2.5 shows the diameter function of a polygon by rotating the figure counterclockwise and finding its height. The angle that gives minimum diameter is a position of the polygon for finding sweep direction. In the figure, the rotation of the polygon that gives maximum diameter is shown in (d) and minimum diameter is shown in (e). The sweeping path for covering the region has the least number of turns when the sweep

direction is perpendicular to x axis. In other words, the sweep line and the rows for covering the polygon are parallel to x axis. Hence, the optimum path for a polygon in figure 2.5 is having the sweep line parallel to x axis in figure (e).

Once the optimal sweep direction is found for each subregion, the path for area coverage can be determined using back and forth motion. The spacing between the back and forth motion depends on the sensing capabilities of the UAVs. The UAV's sensing capability is calculated as a sensing width as presented in equation 2.1.

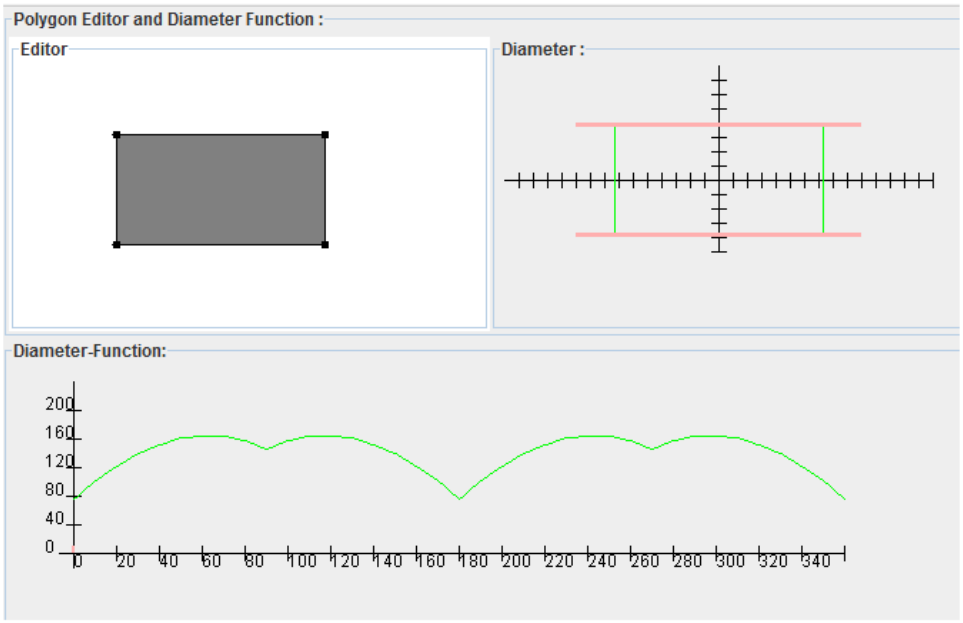

Figure 2.4: An example of a simple diameter function generated using [19].

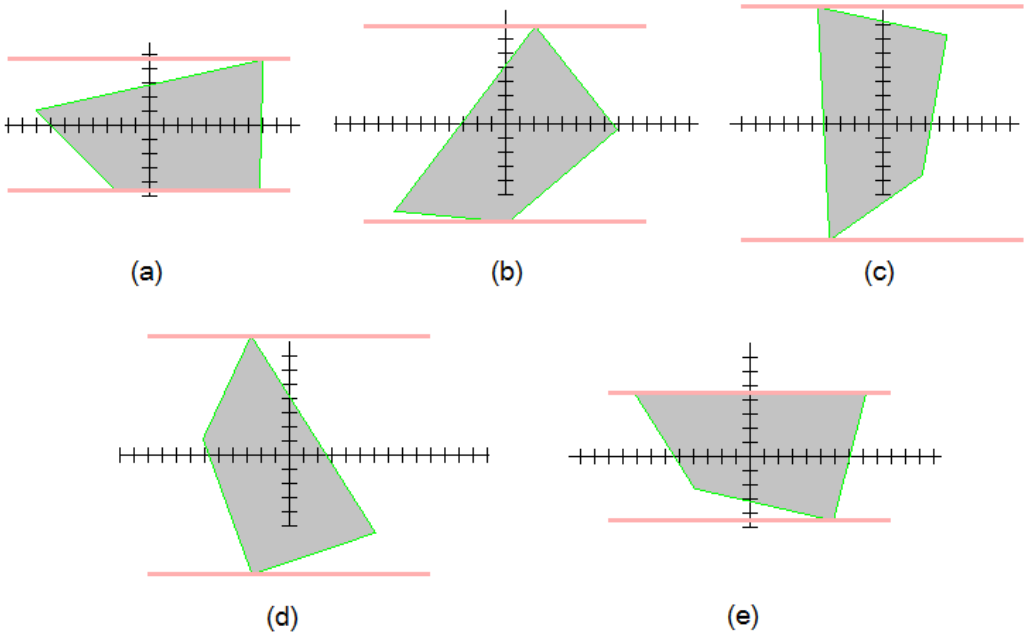

Figure 2.5: Diameter of a polygon rotated counterclockwise. (a) 0 degree (b) 40 degree (c) 80 degree (d) maximum diameter at 289 degree (e) minimum diameter at 347 degree [19].

Chapter 3

Geometry-Based Area Decomposition

This chapter presents a proposed algorithm for area decomposition and algorithms for individual areas coverage. The algorithms are aiming to minimize coverage time spending on the areas by decreasing the number of turns in area coverage, and to minimize area coverage redundancy. It is expected to reduce energy consumption and hovering distance.

As area decomposition algorithm presented in section 2.1.2 does not concern about the shape of the partitioned region, the resultant minimum diameter function is not optimal. The shape of the partitioned region should be considered in area decomposition because it relates to the number of turns when performing area coverage. Also with the location of a base station, it should be considered because it relates to area coverage route and redundancy. Therefore, the decomposition algorithm should concern both shapes of the partitioned regions and location of a base station concurrently.

The proposed area decomposition algorithm in this section concerns with area requirement, location of a base station, and geometry of a polygon which affects shapes of the partitioned regions. The algorithm uses the area requirement to assign the size of a subregion. It makes use of the location of a base station to minimize route redundancy by determining the route that starts and ends at the base station without overlapping the already visited region. It also considers geometry of a polygon to give partitioned regions with smaller minimum diameter functions, in order to decrease the number of turns and increase the distance of straight paths.

The geometry-based area decomposition algorithm is presented in section 3.1. An analysis of the algorithm is presented in section 3.2. Section 3.3 presents routes that give less area coverage redundancy. Section 3.4 presents discussion and comparison of the results from non-communicative area coverage and the proposed algorithms.

3.1 Decomposition Algorithm

The algorithm aims to partition the region of interest such that an individual agent performs area coverage task in the subregion with less number of turns comparing to when it performs in the subregion from section 2.2. The algorithm starts with examining the geometry of a polygon P to find the longest side of the polygon, and then performs polygon decomposition.

As summarized in **Algorithm 2**, the algorithm requires that a line segment L dividing the polygon P into two smaller polygons is parallel to the longest side of the polygon P. There are two cases for the decomposition. One is when the longest side is next to the side of a base station, and another is when it is not. In the first case, the area decomposition is simple. The line segment L that divides the region of interest is parallel to the longest side. It moves toward or away from the longest side to vary the area of a subregion. In the second case, two line segments are used to divide the region. One is the line segment parallel to the longest side and another one is the line segment parallel to the side adjacent to the side of a base station. The reasons behind these two line segments are that the agent has more straight path when sweeping along the longest side, and it can start and end at the base station.

For the area partition of n agents, the algorithm has to perform $n-1$ times for n subregions. The number of vertices of a polygon is denoted by parameter m. The vertices of polygon P in the algorithm are $W(P) = W_i$ where $i = 1,...,m$. In the first case of decomposition, a line segment that divides the polygon P into two convex pieces is $L = (L_s, L_e)$. The segment is oriented from L_s to L_e. In the second case, a line segment L_1 is parallel to the side next to the side of a base station and a segment L_2 is parallel to the longest side of polygon P. An area requirement or desired area of each subregion is denoted by *AreaRequired*.

Figure 3.1 illustrates the steps for dividing the polygon P in the second case where the longest side is not adjacent to the side of a base station. In this example, the polygon P is divided into three polygons with equal areas. Hence, the *AreaRequired* is $\frac{1}{3}$ of area of the polygon P. Figure (a) shows a polygon P with five vertices $(m = 5)$. Its longest side *Long* is (W_4, W_5). Figure (b) shows line segments L_1 and L_2 that are parallel to sides of the polygon P. In the figure, an area of a divided polygon $P_1(W_1, L_s, its, L_e, W_4, W_5)$ is smaller than the *AreaRequired*. Figure (c) shows an interpolation of the line segment L_2 that makes an area of the polygon P_1 equal to the *AreaRequired*. Figure (d) shows the complete polygon area decomposition where the polygon P is divided equally into three polygons.

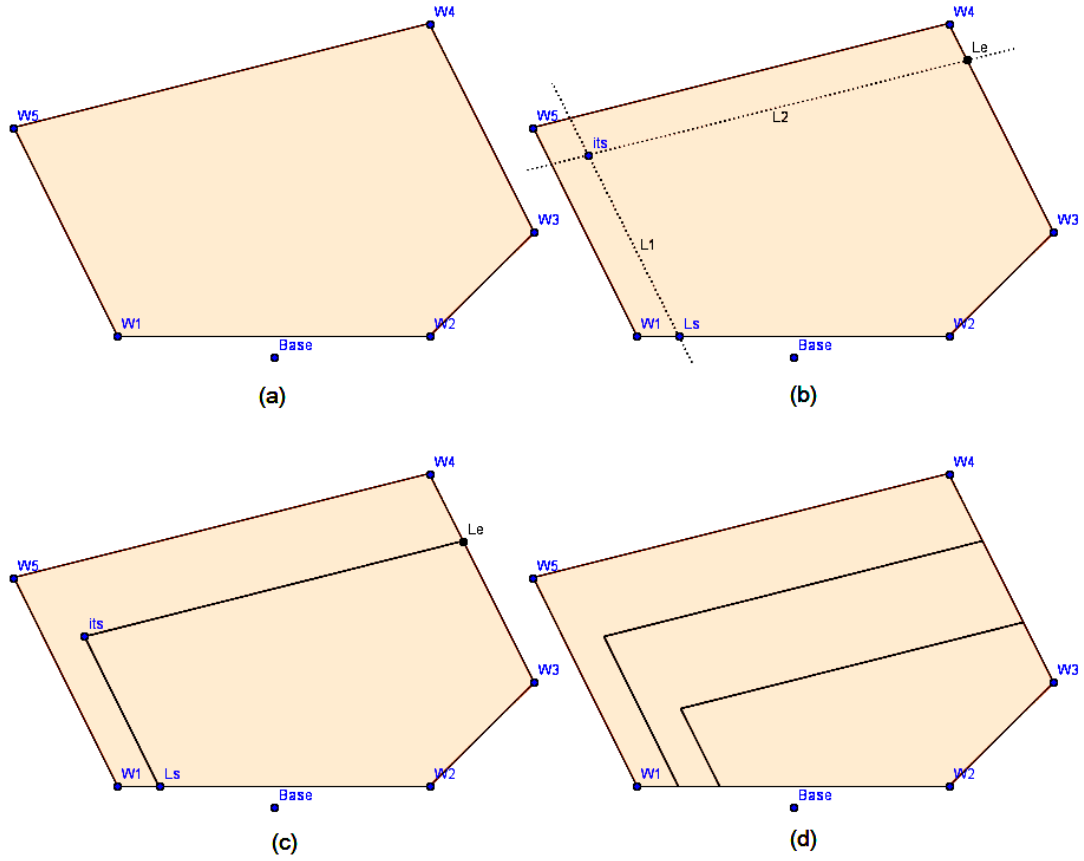

Figure 3.1: Polygon area decomposition using geometry-based algorithm. The figure shows steps for dividing a polygon into three smaller polygons with equal areas.

Algorithm 2 Divide a convex polygon into two smaller polygons

Input: convex polygon P; the list of vertices $W(P) = W_i$, $i = 1, ..., m$ in CCW order, given line segment $l(W_1, W_2)$ is closest to the base station;
Output: two polygons P_1 and P_2

1. Find the longest side $Long = l(W_i, W_{i+1})$ of convex polygon P
2. **if** $i = 2$ **then**
   $L_s \leftarrow W_2$ and $L_e \leftarrow W_3$
   Move L_s CW along P and find new L_e such that $(L_s, L_e) \parallel (W_2, W_3)$ until $Area(P_1) = AreaRequired(P_1)$
   else if $i = m$ **then**
   $L_s \leftarrow W_1$ and $L_e \leftarrow W_m$
   Move L_s CCW along P and find new L_e such that $(L_s, L_e) \parallel (W_1, W_m)$ until $Area(P_1) = AreaRequired(P_1)$
   else
   Initialize line segment L_2; $L_2 \parallel Long$
   if $i \leq (\frac{m}{2} + 1)$ **then**
   Initialize line segment L_1; $L_1 \parallel (W_2, W_3)$. The distance between L_1 and (W_2, W_3) is 2 $\times$ sensing width.
   else
   Initialize line segment L_1; $L_1 \parallel (W_1, W_m)$. The distance between L_1 and (W_1, W_m) is 2 $\times$ sensing width.
   end if
   if $Area(P_1) < AreaRequired(P_1)$ **then**
   Move L_2 away from $Long$ until $Area(P_1) = AreaRequired(P_1)$
   else
   Move L_2 toward $Long$ until $Area(P_1) = AreaRequired(P_1)$
   end if
   end if
3. $P_2 = P - P_1$

3.2 Algorithm Analysis

This section analyzes the geometry-based area decomposition algorithm whether it decreases the number of turns in area coverage for any shape of convex polygons. From the algorithm, there are two cases to consider. One is when the longest side of the region of interest is next to the side of a base station, and another is when it is not, as shown in figure 3.2.

