Metric View Planning Problem with Traveling Cost and Visibility Range

Pengpeng Wang*, Ramesh Krishnamurti†, and Kamal Gupta*
* *RAMP Lab, School of Engineering Science*
Simon Fraser University
Burnaby, BC, CANADA, V5A 1S6
{*pwangf, kamal*}*@cs.sfu.ca*
† *School of Computing Science*
Simon Fraser University
Burnaby, BC, CANADA, V5A 1S6
ramesh@cs.sfu.ca

***Abstract*— In this paper, we consider the problem where a point robot in a 2D or 3D environment equipped with an omnidirectional range sensor of finite range D is asked to inspect a set of surface patches, while minimizing the sum of view cost, proportional to the number of viewpoints planned, and the travel cost, proportional to the length of path traveled. We call it the *Metric View Planning Problem with Traveling Cost and Visibility Range* or Metric TVPP in short. Via an *L-reduction* from the set covering problem to a two-dimensional Metric TVPP, we show that the Metric TVPP cannot be approximated within $O(\log m)$ ratio by any polynomial algorithm, where m is the number of surface patches to cover. We then analyze a natural two-level algorithm of solving first the view planning problem to get an approximate solution, and then solving, again using an approximation algorithm, the Metric traveling salesman problem to connect the planned viewpoints. We show this greedy algorithm has the approximation ratio of $O(\log m)$. Thus, it asymptotically achieves the best approximation ratio one can hope for.**

I. Introduction

Imagine that a robot is asked to autonomously scan the artifacts in a historic site and build their complete surface representations. The robotic artifact "documentation" or virtual reality environment construction ability automates many tedious work done primarily by human thus far. See [1] and the references therein for some of the few existing works on automating this process; and see Fig. 1 for a simple illustration. For such applications, especially in remote missions, the time and energy spent are a critical factor for the tasks to be successfully completed. Thus, we model this robotic object inspection task as an optimization problem of minimizing the corresponding total cost, a weighted sum of both the view cost and the traveling cost. View cost corresponds to the image processing, image registration and geometric model construction after each view is taken and is proportional to the number of viewpoint planned [16]. Travel cost is the cumulative time and energy consumption due to the robot movements and thus is proportional to the length of the total tour the robot travels. We call it the problem of view planning with combined view and traveling costs, denoted by *Traveling VPP* [19]. Note that in [1] and the references therein, this optimization of both view and traveling cost is not addressed.

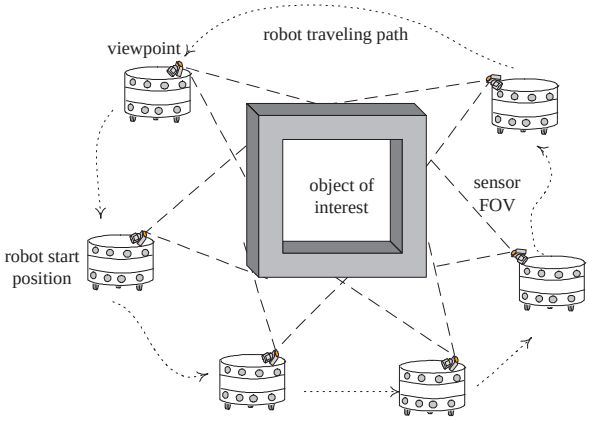

Fig. 1. A Traveling VPP instance. It shows 6 planned sensor viewpoints that totally cover the surface of the object of interest, and the robot traveling tour to realize them.

Traveling VPP combines elements of both the NP-complete view planning problem (VPP) and the watchman route problem, and thus generalizes both. Consequently, Traveling VPP is NP-hard. VPP refers to planning the minimum number of viewpoints to completely inspect an object surface. It is considered in the robot vision area [16], where often a sensor positioning system is used within a well controlled and limited workspace. These formulations do not consider the traveling cost of the robot, a critical cost, particularly for large workspaces and remote autonomous missions, where power consumption is a critical factor. On the other hand, the watchman route problem, considered in the computational geometry area [4], [17], asks for the shortest tour to inspect the interior of a two-dimensional polygonal region. It does not consider view cost, a critical cost, particularly for the inspection tasks considered here, where each sensor view and the consequent processing are time-consuming.

