IEEE Robotics and Automation Letters (RAL) paper presented at the
2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
October 25-29, 2020, Las Vegas, NV, USA (Virtual)

Visual Coverage Path Planning for Urban Environments

Cheng Peng[1] and Volkan Isler[2]

***Abstract*— View planning for visual coverage is a fundamental robotics problem. Coverage for small objects (e.g. for inspection) or small scale indoor scenes have been studied extensively. However, view planning to cover a large scale urban environment remains challenging. Algorithms that can scale up to the size of such environments while providing performance guarantees are missing.**

In this paper, we model urban environments as a set of surfaces with k distinct surface normals whose viewing cones must be visited by a robot. We model the resulting coverage problem as a novel variant of Traveling Salesman Problem with Neighborhoods (TSPN). The neighborhoods are defined as cones, which constrain the path coverage quality. We present a polynomial time algorithm which admits an approximation factor of $O(\frac{k}{\tan(\alpha)}\max[L_B, W_B, H_B])$, where α is the maximum viewing angle, and L_B, H_B, W_B are respectively the length, width, and height of a minimum enclosing box of a city scene B. In addition to the analytical upper bounds, we show in simulations that our method outperforms three baseline methods in both trajectory length and run-time. We also demonstrate our method and evaluate the coverage quality of a city containing more than 70 buildings in a photo-realistic rendering software.

I. Introduction

Capturing detailed views of a large scale city environment is crucial for many applications such as 3D reconstruction, virtual tourism, search and rescue, environmental monitoring, and disaster response. In the case of disaster response, quickly assessing the structural damage and finding possible passage ways are paramount to the rescue team. For a complex shaped building or an entire city, it is also important to optimize the number of viewpoints to efficiently represent the 3D geometry and textures without sacrificing image quality. Therefore, an efficient and scalable data collection method is desirable for large scale coverage and reconstruction tasks.

View planning is a fundamental robotics problem. Earlier work [1]–[5] mainly use manipulators or ground vehicles to cover small objects or to explore indoor environments. With the availability of drones equipped with cameras, it is now possible to take advantage of their versatility to cover large 3D spaces. Recently, many works [6]–[12] address view selection and trajectory planning problems for larger scenes. However the current results are restricted to planar regions (e.g. farms) or a single building. Extensions to large scale environments are non-trivial as they need to account for computational complexity, optimality, and memory efficiency. Using a set of small surface patches to model the geometry of an entire city [8], [9], [11], [12] would be very expensive and memory intensive. When optimizing for discrete viewpoints, methods [11]–[14] that employ linear integer programming or submodular function maximization would also be computationally expensive. Furthermore, heuristic strategies are unable to provide guarantees on the solution quality, which is essential in large scale environments [15].

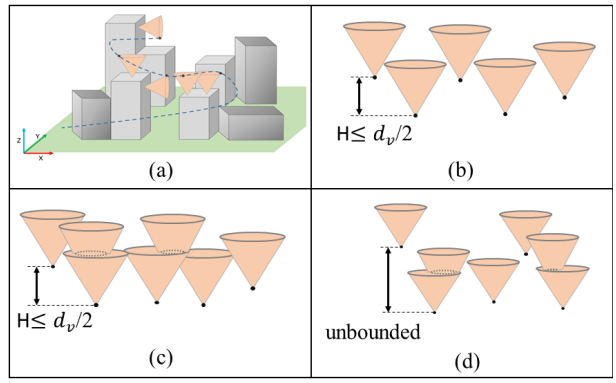


Fig. 1: (a) The city scene is a set of building with surface normals facing different directions. The inverted cones on the building surfaces illustrate a set of good viewing quality regions for the points located at the cone apexes. When a camera pose is within the cone, the point on the apex is assumed to be "well-covered". We generalize this problem into a variant of Traveling Salesman Problem with Neighborhoods and decompose it into the following three subproblems. (b) Problem 1: Disjoint set of cones with constrained apex locations. (c) Problem 2: Non-disjoint set of cones with constrained apex locations. (d) Problem 3: Non-disjoint set of cones with varying apex locations.

To address those challenges when covering city-scale environments, we present a method that splits a city into k groups of surface normals ($k = 5$ for a Manhattan world with axis-aligned buildings). For each group of surfaces, their normals are in the same direction. Our method converts the coverage problem to a novel variant of cone-TSPN [16] problem while providing an approximation factor of $O(\frac{k}{\tan(\alpha)}\max([L_B, H_B, W_B]))$ for the final trajectory, where α is the maximum good quality viewing angle per point and L_B, H_B, W_B are respectively the length, width, and height of the minimum enclosing box of a city scene B. Note that our method allows the city to contain at most k distinct surface normals for all surfaces. Therefore, each building does not have to be rectilinear, which is the case for Cheng et al. [15]. Using this solution as a subroutine, our algorithm is able to achieve a polynomial run-time with a constant

*This work was supported by NSF grant number 1849107.

Cheng Peng[1] and Volkan Isler[2] are with the Department of Computer Science and Engineering, University of Minnesota, 200 Union Street SE, Minneapolis, MN, US, 55108 peng0175, isler@umn.edu

memory space, which significantly improves upon previous methods [10]–[12], [15]. In summary, we study the problem of visually covering a city in order to acquire high quality views and make the following contributions:

- We present a polynomial time $O(\frac{1}{\tan(\alpha)}\max([L_B, H_B, W_B]))$ approximation algorithm for the generalized cone-TSPN problem for the case where the apex locations of the cones can lie on multiple planes with the same orientation.
- We use this result to present a method to cover a large scale city subject to a geometric view quality constraint and present a bound which limits the deviation of our performance from the optimal performance,
- We demonstrate our method and evaluate the coverage quality of a city containing more than 70 buildings in a photo-realistic rendering software.

II. Related Work

In this section, we first review related work on TSP/TSPN, followed by an overview of work in coverage planning.

