On the Problem of Covering a 3-D Terrain

Eduard Eiben,[1] **Isuru S. Godage,**[2] **Iyad Kanj,**[2] **Ge Xia**[3]
[1]Department of Computer Science, Royal Holloway, University of London, UK. Email: Eduard.Eiben@rhul.ac.uk
[2]School of Computing, DePaul University, USA. Emails: igodage@depaul.edu, ikanj@depaul.edu
[3]Department of Computer Science, Lafayette College, USA. Email: xiag@lafayette.edu

Abstract

We study the problem of *covering* a 3-dimensional terrain by a sweeping robot that is equipped with a camera. We model the terrain as a mesh in a way that captures the elevation levels of the terrain; this enables a graph-theoretic formulation of the problem in which the underlying graph is a weighted plane graph. We show that the associated graph problem is NP-hard, and that it admits a polynomial time approximation scheme (PTAS). Finally, we implement two heuristic algorithms based on greedy approaches and report our findings.

Introduction

We study the problem of *covering* a 3-dimensional (3-D) terrain by a sweeping robot that is equipped with a camera; we refer to this problem as the TERRAIN COVERAGE problem. Informally speaking, the TERRAIN COVERAGE problem is to compute a terrain path such that every point in the terrain falls within the camera-range of at least one point of this path. We show how to model the TERRAIN COVERAGE problem as the following graph problem: Given an edge-weighted plane graph, each of whose vertices is associated with a subset of vertices that it can cover, compute a path of minimum weight such that every vertex in the graph is covered by (at least one point of) this path. Variants of this fundamental motion planning problem have been studied, motivated by their robotics applications, which include de-mining (Gage 1994), lawn mowing/harvesting (Arkin, Fekete, and Mitchell 2000; Cao, Huang, and Hall 1988), painting (Atkar et al. 2005), and autonomous underwater inspection of complex structures (Englot and Hover 2012; Hert, Tiwari, and Lumelsky 1996), among others.

In this paper, we study the complexity of the TERRAIN COVERAGE problem and design approximation and heuristic algorithms for it.

Related Work. A significant amount of work has been devoted to variants of the TERRAIN COVERAGE problem, which fall under the umbrella of the so-called Coverage Path Planning problem (CPP). In the CPP problem, given a set of points in the plane, one seeks a path that "covers" all the points in the set, while optimizing a certain function/criterion. Various notions of coverage have been studied, and their corresponding problems turn out to be NP-hard, being variants of the Euclidean Travelling Salesman problem (Euclidean TSP), for which polynomial time approximation schemes (PTAS) exist (Arora 1998; Mitchell 1999). For instance, Arkin and Hassin (Arkin and Hassin 1994) gave constant-ratio approximation algorithms for the Covering Salesman problem, in which we are given a set of points referred to as "buyers", each with a specified set of locations that she is willing to travel to in order to meet the salesman, and the goal is to compute a closed tour of minimum Euclidean length that meets all buyers. Another related problem is the sweeper problem, where one is given a set of points in the plane, and a sweeper's shape (disk/rectangle/etc.), and the goal is to find a path of minimum Euclidean length that sweeps through all given points; constant-ratio approximation algorithms were also given for continuous versions of the sweeper problem, referred to as the milling and mowing problems (Arkin, Fekete, and Mitchell 2000), where the area to be milled/mowed is a continuous region of the plane (*e.g.*, disk or polygon).

Perhaps the most related problem to the TERRAIN COVERAGE problem under consideration, albeit much simpler in terms of its setting, is the Watchman (resp. Sweeper) Route Under Limited Visibility problem, in which we are given a simple polygon and a visibility range, and the goal is to find a tour of minimum Euclidean length such that each point on the boundary of the polygon (resp. in the polygon) is visible from at least one point of the tour. Constant-ratio approximation algorithms were given for the watchman variant of the problem, and a PTAS was given for the simpler sweeper variant (Li and Klette 2008; Ntafos 1991). Other problems that are related to TERRAIN COVERAGE have also been considered, with different notions of coverage and/or optimization criteria; see (Arkin et al. 2011; 2005; Fekete, Mitchell, and Schmidt 2012).

The problem of covering a 3-D terrain has also been extensively studied based on different approaches. One approach relies on using a planar 2-dimensional (2-D) terrain covering algorithm, where the idea is to apply a planar coverage algorithm to successive horizontal planes of the 3-D terrain (Hert, Tiwari, and Lumelsky 1996; Lee et al. 2009). Another valid approach is to use randomized algorithms (*e.g.*, floor-sweeping robots). However, for covering a vast 3-D terrain, randomized algorithms are infeasible, as the energy cost would not be affordable (due to the large amount of coverage overlap which is common in such solutions) (Galceran and Carreras 2013).

Most path coverage algorithms start by decomposing the free space to be covered into subregions, called *cells*. Different decomposition methods have been studied (*e.g.*, trapezoidal (Choset et al. 2005; Latombe 1991); Boustrophedon (Choset et al. 2005); Morse (Acar et al. 2002); grid-based coverage (Lee et al. 2011; Gabriele and Rimon 2002)), and the algorithms are classified accordingly. Graph-based coverage algorithms were also used; they start by formulating the problem as a graph, and then apply heuristics (Xu, Viriyasuthee, and Rekleitis 2011). For more details on coverage path planning and 3-D path planning, we refer the reader to the following surveys (Choset 2001; Galceran and Carreras 2013; Yang et al. 2016).

Results and Techniques. We start by modelling the TERRAIN COVERAGE problem as a graph problem in two phases: in the first phase we create a 2-D mesh to represent the 3-D terrain, using weights to model the terrain elevations, and in the second phase we define a weighted plane graph based on the plane subdivision determined by the mesh. The meshing and camera parameters (*e.g.*, camera range) are captured/modelled through the structural parameters of the defined graph. More specifically, each vertex is associated with a subset of vertices that it can *cover*, and each edge is associated with a weight that reflects the distance between the mesh cells corresponding to its endpoints. The resulting graph problem seeks a path of minimum weight whose vertices cover the vertex-set of the graph.

We first show that the resulting graph problem is NP-hard in the strong sense, even for very restricted instances of the problem; this rules out the existence of a fully polynomial time approximation scheme (FPTAS) for the problem. We then study the approximation properties of this graph problem and show that it admits a polynomial time approximation scheme (PTAS); that is, we show that it can be approximated to within any precision in polynomial time. Unlike many plane graph problems for which Baker's (Baker 1994) folklore layer decomposition method yields PTAS, the TERRAIN COVERAGE problem does not yield to this method, as explained later in this paper. Instead, to design our PTAS, we use an approach similar to that of Klein (Klein 2008), which combines the layer decomposition method with dynamic programming on graphs of bounded treewidth.