For VPP, we refer to [16] for a detailed survey. Here we mention a related theoretical work [15], in which the authors show that VPP is reduced to the well-known NP-complete set covering problem (SCP). SCP refers to, given a universe

of elements and some subsets of the universe, planning the minimum number of such subsets, the union of which is the universe. By regarding each object surface patch as an element in a subset that a viewpoint can cover [15], VPP is reduced to SCP. SCP plays a central role in the combinatorial optimization area. Best known approximation algorithms[1] for SCP proceed greedily according to the (amortized) covering cost, and have an approximation ratio of $O(\log m)$, where m is the number of elements in the universe [18].

There is some existing work on combining the view and traveling cost in the literature, but not in a unified and global fashion. For example, in [8], [11], the authors considered a local version of the robot exploration problem, "to look around a corner", i.e., to detect an object hidden behind a corner while minimizing the sum of the robot traveling distance and the sensor scan time. The problem is considerably simpler since the goal is local, i.e., the objective is not to cover all the object surfaces. In [6], the authors considered the combined problem, however, in a "weak sense", since no view cost is considered, thus corresponding to a special case of Traveling VPP. They proposed to solve the problem by a decoupled two-level approach, i.e., to plan the minimum number of viewpoints without considering robot traveling cost first and then to solve approximately the Metric TSP using the shortest path graph. They used a probabilistic algorithm to plan the minimum number of viewpoints at the first level. We refer [19] for a counter-example where this two-level decoupled approach provides no performance bound (with respect to the optimal solution cost), and can perform arbitrarily poorly.

Our methodology is to design approximation algorithms (used for NP-hard problems) for solving Traveling VPP that guarantee performance bounds. In [19], we gave a unified formulation of the problem with a single objective function that combines both view and traveling costs, and proposed approximation algorithms that are fast and guarantee performances. To abstract out application related issues and concentrate on the combinatorial nature of the problem, in [19], we assume the set of viewpoints, the set of surface patches, and the covering relation between them are given in Traveling VPP. For given scenes, these viewpoints can be derived from the aspect graph of the scene [3], or by randomly sampling the sensor configuration space, the space of the sensor configurations that uniquely determines the viewpoints [10]. Traveling VPP also assumes a general graph that connect the viewpoints is given. This graph would essentially be a roadmap built in the configuration space of the robot. This is a standard and well-studied technique for robot motion planning [12], [14]. In [19], we developed an approximation algorithm for Traveling VPP that has the approximation ratio of the smaller of the *view frequency*, defined as the maximum number of viewpoints that cover a single surface patch, and a poly-log of the input size. We refer [19] for a detailed coverage.

[1]Approximation algorithms are algorithms that run in polynomial time and guarantee the algorithmic solution cost is within a certain ratio of the optimal cost. The ratio is called the approximation ratio [18].

The traveling graph given in [19] is arbitrary and the edge costs do not assume a metric. This implies that the robot used can have arbitrary kinematics and geometry, for example a mobile robot or a mobile-manipulator, while quite often a mobile robot is used for inspection tasks (See Fig. 1) and generally modeled as a point in the same environment as the object. Also, the covering relations between viewpoints and surface patches in Traveling VPP can be arbitrary. While this models well range sensors of omnidirectional and infinite field of view (subject to occlusions from obstacles), realistic sensors usually have limited sensing range and often for better inspection qualities, the range sensors is required to be within a certain range to the surface patch to cover.

To address the above-mentioned realistic scenarios and corresponding constraints, in this paper we introduce a special case of Traveling VPP, in which a point robot equipped with a omnidirectional range sensor with a certain range D, is asked to inspect object surfaces in a two- or three-dimensional environment. We call this special case *Metric View Planning Problem with Traveling Cost and Visibility Range*, or *Metric TVPP* in short. Note that unlike the general Traveling VPP, for Metric TVPP, the robot traveling distances between viewpoints is the shortest path lengths between them, and assume a metric, satisfying the triangular inequality.