A. Traveling Salesman Problem

Traveling salesman problem (TSP) is a classical optimization problem. The objective is to find a shortest tour that visits a given set of points. The Euclidean version of the problem is shown to be NP-Complete [17]. In TSP with neighborhoods (TSPN), the agent is asked to visit a given set of regions instead of points, which is APX-Hard even in the Euclidean space [17]–[19]. For neighborhoods of known or bounded geometry, better approximation factors can be found. Mata and Mitchell [20] provide an $O(\log n)$ approximation for n connected neighborhoods in the plane. Chan and Elbassioni [21] give a quasi polynomial time approximation scheme (QPTAS) for weakly disjoint α fat regions. Bodlaender et al. [22] give a PTAS for TSPN with disjoint fat regions of similar sizes in $\mathbb{R}^d$ for a constant d. Dumitrescu and Toth [23] give an $O(1)$ approximation factor for a set of hyperplanes and unit balls in $\mathbb{R}^d$. Dumitrescu and Toth [24] also achieve a constant factor approximation for disks of varying size on a plane.

More recently, a specific neighborhood geometry of cones (cone-TSPN) has been introduced [16]. Each cone represents the visibility/high-quality region of a ground point using a camera sensor. For a set of cones with the same apex angle, they provide an approximation factor of $O(1+\log(h_{max}/h_{min}))$, where h_{max} and h_{min} is the maximum and minimum cone height. Stefas et al. [25] extend the setup to varying cone orientations and give an approximation factor of $O(\frac{1+\tan\alpha}{1-\tan\varepsilon\tan\alpha}(1+\log(h_{max}/h_{min})))$, where ε is the cone orientation and α is the apex angle.

Both works [16], [25] for TSPN of cones target planar regions, which means all cones originate from the same ground plane. In this paper, the coverage objective is for 3D urban environments, where the surfaces can be at different locations. Therefore, we generalize the problem setup to cones with varying apex locations. We name this problem as "generalized cone-TSPN with varying apex locations" or GC-TSPN in short. This generalization increase the dimensionality of the original problem ($2.5D$ to $3D$) with additional complexity coming from varying cone orientations. Previous methods [16], [21], [23] are unable to address GC-TSPN.

Papadopoulos et al. [26] proposed a sampling based method to cover every point at least once with dynamic constraints. Similarly, works from [27], [28] employs random inspection tree algorithm (RITA) to plan trajectories to cover the region. However, sampling based methods suffer from sampling poses at narrow spaces. It may take long [27], [28] to find path through smaller regions since they have smaller probabilities relative to the entire space.

B. Coverage Path Planning

Coverage path planning (CPP) has been an active research topic for decades. Many works [6]–[9] in CPP focus on planar regions with fixed sensor footprints. Some works [29], [30] decompose the free region into modular shapes, which is referred to as "cellular decomposition". A zig-zag or spiral motion is then employed to cover the free cells [29]–[31]. In the presence of obstacles, an adjacency graph [29] is built from the decomposition such that each node represents a free region to be covered. For a larger planar area, recent methods [32], [33] also propose to revisit the area multiple times efficiently in the presence of energy constrains.

In 3D coverage path planning, instead of covering all free space, most works focus on covering the surface of a 2D object or a scene. For an urban environment, Cheng et al. [15] propose a time-optimal method that executes a spiral motion per building for a complete coverage. However, viewing quality is not specified and the coverage is performed on each building separately. In comparison, our method proposes a coverage quality model using cones, which generalizes the "frontal view" [10]–[12] preference for each surface. For more recent works on aerial 3D coverage, Bircher et al. [13] propose a two-step optimization framework to plan for an efficient trajectory. They first compute a set of viewpoints for a complete coverage, which is formulated as an art gallery problem (AGP) [34]. Then, a tour is computed that visits all viewpoints using a method from Lin et al. [35].

III. Problem Formulation

We assume that a scene contains a ground plane which coincides with the $Z_W = 0$ plane of a world frame $\{W\}$. We model a city as a set of buildings $B = \{b_1, b_2, ..., b_B\}$ as shown in Figure 1 (a). We make the assumption that the bottom surfaces of all buildings lie on the common ground plane. Denote a set of surfaces of a building as $b_i = \{s_1^i, s_2^i, s_3^i, ...\}$ excluding the bottom surface. Given the surface normal $norm(s_1^i)$ of surface s_1^i, we further assume that the surface normals of all surfaces belong to a subset of k directions such that $norm(s_j^i) \in \{N_1, N_2, ..., N_k\},\ \forall i \in B, \forall j \in K$, where $\{N_1, N_2, ..., N_k\}$ are k distinct normal directions. For example, if all boxes are axis-aligned, then there are a total of 5 distinct surface normals.

We use a drone equipped with a pin-hole camera with a 90° field-of-view to cover the city. For a point p on a

surface $s_j \in b$, there exists a good quality viewing region denoted as $\tau(p,[d_v,\alpha])$, where d_v is the maximum viewing distance measured along the surface normal and α is the maximum viewing angle measured from the surface normal $norm(s_j)$. The good-quality region $\tau(p,[d_v,\alpha])$ becomes an inverted cone originated at the point with apex angle α and cone height d_v as shown in Figure 1 (a), which is also used in works from [16], [25]. We define an indicator function $g(s_j,J)=1$ for surface s_j such that there always exist a view in the trajectory J that intersects the quality viewing region for all points on the surface $J \cap \tau(p,[d_v,\alpha]) \neq 0,\ \forall p \in s_j$.

The objective is to find a shortest tour to cover all surfaces of all buildings in a city. Given a scene $B=\{b_1,b_2,...,b_B\}$ with a set of buildings b_i and their corresponding surfaces $b_i=\{s_1^i,s_2^i,s_3^i,...\}$, we would like to find a shortest trajectory J such that $g(s_j^i,J)=1$ for $\forall s_j^i \in B$.

IV. Technical Approach

Our approach represents viewing regions of points to be covered as cones and computes a path which intersects all cones. This formulation naturally leads to a variant of TSP with Neighborhoods (TSPN) where the neighborhoods are cones. Variants of Cone-TSPN have been studied in the literature [16], [25]. However, all existing solutions require cones to have their apexes on a common plane. This assumption is violated for the example when considering the top planes of buildings with non-uniform heights. Therefore, our main technical contribution is to generalize Cone-TSPN to various cone apex locations and orientations. (Section V).

At a high level, our approach is as follows. For a given scene B, we cluster all the surfaces in the scene into k subgroups. For a subgroup $i=[1,2,3,...,k]$, the surfaces share a same surface normal N_i. For each subgroup i, we formulate the coverage path planning problem as GC-TSPN and propose our Algorithm 2 to find an efficient trajectory. The final coverage strategy for the entire scene B then combines k trajectories from all subgroups.