Finally, we implement two efficient heuristic (greedy) algorithms, evaluate their performance empirically on different terrain types and camera ranges, and report our findings.

Due to the space limit, some proofs have been omitted.

Preliminaries

We refer to (Diestel 2012) for more information on graphs.

A graph is *planar* if it can be drawn in the plane without edge intersections (except at the endpoints). A *plane graph* has a fixed embedding in the plane. The unbounded face is called the *outer face*. A *1-outerplane graph*, or an *outerplane graph*, is a planar graph that has a drawing for which every vertex is incident to the outer face. An *i-outerplane graph*, for $i > 1$, is defined inductively as a graph such that the removal of its outer face results in an *$(i-1)$-outerplane graph*. The *layer decomposition* of a plane graph G is a partitioning of $V(G)$ into disjoint layers $(L_1, \ldots, L_r)$, $r \geq 1$, defined inductively as follows. Layer L_1 is the set of vertices that lie on the outer face of G, and layer L_i is the set of vertices that lie on the outer face of $G - \bigcup_{j=1}^{i-1} L_j$ for $1 < i \leq r$. A layer decomposition of a planar graph G can be computed in linear time (Baker 1994).

For an edge $e = uv$ in a edge-weighted graph G with edge-weight function $wt()$, *contracting* e means removing the two vertices u and v from G, replacing them with a new vertex w, and for every vertex y in the neighborhood of v (resp. u) in G, adding in the new graph an edge wy of weight $wt(vy)$ (resp. $wt(uy)$), while allowing the creation of multiple edges but no self-loops. For a subset $S \subseteq E(G)$, G/S denotes the graph resulting from G by contracting the edges in S.

A *tree decomposition* of $G = (V, E)$ is a pair $(\mathcal{V}, \mathcal{T})$, where $\mathcal{V}$ is a collection of subsets of V, referred to each as a *bag*, satisfying that $\bigcup_{X_i \in \mathcal{V}} = V$, and $\mathcal{T}$ is a rooted tree whose node set is $\mathcal{V}$, such that: (1) for every edge $e \in E$, there is a bag in $\mathcal{V}$ containing both endpoints of e; and (2) if $X_i, X_j, X_k \in \mathcal{V}$ such that X_j is on the path between X_i and X_k in $\mathcal{T}$, then $X_i \cap X_k \subseteq X_j$. The *width* of the tree decomposition $(\mathcal{V}, \mathcal{T})$ is defined to be $\max\{|X_i| \mid X_i \in \mathcal{V}\} - 1$. The *treewidth* of the graph G, denoted $\mathrm{tw}(G)$, is the minimum width over all tree decompositions of G.

A tree decomposition $(\mathcal{V}, \mathcal{T})$ is *nice* if it satisfies: (1) Each node in the tree $\mathcal{T}$ has at most two children; (2) if a node X_i has two children X_j and X_k in $\mathcal{T}$, then $X_i = X_j = X_k$, and node X_i is called a *join node*; (3) if a node X_i has only one child X_j in $\mathcal{T}$, then either $|X_i| = |X_j|+1$ and $X_j \subset X_i$, and in this case X_i is called an *insert node*, or $|X_i| = |X_j| - 1$ and $X_i \subset X_j$, and in this case X_i is called a *forget node*; and (4) if X_i is a leaf node or the root, then $X_i = \emptyset$.

A *polynomial-time approximation scheme* (PTAS) for a minimization problem Q is an algorithm that takes as input an instance $\mathcal{I}$ of Q and $\epsilon > 0$, and outputs a solution to $\mathcal{I}$ whose value is at most $(1 + \epsilon) \cdot \mathrm{opt}(\mathcal{I})$, where $\mathrm{opt}(\mathcal{I})$ is the value of an optimal solution for $\mathcal{I}$, and such that the running time of the algorithm is polynomial for every fixed $\epsilon > 0$. A *fully polynomial-time approximation scheme* (FPTAS) is a PTAS whose running time is polynomial in both the input size and $1/\epsilon$.

For two sets A, B, $A \triangle B = (A \setminus B) \cup (B \setminus A)$ denotes the symmetric difference of A and B. For $\ell \geq 1 \in \mathbb{N}$, we write $[\ell]$ for the set $\{1, \ldots, \ell\}$.

Problem Modeling

We model the terrain coverage problem in two phases.

Meshing the terrain. In the first phase, we generate a polygonal 2-D mesh for the 3-D terrain, which is assumed to be a simply-connected surface (*i.e.*, contains no holes or obstructions, such as lakes or other inaccessible areas). For our simulations, we opted for triangular meshes instead of other polygonal meshes, where the former also benefit from faster mesh generating algorithms. However, our results hold for any polygonal meshes.

Geographical terrains are essentially 3-D surfaces, where each point of the terrain can be described by its coordinates x, y, z denoting the latitude, longitude, and elevation, respectively. To capture the complex terrain features in a mesh approximation, it is critical to have uniformly-distributed vertices on the surface of the terrain. A planar triangulation can easily be generated by using the terrain footprint. However, a planar meshing solution on a projected planar area of the terrain can result in a large variety of triangle sizes when lifted back to the Cartesian system, which, in turn, would result in poor terrain approximation where there are non-flat geological features. To circumvent this issue, one has to account for terrain slopes (the surface gradient) and adjust the triangulation such that the spatial triangulation has bounded edge-lengths variation. We applied distmesh2d (Persson 2004) with a nonuniform edge-length function to triangulate the terrains. This function is derived as follows. First, the 2-D gradient of the terrain map is computed. Then, the resulting matrix is used to generate the lengths of the edges (higher gradient implies shorter edge lengths). The vertex coordinates of the triangulation are then used to interpolate the elevation from the original terrain map matrix. Due to the uniform distribution of vertices, this results in triangular meshes of similar triangle sizes in which the minimum-maximum edge-length ratio for all triangles is bounded.