First, we analyze the inapproximability of Metric TVPP, i.e., the best approximation ratio any polynomial algorithm can achieve. We construct a *L-reduction*, a reduction from one problem to the other while the inapproximability is preserved, from the SCP to two-dimensional Metric TVPP and show that Metric TVPP is log-inapproximable, i.e., Metric TVPP cannot be approximated within $O(\log m)$ ratio by any polynomial algorithm, where m is the number of surface patches to cover. Please see the appendix for a quick recap of the inapproximability concept and L-reduction method.

The question we ask in this paper is: Can we find an algorithm for Metric TVPP with better approximation ratio than the one we developed in [19] for the general Traveling VPP? The answer is yes! And this better approximation algorithm follows the two-level approach mentioned earlier, i.e., to first solve the minimum number of viewpoints problem and then solve the Metric TSP to connect them. Our contribution here is to show that this two-level approach, although shown to perform arbitrarily poorly for the general Traveling VPP in [19], achieves an approximation ratio of $O(\log m)$, which is of the same order as the inapproximability result of Metric TVPP. Intuitively, the key insight is that the sensor range constraint in Metric TVPP implicitly couples the traveling and view components: in order to cover a surface patch, the robot has to travel to at least within the sensor range of it. Thus the two-level approach is no longer *decoupled*.

The rest of the paper is organized as follows: first, we give notations and the problem formulation for the Metric TVPP; second, we give a L-reduction to show the inapproximability of *Metric TVPP*; we then recap the two-level algorithm in [6] and analyze its approximation ratio; finally, we conclude

and discuss future work.

II. Notations and Problem Formulation

Let $\mathcal{W}$ denote the environment where both robot and surface patches reside. We assume $\mathcal{W}$ is either the two- or three-dimensional Euclidean space populated by obstacles. For any two points x_1 and x_2 in $\mathcal{W}$, we denote the Euclidean distance between them by $\|x_1, x_2\|_{\mathcal{W}}$.

We denote the set of all viewpoints by $\mathcal{V}$ and index them by i. We denote the set of surface patches by $\mathcal{S}$ and index them by j. We use the notations $i \in \mathcal{V}$ and $j \in \mathcal{S}$ to imply the "ith viewpoint" and "jth surface patch", respectively. For $i \in \mathcal{V}$, let $\mathcal{S}(i)$ denote the subset of the surface patches that viewpoint i covers; and for $j \in \mathcal{S}$, let $\mathcal{V}(j)$ denote the subset of viewpoints that cover surface patch j. By definition, the necessary condition for a viewpoint i to cover a surface patch j is that the distance between them is upper bounded by D. With a slight abuse of notation, we use $\|i, j\|_{\mathcal{W}}$ to denote this distance. Formally, for a surface patch j that occupies a region $\mathcal{R}(j)$ (of co-dimension 1) in $\mathcal{W}$, the distance $\|i, j\|_{\mathcal{W}}$ is the supremum of the distances between points belonging to $\mathcal{R}(j)$ and the viewpoint i, i.e., $\|i, j\|_{\mathcal{W}} = \sup_{x \in \mathcal{R}(j)} \|x, i\|_{\mathcal{W}}$. Let $G = (\mathcal{V}, E)$ denote the traveling graph. The edge cost $c_e, e = (i_1, i_2)$ between two viewpoints $i_1, i_2 \in \mathcal{V}$ is the shortest distance the robot travels between them. This implies that any free path between i_1 an i_2 has length lower bounded by c_e For a subset $\mathcal{U}$ of viewpoints, i.e., $\mathcal{U} \subseteq \mathcal{V}$, let $Tour(\mathcal{U})$ and $\|Tour(\mathcal{U})\|$ denote a tour connecting these viewpoints and its total length, i.e., the sum of edges costs on the tour, respectively. Let w_v and w_p denote the unit view cost, or cost per viewpoint, and the unit traveling cost, or cost per unit traveling distance respectively. Let $|\mathcal{A}|$ denote the cardinality of set $\mathcal{A}$.