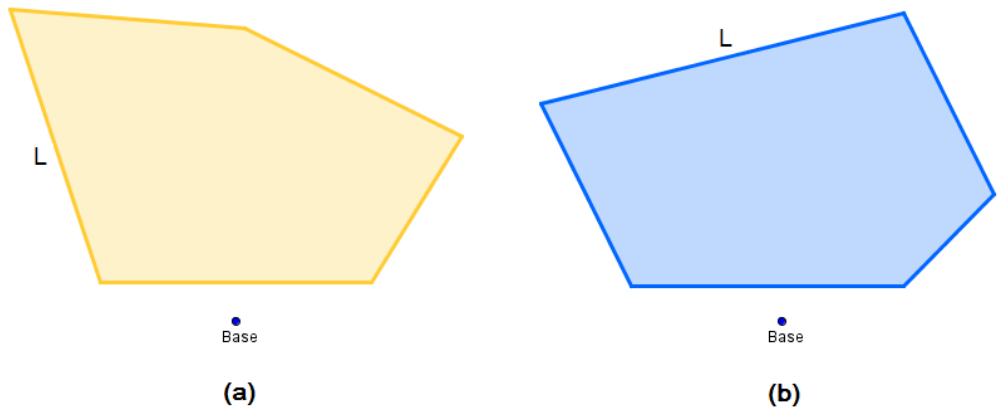

Figure 3.2: Regions of interest with different side of the longest side L.

Since an agent sweeps back and forth to perform area coverage, the number of turns relates to the number of rows it has to cover. Figure 3.3 shows partitioned regions from the polygons in figure 3.2. Figure (a) shows the first case of the decomposition. The number of turns is determined from

$$\text{number of turns} = 2\left(\frac{d_1}{sensing\,width} - 1\right) \tag{3.1}$$

where d_1 is a minimum diameter function of the partitioned region and

$$\text{number of rows} = \frac{d_1}{sensing\,width}. \tag{3.2}$$

Figure (b) shows the decomposition where the subregion gives a better coverage path when assigning it with different sweep directions. The diameter function considered in this case is from the region connected to the longest side, since the region varies

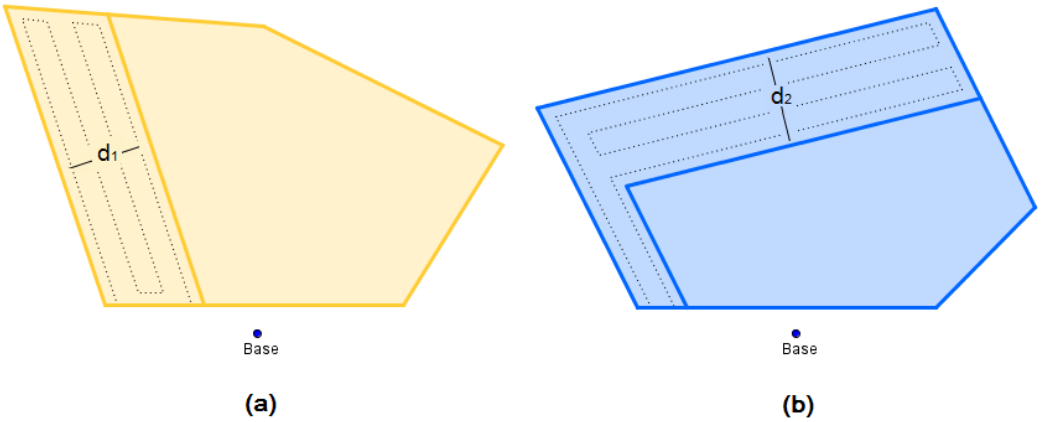

Figure 3.3: The regions with different longest side are partitioned differently .

according to the area required. The number of turns in this subregion is determined from

$$\begin{aligned} \text{number of turns} &= 2\left(\frac{d_2}{sensing\,width} - 1\right) + 2 \\ &= 2\left(\frac{d_2}{sensing\,width}\right) \end{aligned} \tag{3.3}$$

where d_2 is a minimum diameter function of the region connected to the longest side.

However, the number of turns in this case valid only on the condition that the number of rows or $\frac{d_2}{sensing\,width}$ is an even number. Otherwise, it results in two additional number of turns and route redundancy.

The diameter function is determined from [17] and sensing width is determined using equation 2.1. If the UAVs are homogeneous, the sensing widths are the same. The number of turns varies upon the size of the diameter function.

Figure 3.4 presents diameter functions of partitioned regions with equal area. The subregion in figure (a) is from an algorithm presented in section 2.1.2. The subregion in figure (b) is from the proposed algorithm. The number of turns in both figure (a) and (b) are calculated using equation 3.1.

From the figure, the diameter function of a subregion in figure (b) is shorter than in figure (a). This is due to the constraint in the proposed decomposition algorithm that

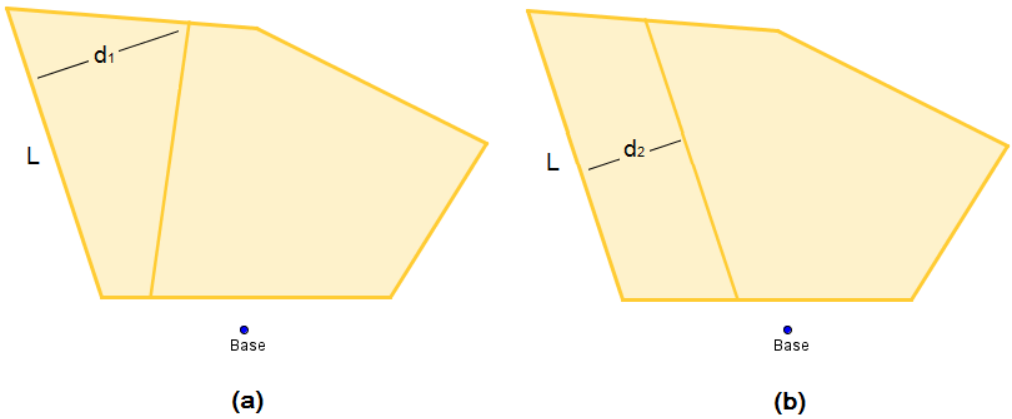

Figure 3.4: Diameter functions of subregions with equal area but are partitioned using different algorithm.

a partitioning line segment must be parallel to a longest side L. Accordingly, in any partitioned regions

$$d_2 \leq d_1 \tag{3.4}$$

where d_2 is the diameter function of a subregion from the proposed algorithm and d_1 is the diameter function of a subregion from section 2.1.2 algorithm.

The number of turns is an integer. From the numerical rounding, the number of turns of a subregion from the proposed algorithm is less than from algorithm presented in section 2.1.2 if

$$d_1 - d_2 > \frac{sensing\,width}{2}. \tag{3.5}$$

If the longest side of a polygon is not adjacent to the side of a base station, the number of turns of a subregion from the proposed algorithm is calculated using equation 3.3 and the subregion from section 2.1.2 is calculated using equation 3.1. Figure 3.5 shows the partitioned regions from both algorithms and their diameter functions. The partitioned regions have the same area, but with different diameter function. According to equation 3.1 and 3.3, the number of turns is improved if

$$d_1 - d_2 > 2 \times sensing\,width. \tag{3.6}$$

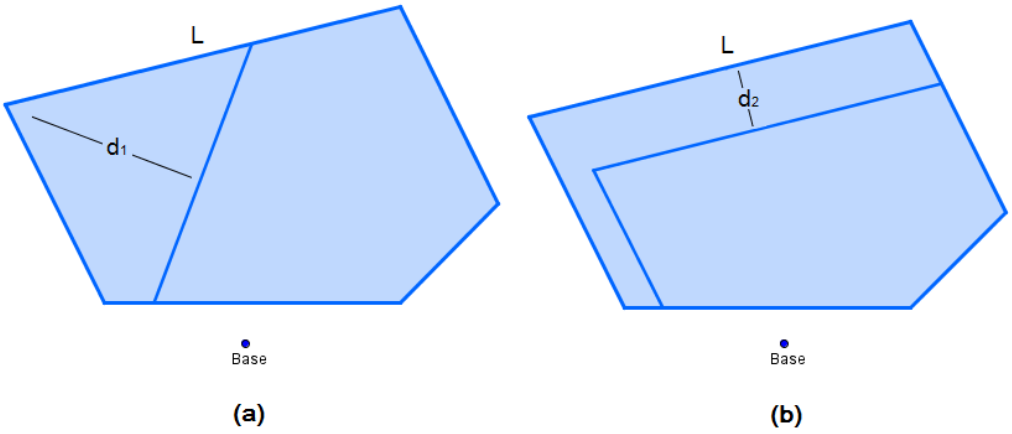

Figure 3.5: Diameter functions of subregions with equal area but are partitioned using different algorithm.

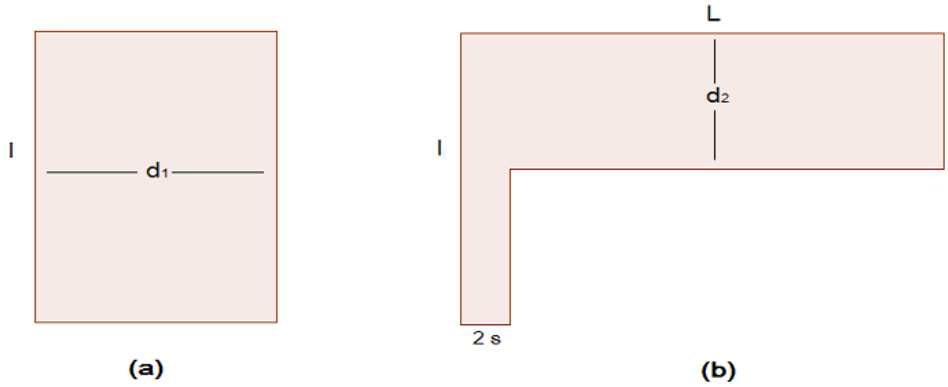

Figure 3.6: Diameter function of a partitioned region from 2.1.2 is the shortest in figure (a). Figure (b) is the partitioned region from proposed algorithm with equal area to (a).

Consider partitioned regions in figure 3.6 where both the regions have the same area. In figure (a), the diameter function d_1 is the shortest when using area decomposition presented in section 2.1.2. Figure (b) is the partitioned region from the proposed algorithm with the same area as in (a).

$$\begin{aligned} \text{Area} = d_1 l &= d_2 L + (l - d_2)(2s) \\ d_1 &= \frac{d_2 L + 2ls + 2d_2 s}{l} \end{aligned} \tag{3.7}$$

where s is the sensing width.

Substitute eq. 3.7 in eq. 3.6 gives

$$L - l > 2 \times sensing\,width. \tag{3.8}$$

Hence, in the second case, the number of turns is improved if the longest side is longer than the side adjacent to the base station twice the sensor width. If not, the region should be decomposed as if the longest side is l and follow the first case.

3.3 Area Coverage Redundancy

Area coverage redundancy is overlapped areas when the agents perform area coverage task, or covered areas that are not in a region of interest. The redundancy does not give productive surveillance. It wastes both time and energy consumption.

In individual areas coverage algorithm presented in section 2.2, the algorithm applies an optimal line-sweep-based decomposition [17] which gives area coverage in each sub-region with minimum number of turns. However, the algorithm does not concern over area coverage redundancy. The redundancy from overlapped areas usually occurs when an agent are obliged to start and end at the same base station. When an agent completed its area coverage task, it returns to the base station. Its return route repeats the already visited region and causes overlapped area if the final area coverage position is not at the base station.

The overlapped area is determined from the distance between the agent's final area coverage position and the base station. This final area coverage position depends on the start position, the size of the diameter function, and route generation of the UAV as shown in figure 3.7, 3.8, and 3.9 respectively.

In figure 3.7, the final area coverage positions are different as the agent starts its route at different location. There are route overlaps in both figure (a) and (b). The overlap in figure (b) is less than in (a) since the final position in (b) is closer to the base station than in (a).