Graph formulation. Once we have generated the 2-D mesh as above, in the second phase we define a plane graph as follows. For every cell in the mesh, we associate a vertex in the graph located at the center of (gravity of) the cell. Two vertices are adjacent if and only if their corresponding cells are adjacent (*i.e.*, share an edge). The weight of an edge is the Euclidean distance between the 3-D points in the terrain that correspond to the 3-D images of the cell centers. Since the mesh is assumed to be a triangular mesh, the graph obtained has bounded maximum degree. Moreover, based on the meshing process (*i.e.*, similarities between the triangles in the mesh), the length of any edge in the mesh is both lower bounded and upper bounded by some constants; that is, the weights of the edges of the plane graph are between 1 (unit) and some constant $\lambda \geq 1$. Since the camera has some fixed range based on its specification, if the camera is located at a certain vertex v, then the camera will be able to cover a subset of vertices $\gamma(v)$, each of distance at most some integer constant ρ (depending on the camera range) to v, and such that the subgraph induced by $\gamma(v)$ is connected (we assumed that the robot can travel between any two points that are visible from each other). Next, we define these concepts formally.

Let G be an edge-weighted bounded-degree plane graph of maximum degree Δ, with weight function $wt : E(G) \longrightarrow \mathbb{Q}$. For a rational constant $\lambda \geq 1$, we say that G is λ-*proportionally weighted*, or λ-weighted for a shorthand, if the weight of the maximum-weight edge in G is at most λ times the weight of the minimum-weight edge in G; or equivalently, assuming that the edge weights are normalized so that the minimum weight is 1, for any edge $e \in E(G)$, we have $1 \leq wt(e) \leq \lambda$. Each vertex $v \in V(G)$ is associated with a set of vertices $\gamma(v) \supseteq \{v\}$, where the subgraph $G[\gamma(v)]$ induced by $\gamma(v)$ is connected, and its radius w.r.t. v is at most ρ, where $\rho \geq 1$ is an integer constant. In the context of the TERRAIN COVERAGE application, λ is a parameter determined by the meshing process, ρ is a parameter determined by the robot's camera specifications, and the sets $\gamma(v)$, for $v \in V(G)$, are determined by both the camera specifications and the topography of the terrain, as discussed in the previous subsection. Without loss of generality, we assume that, for any two vertices $u, v \in V(G)$, if $u \in \gamma(v)$ then it holds that $v \in \gamma(u)$. For $S \subseteq V(G)$, we define $\gamma(S) = \bigcup_{v \in S} \gamma(v)$.

A subset $S \subseteq V(G)$ *covers* a subset $S' \subseteq V(G)$, or is a *covering set* for S', if for each $u \in S'$, there exists $v \in S$ such that $u \in \gamma(v)$. S covers G if it covers $V(G)$. A *tour* is a walk in G. The weight of the tour is the sum of the weights of all edges in it, including multiplicities of edges. The subset of vertices covered by a tour τ is the subset of vertices in G covered by the vertices that appear in τ.

From the above, we can formulate the 3-D terrain coverage problem as the following graph optimization problem:

TERRAIN COVERAGE
Given: A λ-weighted plane graph G ($\lambda \geq 1$) of bounded degree Δ, in which each vertex v is associated with a connected set of vertices $\gamma(v)$ of radius at most ρ; and a vertex $s \in V(G)$.
Goal: Compute a tour τ of minimum weight that starts at s and visits a covering set for G.

Complexity and Approximation

In this section, we study the complexity and approximation of TERRAIN COVERAGE. We start with the following result:

Theorem 1. *The* TERRAIN COVERAGE *problem is* NP*-hard in the strong sense, even when $\lambda = \rho = 1$ and $\Delta = 4$.*

The proof of Theorem 1 is via a reduction from the HAMILTONIAN PATH problem on planar cubic graphs, and results in instances of TERRAIN COVERAGE in which the maximum edge weight is 1. This shows that the problem is NP-hard in the strong sense (Garey and Johnson 1979), which (assuming P$\neq$ NP) rules out the existence of fully polynomial time approximation schemes (FPTAS) for the problem (Garey and Johnson 1979). Given the above, we now switch our attention to designing PTAS for TERRAIN COVERAGE.

Let G be a plane graph with layer decomposition $(L_1, \ldots, L_r)$. Baker (Baker 1994) used the layer decomposition of a plane graph to design PTAS for many graph problems based on a planar separator approach. The idea is to select a separator, consisting of a group of layers, that contains a small fraction (depending on the desired approximation error) of the optimal solution, whose removal breaks the graph into chunks, each with bounded outerplanarity. More specifically, for a given approximation error ϵ, a number $k \in \mathbb{N}$

that depends on ϵ is chosen properly. Then the layers are partitioned into k groups, each consisting of all layers whose indices are congruent to the same number modulo k. Using an averaging argument, one of these groups has weight at most a "small" fraction of that of an optimal solution, and hence, by discarding all vertices in the group we do not "lose much". Discarding all layers in a group results in separating the graph into chunks, each of outerplanarity at most k, and hence of treewidth at most $3k - 1$ (Robertson and Seymour 1991; Biedl 2015). A dynamic programming approach then computes an optimal solution of each chunk in polynomial time, based on which an approximate solution is returned.

While this approach works for many graph problems, it does not work for TERRAIN COVERAGE. An approach of similar flavor was proposed by Klein (Klein 2008) for planar TSP. In this approach, instead of removing the layers in the chosen separator group, the edges in these layers are contracted. This still yields a graph of bounded treewidth ($\leq 3k + 2$), enabling dynamic programming on the contracted graph. An optimal solution for the contracted graph is then lifted to an approximate solution of the original graph.

As noted before, the TERRAIN COVERAGE problem is more complex than the TSP, in the sense that the sought tour is required to only *cover* all the vertices, not *visit all* of them. This poses several complications in multiple stages of this approach. We show how to circumvent these complications to make this approach work for TERRAIN COVERAGE.

First, we refine a result in (Klein 2008), which is a primary ingredient in our approach. Let $(G, \lambda, \Delta, \gamma, \rho, s)$ be an instance of TERRAIN COVERAGE, where each $\gamma(v)$ has radius at most ρ. In (Klein 2008) (implied from Corollary 7.2), they showed that for a weighted plane graph G with layer decomposition $(L_1, \ldots, L_r)$, where $r \geq 1$, there is a linear time algorithm that, for any integer $k \geq 1$, computes a set of layers $\mathcal{S}_k$ with edge set $E(\mathcal{S}_k)$ such that $wt(E(\mathcal{S}_k)) \leq wt(G)/k$ and $tw(G/E(\mathcal{S}_k)) \leq 3(k+2)$. We refine the set $\mathcal{S}_k$ of layers as follows. First, we remove from $\mathcal{S}_k$ every layer (if any) that is among the first or last $\rho + 1$ layers in G (*i.e.*, we remove every L_i where $1 \leq i \leq \rho + 1$ or $r - \rho \leq i \leq r$). Second, if there are three layers in $\mathcal{S}_k$ such that the number of layers between two of them is $\leq 2\rho$, then we remove the intermediate layer among the three from $\mathcal{S}_k$ (*i.e.*, the one whose index is comprised between the two indices of the others). It is easy to see that by doing so, we may have increased the outerplanarity of each chunk of the graph resulting from removing the layers in $\mathcal{S}_k$ by at most $2\rho + 1$. Using the same arguments as in (Klein 2008), the above discussion yields the following modification of the result in (Klein 2008):