Metric TVPP is to plan a subset of viewpoints, denoted by $\mathcal{V}'$, and a connecting tour $Tour(\mathcal{V}')$, such that the total cost, $w_v|\mathcal{V}'| + w_p\|Tour(\mathcal{V}')\|$, is minimized, under the constraint that all the surface patches are covered, i.e., $\forall j \in \mathcal{S},\ \mathcal{V}(j) \cap \mathcal{V}' \neq \emptyset$.

III. Inapproximability of *Metric TVPP*

In this section we give an L-reduction from SCP to the Metric TVPP. Given an arbitrary SCP instance, we construct a two-dimensional Metric TVPP instance where the unit view cost is 1 and the traveling cost is negligible, i.e., $w_p \ll w_v = 1$. In the constructed Metric TVPP instance, the number of viewpoints and surface patches are respectively the same as the number of subsets and elements in the SCP instance. Any solution to the constructed Metric TVPP instance correspond to a solution to the SCP instance with the same cost. This L-reduction extends the inapproximability result for SCP to Metric TVPP.

Theorem 1: Metric TVPP is $O(\log m)$ inapproximable.

Proof: Given an arbitrary SCP instance, we denote the universe of elements by $\mathcal{S} = \{s_j, j = 1, \ldots, m\}$, and a collection of its subsets by $\mathcal{V} = \{v_i : v_i \subseteq \mathcal{S}, j = 1, \ldots, n\}$. We construct a two-dimensional Metric TVPP instance as in Fig. 2. We first draw a circle C of radius R_C and let L be a diameter dividing the boundary of the circle into two halves. For one half of the circle, we divide it equally into three parts as shown and put m equally-sized surface patches in the perimeter of the middle part, each of which corresponds to an element of the universe of the SCP instance. This middle part can guarantee any viewpoint surface relationships as we show below. The size of each is less than $\frac{R_C \pi}{3m}$. We denote the surface patches using the same labels as those for the elements in the SCP instance, $s_j, j = 1, \ldots, m$, to imply this correspondence. We then create viewpoints, corresponding to the subsets in the SCP instance, and put them on L with distances between two consecutive ones greater than $2R_O$. (See Fig. 2(c).) We construct a half ring (cut by L) of obstacles of radius R_O and negligible thickness. To accommodate n viewpoints, we require that $n \cdot 2R_O < 2R_C$.

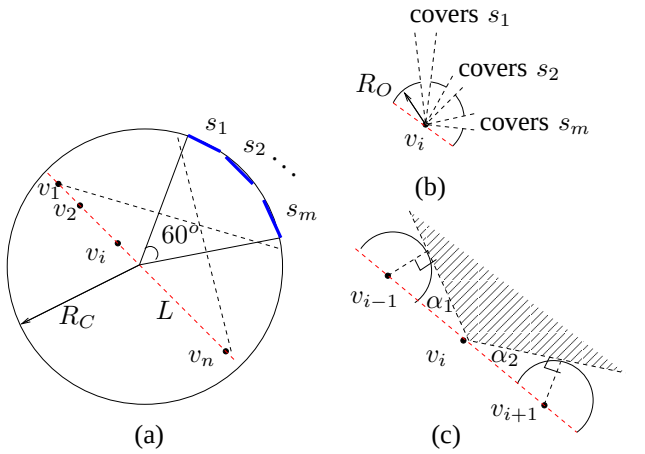

Fig. 2. Reduction from an arbitrary SCP instance to a VPP instance. See the text for the explanation.