To solve GC-TSPN, we further divide GC-TSPN into 3 subproblems as shown in Figure 1 (b)(c)(d). For the first problem "Disjoint set of cones with constrained apex locations" (Problem 1) shown in Figure 1(b), the constrained apex location refers height differences less than $d_v/2$ so that there always exists a plane orthogonal to the group normal N_i that can intersect all cones. Using methods from [24], we can find a constant approximation factor trajectory for disjoint disks of varying sizes on the plane that intersects all cones. Next, we generalize the special case to non-disjoint set of cones (Problem 2) as shown in Figure 1(c). To solve this problem, we generate a maximal independent set (MIS) of all cones. By adding detours to the trajectory to the previous problem, the resulting trajectory visit the neighborhoods of the MIS and admits an $O(\frac{1}{\tan(\alpha)})$ approximation factor. At last, we generalize the problem to non-disjoint cones with varying apex locations (Problem 3). By dividing the unbounded height differences into subgroups, we can solve this problem by solving a set of Problem 2. To cover all surfaces in a city, we execute our method for each group with N_k surface normals and combine the k trajectories into a final trajectory. The final trajectory admits an $O(\frac{k}{\tan(\alpha)}\max([L_B,H_B,W_B]))$ approximation.

V. Generalized cone-TSPN

In this section, we convert a coverage path planning problem of a set of surfaces with the same surface normals into GC-TSPN, which is a variant of cone-TSPN [16], [25]. To approach the general problem, we first solve Problem 1 of disjoint cones with constrained apex locations.

A. Problem 1: Disjoint cones of constrained apex locations

Denote a set of cones defined as $C=\{c_1,c_2,...,c_n\}$ with the same apex angle α and cone height d_v. The bisector of each cone is the same normal vector N. Without loss of generality, we assume cone bisector faces upward ($N=[0,0,1]^T$) and the minimum height cone is always located at the ground plane ($\min(H(C))=0$). Denote the apex height of a cone as $H(c_i) \in H(C)$. The constrained apex location is then defined as $\max(H(C))-\min(H(C)) \leq d_v/2$ as shown in Figure 1(b). It means that we can always find an intersection plane parallel to the ground plane such that all cones will be intersected and the resulting disk radius from the intersection plane is non-zero.

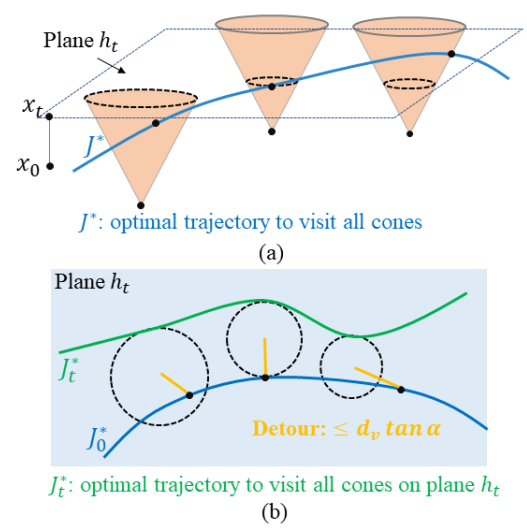


Fig. 2: (a) Optimal trajectory J^* that covers the cones at different location with starting location x_0 and $x_t = x_0 + [0,0,h_t]^T$. (b) The optimal trajectory J_t^* that covers the cones at plane h_t with maximum detour length for each circle $2d_v\tan(\alpha)$. J_0^* is the projection of J^* on plane h_t.

Lemma 5.1: The Minkowski sum $M(J^*,1.5d_v,4d_v\tan(\alpha))$ of the optimal tour J^* when sweeps a cylinder of height $1.5d_v$ and diameter $4d_v\tan(\alpha)$ is guaranteed to enclose all cones that the optimal tour visits.

$$M(J^*,1.5d_v,4d_v\tan(\alpha)) \leq 6d_v^2\tan(\alpha)|J^*| \quad (1)$$

Proof: When projecting the optimal tour J^* to the ground plane denoted as J_0^*, the apex location projected on to the ground plane will be at most $d_v\tan(\alpha)$ away from J_0^*.

Since the optimal tour J^* is guaranteed to visit all cones in C, we can sweep a cylinder of height $1.5d_v$ with diameter $4d_v\tan(\alpha)$ to enclose all cones. Given the height and the diameter of the cylinder, the maximum sweeping volume along the optimal tour J^* is then $(1.5d_v)(4d_v\tan(\alpha))|J^*| = 6d_v^2\tan(\alpha)|J^*|$. ■

Lemma 5.2: Given the Minkowski sum $M(J^*, 1.5d_v, 4d_v\tan(\alpha))$ of the optimal tour J^* that visits a disjoint set of cones C, the total number of all cones n is upper bounded as:

$$n \leq \frac{18|J^*|}{\pi d_v \tan(\alpha)} \tag{2}$$

Proof: Since all cones are disjoint, the Minkowski sum must also enclose all disjoint cones.

$$M(J^*, 1.5d_v, 4d_v\tan(\alpha)) \geq n\frac{\pi}{3}d_v^3\tan^2(\alpha)$$

By substituting results from Eq 1, we can get the following relationship.

$$6d_v^2\tan(\alpha)|J^*| \geq n\frac{\pi}{3}d_v^3\tan^2(\alpha)$$
$$n \leq \frac{18|J^*|}{\pi d_v\tan(\alpha)}$$

■

Algorithm 1 Slice-and-Visit

Input: $x_0 \in \mathbf{R}^3$, $C = \{c_1, c_2, ..., c_n\}$, d_v
Output: J_t

1: Set $h_t = d_v$
2: Set $C_{h_t} = h_t \cap C$, which is a set of circles on plane h_t
3: Set $x_t = x_0 + [0, 0, h_t]^T$
4: Plan a tour J_t for C_{h_t} starting at x_t using method from [24].

If we choose a specific height h_t that intersects all cones at non-apex location, we can use the algorithm in [24] to visit disks (non-zero radius) with an $O(1)$ approximation. Denote J_t as a tour that visits all the cones at height h_t using algorithm in [24]. The constant in $O(1)$ approximation is denoted as $B(J_t)$. Here, we present our next Lemma that defines the relationship between the trajectory J_t and the optimal trajectory J^* that visits all cones C.