Lemma 2. *Let G be a weighted plane graph with layer decomposition $(L_1, \ldots, L_r)$, where $r \geq 1$. There is a linear time algorithm that, for any integer $k \geq 1$, computes a set of layers $\mathcal{S}_k$ with edge set $E(\mathcal{S}_k)$ such that $wt(E(\mathcal{S}_k)) \leq wt(G)/k$ and $tw(G/E(\mathcal{S}_k)) \leq 3(k+2\rho+3)$. Moreover, $\mathcal{S}_k$ does not contain any of the first $\rho + 1$ or last $\rho + 1$ layers in G, and between any two "consecutive" layers L_i and L_j in $\mathcal{S}_k$ (*i.e.*, there is no layer L_q in $\mathcal{S}_k$ with $i < q < j$), we have $j - i > 2\rho + 1$; that is, any two consecutive layers in $\mathcal{S}_k$ are separated (in G) by at least $2\rho + 1$ layers.*

Let $(G, \lambda, \Delta, \gamma, \rho, s)$ be an instance of TERRAIN COVERAGE, $k \geq 1$ be an integer, and let $\mathcal{S}_k$ be the set of layers stipulated by Lemma 2. We first explain how to obtain the graph resulting from contracting an edge in $E(\mathcal{S}_k)$. We already explained in the preliminaries section how weights are assigned to the edges resulting from an edge contraction. Therefore, what remains to be explained is how the $\gamma()$ sets are defined for the new vertex resulting from a contraction. Basically, whenever we contract an edge $e = uv$ to obtain a new vertex w, we set $\gamma(w) := \gamma(v) \cup \gamma(u)$. Note that the resulting graph may no longer satisfy the requirements of an instance of TERRAIN COVERAGE. In particular, the maximum degree of the resulting graph may exceed Δ. Notice, however, that the only vertices in the resulting contracted graph that may have degree more than Δ are the new vertices resulting from the contractions. Notice also that the radius of each $\gamma()$ set remains upper bounded by ρ.

We start with the following two simple results:

Observation 3. *For any covering tour τ of G, we can assume that any edge in τ appears at most twice.*

The next lemma lower bounds the weight of an optimal solution by a constant fraction of the weight of the graph, and follows since the degree and the covering radius are upper bounded by constants:

Lemma 4. *Let $(G, \lambda, \Delta, \gamma, \rho, s)$ be an instance of* TERRAIN COVERAGE*, and denote by $opt(G)$ the weight of an optimal solution of G. Then $opt(G) \geq c_0 \cdot wt(G)$, where $c_0 = 1/(3\lambda \cdot \Delta^\rho)$.*

Lemma 5. *Let $(G, \lambda, \Delta, \gamma, \rho, s)$ be an instance of* TERRAIN COVERAGE*, $k \geq 1$ be an integer, and let $\mathcal{S}_k$ be the set of layers stipulated by Lemma 2. Let G' be the graph obtained from G by contracting the edges in $E(\mathcal{S}_k)$, and let C be the set of vertices in G' resulting from contracting the edges in $E(\mathcal{S}_k)$. Suppose that τ' is an optimal solution for G' subject to the condition that it contains all vertices in C. Given τ', in linear time we can compute a covering tour for G of weight at most $opt(G) + 2wt(G)/k$.*

Proof. Let τ be an optimal solution for G, and τ' be a covering tour for G' with minimum weight subject to the condition that it contains all vertices in C. First, note that τ must visit at least one vertex in each layer of $\mathcal{S}_k$. This follows from the properties of $\mathcal{S}_k$, namely that (1) each layer L_i is a separator in G, (2) $\mathcal{S}_k$ does not contain any of the first or last $\rho + 1$ layers of G, and (3) any two consecutive layers in $\mathcal{S}_k$ are separated by at least $2\rho + 1$ layers in G. This implies that if τ does not visit a layer in $\mathcal{S}_k$, then it cannot be a covering tour.

Now let τ'' be the tour in G' obtained by updating τ according to the contractions; that is, all contracted edges in $E(\mathcal{S}_k)$ are removed from τ, contracted vertices are replaced with the corresponding new vertices and their incident edges are modified accordingly, and adjacent duplicate vertices are removed. Based on how we contract edges, it is clear that τ'' covers G' and that $wt(\tau'') \leq wt(\tau)$. Moreover, since τ must contain at least one vertex from each layer in $\mathcal{S}_k$, it follows that τ'' must contain all vertices in C. Since τ' is a covering tour for G' with minimum weight subject to the constraint of containing all vertices in C, we have $wt(\tau') \leq wt(\tau'') \leq wt(\tau)$.

We define a covering tour τ_{apx} for G from τ' as follows. First, we decontract the edges in $E(\mathcal{S}_k)$ and double these edges. The tour τ_{apx} follows τ', and for each decontracted component Q that corresponds to a vertex, say w, in C, when the tour τ' visits w for the first time, we perform in τ_{apx} a depth-first traversal of Q to ensure that all vertices in Q are visited, and hence covered by τ_{apx}. Then, whenever τ' uses an edge wv, where w is a new vertex resulting from the contraction of edges in $E(\mathcal{S}_k)$, we replace the edge with the corresponding path in the component whose contraction results in w. (For instance, if xw is an edge in τ', where w results from contracting Q, let wy be the edge after xw in τ, and replace xw, wy by the path $(xu, uz_1, z_1z_2, \ldots, z_qv, vy)$, where $uz_1, z_1z_2, \ldots, z_qv$ is a path in Q. The case is treated in a similar fashion if xw is the last edge in τ', or if the edge is wx and it is the first edge in τ'.) Afterwards, we apply Observation 3 to the resulting tour; w.l.o.g., call τ_{apx} the final tour obtained. Notice that τ_{apx} is a covering tour for G. This follows since τ_{apx} visits all vertices in C, and since τ' is a covering tour for G'. Since each edge in $E(\mathcal{S}_k)$ appears at most twice, it is easy to see that $wt(\tau_{apx}) \leq wt(\tau') + 2wt(E(\mathcal{S}_k)) \leq wt(\tau) + 2wt(G)/k$. ☐

The final piece of our solution is to show the following. Given a weighted planar graph of treewidth bounded by $3(k + 2\rho + 3)$, in which every vertex has bounded degree Δ, except those belonging to the set of vertices C resulting from contracting $E(\mathcal{S}_k)$, we can compute in polynomial time an optimal covering tour for the graph subject to the condition that the tour visits all vertices in C. We define the problem formally below. (For convenience and ease of terminology, in what follows we refer to the instance graph as G.)