The size of a subregion's minimum diameter function should also be concerned when decomposing a region of interest. It defines the number of rows an agent has to cover when performing area coverage. The number of rows is determined from equation 3.2. It affects the agent's final area coverage position and hence route redundancy. When decomposing the region, the size of minimum diameter function should be

$$2n \times sensing\,width \tag{3.9}$$

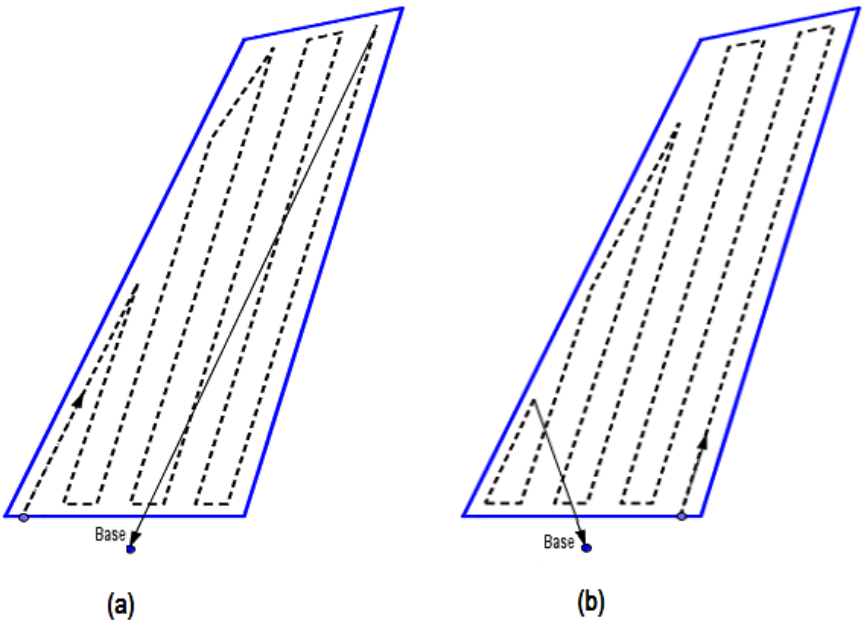

Figure 3.7: Route comparison of the same polygon but with different start position.

where $n = 1, 2, ..., n$. The sensing width defines the distance between back and forth motion which is determined from equation 2.1. The diameter function of a subregion according to equation 3.9 allows the agents to start and end the area coverage task at the same base station and eliminates the area coverage redundancy.

Figure 3.8 shows an example of a polygon with different minimum diameter function of partitioned regions. In figure (a), the diameter function is equal to five sensing width. The final area coverage position of a UAV is not on the side closest to a base station. Hence, the return route will overlap an already covered area. In figure (b), the subregion's minimum diameter function is according to equation 3.9. The area coverage route sweeps the subregion completely without route overlap. The entire route results in productive area coverage.

When decomposing the region considering this criterion, the area of the partitioned region might contradict the *AreaRequired*. The area of the partitioned region can be less than the *AreaRequired*. However, it should not be more than the *AreaRequired* since it might exceed the agent's capability. This criterion also affects the area of other partitioned regions, hence it should be considered concurrently when performing area

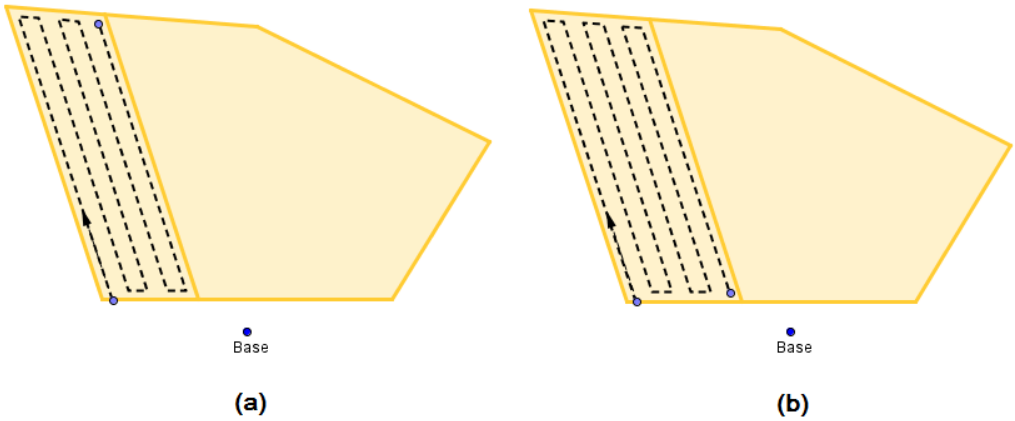

Figure 3.8: Different width of partitioned regions result in different overlapped area.

decomposition algorithms.

When decomposing a region, there is one last subregion that may not fit into the presented scheme, i.e. no side parallel to the partitioning line segment. The route for this subregion is determined from the direction of minimum diameter function and the position of the base station. To remove route overlap, both start and end positions of the route should be close to the base station or the return route should be part of the area coverage path.

Figure 3.9 shows an example of possible route generation. The routes in both figure (a) and (b) are generated toward the direction of minimum diameter function for minimum number of turns. However, a route overlap is also concerned in figure (b). The route in figure (b) allows an agent to start and end at the base station without overlap.

3.4 Discussion and Results

There are various factors that influence the performance of non-communicative area coverage. The ones already been examined in this thesis are sensor range, area decomposition, and individual areas coverage algorithm. The efficiency of the UAVs is depending on the design or purpose of each UAV which can be various in speeds, stability, battery lifetime, and etc. Unpredictable factors such as environment of a region

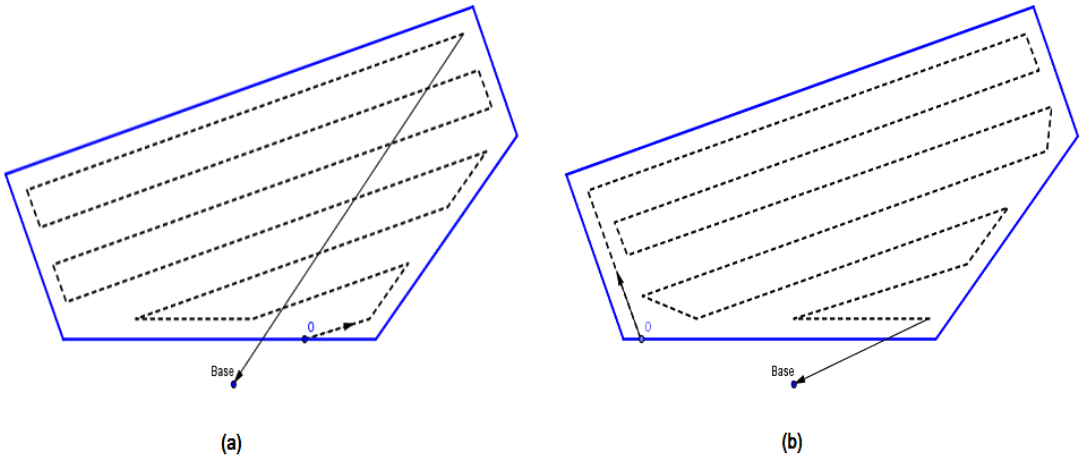

Figure 3.9: Route comparison of the same polygon but with different algorithm.

of interest is depending on a moment when a survey takes place. The unpredictable parameters that affect the environment are such as brightness, wind and weather condition.

The time spending on area coverage can be minimized by removing an unnecessary path and decreasing the number of turns. In section 3.3, the unnecessary path is taken away by rearranging the area coverage route such that the UAV does not repeat any of its path or other UAVs path when performing area coverage mission.

Figure 3.10 compares a polygon area decomposition and individual areas coverage algorithms between (a) algorithms demonstrated by Maza and Ollero [22] as presented in section 2 and (b) the proposed algorithms. Figure (a) illustrates that the polygon area decomposition and individual areas coverage algorithms are considered separately. First, the polygon is divided according to the required area of each agent. Then, the sweep direction in each subregion is determined for area coverage route with the minimum number of turns. In figure (b), the geometry of a polygon and the position of a base station are considered before the area decomposition. The longest side of the polygon is determined, then the polygon is partitioned. This results in the area coverage route without any overlapped areas, shorter traveling distance, and fewer number of turns compared to the route from figure (a).

The geometry-based area decomposition algorithm improves the area coverage perfor-

mance most when a polygon P has one side apparently longer than the others. With this polygon, the UAVs can sweep along the longest side having longer distance on straight paths and less number of turns.

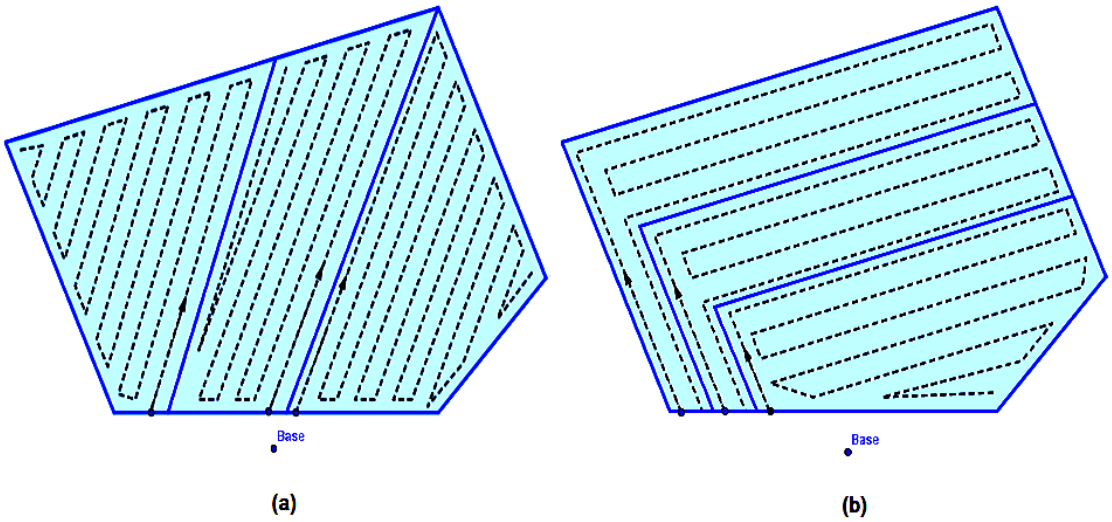

Figure 3.10: Area decomposition and route generation of a polygon using different algorithms.

Chapter 4

Communicative Area Coverage

Communications plays a much more important part in the overall operation of a UAV than it does for manned aircraft because the men-in-the-loop are on the ground [7]. Communications gives more robust area coverage algorithm. It allows a base station to know the location and situation of each agent. The base station is able to obtain images from the observed area in near real time using wireless communication system. This is helpful for the base station, especially if the data is critical and time-sensitive. Moreover, it can command the agents to go to specific locations or return to the base station.

Given that the region of interest is in a disaster area or there exists no networking infrastructure, mobile ad hoc networks (MANETs) offer a way to maintain connectivity between the agents. The MANETs communications are performed over radio and do not require any infrastructure. They are very flexible and suitable for several types of applications as they allow the establishment of temporary communication without any pre-installed infrastructure [12].

With MANETs, the UAVs nodes can dynamically located into arbitrary and temporary network topologies. The networks provide the most mobile and least purpose-specific type of wireless ad hoc networks [26]. They are meant to provide an immediate connectivity in any situation. However, as it uses wireless communication as a medium, characteristics such as high bit error rate, path loss or signal strength attenuation, multipath fading, interference, delay spread, and penetration loss should be considered [23], [24].

The purpose of this chapter is to place a certain number of nodes such that a region of interest is completely covered and all the nodes and a base station can communicate. The node placement can be classified into static positioning and dynamic positioning scheme. In static positioning, the placement of the nodes is determined by the base station before a task begins. These UAV nodes will go to the predefined locations when

the task begins. The placement calculation is based on the region of interest and sensing range of the nodes.

In dynamic positioning, the placement is determined by each node during the task. The placement calculation for each node is based on current locations of the node and its neighbors, communication range, sensing range and the region of interest. In this dynamic scheme, the nodes find their new locations autonomously and they can reposition themselves when there is one or more missing nodes in the region.

The UAV nodes are equipped with a low-power Global Positioning System (GPS) receiver to obtain their positions. The UAVs use GPS to navigate them to the predefined positions in static positioning, and to determine their positions and the positions of their neighbors in autonomously dynamic positioning.

In this thesis, all the nodes are assumed to have the same sensing range and maximum transmission range. The transmission range is at least twice the size of the sensing range. Each of the nodes has distinct identity. Uniform node distribution can be applied to spread the UAVs over a region of interest evenly and achieve maximum coverage. The complete coverage of the nodes allows the base station to obtain the whole image of the region of interest and lets all the agents to stay connected.

Section 4.1 presents static positioning of the nodes and minimum number of nodes required to completely cover the region of interest. Section 4.2 presents three methods for dynamic repositioning; DSSA, VDDA, and the centroid-based scheme. Simulation results and analysis when applying the algorithms to a given situation is presented in section 4.3.

4.1 Static Positioning of Nodes

When the environment is sufficiently known, the positions of the nodes can be predetermined. Different clustering shapes could be used to subdivide a region of interest and determine locations of the nodes. The region is subdivided using identical clustering cells and with no overlaps between the cells. The popular clustering shapes for the nodes deployment are circle, square, and hexagon. The circle shape is the "most natural" cluster because it coincides with the shape of the radio transmission range [27]. However, these clusters have to be overlapped to completely cover the region of interest. There is no hint on the circles how much they should overlap. Hence, it is more complicated than the other clustering shapes in determining uniform locations of the nodes that can completely cover the region of interest.

An ideal shape for clustering is the one that tessellating cells are adjacent to each other without any holes and overlapping areas. The shapes that have this property are triangle, square, and hexagon. Among these shapes, hexagon is the best clustering shape because it is the largest polygon (in terms of number of sides) that has this property [27] and the distances between the node and all the adjacent neighbors are equal.

Figure 4.1 illustrates different clustering shapes for node deployment. In figure (a), the circle cells could not cover the entire network without overlap. Figure (b) shows square clusters where there is no uncovered area, but the distance between two adjoining nodes and diagonal nodes are not equal. Figure (c) demonstrates hexagonal clustering nodes where the distances between the node and its neighbors are the same in all direction.