Lemma 5.3: Denote h_t as a specific plane orthogonal to the cone bisector that intersects all cones in C at non-apex location. There exists a trajectory J_t that visits all the disjoint cones on the plane h_t, where the trajectory length comparing to optimal trajectory is bounded as follows.

$$\frac{|J_t|}{B(J_t)} - |J^*| \leq 2nd_v\tan\alpha + 2d_v \tag{3}$$

Proof: We know the optimal trajectory length that visits all disjoint disks of varying size is $|J_t^*| = \frac{|J_t|}{B(J_t)}$. The projection of the optimal trajectory J^* on the plane h_t is denoted J_0^*. Note that J_0^* may not intersect all disks on the plane h_t as shown in Figure 2 (b). By adding a detour of at most $d_v\tan(\alpha)$ per disk, which is the radius of a cone (Figure 2 (b)), trajectory J_0^* is guaranteed to visit the center of a disk. Therefore, the additional tour should visit n disk centers and come back to J_0^*, which results in a detour length of at most $(2d_v\tan(\alpha)) * n$. There is also an additional cost to move up to a point x_t on plane h_t from the starting location x_0, which results in an additional trip of at most $2h_t \leq 2d_v$.

$$\frac{|J_t|}{B(J_t)} \leq |J_0^*| + 2nd_v\tan(\alpha) + 2h_t$$
$$\leq |J^*| + 2nd_v\tan(\alpha) + 2d_v$$

■

Using Lemma 5.1, 5.2, and 5.3, we can bound the length of trajectory J_t that visits all cones at height h_t.

Theorem 5.4: Denote a set of disjoint cones as $C = \{c_1, c_2, ..., c_n\}$ with the same apex angle α and cone height d_v. The apex location of the cones are constrained to be $\max(H(C)) - \min(H(C)) \leq d_v/2$. Our algorithm Slice-and-Visit(Algorithm 1) that visits the cone at height $h_t = d_v$ produce a trajectory J_t with bound

$$|J_t| \leq B(J_t)(2 + \frac{36}{\pi})|J^*| \tag{4}$$

Proof: Since $\max(H(C)) - \min(H(C)) \leq d_v/2$ and $\min(H(V)) = 0$, the plane at height d_v is guaranteed to slice all cones at non-apex locations (disks with non-zero radius). We can substitute the upper bound for n in Eq 2 to Eq 3.

$$\frac{|J_t|}{B(J_t)} - |J^*| \leq 2nd_v\tan(\alpha) + 2d_v$$
$$\frac{|J_t|}{B(J_t)} - |J^*| \leq \frac{36|J^*|}{\pi} + 2d_v$$

Since the optimal trajectory have to travel at most $d_v/2$ to visit the apex of the highest cone from the ground, a close tour doubles this value and can be bounded as $d_v \leq |J^*|$.

$$\frac{|J_t|}{B(J_t)} - |J^*| \leq \frac{36|J^*|}{\pi} + 2d_v$$
$$\frac{|J_t|}{B(J_t)} - |J^*| \leq \frac{36|J^*|}{\pi} + 2|J^*|$$
$$|J_t| \leq B(J_t)(3 + \frac{36}{\pi})|J^*|$$

■

B. Problem 2: Non-disjoint cones of constrained apex heights

To visit a set of non-disjoint cones, we can borrow the idea from [24] that first finds a maximal independent set of the cones and modifies the trajectory to visit the neighborhoods.

Theorem 5.5: Denote a set of non-disjoint cones as $C = \{c_1, c_2, ..., c_n\}$ with the same apex angle α and height d_v. The apex location of the cones are constrained to be $\max(H(C)) - \min(H(C)) \leq d_v/2$. Our algorithm General-Slice-and-Visit (Algorithm 2) generates a trajectory J_t^{non} with an approximation factor of $O(\frac{1}{\tan(\alpha)})$.

$$|J_t^{non}| \leq B(J_t^{non})(\frac{36}{\pi\tan(\alpha)} + 50)|J^*| \tag{5}$$

Before proving this theorem, we first need to show that the detour around the edges of all cones in C_{MIS} can also

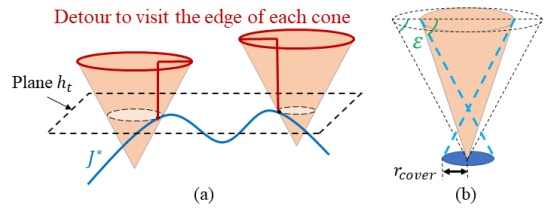


Fig. 3: (a) A detour that visits the neighborhoods of a set of disjoint cones is generated by moving vertical upward and circulating the edge (red ellipse) of each cone. (b) Converting coverage of a single point to a circular patch of radius r_{cover} by shrinking the cone's apex angle.

visit their neighborhoods. We define the edge of a cone as the perimeter on the top-most surface shown as red circles in Figure 3 (a).

Lemma 5.6: There exists a maximal independent set C_{MIS} of C such that a trajectory that contains the edge of each cone in C_{MIS} is guaranteed to intersect all cones in C.

Proof: We prove this lemma by contradiction. Suppose there exists a cone $c_y \notin C_{MIS}$ and c_y does not intersect any other cones in C_{MIS}. The first case is when c_y does not intersect any other cones in C. It is clear that the algorithm will find $c_y \in C_{MIS}$, which contradicts the assumption. The second case is when c_y intersects another cone c_x through the edge of c_y, which is equivalent to $H(c_x) > H(c_y)$ and $c_x \cap c_y \neq 0$. Given that our algorithm finds C_{MIS} from the lowest height, it means that $c_y \in C_{MIS}$, which contradicts the assumption. The third case is then c_y intersects another cone c_x through the edge of c_x. We can use the proof for the second case by switching c_x and c_y. ■

For each cone c_i that the trajectory visit at height h_t, we add an additional detour that visits the edge of c_i as shown in Figure 3. Therefore, we can bound the trajectory length with the following proof for Theorem 5.5. *Proof:* The additional trajectory cost is at most $n[2d_v + 2d_v\tan(\alpha) + 2\pi d_v\tan(\alpha)]$, where d_v for going vertically up to the cone surface, $d_v\tan(\alpha)$ is the distance to the edge of the cone, and $2\pi d_v\tan(\alpha)$ is the edge circle circumference. Using similar deriving process in Theorem 5.5, we can get the following.