BOUNDED-TW-TERRAIN COVERAGE

Given: A λ-weighted plane graph G ($\lambda \geq 1$) of treewidth at most θ in which each vertex v is associated with a (connected) set of vertices $\gamma(v)$ each with radius at most ρ; and a starting vertex $s \in V(G)$ and a terminal vertex $t \in V(G)$.

Goal: Compute a tour τ of minimum weight that starts at s, visits all vertices of degree at least $\Delta + 1$ and a covering set for G, and finishes at t.

The above problem formulation assumes that the destination vertex t is given. This assumption can easily be lifted by trying each vertex as the destination t, and selecting the tour of minimum cost over all possible destination vertices, which adds an $\mathcal{O}(|V(G)|)$ factor to the running time.

Lemma 6. *Let $(G, \lambda, \Delta, \gamma, \rho, \theta, s, t)$ be an instance of* BOUNDED-TW-TERRAIN COVERAGE. *Let G' be the graph obtained from G by doubling every edge. Let $F \subseteq E(G')$ such that $G'[F]$ is connected, s and t are both incident to an odd number of edges in F, and every vertex in $V(G') \setminus \{s, t\}$ is incident to an even (possibly 0) number of edges in F. Then in linear time we can compute a tour τ of weight $wt(F)$ that visits all vertices incident to an edge in F.*

Proof. The graph induced by $E(F)$ has precisely two vertices of odd degree, s and t, and hence has an Eulerian trail between s and t. Therefore, we can compute the Eulerian s-t-tour τ of weight $wt(F)$ in linear time (Diestel 2012). ☐

Lemma 7. *Given an instance $(G, \lambda, \Delta, \gamma, \rho, \theta, s, t)$ of* BOUNDED-TW-TERRAIN COVERAGE, *there is an algorithm that outputs an optimal tour τ for the instance in time $2^{\mathcal{O}(g \cdot \theta \cdot \log \theta)} \cdot n^{\mathcal{O}(1)}$, where g is the maximum of $|\gamma(v)|$ over all vertices v of degree at most Δ and $n = |V(G)|$.*

Proof. Let G' be the graph obtained from G by doubling every edge. Herein, we assume that each edge has its unique identifier so that we can distinguish multiple edges between the same pair of vertices. From Observation 3 and Lemma 6, it suffices to find a set of edges $F \subseteq E(G')$ of minimum weight such that $G'[F]$ is connected, the vertices incident on F are a covering set for G, s and t are both incident with an odd number of edges in F, and every vertex in $V(G') \setminus \{s, t\}$ is incident with an even (possibly 0) number of edges in F. (The aforementioned statement is true because any covering tour of G corresponds to such a set of edges F and vice versa.) Note that the treewidth of G' is the same as that of G, as every tree decomposition of G is also tree decomposition of G' and vice versa. Now let $(\mathcal{V}, \mathcal{T})$ be a nice tree decomposition of G'. We note that there exists a polynomial-time algorithm that computes a tree decomposition of a graph of bounded treewidth (Biedl 2015; Bodlaender 1996), which can then be converted in linear time into a nice tree-decomposition of the same width (Kloks 1994). For convenience, let us add s and t to every bag in $\mathcal{V}$.

Given a bag X_i, we define a *configuration* w.r.t. X_i to be a tuple $(A, \bar{A}, \mathcal{P}, B, \sigma)$, where $A \subset X_i$ such that A contains s, t, and all vertices of degree at least $\Delta + 1$ in X_i, $\bar{A}$ is a subset of A, $\mathcal{P}$ is a partition of $\bar{A}$, B is a subset of $\gamma(X_i \setminus A)$ and $\sigma : \bar{A} \to \{0, 1\}$. Note that, since A contains all vertices of degree at least $\Delta + 1$ and $|X_i| \leq \theta + 1$, it is not difficult to verify that the number of configurations w.r.t. X_i is upper bounded by $2^{\mathcal{O}(g \cdot \theta \cdot \log \theta)}$.

For a bag $X_i \in \mathcal{V}$, denote by G'_i the subgraph of G' induced by the vertices in the bags of the subtree of $\mathcal{T}$ rooted at X_i. We will use dynamic programming along the nice tree decomposition $(\mathcal{V}, \mathcal{T})$ of G' to compute a table Γ_i for each bag X_i, such that for each configuration $C = (A, \bar{A}, \mathcal{P}, B, \sigma)$ w.r.t. X_i, the entry $\Gamma_i[C]$ contains a minimum-weight set of edges $F \subseteq E(G'_i)$ satisfying that:

- no vertex in $X_i \setminus \bar{A}$ is incident to an edge in F;
- for each vertex $v \in \bar{A}$, the number of edges in F incident to v modulo 2 is $\sigma[v]$;
- every vertex in $\bar{A}$ is incident to an edge in F;
- if $u \in A$ such that all edges incident to u are in $E(G'_i)$, then $u \in \bar{A}$;
- the vertices incident to F together with the set A cover the set $V(G'_i) \triangle B$; and
- $u, v \in \bar{A}$ are in the same connected component of $G'[F]$ if and only if there exists $P \in \mathcal{P}$ such that $\{u, v\} \subseteq P$.

Clearly, every feasible solution satisfies the above conditions with respect to the configuration $C_{sol} = (\{s, t\}, \{s, t\}, \{\{s, t\}\}, \emptyset, \{s \to 1, t \to 1\})$ at the root node of the tree decomposition. Hence, if we correctly compute the table Γ_r for the root node X_r of $\mathcal{T}$, then we can compute an optimal tour τ from $\Gamma_r[C_{sol}]$ using Lemma 6. Therefore, to prove the lemma, it remains to show that we can compute, for each bag $X_i \in \mathcal{V}$, the table Γ_i, based on the tables stored at the children bags of X_i. For simplicity, if for a configuration C w.r.t. X_i there does not exist a set of edges respecting C as discussed above, then we write $\Gamma_i[C] = \infty$. We extend the weight function by letting $wt(\infty) = \infty$.