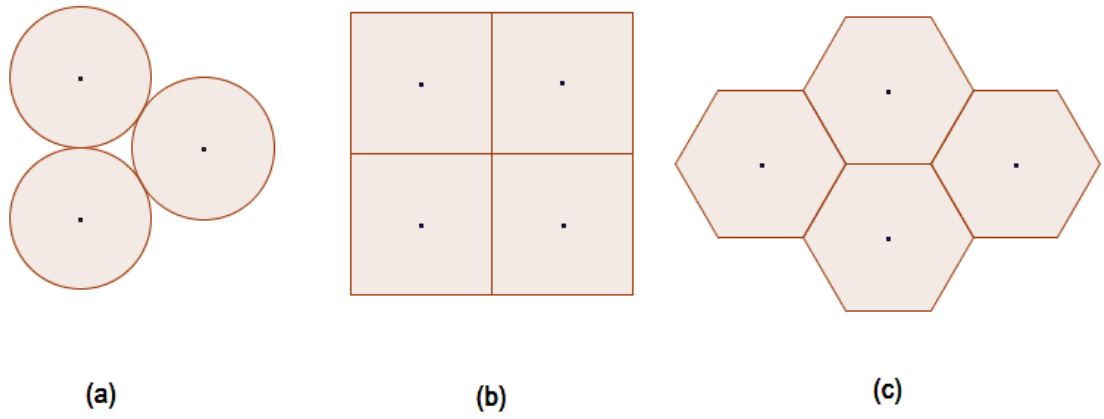

Figure 4.1: The clustering shapes of circle (a), square (b), and hexagon (c).

4.1.1 Minimum number of nodes

The number of agents required to completely cover the region of interest depends on the sensing range and the shape of the region. The distance between the agent and its adjacent neighbors or between the agent and its adjacent boundary must not exceed the sensing range so that the entire region is covered. There is no general solution to find the minimum number of agents required to completely cover the region. The regions with equal area might require different number of agents, if the shapes of the two regions

were different.

To uniformly place the nodes in the region, regular hexagonal tiling could be applied. The nodes are placed at the center of the hexagons. The size of the hexagons corresponds to the node's maximum sensing range to reduce overlapped sensing regions. The hexagons are considered in node positioning if their centers lie inside the region of interest. These hexagons give approximate number and positions of node positioning.

Figure 4.2 demonstrates the uniform distribution of the nodes that lie inside the region of interest. Figure (a) shows the hexagonal tiling and the positions of the nodes. Figure (b) shows the sensing regions according to the node positions in figure (a). The number of agents in this figure is 17, however, the region is not entirely covered.

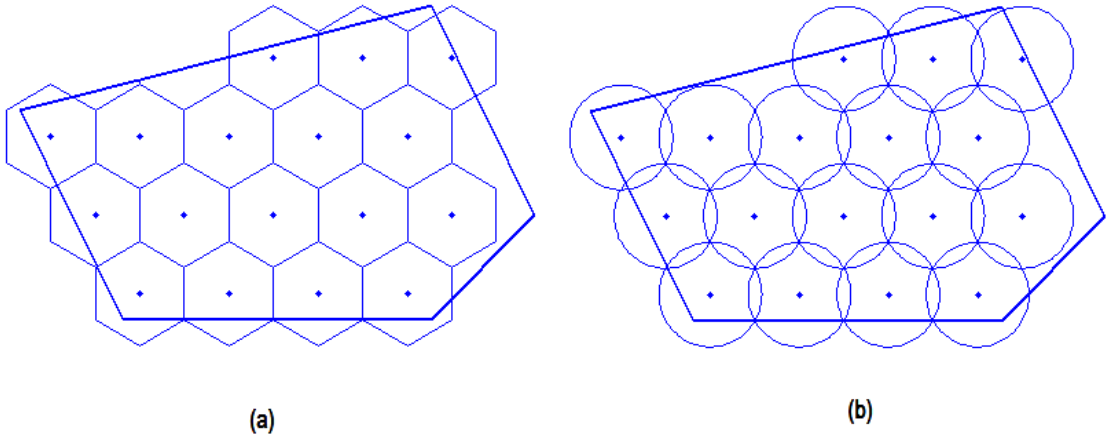

Figure 4.2: Regular hexagonal tiling inside the region of interest (a) and the nodes sensing regions (b).

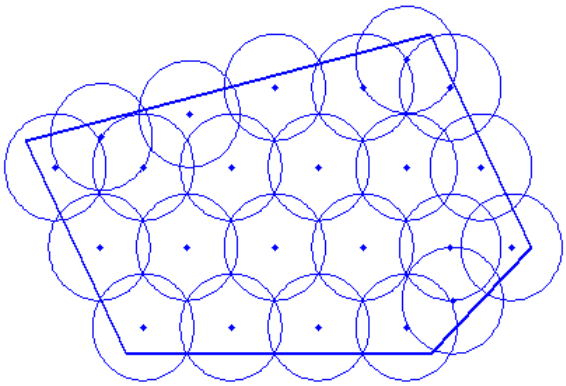

Figure 4.3: Node locations and their sensing regions.

Additional nodes are required to cover the whole region as shown in figure 4.3. In the figure, the number of agents in the region is 23. The number of the nodes is not minimized since additional nodes give large redundant covered regions. The nodes could be relocated to reduce the number needed.

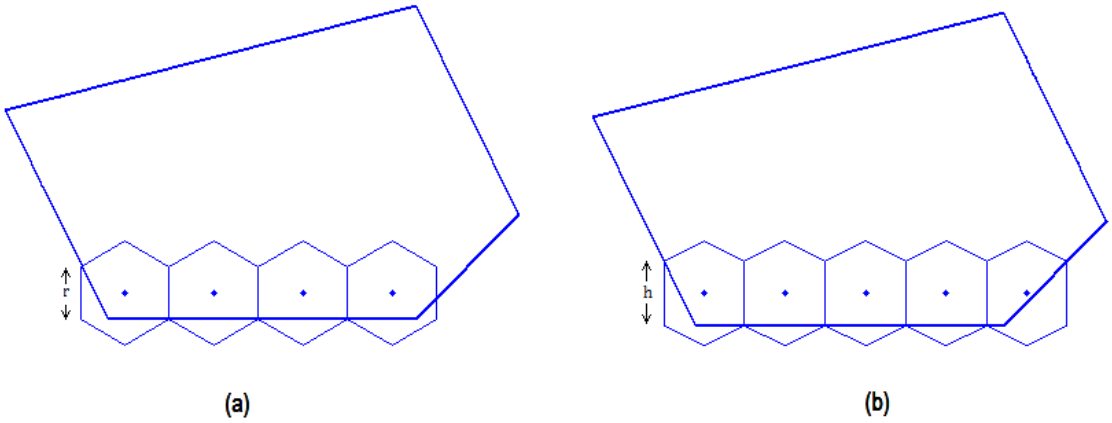

Figure 4.4: Regular hexagonal tiling (a) and hexagonal tiling adjusted according to the width of the region (b).

The nodes relocation can be done by changing the dimension of the hexagonal tile. Figure 4.4 illustrates the first layer of the hexagonal tiling. In figure (a), the nodes are located according to regular hexagonal tiling where all the sides are equal. There is

uncovered region in this first layer figure since the width of the hexagonal tiles and the region are not proportional. In figure (b), the width of the tiles is adjusted according to the size of the region. The width is smaller but the vertical sides are longer. The nodes cover the whole layer in this figure.

To adjust the width of the hexagon, the number of the tiles required in the first layer is determined from

$$n = \lceil \frac{w_r}{r\sqrt{3}} \rceil \tag{4.1}$$

where r is the maximum sensing range and w_r is the width of a region when $y = r$. Then, as shown in figure 4.5, find $y = h$ that makes

$$\frac{h}{2} = \sqrt{r^2 - \frac{d}{2}^2} \tag{4.2}$$

where d is the width of each hexagonal tile and $d < r\sqrt{3}$. d is determined from the ratio of the width of the region w_h at $y = h$ and the number of required tiles n:

$$d = \frac{w_h}{n}. \tag{4.3}$$

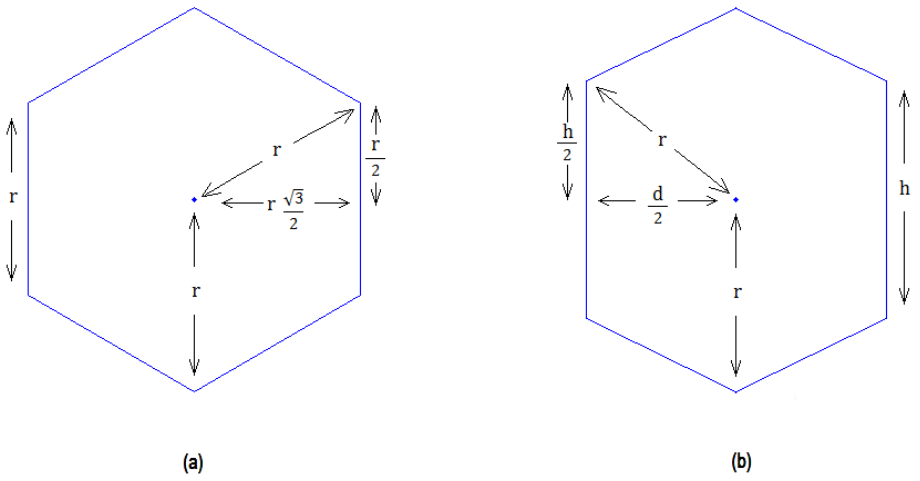

Figure 4.5: Hexagon and the length of each side. Figure (a) is a regular hexagon. Figure (b) is the hexagon adjusted according to the width of the region.

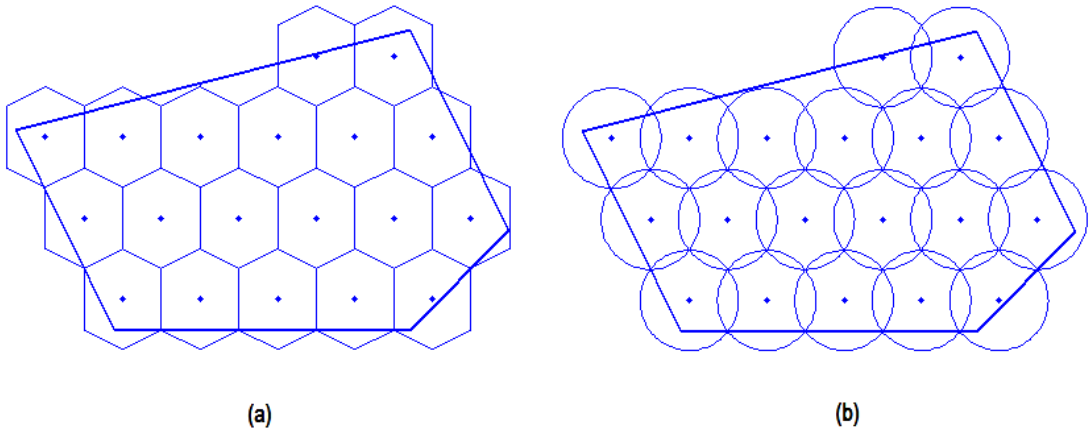

Figure 4.6: Adjusted hexagonal tiling in the region (a) and sensing regions from the tiling (b).

The adjusted hexagonal tiling, the nodes location and their sensing region are shown in figure 4.6. The distance between the nodes in the same layer are closer, but the distance of the nodes in different layers are farther when comparing to regular hexagonal tiling in figure 4.2. With this adjusted tiling, the number of nodes that covers the entire region of interest is 21 as illustrated in figure 4.7.

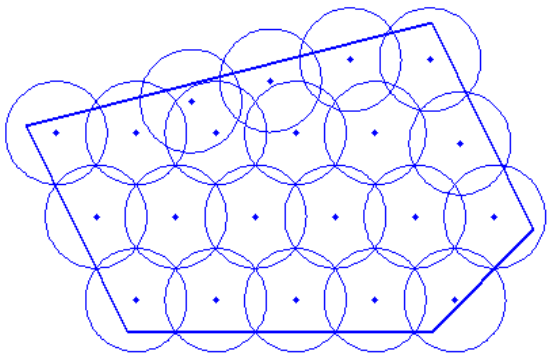

Figure 4.7: Node placement using adjusted hexagonal tiling

The width of the hexagonal tile is adjusted according to the first layer width of the region. When consider the first layer at different side of the region, the width of the tiles change. For example, the width of the tile when consider the top left side as the first layer is different from when consider the bottom right side of the region as the first layer, as shown in figure 4.8. Therefore, all the sides of the region should be determined in order to find the minimum number of the nodes.

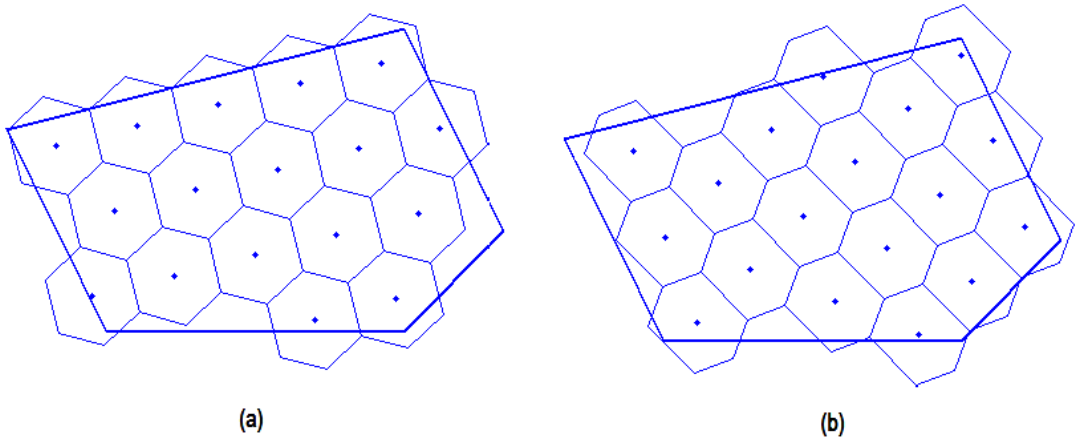

Figure 4.8: Hexagonal tiling when considers different sides of a region as the first layer.