$$\frac{|J_t^{non}|}{B(J_t^{non})} - |J^*| \leq n[2d_v + 2d_v\tan(\alpha) + 2\pi d_v\tan(\alpha)] + 2d_v$$
$$\frac{|J_t^{non}|}{B(J_t^{non})} - |J^*| \leq d_v[2 + 2\tan(\alpha) + 2\pi\tan(\alpha)]\frac{18|J^*|}{\pi d_v\tan(\alpha)} + 2|J^*|$$
$$|J_t^{non}| \leq B(J_t^{non})(\frac{36}{\pi\tan(\alpha)} + 50)|J^*|$$

■

C. Problem 3: Non-disjoint cones of varying apex locations

Now, we can generalize the apex location to varying heights, which means that $\max(H(C)) - \min(H(C))$ is unbounded (Figure 1 (d)). Our strategy for this case is to separate the cones into different height groups. Each group will be visited using our General-Slice-and-Visit (Algorithm 2) method. The different height groups will be connected via a vertical line linking all sub-tours at the starting location x_0.

Theorem 5.7: Denote a set of non-disjoint cones as $C = \{c_1, c_2, ..., c_n\}$ with the same apex angle α and height d_v. The height of the apex location is defined as $H(c_i) \in H(C)$ for cone $c_i \in C$. There exist a tour J that visits C with an approximation factor of $O(\frac{\max(H(C))}{\tan(\alpha)})$ compared to the optimal trajectory J^*.

$$|J| \leq (B(J_t)(\frac{36}{\pi\tan(\alpha)} + 50)\frac{\max(H(C))}{1.5d_v} + 2)|J^*| \quad (6)$$

Proof: We first classify cones in C into different groups of apex height denoted as G with height range $[h_{min}, 1.5d_v + h_{min}), [1.5d_v + h_{min}, 3d_v + h_{min}), [3d_v + h_{min}, 4.5d_v + h_{min}), ..., [1.5md_v + h_{min}, 1.5(m+1)d_v + h_{min})$ until the last group contains h_{max}. Therefore, the number of groups in G is $(\max(H(C)) - \min(H(C)))/(1.5d_v)$. There exists an additional tour for our trajectory J to visit all groups in G, which is $2(\max(H(C)) - \min(H(C)))$ at max starting from x_0 for a round trip. We also know that $\min(H(C)) = 0$. Similar to proof for Theorem 5.5, we can bound the maximum height as $\max(H(C)) \leq |J^*|$.

$$|J| \leq B(J_t)(\frac{36}{\pi\tan(\alpha)} + 50)\frac{\max(H(C))}{1.5d_v}|J^*| + 2\max(H(C))$$
$$|J| \leq (B(J_t)(\frac{36}{\pi\tan(\alpha)} + 50)\frac{\max(H(C))}{1.5d_v} + 2)|J^*|$$

■

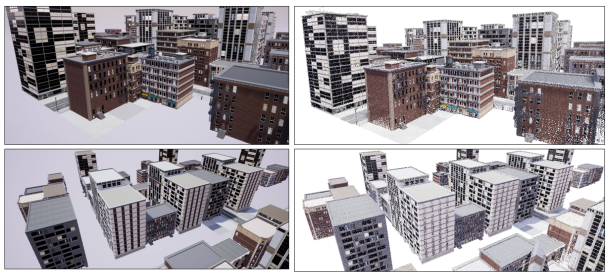
D. Coverage path planing to GC-TSPN

The coverage path planning problem can be converted into a traveling salemans problem with neighborhoods of cones. By changing the cone apex angle α into $\alpha_{mod} = \alpha - \varepsilon$, where $0 < \varepsilon < \alpha$, we can cover a circular patch with radius $r_{cover} = d_v(\tan(\alpha) - \tan(\alpha_{mod}))$ around the apex location instead of a single point as shown in Figure 3 (b). Such modification also allows our method to deal with occlusion. For a scenario that a short building is surrounded by a set of tall buildings, arbitrary cone heights d_v and apex angle α of the short building can be partially blocked by side buildings. We shrink both d_v and α so that the cones are no longer

Algorithm 2 General-Slice-and-Visit

Input: $x_0 \in \mathbf{R}^3$, $C = \{c_1, c_2, ..., c_n\}$, d_v
Output: J_t^{non}

Denote C_{sort} as sorting $\{c_1, c_2, ..., c_n\}$ with ascending $H(c_i)$
Set $C_{MIS} = \{\}$
for $c_i \in C_{sort}$ **do**
  $C_{MIS} = C_{MIS} \cup c_i$
  Set $Neighbors(c_i) = \{c_k \mid c_k \ intersect \ c_i, \ c_k \in C_{sort}\}$
  $C_{sort} = C_{sort} \setminus Neighbors(c_i)$
end for
Find J_t using Algorithm 1 with input C_{MIS}
For each c_i that J_t visits, add a detour that moves up to the c_i surface and circulates along the edge of c_i
Denote J_t^{non} as the new trajectory for non-disjoint cones



(a) (b)

Fig. 4: Qualitative comparison of the coverage results. (a) Images of the buildings in unreal simulation. (b) Corresponding images of the registered 3D point cloud from our trajectory.

in obstacles. Since the modified cone is smaller, multiple circular detours around C_{MIS} in Line 9 of Algorithm 2 are required to cover the neighborhoods. However, the additional detours in Theorem 5.5 does not change the order of the approximation factor.

VI. Path Planning for a City

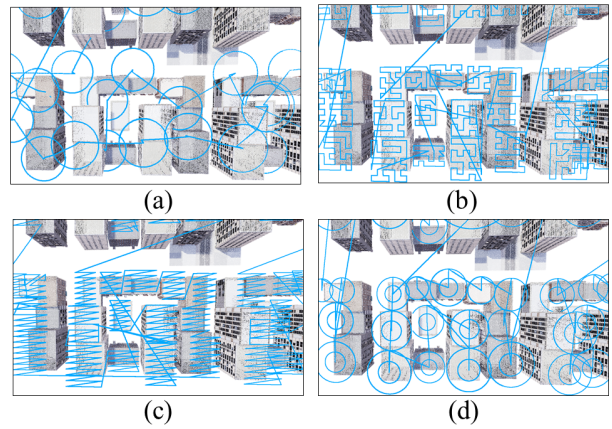
We have established how to plan a trajectory for a set of surfaces that are facing the same direction. In this section, we generalize this method to a city B that contains k distinct surface normals. To plan for surfaces of different orientations, we first group the surfaces into k subgroups based on their normal directions. For each group $S_i \subseteq B$ and $i = [1,2,...,k]$, Divide-and-Visit (Algorithm 2) is used to compute a trajectory J_i. To combine all the trajectories $J_i,\ i \in [1,2,...,k]$, we connect the starting location of each J_i and denote the resulting tour as J.