For the half ring of obstacles around each viewpoint, we cut some openings, such that the viewpoint covers only the surface patches corresponding to those elements in the viewpoint's corresponding subset. For example, in Fig. 2(b), for viewpoint v_i corresponding to subset $\{s_1, s_2, s_m\}$, we cut three openings such that only the three surface patches corresponding to s_1, s_2, and s_m are visible from v_i. As shown in Fig. 2(c), for viewpoint v_i, the obstacles around viewpoints other than v_i cannot occlude the shaded area from v_i. The area they can occlude is bounded by the two angles α_1 and α_2, and $\alpha_1, \alpha_2 \leq \tan^{-1} \frac{R_O}{2R_O} = 30^o$. Thus, we can achieve any covering relationships between v_i and the surface patches if the patches are put on the part of circle C in the shade area. By putting the surfaces patches on the common "shaded areas" of all viewpoints, we can construct any SCP instance. By simple planar geometries, this common area is bounded by the dotted lines from v_1 and v_n respectively, and thus occupies the middle part as shown.

The resulting Metric TVPP instance is to plan the minimum number of viewpoints and a connecting tour that cover all the surface patches. Since the traveling cost is negligible, the cost to minimize is just the number of viewpoints planned. The solution to the Metric TVPP instance implies

a solution to the SCP instance, i.e., to choose the subsets corresponding to those viewpoints chosen. The two solutions have the same cost. Thus, the L-reduction from SCP to Metric TVPP is constructed.

The above reduction implies that Metric TVPP is at least as hard as the SCP, and any approximation algorithm for the Metric TVPP cannot have a better approximation ratio than the best approximation ratio for the SCP. Thus the inapproximability result for SCP [7] implies this lemma. ■

IV. Two-level Algorithm for *Metric TVPP*

In this section, we give the the pseudo code of a two-level algorithm for *Metric TVPP*.

Algorithm 1: Two-level Algorithm for Metric TVPP

Step 1. Solve the SCP greedily:

Iteratively choose the viewpoint that covers the most uncovered *surface patches until all surface patches are covered.*

Output the chosen viewpoint set $\mathcal{V}'$.

Step 2. Solve the Metric TSP to connect $\mathcal{V}'$

Construct the shortest path graph G' on $\mathcal{V}'$, i.e., the complete graph on $\mathcal{V}'$ where the edge cost is the corresponding shortest path length on G

Use Christofides' algorithm [5] to construct the tour on G' to connect $\mathcal{V}'$ [2].

This algorithm solves the problem in two steps. In the first step, it solves the VPP or SCP part of Metric TVPP greedily. In the second step, it solves the Metric TSP to connect these picked viewpoints using the Christofides' algorithm [5].

V. Algorithm Analysis

To analyze the approximation ratio of the two-level algorithm, Algorithm 1, we present an alternative algorithm, Algorithm 2, whose performance is no better than Algorithm 1. We emphasize that Algorithm 2 is only for analysis purpose and does not need to be implemented. We then give the approximation ratio of this alternative algorithm. This ratio thus also serves as the approximation ratio for Metric TVPP. In the following, we give the algorithm after a few definitions, and then analyze its performance.

A. Alternative algorithm

For any surface patch j, any two covering viewpoints must lie within $2D$ distance of each other, $\forall i_1, i_2 \in \mathcal{V}(j), e = (i_1, i_2), \quad c_e \leq 2D$. This is because there exists a free path between i_1 and i_2 having distance less than $2D$, as shown in Fig. 3. This path follows first the free visibility line from i_1 to surface j, and then the visibility line from j to i_2. For viewpoint i, we define the free region within its $2D$ distance, i.e., the set of points i can reach using the shortest path of length less than $2D$, the *domain* of i. We call two viewpoints *dependent* if one lies in the other's domain, and *independent* otherwise.

[2]Christofides' algorithm first constructs the minimum spanning tree T on G'; second, it computes the minimum cost perfect matching between vertices of T with odd degrees; third, it adds edges corresponding to this matching to T to make it Eulerian; last, it computes a tour on the resulting Eulerian graph [5].