4.2 Dynamic Repositioning of Nodes

In static positioning, the nodes go to the predefined locations without the calculations from the nodes. It gives fast deployment, but the network topology is non-adaptable. When there is a node failure, the region under this failed node is not observed and the base station could not obtain images of the entire region. Dynamic repositioning relocates the working nodes to completely cover the region.

In dynamic positioning, the self-deployment algorithm is considered. The nodes do not rely on the base station for their relocation, but they determine the new locations themselves. The nodes should reach their new locations with minimum time and energy consumption.

The situation for dynamic repositioning considered in this thesis is when the nodes are already in the positions that cover the entire region of interest, but there is a lost or broken node that makes incomplete area coverage. The purpose of this section is to find the reposition algorithm that makes the nodes once again completely cover the region of

interest. The nodes should be able to reposition themselves when they sense that their neighboring node is lost or broken down. Repositioning compensates the lost in coverage and/or network connectivity of a missing node. It is necessary in communicative surveillance task, especially in a situation where the number of mobile agent changes with time; e.g. due to forest fire, hostile attacks, or malfunctions.

There are many algorithms for the nodes dynamic repositioning as presented in [28]. The nodes are repositioned based on the quality of their performance metrics such as the data rate, sensing range, path length in terms of the number of hops from a sensor node to the base station, etc. This thesis only considers repositioning when there is a region out of the nodes' sensing ranges.

The algorithms presented in this section are DSSA [13], VDDA [14], and the centroid-based scheme [20]. The DSSA algorithm which applies the concept of force in physics for node positioning is presented in section 4.2.1. Section 4.2.2 presents the VDDA algorithm where local Voronoi diagram is used to calculate the effective area of the nodes. Section 4.2.3 presents the centroid-based scheme which used Voronoi diagram and centroid of the polygon for node relocation.

4.2.1 Distributed Self-Spreading Algorithm (DSSA)

This algorithm is introduced by Heo and Varshney [13]. It is the peer-to-peer algorithm inspired by the equilibrium of molecules. The algorithm applies inter-nuclear repulsion and attractions between molecules. If the nodes are located too close to each other, the coverage gaining from these nodes is not high. On the contrary, if the nodes are located too far from each other, the coverage regions may not overlap and the region is not entirely covered. The optimal spacing between the nodes can be determined from a process similar to the equilibrium of molecules.

The algorithm begins with a specified number of nodes scattered randomly in a given region. These nodes have communication range and sensing range to sense an environment within its sensing region. Any pair of nodes within their communication range are considered neighborhood and can communicate with each other. Within neighborhood, locations of the nodes can be obtained and the sensed data can be received and transmitted. **Algorithm 3** [14] shows the pseudocode of the algorithm executed at each node i. The algorithm contains four parts.

Algorithm 3 Distributed Self Spreading Algorithm [14]

1. **Initialization**

   initial node locations p_0;
   sensing range sR;
   communication range cR;
   calculate local density D;
   calculate expected density μ;

   While (Not(Oscillation occured OR In a region of stability))

2. **Partial Force Calculation**

   calculate partial force $f_n^{i,j}(\mu, D, cR, p_n)$;
   update temporary position p_{n+1}^i;

3. **Oscillation Check**

   If $(|p_{n-1}^i - p_{n+1}^i| < threshold_1)$

   Increase oscillation count by 1;
   If (oscillation count < oscillation limit)

   Update next location to the temporary position;
   Update local density D;

   Else

   Move to the centroid of oscillating points;
   Update local density D;

   Else

   Stop node i's movement;

   Update next location to the temporary position;
   Update local density D;

4. **Stability Check**

   If $(|p_{n+1}^i - p_n^i| < threshold_2)$

   Increase stability count by 1;
   If (stability count < stability limit);

   Go to while loop;

   Else

   Stop node i's movement;

   Else

   Go to while loop;

1. *Initialization*: First, each node specifies the given communication range (cR) and sensor range (sR). The initial node locations (p_0) are specified in terms of a vector that contains the longitude component and the latitude component of each node location. Then expected density which is a rough estimation of the desired density is calculated from

$$\mu(cR) = \frac{N \cdot \pi \cdot cR^2}{A} \tag{4.4}$$

where N is the number of nodes, cR is the communication range, and A is the area of the region of interest. The expected density μ is the average number of nodes required to uniformly cover the entire area. The number of nodes within a node's communication range is local density (D). D_0 is an initial local density. These densities will be used when decisions regarding positions of nodes are made.

2. *Partial Force Calculation*: The algorithm applies the concept of force to define the movement of nodes during node placement method. The concept depends not only on the distance between a node and its neighboring nodes within its communication range, but also on the current local density. The force resulting from high local density is greater than the force resulting from low local density. Also, the force from a closer neighbor is greater than the force from a farther neighbor. The force function is defined to satisfy the following conditions:

   (a) Inverse relation: $f(d_1) \geq f(d_2)$, when $d_1 \leq d_2$, where d_1 and d_2 are node separations from the origin.

   (b) Upper bound: $f(0^+) = f_{max}$.

   (c) Lower bound: $f(d) = 0$, where $d > cR$, d is the node separation and cR is the communication range of each node.

   Condition (a) is the same as in physics that follow Coulomb's law. Condition (b) and (c) are included to incorporate the notion of locality. They are a limiting function applied to a node's neighborhood.

   The partial force on node i from its neighboring node j at time step n is calculated to be a repulsive force as

$$f_n^{i,j} = \frac{D_n^i}{\mu^2}(cR - |p_n^i - p_n^j|)\frac{p_n^j - p_n^i}{|p_n^i - p_n^j|} \tag{4.5}$$

   where

   cR stands for communication range;

p_n^i stands for the location of i^{th} node at time step n;
D_n^i stands for the local density of i^{th} node at time step n.

From equation 4.5, the distance between a node and its neighboring nodes affects the partial force inversely. Closely located nodes impose larger partial forces while sparsely located nodes induce smaller partial forces. The magnitude of the partial force exerted by a pair of nodes on each other are the same but with the directions opposite to each other. The ratio of the local density (D) and the square of the expected density (μ) at each node define the density factor ($\frac{D}{\mu^2}$). This density factor is small in sparse regions and is large in dense regions.

Each node can decide its next movement, after calculating the partial forces for all nodes at time step n. The node's movement is decided from the collection of forces due to the nodes in its neighborhood. After the node moves to new position at time step $n+1$, it updates the local information at this position. The local information is collected from the nodes that are within the communication range and will be used for the calculation of the local density at each node. Again, the partial force is calculated and the next movement is generated.

3. *Oscillation Check*: The nodes will stop their movements from two criteria. The first criterion is when a node moves back and forth between almost the same locations many times. This node is regarded as in the oscillation state. To determine whether the oscillation is going on or not, the history of a node's movement is examined. If the number of node's oscillations (O_{count}) is over the oscillation limit (O_{lim}), the node's movement is stopped at the center of gravity of the oscillating points.

4. *Stability Check*: The second criterion for stopping a node's movement is when the node moves less than $threshold_2$ for the time duration $Stability\_limit(S_{lim})$. This is due to three reasons.

   - The node has reached its stable state
   - Fuel exhaustion
   - Broken mobile units

Figure 4.9 illustrates an example of a node movement when applies DSSA algorithm to node A. Nodes B, C, D, E are neighboring nodes. They are located within node A's communication range. The partial forces f_B, f_C, f_D, and f_E of the neighboring nodes are calculated using equation 4.5. The total force F on node A is obtained from the vectors

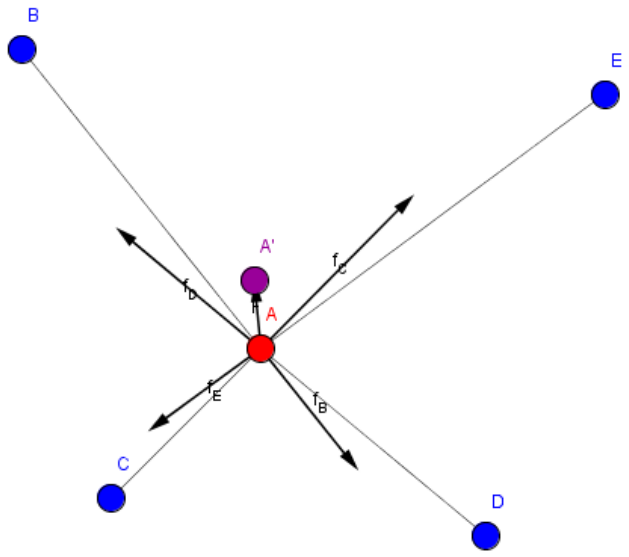

Figure 4.9: Example of DSSA node movement [14].

summation of its neighboring nodes' partial forces, i.e., F = $f_B + f_C + f_D + f_E$. The resulting total force defines a new position of node A; Á. From the figure, the partial force corresponding to node C is the largest since the location of node C is the closest to A comparing to other neighboring nodes.

Figure 4.10 illustrates a case where a neighboring node loses its communication functionality. In this figure, node C is assumed to be broken during the deployment period. Consequently, it is not considered in the calculations of its neighboring nodes for their next movements. The total force now excludes the largest partial force f_C from the example in figure 4.9, i.e., F = $f_B + f_D + f_E$. The next position for node A is Á which compensates the missing coverage from the loss of node C.

Since the sensor nodes can be adversely affected when deployed in a remote and hostile region, a robust method for node deployment is needed. This DSSA algorithm exhibits robustness over lack of mobility and lack of communication in some nodes. When the node loses its mobility, that node is considered to be an early stopped node and becomes a static node in the sensor network. Neighboring nodes that does not lose their mobility may still change their locations to improve the irregular topology. Nevertheless, if a node loses its communication capability, that node becomes of no use in the sensor net-

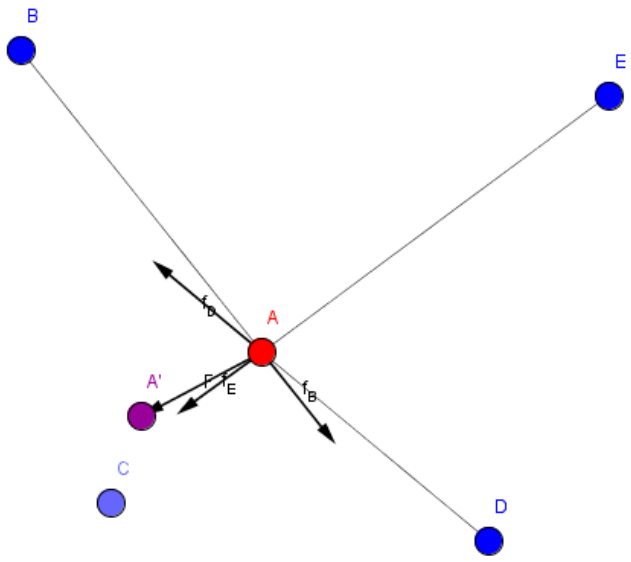

Figure 4.10: Example of DSSA node movement with a failed node [14].

work. Its neighboring nodes have to rearrange the locations to cover the region instead of the broken node.

4.2.2 VD-Based Deployment Algorithm (VDDA)

Voronoi diagram is one of the most fundamental data structures in computational geometry. Many researchers have demonstrated the importance and usefulness of the Voronoi diagram in a wide variety of fields including mathematics, computational geometry, biology, chemistry, geography, communications, and coding theory [3].

Figure 4.11 shows an example of a Voronoi diagram. Given a number of points in the region of interest, Voronoi diagram can be used to decompose these points into non-overlapping Voronoi cells (Voronoi region). The cells are divided according to the nearest-neighbor rule. Each given point in the Voronoi cell is associated with the cells of points in the region of interest that are closest to it. These given points and their corresponding Voronoi cells can be applied to the node placement problem by defining

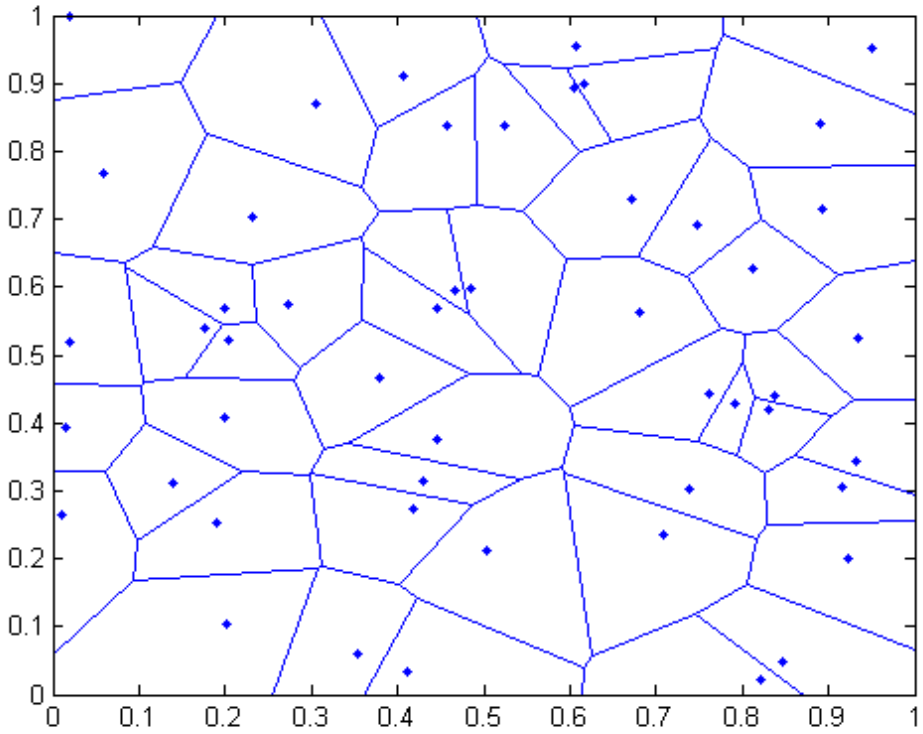

Figure 4.11: Voronoi diagram

the points as the nodes and the Voronoi cells as the communication ranges. Hence, a deployment algorithm for communication can be developed using the notion of Voronoi regions.