Theorem 6.1: Denote a city $B = \{s_1, s_2, ..., s_m\}$ that contains surfaces on each building. The surface normal is defined as $norm(s_i) \in \{N_1, N_2, ..., N_k\}\ \forall s_i \in B$, where $\{N_1, N_2, ..., N_k\}$ is a set of k distinct normal directions. The final trajectory J that covers B with high quality such that $g(s_i, J) = 1,\ \forall s_i \in B$ admits an approximation factor $O(\frac{k}{\tan(\alpha)} \max([L_B, W_B, H_B]))$, where $[L_B, W_B, H_B]$ are the length, width and height of the minimum enclosing box of B.

Proof: For each subgroup $S_i \subseteq B$ that share the same surface normals, we use Divide-and-Visit to compute a trajectory that is at most $O(\frac{1}{\tan(\alpha)} \max(H(S_i)))$ comparing to the optimal trajectory. For any pair of subgroups $S_i \subseteq B$ and $S_j \subseteq B$ such that $S_i \cap S_j = \emptyset$, we assume Algorithm 2 produces trajectories J_i and J_j respectively. The distance between the closest point of J_i and J_j is at most $\sqrt{L_B^2 + W_B^2 + H_B^2} \leq \sqrt{3} \max([L_B, W_B, H_B])$. Therefore, the resulting trajectory that visits all the surfaces of a city B admits an approximation factor of $O(\frac{k}{\tan(\alpha)} \max([L_B, W_B, H_B]))$. ■

VII. Experiments

Here, we provide the implementation details of our method. For a city B, we are first given a discrete set of surfaces and their normals. They are then clustered into k subgroups. Since a cone defines a set of good viewing region for the point on the cone apex, to cover an entire surface



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

Fig. 5: Comparisons among trajectories of the top surfaces of the buildings in unreal simulation. Our method is more efficient because it allows coverage of multiple surfaces from a single view (Algorithm 2 Line 9), whereas other methods dedicate a single view for each patch. (a) Our method (b) TSP-Grid method (c) TSP-Grid-Fast method (d) Time-optimal method [15]

requires infinitely many cones. Therefore, using strategy in Section V-D, a discrete set of cones per surface can be generated with apexes located at a grid with resolution of r_{cover} (Sec V-D). Since a small ε will result in a large number of cones in C, while a large ε can over populate C_{MIS}, we set $\varepsilon = 10^\circ$ ($r_{cover} = 2.5$ meters), which offers a good trade-off between computational time and coverage footprint.

For each subgroup of cones, we use TSP-GA [1] to compute a TSP tour that visits the center of a maximal independent set C_{MIS} of the cones. To avoid occlusion, cone heights and apex angles will be modified to fit the free space in narrow regions, which is also explained in Section V-D. For the trajectory to be valid, if any point on the trajectory is inside an obstacle, a closest point on the boundary of the obstacle will be used instead. We compare our method with three different baseline method quantitatively on a set of buildings simulated in matlab. We test the trajectory length and runtime of all, where we modify the scale of the city (50 to 500 buildings) and the orientation of each building. We evaluate[2] our method in a photo-realistic rendering software "Unreal Engine" [36] to show the coverage quality.

A. Comparison methods

The first baseline method "TSP-Grid" (TG) generates a grid of view points for each surface of the city. The viewing distance for each viewpoint is fixed to d_v and the grid resolution is fixed to r_{cover}. A final TSP tour is then estimated using TSP-GA method.

For a large number of discrete viewpoints, this method is too expensive. Therefore, we also present a second baseline method "TSP-Grid-Fast" (TGF) that uses the same set of discrete viewpoints. Instead, TGF estimates a TSP tour using

[1] https://www.mathworks.com/matlabcentral/fileexchange/13680-traveling-salesman-problem-genetic-algorithm

[2] Evaluation with an I5-4460 CPU and 8GB of ram.

		50 buildings			250 buildings			500 buildings		
		mean $\|J\|$ (10^5 m)	STD (10^5 m)	mean time (s)	mean $\|J\|$ (10^5 m)	STD (10^5 m)	mean time (s)	mean $\|J\|$ (10^5 m)	STD (10^5 m)	mean time (s)
$[\pi/4, 10]$	TG	1.79	0.14	3841	7.69	0.26	16037	32.4	0.21	36023
	TGF	1.81	0.17	685	8.69	0.24	1978	29.6	0.35	3982
	TO	1.83	0.12	**192**	8.53	0.23	**403**	28.9	0.24	**1201**
	Ours	**1.76**	0.11	215	**5.12**	0.28	564	**13.2**	0.32	1342
$[\pi/4, 15]$	TG	1.64	0.11	3027	7.43	0.22	9436	31.5	0.26	21023
	TGF	1.72	0.13	509	8.13	0.25	1652	24.6	0.42	3669
	TO	1.63	0.14	**147**	8.02	0.21	469	22.8	0.35	**948**
	Ours	**1.30**	0.13	169	**4.58**	0.24	**454**	**11.8**	0.36	1190
$[\pi/8, 10]$	TG	3.12	0.35	6427	9.21	0.53	20315	$\sim$	$\sim$	$\sim$
	TGF	2.29	0.31	923	9.43	0.57	2658	30.5	0.74	4573
	TO	2.78	0.36	**341**	9.23	0.56	778	27.9	0.69	1862
	Ours	**2.74**	0.28	357	**6.43**	0.58	**744**	**14.7**	0.71	**1634**
$[\pi/8, 15]$	TG	3.07	0.33	6839	$\sim$	$\sim$	$\sim$	$\sim$	$\sim$	$\sim$
	TGF	2.65	0.29	509	9.19	0.51	2975	28.3	0.68	4617
	TO	2.61	0.28	452	8.92	0.49	825	29.2	0.69	1869
	Ours	**2.35**	0.30	**416**	**5.68**	0.53	**814**	**13.8**	0.74	**1741**

TABLE I: A comparison of the trajectory length $|J|$ (10^5m) and the run-time (s) with apex angle and cone height shown in the first column. The building orientations are fixed to $k = 5$. $\sim$ means that the computation time is far too large.