Claim 1. *If X_i is a leaf, an insert or a forget node, then Γ_i can be computed in $2^{\mathcal{O}(g \cdot \theta \cdot \log \theta)} \cdot n^{\mathcal{O}(1)}$ time.*

Claim 2. *If X_i is a join node with children X_j and $X_{j'}$, then Γ_i can be computed in $2^{\mathcal{O}(g \cdot \theta \cdot \log \theta)} \cdot n^{\mathcal{O}(1)}$ time.*

Proof. The main idea is that if we have an optimal set of edges F for a configuration $C = (A, \bar{A}, \mathcal{P}, B, \sigma)$, then we can split it into the disjoint union of two edge sets, F_1 and F_2, such that F_1 and F_2 are feasible solutions for configurations C_1 and C_2 w.r.t. X_j and $X_{j'}$, respectively. This follows since X_i separates $V(G'_{j'})$ from $V(G'_j)$, and hence, any vertex in $V(G'_{j'})$ (resp. $V(G'_j)$) that is covered by a vertex in $V(G'_j)$ (resp. $V(G'_{j'})$) must be covered by a vertex in X_i. (Note that if a vertex $u \in \gamma(v)$ then $v \in \gamma(u)$; see the "Graph formulation" subsection.) Since the two sets of edges are disjoint, it follows that $\Gamma_i[C] = \Gamma_j[C_1] \cup \Gamma_{j'}[C_2]$. Now we just need to go through all possible pairs of configurations in X_j and $X_{j'}$ that could be combined to yield C.

Let $C_1 = (A_1, \bar{A}_1, \mathcal{P}_1, B_1, \sigma_1)$ and $C_2 = (A_2, \bar{A}_2, \mathcal{P}_2, B_2, \sigma_2)$ be two configurations w.r.t. X_j and $X_{j'}$, respectively. The necessary and sufficient conditions for C_1 and C_2 to combine into C are:

- $A = A_1 = A_2$ (as A represents the subset of $X_i = X_j = X_{j'}$ which will be adjacent to an edge in the final solution).
- $\bar{A} = \bar{A}_1 \cup \bar{A}_2$ (if a vertex is adjacent to an edge in $\Gamma_i[C]$, then it should be adjacent to an edge in either $\Gamma_j[C_1]$ or $\Gamma_{j'}[C_2]$).
- $\mathcal{P}$ is a join of partitions $\mathcal{P}_1$ and $\mathcal{P}_2$; that is, for all $u, v \in \bar{A}$ and $P \in \mathcal{P}$, it holds that $\{u, v\} \subseteq P$ if and only if there exit vertices $u = u_0, u_1, \ldots, u_{k-1}, u_k = v$ such that for all $\ell \in [k]$, there exists $P \in \mathcal{P}_1 \cup \mathcal{P}_2$ with $\{u_{\ell-1}, u_\ell\} \subseteq P$. Observe that if u, v are in the same component of $G'[\Gamma_i[C]]$, then there has to be a path from u to v such that every two consecutive vertices of the path are either in the same component of $G'[\Gamma_j[C_1]]$ or in the same component of $G'[\Gamma_{j'}[C_2]]$.
- $V(G_i) \triangle B \subseteq (V(G_j) \triangle B_1) \cup (V(G_{j'}) \triangle B_2)$ since a vertex that is covered by $A \cup \Gamma_i[C]$ has to be also covered in one of the children bags.
- $\sigma[u] = (\sigma_1[u] + \sigma_2[u]) \mod 2$ where $\sigma_k[u] = 0$ for $k \in \{1, 2\}$, if $u \notin \bar{A}_k$. That is, the parity of the edges incident to u in $\Gamma_i[C]$ is the sum of the parities of edges incident to u in the children bags.

It follows that we can compute $\Gamma_i[C]$ by going through all pairs of C_1 and C_2 satisfying the above conditions and choosing the pair with the minimum sum of weights, which can be done in $2^{\mathcal{O}(g \cdot \theta \cdot \log \theta)} \cdot n^{\mathcal{O}(1)}$ time. □

This completes the proof of Lemma 7. □

Table 1: Grid-based heuristic algorithm.

terrain type	camera range	# dominators / # vertices	tour wt / total wt	tour wt / MST wt	max edge wt / min edge wt
10	4	0.3774	0.6317	0.9735	2.3893
10	12	0.1242	0.3267	0.5034	2.3893
40	4	0.3750	0.6257	0.9678	2.6736
40	12	0.1263	0.3269	0.5056	2.6736
80	4	0.4246	0.6878	1.0904	3.8749
80	12	0.1417	0.3518	0.5576	3.8749

Table 2: Global heuristic algorithm.

terrain type	camera range	# dominators / # vertices	tour wt / total wt	tour wt / MST wt	max edge wt / min edge wt
10	4	0.3445	0.6079	0.9368	2.3893
10	12	0.1033	0.3077	0.4742	2.3893
40	4	0.3413	0.6031	0.9328	2.6736
40	12	0.1058	0.3093	0.4783	2.6736
80	4	0.3944	0.6662	1.0559	3.8749
80	12	0.1211	0.3336	0.5289	3.8749

Theorem 8. TERRAIN COVERAGE *admits a PTAS.*

Proof. Let $(G, \lambda, \Delta, \gamma, \rho, s)$ be an instance of TERRAIN COVERAGE and $\epsilon > 0$. By Lemma 4, $opt(G) \geq c_0 \cdot wt(G)$ for some constant $c_0 \leq 1$. Choose $k \in \mathbb{N}$ such that $k > 2/(c_0\epsilon)$. Let $G = (L_1, \ldots, L_r)$ be a layer decomposition of G, where $r \geq 1$, and apply Lemma 2 to find in linear time a set $\mathcal{S}_k$ of layers satisfying: (1) $wt(E(\mathcal{S}_k)) \leq wt(G)/k$; (2) $tw(G/E(\mathcal{S}_k)) \leq 3(k + 2\rho + 3)$; (3) $\mathcal{S}_k$ does not contain any of the first $\rho + 1$ or last $\rho + 1$ layers in G; and (4) any two consecutive layers in $\mathcal{S}_k$ are separated (in G) by at least $2\rho + 1$ layers. Now let G' be the graph obtained from G by contracting the edges in $E(\mathcal{S}_k)$, and let C be the set of vertices in G' resulting from contracting the edges in $E(\mathcal{S}_k)$. From the above, we have $tw(G') \leq 3(k + 2\rho + 3)$. By Lemma 7, and since both k and ρ are constants, in polynomial time, we can compute an optimal covering tour τ' for G' that contains all vertices in C. By Lemma 5, in linear time we can compute a tour τ_{apx} from τ' satisfying that $wt(\tau_{apx}) \leq opt(G) + 2wt(G)/k$. Since $opt(G) \geq c_0 \cdot wt(G)$, it follows that $wt(\tau_{apx}) \leq opt(G) + 2opt(G)/(c_0 k) \leq opt(G) + \epsilon \cdot opt(G) = (1 + \epsilon)opt(G)$. Therefore, for any given $\epsilon > 0$, we have a polynomial time algorithm that computes a solution for a given instance of TERRAIN COVERAGE whose weight is within ratio $1 + \epsilon$ from that of an optimal solution of the instance. The theorem follows. □

Heuristics and Empirical Results

We report on the implementation of two heuristic approximation algorithms for TERRAIN COVERAGE.