This VD-based deployment algorithm (VDDA) is from [14]. The goal of this algorithm is to have the Voronoi cell corresponding to the node's sensing range. The Voronoi cell that matches with a node's sensing range at a certain time instance is considered to be the desired solution in terms of coverage. If the difference between the node's sensing range and the Voronoi cell corresponding to that node are not within the predefined tolerance, the node is moved. It is moved until the sensing range and the Voronoi cell are aligned, then the nodal movement is stopped and the resulting solution is accepted. The algorithm from [14] for VD-based deployment is presented in **Algorithm 4**. This distributed algorithm is executed at each node i. The algorithm comprises of two steps: initialization and finding the best energy-utilization point.

Algorithm 4 Voronoi Diagram based Deployment Algorithm [14]

1. **Initialization**

   initial node locations p_0;
   sensing range sR;
   communication range cR;
   initial energy E_0^i;
   energy consumption rate M_0^i, C_0^i, S_0^i;
   calculate initial $VD(p_0, cR)$;
   calculate initial effective area A_0^i from $VD(p_0, cR)$ and $SensingArea(p_0^i, sR)$;
   calculate initial estimated lifetime T_0^i;
   calculate initial node utility $U_0^i = A_0^i \times T_0^i$;
   While (Not(utility gain > a predefined threshold))

2. **Find the maximal energy utilization point**

   Set the current solution as the last solution of previous step: $U_{max,n}^i = U_{n-1}^i$
   calculate $VD(p_n, cR)$
   find the centroid of VD: VD_1
   { search max. energy utilization point among points linearly spaced between p_i and VD_1
   calculate effective area $A_{t,n}^i$ from $VD(p_n, cR)$ and $SensingArea(p_{t,n}^i, sR)$
   calculate energy consumption rate $M_{t,n}^i$, $C_{t,n}^i$, $S_{t,n}^i$
   calculate estimated lifetime $T_{t,n}^i$
   if $U_{t,n}^i > U_{max,n}^i$
   $U_{max,n}^i = U_{t,n}^i$
   else
   keep searching
   end
   update p_i **}**
   find the mean of maxVD and min VD: VD_2
   { search max. energy utilization point among points linearly spaced between p_i and VD_2
   calculate effective area $A_{t,n}^i$ from $VD(p_n, cR)$ and $SensingArea(p_{t,n}^i, sR)$
   calculate energy consumption rate $M_{t,n}^i$, $C_{t,n}^i$, $S_{t,n}^i$
   calculate estimated lifetime $T_{t,n}^i$
   if $U_{t,n}^i > U_{max,n}^i$
   $U_{max,n}^i = U_{t,n}^i$
   else
   keep searching
   end **}**
   update p_i
   update $U_n^i = U_{max,n}^i$
   update current energy E_n^i
   End of while loop

1. *Initialization*: The algorithm starts with specifying the communication range (cR), the sensor range (sR), and the initial node locations (p_0). Initial energy for the i^{th} node (E_0^i) and energy consumption for different activities are also specified. The node consumes energy for activities namely node movement, node communication, and sensing. Energy consumption for movement of the i^{th} node (M^i) is the cost for movement per unit distance. Accordingly, the total cost for movement of the i^{th} node is equal to the product of (M^i) and the distance. Energy consumption for communication by the i^{th} node per unit time (C^i) is a function of the largest distance between itself and its neighbors. Energy consumption for sensing and computation by the i^{th} node per unit time (S^i) is assumed to have a fixed cost. In addition, all the nodes are assumed to have the same amount of energy (E_0) at the beginning.

   Next, the initial local VD (VD_0) is calculated using p_0 and cR. The local VD is used to determine the effective area of each node. The initial effective area (A_0) is obtained from the intersection of VD and sensing area of the i^{th} node, where VD is the Voronoi region of the i^{th} node and sensing area is the circular area of i^{th} node centered at p_0 with the radius sR.

   The effective area of the i^{th} node is the covered area based on the nearest node concept, since each node has only local information from its neighboring nodes. Accordingly, the node can calculate its VD with the information from the neighbors. If a different communication range is used, the Voronoi region corresponding to the node and the size of its neighborhood are different.

   The estimated lifetime (T_n^i) of i^{th} node at time step n is defined as

$$T_n^i = \frac{E_{n-1}^i - M_n^i \times d_{n-1,n}^i}{C_n^i + S_n^i} \tag{4.6}$$

   where

   E_{n-1}^i stands for the energy remaining at the i^{th} node after time step $n-1$;
   M_n^i stands for the energy consumption per unit distance for movement of the i^{th} node at time step n;
   $d_{n-1,n}^i$ stands for the distance moved by the i^{th} node between time step $n-1$ and time step n;
   C_n^i stands for the energy consumption per unit time due to communication by the i^{th} node;
   S_n^i stands for the energy consumption per unit time for sensing and computation by the i^{th} node.

   To determine the degree of coverage achieved by each node in terms of energy

efficiency, a node utility metric is considered. The node utility metric (U) indicates how well the node is utilized to cover its effective area during the lifetime. Consequently, it is used to decide energy-efficient movements by each node. The node utility metric (U_n^i) of the i^{th} node at time step n is defined as

$$U_n^i = A_n^i \times T_n^i \tag{4.7}$$

where
A_n^i represents the effective area covered by the i^{th} node at time step n;
T_n^i represents the estimated lifetime of the i^{th} node at time step n.

2. *Finding the Best Energy-Utilization Point*: The best energy-utilization point of each node is obtained by comparing utility gains that a node would achieve when moving to different possible node locations. A movement from the current location to different node locations may or may not reduce the communication cost, but it will cost energy consumption depending on the node destination.

   In VDDA, the local VDs are used to reduce the search space from infinite number when considering continuous coordinates. The local Voronoi region can be considered as the estimated or desired coverage by the local node due to the nearest neighbor relation of VDs. Since coverage area of a node is circular, moving the node to the centroid of the Voronoi region would enhance the nodes coverage and/or uniformity. Another node location that can be used to guide the search is the center of the Voronoi range. It is defined as the midpoint of the maximum and minimum along the x and y coordinates in the Voronoi region.

   The local VDs reduce search space to several points linearly spaced. The search space starts from the current location to the centroid of the Voronoi region, and then from the centroid to the center of the Voronoi range. In these searched points, the node utility metric is evaluated and the best action is determined. From the algorithm, the node utility metric is iteratively calculated and compared to the previous value. During the search process, only the best solution is kept. Therefore, a locally optimal solution is obtained after the search process.

   Once the best energy-utilization point is found, actual movement of each node occurs. The movement takes place only after the best solution is found in order to save energy consumption.

4.2.3 The Centroid-based Scheme

This algorithm is from [20]. The algorithm uses the Voronoi diagram to determine the region that the node has to cover. It uses centroid of a polygon to decide the nodes' next movement. The algorithm is similar to VDDA that the nodes have to calculate their Voronoi regions at the beginning of each round of nodes moving. However, it does not search for the maximum gain among the points linearly spaced between the current node location and the centroid of Voronoi region. The node simply moves to the centriod of the region if it gives better local coverage.

The algorithm starts with initializing all the node locations and their sensing range and communication range. It then calculates the Voronoi region of a node using the node location and its communication range. If the Voronoi region is fully covered by the node's sensing range, the node does not move at this round. The Voronoi region is fully covered when all distances from the local Voronoi vertices to the nodes are less than the node's sensing range.

If the node's sensing range does not fully cover the Voronoi region, the centroid of the Voronoi region is determined. The node checks whether the local coverage will increase by its movement to the new location. If the node covers more Voronoi region at the new location, it will move; otherwise, it will stay at its current location. The algorithm is summarized as shown in **Algorithm 5**.

Algorithm 5 The Centroid Scheme

1. **Initialization**

   initial node locations p_0;

   sensing range sR;

   communication range cR;

   calculate initial $VD(p_0, cR)$;

   calculate all the distances between Voronoi vertices and node locations $D(VD, p_0)$;

   While $(D(VD, p_0) > sR)$

2. **Relocation**

   calculate centroid of the Voronoi region $C(VD)$

   if (moving node to $C(VD)$ increases local coverage)

   move the node to $C(VD)$

   update node location p_n

   calculate $VD(p_n, cR)$

   calculate $D(VD, p_n)$

   else

   break

   end

   End of while loop

4.3 Simulation Results and Analysis

This chapter has presented the methods for nodes positioning both statically and dynamically. Given a situation where a region of interest needs to be observed as long as possible, a sufficient number of nodes have to spread over the region of interest. The minimum number of nodes from section 4.1.1 is enough to completely cover the region of interest, however, the coverage is not robust against potential vehicle failures.

An extra number of nodes is added to the region in case of potential node failure. The number of nodes that completely cover the region of interest and robust against the failure is shown in figure 4.12. The figure shows an example of predetermined positions of

the nodes when the sensing range is 80% of its maximum sensing range. In figure (a) the region is fully covered by the nodes. There is one missing node in figure (b).

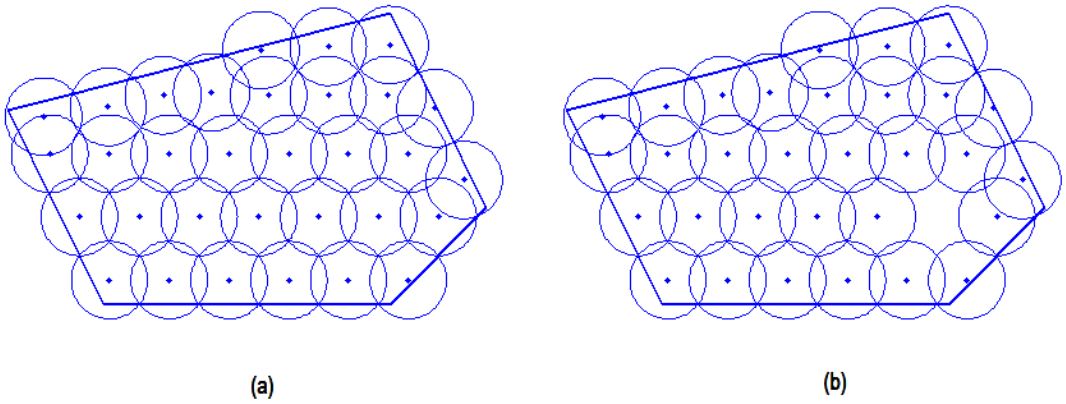

Figure 4.12: Node placement where the sensing range is 80% of its maximum sensing range. All the agents function properly in (a). In (b) there is one missing agent.

When there is one or more missing nodes, dynamic repositioning can be used to relocate the nodes autonomously. In this thesis, the dynamic repositioning algorithms that are applied to the situation are DSSA, VDDA, and the centroid scheme.

When applying DSSA algorithm to the situation, the nodes move according to the positions of their neighbors. The neighbors of the missing node tend to move toward the missing node's direction since there is no force exerting from that direction. Owing to this, the nodes located near the boundary are also likely to move out of the region of interest. This is because the DSSA algorithm only concerns with area of the region, but not the shape or boundary of the region.

Figure 4.13 shows the positions of the nodes and their next movements using DSSA algorithm. Figure (b) illustrates the next movements of the nodes at five time steps which indicated as pink circles. The original node locations are indicated as blue dots as shown in figure (a). The direction of the node's next position is according to the position of its neighbors. The distance between a current node location and its next position depends on the density of the nodes within its communication range. From the figure, the DSSA algorithm does not perform well in this setting since the nodes located near the boundary tend to move out of the region.

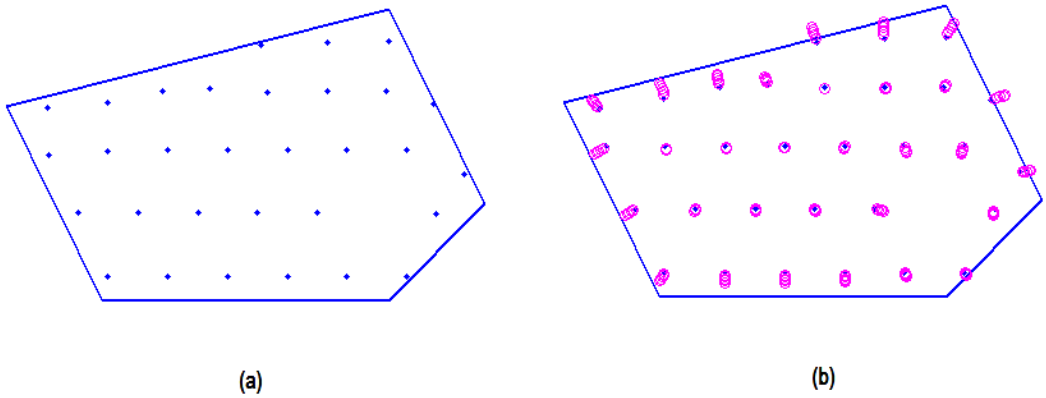

Figure 4.13: Node repositioning using DSSA algorithm. Figure (a) is the original node locations. Figure (b) shows node locations after five time steps.