	$k = 9$			$k = 13$			$k = 17$		
	mean $\|J\|$ (10^5 m)	STD (10^5 m)	mean time (s)	mean $\|J\|$ (10^5 m)	STD (10^5 m)	mean time (s)	mean $\|J\|$ (10^5 m)	STD (10^5 m)	mean time (s)
TG	1.74	0.15	3134	1.69	0.17	3215	1.81	0.20	3154
TGF	1.83	0.21	584	1.79	0.22	595	1.91	0.19	568
TO	1.71	0.16	**179**	1.82	0.23	239	1.83	0.25	267
Ours	**1.36**	0.14	184	**1.41**	0.19	**205**	**1.53**	0.21	**243**

TABLE II: A comparison of the trajectory length $|J|$ (10^5m) and the run-time (s) with $k = [9, 13, 17]$, 50 buildings, and $[\alpha = \pi/4, d_v = 15]$.

TSP-GA for each building. To connect all the buildings, a minimal spanning tree of the buildings is computed using the closest distance between buildings as metric. A depth-first-search strategy on the minimal spanning tree gives a visiting order for all the buildings. To compare with existing methods, we also implemented "Time-optimal" (TO) method [15], which uses a spiral motion to cover each building separately. The field-of-view for this method is set to 90° with the same viewing distance to our method. The experiments are repeated 30 times for each set of comparison and the standard deviations for the trajectory length are shown in Table I and Table II. The computation time standard deviations are negligible and are not shown in the tables.

B. Quantitative comparison

We compare our method with three baseline methods in a simulated city environment. For each building, the size of the buildings are sampled from a uniform distribution ranges from $[20 \sim 50]$ meters independently for width and length. The building's height is sampled from a uniform distribution ranges from $[50 \sim 100]$ meters. We randomly distribute $[50, 250, 500]$ buildings in a 1500×1500 environment with a minimum street width of 15 meters. The comparisons are also among varying apex angle and cone height $[\alpha, d_v]$ as shown in Table I. It is clear that our method out-performs the three baseline methods in most cases.

We also assign different surface normals to evaluate the effectiveness of our method. We distribute 50 buildings with at most 3 different orientations where $k = [9, 13, 17]$. The resulting comparison is shown in Table II. Our method also out-performs the baseline methods in most cases. More importantly, the trajectory length using our method does not diverge much for different k.

C. Qualitative comparison

To demonstrate our strategy, we implement our method in a photo-realistic rendering engine called "Unreal Engine" [36]. For better visualization, we only show the top trajectories for the four different methods as shown in Figure 5. Our method is more efficient because it allows coverage of multiple surfaces from a single view (Algorithm 2 Line 9), whereas other methods dedicate a single view for each patch. To show the detailed coverage quality, we also register the depth image with RGB color data to show the completeness and consistency comparing to the original structure as shown in Figure 4. For a real scene, the street width is not consistent. Therefore, we ignore sides where the street width is smaller than 5 meters.

VIII. Conclusion

This paper studies the problem of visual coverage path planning of a large scale urban environments. We present a polynomial time $O(\frac{1}{\tan(\alpha)} \max([L_B, H_B, W_B]))$ approximation algorithm for the generalized cone-TSPN problem for the case where the apex locations of the cones can lie on multiple planes. Our method provides a strategy to cover a large

scale urban environment efficiently. The resulting algorithm returns a solution that is at most $O(\frac{k}{\tan(\alpha)}\max([L_B, H_B, W_B]))$ of the optimal trajectory. The algorithm is well-suited for large scale environments since it runs in polynomial time with a constant memory consumption. To demonstrate the performance of our method, we compare it with two baselines methods with varying city scale (50 buildings to 500 buildings), view quality cone geometry (α and d_v), and building surface orientations ($k = [5, 9, 13, 17]$). Our method outperforms the other methods in both run-time and trajectory length. To evaluate our method in a realistic setting, we also demonstrate it in a city scene that contains more than 70 buildings using a photo-realistic rendering software. The resulting reconstruction quality is comparable to the original scene.

ACKNOWLEDGMENT

The authors would like to thank Shan Su, Wenbo Dong, Selim Engin, and Nicolai Haeni for their valuable discussions.

REFERENCES

[1] W. R. Scott, G. Roth, and J.-F. Rivest, "View planning for automated three-dimensional object reconstruction and inspection," *ACM Computing Surveys (CSUR)*, vol. 35, no. 1, pp. 64–96, 2003.

[2] J. I. Vasquez-Gomez, L. E. Sucar, R. Murrieta-Cid, and E. Lopez-Damian, "Volumetric next-best-view planning for 3d object reconstruction with positioning error," *International Journal of Advanced Robotic Systems*, vol. 11, no. 10, pp. 159–172, 2014.

[3] A. Bircher, M. Kamel, K. Alexis, H. Oleynikova, and R. Siegwart, "Receding horizon" next-best-view" planner for 3d exploration," in *international conference on robotics and automation*. IEEE, 2016, pp. 1462–1468.

[4] S. Isler, R. Sabzevari, J. Delmerico, and D. Scaramuzza, "An information gain formulation for active volumetric 3d reconstruction," in *International Conference on Robotics and Automation*. IEEE, 2016, pp. 3477–3484.

[5] X. Fan, L. Zhang, B. Brown, and S. Rusinkiewicz, "Automated view and path planning for scalable multi-object 3d scanning," *ACM Transactions on Graphics (TOG)*, vol. 35, no. 6, pp. 1–13, 2016.

[6] A. Xu, C. Viriyasuthee, and I. Rekleitis, "Efficient complete coverage of a known arbitrary environment with applications to aerial operations," *Autonomous Robots*, vol. 36, no. 4, pp. 365–381, 2014.

[7] A. Nedjati, G. Izbirak, B. Vizvari, and J. Arkat, "Complete coverage path planning for a multi-uav response system in post-earthquake assessment," *Robotics*, vol. 5, no. 4, pp. 26–41, 2016.