Heuristic algorithms. While the PTAS developed in the previous section settles the approximation properties of TERRAIN COVERAGE (given that no FPTAS exists), it is impractical to implement due to its technicality and complexity (resulting in a high-degree polynomial). Therefore, we implemented two constant-ratio approximation algorithms for that

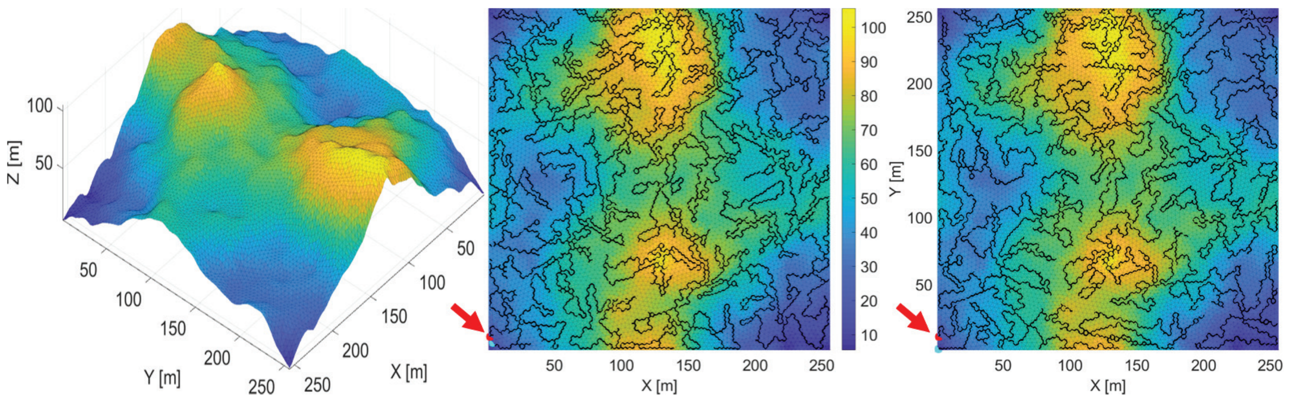

Figure 1: Left: A $256m \times 256m$ random terrain considered in this study with $80m$ elevation range; middle: the path generated using a grid; and right: the path generated without using a grid. The red arrow shows the starting point.

are similar to the folklore greedy algorithm for the DOMINATING SET, whose greedy step chooses a vertex that dominates the maximum number of vertices.

The first algorithm we implemented starts by superimposing a grid on the plane graph G, whose cell-size is large enough so that each cell is guaranteed to contain at least one vertex from any covering set; the cell size is a constant that is determined based on the camera range. Then a greedy algorithm is applied to each grid cell to chose a covering set of the cell; this greedy algorithm is similar to the folklore greedy algorithm for DOMINATING SET. For a given cell C, until all vertices in C are covered, the algorithm repeats the following greedy step: choose a vertex v in C that (among all unchosen vertices in C) covers the maximum number of vertices in C that have not been covered yet; mark v as chosen, and mark all vertices covered by v *in* G as covered. The algorithm then proceeds to a new cell, taking into account that some vertices in the new cell are already covered. When a covering set for each cell has been computed, denote by κ their union. The algorithm computes a minimum spanning tree T of the subgraph of $\mathcal{E}$ (the complete graph whose vertex-set is $V(G)$) induced by the vertices in κ. Then a tour τ is computed based on a depth-first traversal of T, similarly to how the ratio-2 approximation algorithm for the metric TSP is obtained, albeit without the need to return to the starting vertex in the tour. Finally, for each two consecutive vertices u, v on the tour, the direct edge uv in $\mathcal{E}$ is replaced by the path in G formed by the vertices whose corresponding triangles in the mesh are intersected by the straight-line segment between u and v.

The second algorithm is similar to the first, but instead of superimposing a grid and computing a covering set for each grid-cell, a covering set κ for the whole graph is computed by applying the same greedy algorithm to G. The rest is the same: a minimum spanning tree T of κ is computed and a tour based on a traversal of T is obtained.

Since G is a bounded-degree λ-weighted plane graph, and since ρ is a constant, one can easily show that both algorithms are constant-ratio approximation algorithms. This mainly follows from the observation that an optimal solution contains $\Omega(n)$ vertices and edges, and hence its weight is a constant fraction of the weight of G. Next, we evaluate the performance of both algorithms.

Experimental evaluation. To test the two heuristic algorithms, we generated random landscapes using the fractal landscape generation function in (Kaya 2013). The function takes three arguments: an argument that defines the size of the terrain (edge length), an argument that defines the smoothing parameter, and an argument that defines the elevation range. We generated 10 random landscapes for each range between 10 and 80, in multiples of 10. Tables 1 and 2 show the simulation results reported only for the three terrain ranges $10, 40, 80$ and two camera ranges $4, 12$, where the data was aggregated over the 10 terrains for each range. Table 1 shows the results for the heuristic algorithm based on a superimposed grid, and Table 2 shows the results for the other algorithm. The first column in the tables is the terrain range (difference between maximum and minimum elevation). The second column shows the camera range. The third column shows the ratio of the number of covering vertices to the total number of vertices. The fourth column is the ratio between the weight of the solution to the total weight of the edges in the graph. The fifth column is the ratio between the weight of the solution and that of a minimum spanning tree of the graph. The last column shows the ratio of the maximum weight edge to that of the minimum weight edge.