In VDDA algorithm, the goal is to have the Voronoi region corresponding to a node to coincide with the node's sensing region. The algorithm also considers energy consumption and moving distance in the relocation calculation. The node determines its utility gain over several possible locations and moves to the next position according to the best utility gain. The node requires less energy consumption for the movement and goes to the position that gives better effective area. However, the algorithm spends more time in determining the next position, since it has to calculate the utility gains from several points before making a decision.

For randomly dispersed nodes, the Voronoi regions are with various shapes and sizes as shown in figure 4.11. If a Voronoi region corresponding to a node is much bigger than the node's sensing region, the effective areas are the same in many searched positions. Estimated lifetime in each position distinguishes the utility gain between these same effective areas. The position that gives the maximum product of effective area and estimated lifetime is selected as the next position.

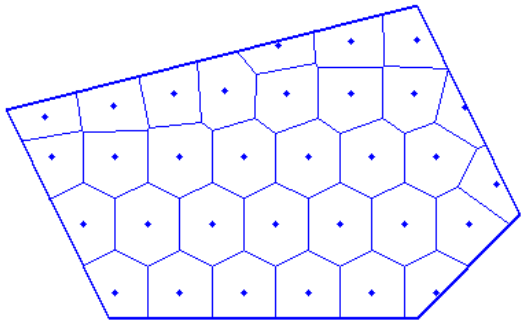

Figure 4.14: Voronoi diagram of the nodes at predetermined locations

When consider a situation where the nodes are in the predetermined locations that they entirely cover the region of interest, the Voronoi regions corresponding to the nodes are smaller than their sensing regions as shown in figure 4.14. When there is one missing node, the Voronoi regions around the missing node are bigger than their sensing regions. The region of interest is not entirely covered. The Voronoi diagram of this situation is shown in figure 4.15. From this situation, node relocation is applied to compensate the omitted region and make the region of interest entirely covered again.

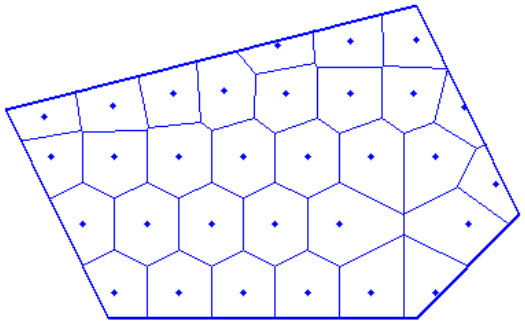

Figure 4.15: Voronoi diagram of predetermined locations with one missing node

Since most of the nodes in figure 4.15 are already in the positions giving them maximal effective area, nodes movement and utility gains iteration as performed in VDDA

algorithm are not necessary. However, when applying VDDA algorithm to the situation as shown in figure 4.15, the neighbors of the missing node move only slightly since the utility gain concerns with the node's energy consumption and effective area covered by it. The node consumes less energy if it moves less. The algorithm searches for maximal utility gain before every node movement, hence the nodes take long time to compensate the missing node.

It can be seen from the figure that moving the nodes toward the missing node's direction assists the missing area coverage. The neighbors of the missing node are closer to the missing node's position when they move toward centroid of their Voronoi regions. Hence, the centroid-based scheme could efficiently relocate the nodes in a given situation.

In the centroid-based scheme, a node moves if the Voronoi region corresponding to the node is smaller than its sensing region. The node relocates when the distance between the node location and one of the Voronoi vertices is longer than its sensing range. After the movement, the node updates its location and Voronoi region. It repeats the relocation if the distance is still longer than its sensing range.

Figure 4.16 shows the nodes relocation of four time steps from figure (a) to (d) respectively. Blue dots indicate the current locations. Red dots indicate the next movements. The relocation in this situation stops after four time steps. There is no distance between the vertices and the node locations longer than the sensing range at the fourth step. In figure (a), the neighbors of the missing node move toward the missing node's direction. In figure (b), the nodes around the neighbors also move toward the missing node's direction. The node locations and their sensing regions after the fourth time step is shown in figure 4.17. The entire region of interest is now covered by the nodes.

This node relocation also applies to a situation where there are more than one missing nodes. The relocation is effective as long as the number of working nodes is not less than the minimum number of nodes required to fully cover the region of interest. For example, in a situation that has four nodes missing in the region as shown in figure 4.18. The green dots in figure (a) are positions of the missing nodes. The sensing regions of the operating nodes are as shown in figure (b).

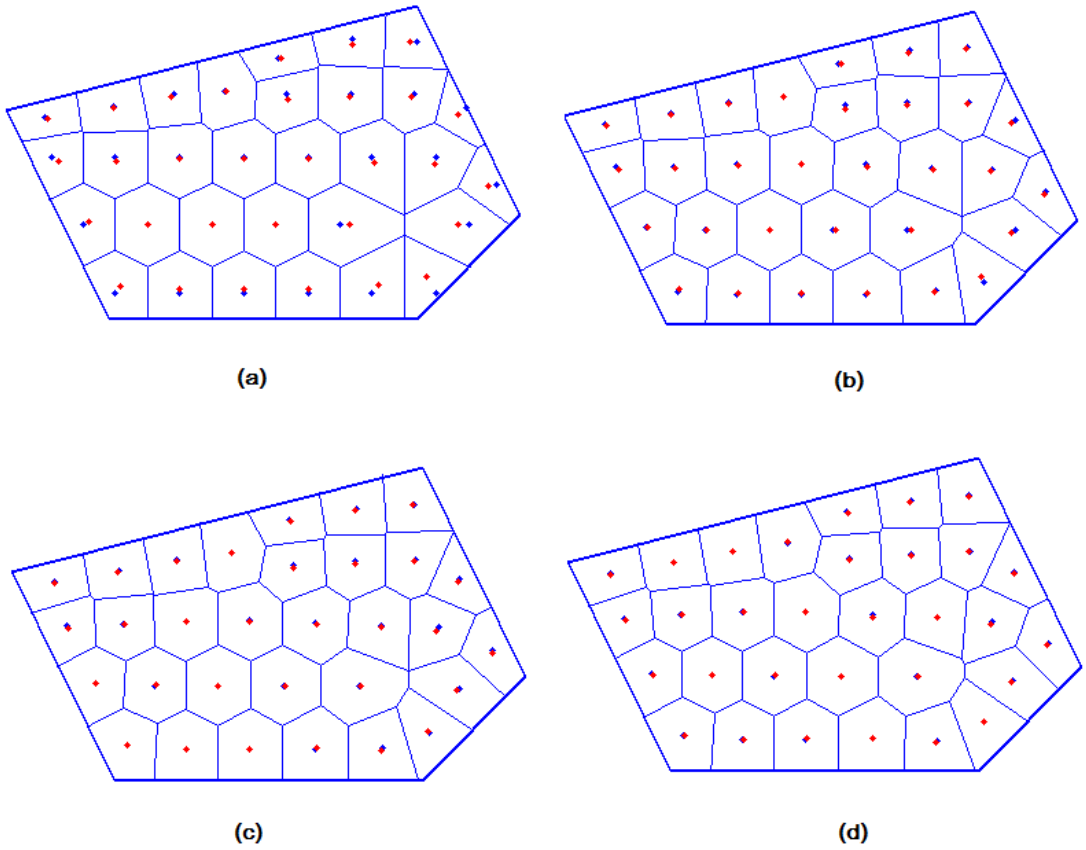

Figure 4.16: Nodes relocation at four time steps from figure (a) to (d) respectively. Blue dots indicate the current node locations. Red dots indicate the next movements.

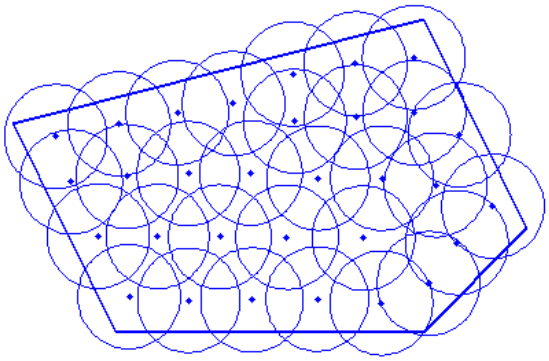

Figure 4.17: Nodes locations and their sensing regions after applying the algorithm. The region is entirely covered by the nodes again

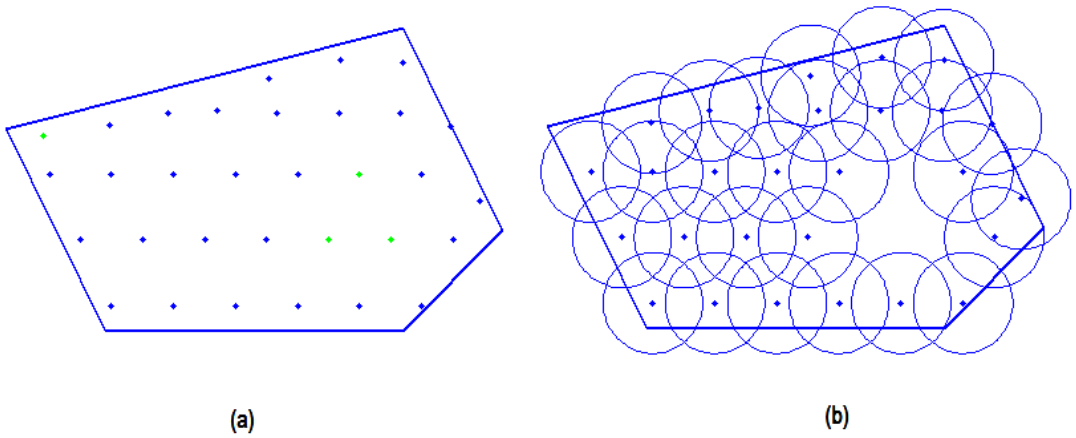

Figure 4.18: Predetermined node locations with four missing nodes

Figure 4.19 shows the Voronoi diagram and relocation algorithm of the node locations in figure 4.18. Figure 4.19 (a) is the node movement at the first time step after applying the algorithm. Figure (b), (c), (d), and (e) are at second, fourth, sixth, and eighth time step respectively. After eighth time step of the algorithm, the nodes entirely cover the region as can be shown in figure (f).

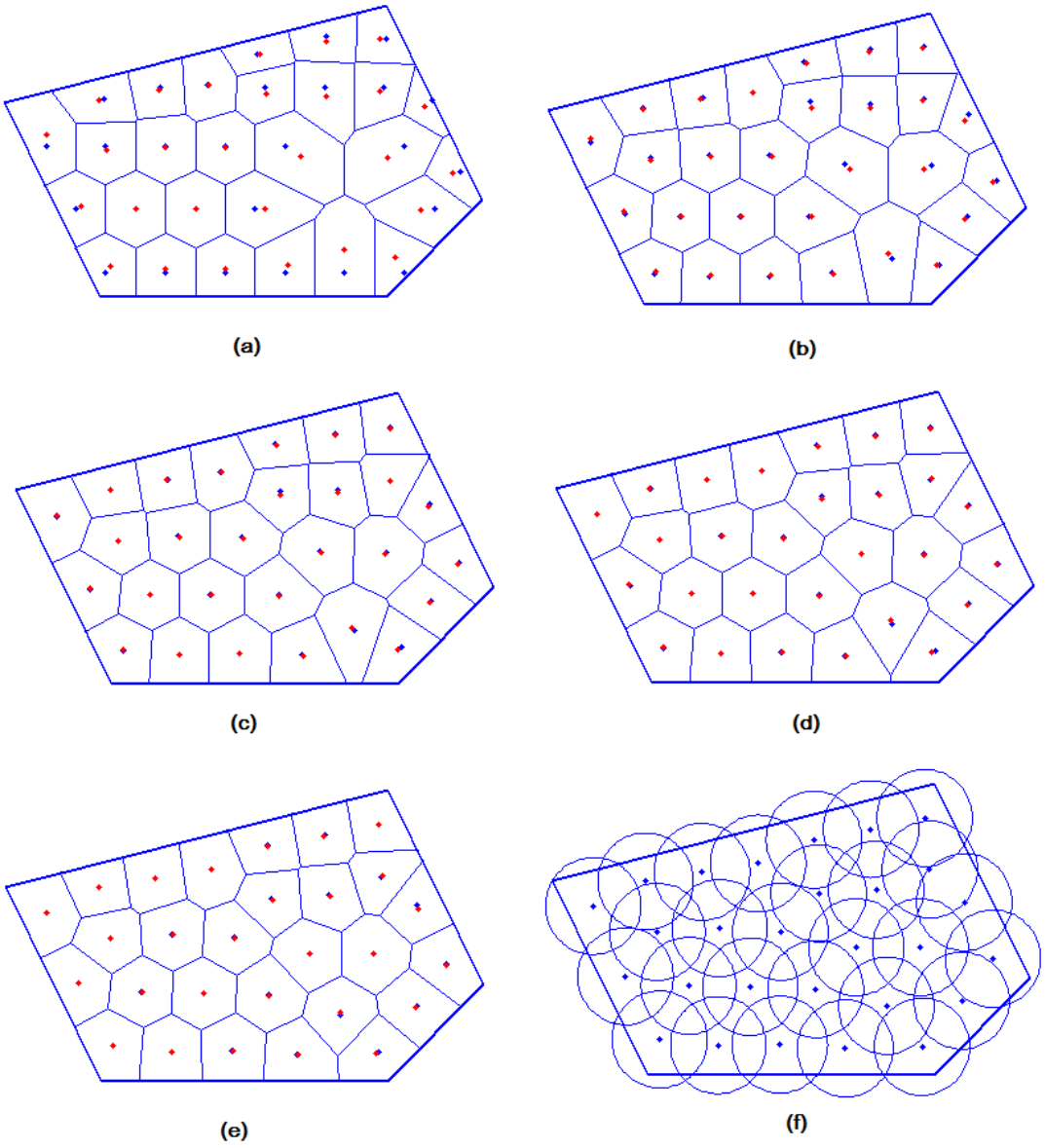

Figure 4.19: Voronoi diagram and node relocation algorithm when there are four missing nodes. The figures (a) to (e) are at 1,2,4,6,8 time step respectively. The relocation completes after eight time step as shown in figures (e) and (f).