[8] Y.-H. Choi, T.-K. Lee, S.-H. Baek, and S.-Y. Oh, "Online complete coverage path planning for mobile robots based on linked spiral paths using constrained inverse distance transform," in *International Conference on Intelligent Robots and Systems*. IEEE, 2009, pp. 5788–5793.

[9] C. Peng and V. Isler, "View selection with geometric uncertainty modelling," in *Robitcs: Science and Systems*, 06 2018, pp. 1–11.

[10] P. Cheng and I. Volkan, "Adaptive view planning for aerial 3d reconstruction," in *2019 International Conference on Robotics and Automation (ICRA)*, May 2019, pp. 2981–2987.

[11] M. Roberts, D. Dey, A. Truong, S. Sinha, S. Shah, A. Kapoor, P. Hanrahan, and N. Joshi, "Submodular trajectory optimization for aerial 3d scanning," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 5324–5333.

[12] B. Hepp, M. Nießner, and O. Hilliges, "Plan3d: Viewpoint and trajectory optimization for aerial multi-view stereo reconstruction," *ACM Transactions on Graphics (TOG)*, vol. 38, no. 1, pp. 1–17, 2018.

[13] A. Bircher, M. Kamel, K. Alexis, M. Burri, P. Oettershagen, S. Omari, T. Mantel, and R. Siegwart, "Three-dimensional coverage path planning via viewpoint resampling and tour optimization for aerial robots," *Autonomous Robots*, vol. 40, no. 6, pp. 1059–1078, 2016.

[14] W. Jing, J. Polden, P. Y. Tao, W. Lin, and K. Shimada, "View planning for 3d shape reconstruction of buildings with unmanned aerial vehicles," in *2016 14th International Conference on Control, Automation, Robotics and Vision (ICARCV)*. IEEE, 2016, pp. 1–6.

[15] P. Cheng, J. Keller, and V. Kumar, "Time-optimal uav trajectory planning for 3d urban structure coverage," in *International Conference on Intelligent Robots and Systems*. IEEE, 2008, pp. 2750–2757.

[16] P. A. Plonski and V. Isler, "Approximation algorithms for tours of height-varying view cones," *The International Journal of Robotics Research*, vol. 38, no. 2-3, pp. 224–235, 2019.

[17] C. H. Papadimitriou, "The euclidean travelling salesman problem is np-complete," *Theoretical Computer Science*, vol. 4, pp. 237–244, 1977.

[18] M. de Berg, J. Gudmundsson, M. J. Katz, C. Levcopoulos, M. H. Overmars, and A. F. van der Stappen, "Tsp with neighborhoods of varying size," *Journal of Algorithms*, vol. 57, no. 1, pp. 22–36, 2005.

[19] S. Safra and O. Schwartz, "On the complexity of approximating tsp with neighborhoods and related problems," *computational complexity*, vol. 14, no. 4, pp. 281–307, 2006.

[20] C. S. Mata and J. S. Mitchell, "Approximation algorithms for geometric tour and network design problems," in *Proceedings of the eleventh annual symposium on Computational geometry*, 1995, pp. 360–369.

[21] T.-H. H. Chan and K. Elbassioni, "A qptas for tsp with fat weakly disjoint neighborhoods in doubling metrics," *Discrete & Computational Geometry*, vol. 46, no. 4, pp. 704–723, 2011.

[22] H. L. Bodlaender, C. Feremans, A. Grigoriev, E. Penninkx, R. Sitters, and T. Wolle, "On the minimum corridor connection problem and other generalized geometric problems," *Computational Geometry*, vol. 42, no. 9, pp. 939–951, 2009.

[23] A. Dumitrescu and C. D. Tóth, "The traveling salesman problem for lines, balls, and planes," *ACM Transactions on Algorithms (TALG)*, vol. 12, no. 3, pp. 1–29, 2016.

[24] A. Dumitrescu and C. D. Toth, "Constant-factor approximation for tsp with disks," in *A Journey Through Discrete Mathematics*. Springer, 2017, pp. 375–390.

[25] N. Stefas, P. A. Plonski, and V. Isler, "Approximation algorithms for tours of orientation-varying view cones," in *2018 IEEE International Conference on Robotics and Automation*. IEEE, 2018, pp. 1–6.

[26] G. Papadopoulos, H. Kurniawati, and N. M. Patrikalakis, "Asymptotically optimal inspection planning using systems with differential constraints," in *2013 IEEE International Conference on Robotics and Automation*. IEEE, 2013, pp. 4126–4133.

[27] A. Bircher, K. Alexis, U. Schwesinger, S. Omari, M. Burri, and R. Siegwart, "An incremental sampling-based approach to inspection planning: the rapidly exploring random tree of trees," *Robotica*, vol. 35, no. 6, pp. 1327–1340, 2017.

[28] P. Kafka, J. Faigl, and P. Váňa, "Random inspection tree algorithm in visual inspection with a realistic sensing model and differential constraints," in *2016 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2016, pp. 2782–2787.

[29] H. Choset and P. Pignon, "Coverage path planning: The boustrophedon cellular decomposition," in *Field and service robotics*. Springer, 1998, pp. 203–209.

[30] E. U. Acar, H. Choset, A. A. Rizzi, P. N. Atkar, and D. Hull, "Morse decompositions for coverage tasks," *The international journal of robotics research*, vol. 21, no. 4, pp. 331–344, 2002.

[31] V. J. Lumelsky, S. Mukhopadhyay, and K. Sun, "Dynamic path planning in sensor-based terrain acquisition," *IEEE Transactions on Robotics and Automation*, vol. 6, no. 4, pp. 462–472, 1990.

[32] M. Wei and V. Isler, "Coverage path planning under the energy constraint," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2018, pp. 368–373.

[33] C. Di Franco and G. Buttazzo, "Coverage path planning for uavs photogrammetry with energy and resolution constraints," *Journal of Intelligent & Robotic Systems*, vol. 83, no. 3-4, pp. 445–462, 2016.

[34] H. González-Banos, "A randomized art-gallery algorithm for sensor placement," in *Proceedings of the seventeenth annual symposium on Computational geometry*, 2001, pp. 232–240.

[35] S. Lin and B. W. Kernighan, "An effective heuristic algorithm for the traveling-salesman problem," *Operations research*, vol. 21, no. 2, pp. 498–516, 1973.

[36] UnrealEngine, "Unreal Engine 4," https://www.unrealengine.com/en-US/blog, 2017.