For Table 1, we notice that the ratio between the number of covering vertices and the total number of vertices is roughly $3/8$ for camera range 4 and elevation range 10, and it decreases to roughly $1/8$ for camera range 12. The ratio is slightly worse for higher elevation range. The ratio of the weight of the solution to the total edge weight of the graph for camera range 4 and elevation 10 is roughly 2/3, and it improves to roughly 1/3 for camera range 12. This ratio gets slightly worse as the elevation range increases. Note that this ratio is very conservative, as it is computed based on the total edge weight of the graph. The results in Table 2 are slightly better. This is justifiable as the tour was computed based on a minimum spanning tree (MST) for the whole graph, rather than for the individual grid cells.

Figure 1 shows a randomly-generated terrain and the paths generated by the two algorithms when applied to this terrain.

Conclusion

In this paper, we studied the complexity of the TERRAIN COVERAGE problem, and designed a PTAS and heuristic algorithms for it. The PTAS we designed relies heavily on the connectivity assumption about the $\gamma()$ sets, which results from the assumption that the terrain is simply connected; this is a property that may be violated if the terrain contains topographical barriers (*e.g.*, holes). An interesting problem that ensues from our work is to investigate if we can still obtain PTAS without the connectivity assumption about the $\gamma()$ sets, as this would enable modeling terrains with more versatile topographical properties.

References

Acar, E.; Choset, H.; Rizzi, A.; Atkar, P.; and Hull, D. 2002. Morse decompositions for coverage tasks. *I. J. Robotics Res.* 21(4):331–344.

Arkin, E., and Hassin, R. 1994. Approximation algorithms for the geometric covering salesman problem. *Discrete Applied Mathematics* 55(3):197–218.

Arkin, E.; Bender, M.; Demaine, E.; Fekete, S.; Mitchell, J.; and Sethia, S. 2005. Optimal covering tours with turn costs. *SIAM J. Comput.* 35(3):531–566.

Arkin, E.; Bender, M.; Mitchell, J.; and Polishchuk, V. 2011. The snowblower problem. *Comput. Geom.* 44(8):370–384.

Arkin, E.; Fekete, S.; and Mitchell, J. 2000. Approximation algorithms for lawn mowing and milling. *Comput. Geom.* 17(1-2):25–50.

Arora, S. 1998. Polynomial time approximation schemes for Euclidean traveling salesman and other geometric problems. *J. ACM* 45(5):753–782.

Atkar, P.; Greenfield, A.; Conner, D.; Choset, H.; and Rizzi, A. 2005. Uniform coverage of automotive surface patches. *I. J. Robotics Res.* 24(11):883–898.

Baker, B. 1994. Approximation algorithms for NP-complete problems on planar graphs. *J. ACM* 41(1):153–180.

Biedl, T. 2015. On triangulating k-outerplanar graphs. *Discrete Applied Mathematics* 181:275 – 279.

Bodlaender, H. L. 1996. A linear-time algorithm for finding tree-decompositions of small treewidth. *SIAM J. Comput.* 25(6):1305–1317.

Cao, Z.; Huang, Y.; and Hall, E. 1988. Region filling operations with random obstacle avoidance for mobile robots. *J. Field Robotics* 5(2):87–102.

Choset, H.; Lynch, K.; Hutchinson, S.; Kantor, G.; Burgard, W.; Kavraki, L.; and Thrun, S. 2005. *Principles of Robot Motion: Theory, Algorithms, and Implementations*. MIT Press.

Choset, H. 2001. Coverage for robotics – a survey of recent results. *Annals of Mathematics and Artificial Intelligence* 31(1):113–126.

Diestel, R. 2012. *Graph Theory, 4th Edition*. Springer.

Englot, B., and Hover, F. 2012. Sampling-based coverage path planning for inspection of complex structures. In *ICAPS*. AAAI.

Fekete, S.; Mitchell, J.; and Schmidt, C. 2012. Minimum covering with travel cost. *J. Comb. Optim.* 24(1):32–51.

Gabriele, Y., and Rimon, E. 2002. Spiral-STC: An on-line coverage algorithm of grid environments by a mobile robot. In *ICRA*, 954–960. IEEE.

Gage, D. 1994. Randomized search strategies with imperfect sensors. In *Proc. SPIE*, volume 2058, 270–279.

Galceran, E., and Carreras, M. 2013. A survey on coverage path planning for robotics. *Robot. Auton. Syst.* 61(12):1258–1276.

Garey, M., and Johnson, D. 1979. *Computers and Intractability*. W.H. Freeman.

Hert, S.; Tiwari, S.; and Lumelsky, V. 1996. A terrain-covering algorithm for an AUV. *Autonomous Robots* 3(2):91–119.

Kaya, H. 2013. Fractal landscape generation with diamond-square algorithm. Available at: https://www.mathworks.com/matlabcentral/.

Klein, P. 2008. A linear-time approximation scheme for TSP in undirected planar graphs with edge-weights. *SIAM J. Comput.* 37(6):1926–1952.

Kloks, T. 1994. *Treewidth, Computations and Approximations*, volume 842 of *LNCS*. Springer.

Latombe, J. C. 1991. *Robot Motion Planning*. Norwell, MA, USA: Kluwer Academic Publishers.

Lee, T.-S.; Choi, J.-S.; Lee, J.-H.; and Lee, B.-H. 2009. 3-D terrain covering and map building algorithm for an auv. In *IEEE/RSJ*, 4420–4425.

Lee, T.; Baek, S.; Choi, Y.; and Oh, S. 2011. Smooth coverage path planning and control of mobile robots based on high-resolution grid map representation. *Robotics and Autonomous Systems* 59(10):801–812.

Li, F., and Klette, R. 2008. An approximate algorithm for solving the watchman route problem. In *RobVis*, volume 4931 of *LNCS*, 189–206. Springer.

Mitchell, J. 1999. Guillotine subdivisions approximate polygonal subdivisions: A simple polynomial-time approximation scheme for geometric TSP, k-MST, and related problems. *SIAM J. Comput.* 28(4):1298–1309.

Ntafos, S. 1991. Watchman routes under limited visibility. *Comput. Geom.* 1:149–170.

Persson, P. O. 2004. DISTMESH-a simple mesh generator in matlab. http://persson.berkeley.edu/distmesh/.

Robertson, N., and Seymour, P. 1991. Graph minors. X. obstructions to tree-decomposition. *Journal of Combinatorial Theory, Series B* 52(2):153 – 190.

Xu, A.; Viriyasuthee, C.; and Rekleitis, I. 2011. Optimal complete terrain coverage using an unmanned aerial vehicle. In *ICRA*, 2513–2519. IEEE.

Yang, L.; Qi, J.; Song, D.; Xiao, J.; Han, J.; and Xia, Y. 2016. Survey of robot 3D path planning algorithms. *Journal of Control Science and Engineering* 2016(5):1–22.