Chapter 5

Conclusion and Future Work

This thesis has presented different algorithms for area coverage problem. The algorithms are divided into non-communicative and communicative area coverage. In non-communicative, the UAV agents perform the assigned tasks disjointedly by sweeping over the region of interest. They follow the partitioned regions and routes assigned by the base station. In communicative area coverage, the agents spread over the region of interest. They sense regions within their sensing range and are able communicate to the nodes within their communication range. This area coverage can monitor the region at longer time and send the data to the base station in near real time. However, it requires more agents to perform the task.

In non-communicative area coverage, a polygon area decomposition algorithm is proposed. The algorithm intends for a setting where the UAV agents have to start and end the area coverage at the same base station. The region of interest is a simple, non-convex polygon. The algorithm considers geometry of the polygon before decomposing the region. It starts the decomposition at the longest side of the polygon with partitioning line segment parallel to the longest side. The algorithm assigns the partitioning line segment to be parallel to the longest side of the polygon for the smaller minimum diameter function. The results show that a partitioned region always gives less or equal number of turns compared to a partitioned region from [15]. Moreover, the algorithm also gives less overlapped areas since all the partitioned regions have one side connected to the base station.

The proposed algorithm is for a situation where a region of interest is a convex polygon without any holes or obstacles. However, there could be a situation where the agents have to avoid obstacles in some area inside the region of interest. There exist algorithms for partitioning the polygons with obstacles, but none of them considers a situation where the agents have to start and end the routes at the same base station. Figure 5.1 illustrates examples of a polygon with an obstacle. The figures show possible ways to

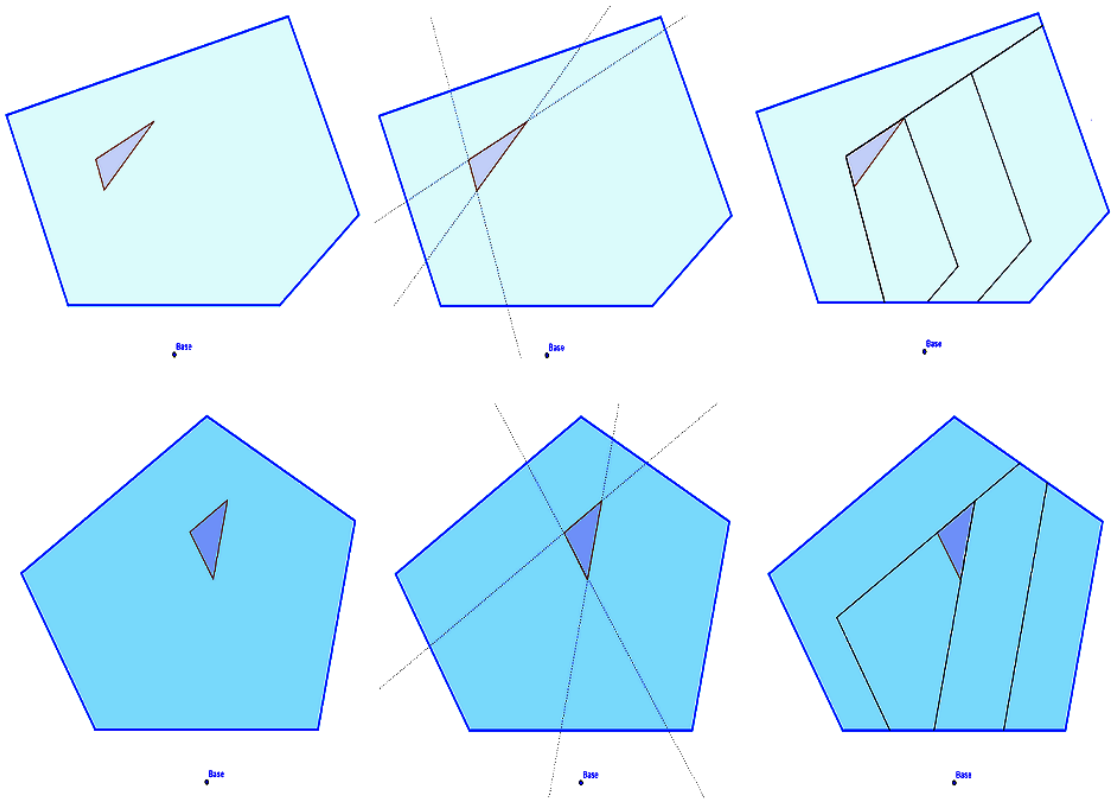

Figure 5.1: Examples of partition of a polygon with obstacle.

partition the polygon.

The potential future work of this area decomposition with or without obstacles could be simulating the sweeping area coverage in virtual situation using simulating tools such as USARSim. The simulation would give quantitative comparison between the two algorithms and yield parameters such as appropriate flying height or maximum flying distance.

In non-communicative area coverage, the overall task cannot be completed if one agent fails to perform its mission. The agent might be broken down or lost without any awareness from its team member or the base station. On the contrary, all the agents and the base station are connected in communicative area coverage. The base station knows the

agents' status and locations for more robust and real time area coverage.

The algorithms presented in communicative area coverage are static and dynamic positioning. Static positioning predetermined the node positions according to sensing range of the node and region of interest. The positioning gives fast but non-adaptable node deployment. In finding minimum number of nodes that fully cover the region of interest, hexagonal tiling is used to provide approximate number and positions of the nodes. The distance between a node at the center of a hexagon and its adjacent neighbors are equal in all directions.

Dynamic positioning assists the area coverage when the nodes are randomly distributed or when there is a potential node failure. In this work, three algorithms were studied and applied to predetermined node locations where there is one or more nodes missing. The algorithms are DSSA, VDDA, and the centroid-based scheme. DSSA relocation is not suitable for the mentioned setting since boundary of the region of interest is not considered in the algorithm. The nodes near the boundary tend to move out of the region according to partial forces exerted from their neighbors.

VDDA and centroid-based scheme algorithms use Voronoi diagram to determine the region the node has to cover. Hence, the algorithms can limit the node movement to within a region of interest by considering only the Voronoi diagram that intersects with the region of interest. VDDA algorithm searches for the location that gives maximal utility gain which is the product of effective area covered by the node at its new location and the estimated lifetime when the node moves to the new location. When applying VDDA to the predetermined node locations with one or more missing nodes, most of the nodes remain at their current locations. This is because the locations are predetermined and the current locations already give maximal utility gain.

The centroid-based scheme is applicable to this setting. When there is a missing node, the Voronoi region of its neighboring node is bigger than its sensing region. The neighboring node moves to the centroid of its Voronoi region. The direction of the centroid is toward missing the node's direction. Hence, the nodes compensate the missing one and fully cover the region of interest after some rounds of relocation. The time spending on node relocation depends on the number of missing nodes and position of the missing nodes. The location of the missing node that has more adjacent neighbors requires less time to relocate than the location that has fewer adjacent neighbors. This relocation algorithm functions properly as long as the number of working nodes is not less than the minimum number of nodes required to completely cover the region of interest.

The communicative area coverage in this thesis considers only the positioning algorithm. Nevertheless, the aspects related to communication should also be concerned in the actual situation. For example, the number of hops in multi-hop wireless ad hoc network, the communication delay, ad hoc routing algorithm, and etc. These factors affect the efficiency of the area coverage and should be studied further.

List of Figures

Bibliography

[1] A. Agarwal, L. M. Hiot, E. M. Joo, and N. T. Nghia. Rectilinear workspace partitioning for parallel coverage using multiple uavs.

[2] T. Asano and T. Asano. Minimum partition of polygonal regions into trapezoids. In *Proceedings of the 24th Annual Symposium on Foundations of Computer Science*, SFCS '83, pages 233–241, 1983.

[3] F. Aurenhammer. Voronoi diagrams a survey of a fundamental geometric data structure. *ACM Comput. Surv.*, 23(3):345–405, Sept. 1991.

[4] Y. Bouktir, M. Haddad, and T. Chettibi. Trajectory planning for a quadrotor helicopter. *2008 16th Mediterranean Conference on Control and Automation*, pages 1258–1263, 2008.

[5] B. M. Chazelle. *Computational geometry and convexity*. PhD thesis, New Haven, CT, USA, 1980. AAI8110021.

[6] C. Chen and R.-C. Chang. On the minimality of polygon triangulation. *BIT*, 30(4):570–582, 1990.

[7] A. Clot and R. S. L. M. G. Britain). *Communications Command and Control*. Defense Technical Information Center, 2000.

[8] L. de Floriani and E. Puppo. An On-Line Algorithm for Constrained Delaunay Triangulation. *GMIP*, 54(4):290–300, July 1992.

[9] P. A. Devijver and S. Maybank. On the computation of the delaunay triangulation of a convex polygon. In *Proc. 6th IEEE Internat. Conf. Pattern Recogn.*, pages 420–422, 1982.

[10] S. Fowers. *Stabilization and control of a quad-rotor micro-uav using vision sensors*. PhD thesis, 2008.

[11] D. H. Greene. The decomposition of polygons into convex parts. *Computational Geometry, Advances in Computing Research*, 1:235–259, 1983.

[12] M. Günes and O. Spaniol. Routing Algorithms for Mobile Multi-Hop Ad-Hoc Networks. In H. Turlakov and L. Boyanov, editors, *Proceedings of Next Generation Network Technologies International Workshop*, pages 10–24 October, October 2002.

[13] N. Heo and P. Varshney. A distributed self spreading algorithm for mobile wireless sensor networks. In *Wireless Communications and Networking, 2003. WCNC 2003. 2003 IEEE*, volume 3, pages 1597 –1602 vol.3, 2003.

[14] N. Heo and P. Varshney. Energy-efficient deployment of intelligent mobile sensor networks. *Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on*, 35(1):78 – 92, 2005.

[15] S. Hert and V. Lumelsky. Polygon area decomposition for multiple-robot workspace division. *International Journal of Computational Geometry and Applications*, 8:437–466, 1998.

[16] G. M. Hoffmann, S. L. Waslander, and C. J. Tomlin. *Electrical Engineering*, 44(August):1–14, 2008.

[17] W. H. Huang. Optimal line-sweep-based decompositions for coverage algorithms. In *Proceedings IEEE International Conference on Robotics & Automation*, pages 27–32, 2001.

[18] J. M. Keil. Decomposing a polygon into simpler components. *SIAM Journal on Computing*, 14:799–817, 1985.

[19] P. D. R. Klein. Geometry lab: The diameter function of an arbitrary polygon. `http://web.cs.uni-bonn.de/I/GeomLab/Diameter/index.html.en`. [Online; accessed 28-Dec-2011].

[20] H.-J. Lee, Y. hwan Kim, Y.-H. Han, and C. Y. Park. Centroid-based movement assisted sensor deployment schemes in wireless sensor networks. In *VTC Fall*, pages 0–, 2009.

[21] C. Levcopoulos and A. Lingas. Bounds on the length of convex partitions of polygons. In *Proceedings of the Fourth Conference on Foundations of Software Technology and Theoretical Computer Science*, pages 279–295, London, UK, UK, 1984. Springer-Verlag.

[22] I. Maza and A. Ollero. Multiple uav cooperative searching operation using polygon area decomposition and efficient coverage algorithms. In *Proceedings of the 7th International Symposium on Distributed Autonomous Robotic Systems*, pages 211–220, 2004.

[23] A. Mukherjee, S. Bandyopadhyay, and D. Saha. Radio propagation. In *Location management and routing in mobile wireless networks*, Artech House mobile communications series, pages 32–37. Artech House, 2003.

[24] I. Redbooks. Wireless lan performance. In *Local Area Network Concepts and Products: Adapters, Hubs and Atm*. Vervante, 1996.

[25] S. B. Tor and A. E. Middleditch. Convex decomposition of simple polygons. *ACM Trans. Graph.*, 3(4):244–265, 1984.

[26] H. ur Rehman. *Multihop connectivity in wireless ad hoc networks*. PhD thesis.

[27] D. Wang, L. Xu, J. Peng, and S. Robila. Subdividing hexagon-clustered wireless sensor networks for power-efficiency. In *Proceedings of the 2009 WRI International Conference on Communications and Mobile Computing - Volume 02*, CMC '09, pages 454–458, Washington, DC, USA, 2009. IEEE Computer Society.

[28] M. Younis and K. Akkaya. Strategies and techniques for node placement in wireless sensor networks: A survey. *Ad Hoc Netw.*, 6(4):621–655, June 2008.