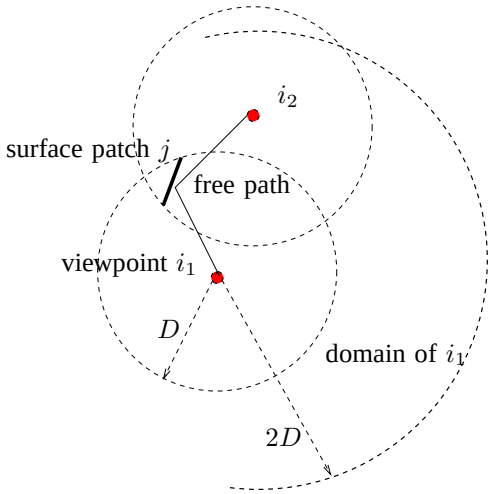

Fig. 3. Virtual domain and dependence. Viewpoints i_i and i_2 are dependent and have a free path of length $\leq 2D$, comprising of solid line segments joining the two viewpoints.

In the following, we give the pseudo codes of the alternative algorithm.

Algorithm 2: Alternative Algorithm for Metric TVPP

Step 1. Solve the SCP greedily to get $\mathcal{V}'$.

(This step is exactly the same as Step 1 of the two-level algorithm Algorithm 1.)

Step 2. Choose independent viewpoints:

Iteratively choose a viewpoint from $\mathcal{V}'$ that is independent *from already chosen ones.*

Output the chosen viewpoint set $\mathcal{V}''$.

Step 3. Solve the Metric TSP to connect $\mathcal{V}''$:

Construct the shortest path graph G'' on $\mathcal{V}''$.

Use Christofides' algorithm [5] to construct the tour on G'' to connect $\mathcal{V}''$.

Step 4. Connect the viewpoints in $\mathcal{V}' \setminus \mathcal{V}''$ to its nearest neighbor in $\mathcal{V}''$ using the shortest path on G.

Note that Algorithm 2 chooses exactly the same viewpoints, $\mathcal{V}'$, as Algorithm 1. However, it constructs the tour differently: it first constructs the tour to connect only the viewpoints in $\mathcal{V}''$, called *centers*; and for $\mathcal{V}' \setminus \mathcal{V}''$, it constructs "detours" to their nearest centers. So clearly, the solution cost by Algorithm 2 is no better than that by Algorithm 1.

B. Analysis

In this section, after giving some notations, we first show that the optimal solution to Metric TVPP has to pass through as least a viewpoint in every domain of the centers, Lemma 2. This observation helps lower bound the optimal solution cost, Lemma 3, and upper bound the algorithmic solution cost, Lemma 4. Combining both bounds gives us the algorithmic approximation ratio.

Let OPT_{TVPP} denote the optimal solution cost to Metric TVPP. We denote the algorithmic solution cost (Algorithm 2) by $cost'$, the view cost and traveling parts of which are denoted by $cost'_{view}$ and $cost'_{travel}$ respectively. We use OPT_{SCP} to denote the optimal SCP solution cost to cover

all the surface patches, i.e.,

$$OPT_{SCP} = \min_{\mathcal{U}\subseteq\mathcal{V}:\ \cup_{i\in\mathcal{U}}\mathcal{S}(i)=\mathcal{S}} |\mathcal{U}|.$$

We call the tour that connects at least one point from every domain of the centers a *domain tour*. We use $OPT_{DOMtour}$ to denote the shortest distance of such tours.

Lemma 2: The tour in the optimal solution to Metric TVPP has to visit the domains of all centers, and is hence a feasible domain tour.

Proof: For any center $i \in \mathcal{V}''$, we arbitrarily pick one surface patch from its surface patch set $j \in \mathcal{S}(i)$. Note that due to visibility range constraints and the triangular inequality, any viewpoint i' of surface patch j's viewpoint set, i.e., $\forall i' \in \mathcal{V}(j)$, has to lie in the domain of i, i.e., $\|i, i'\| \leq \|j, i\| + \|j, i'\| \leq D + D = 2D$. At least one viewpoint from $\mathcal{V}(j)$ is chosen and visited by the constructed tour. ■

Lemma 2 leads to the following upper bound result for the cost of the optimal solution to Metric TVPP.

Lemma 3: The cost of the optimal solution to Metric TVPP is at least the sum of the minimum view cost (ignoring traveling) and the optimal tour cost to visit every domain of the centers, i.e.,

$$OPT_{TVPP} \geq w_v OPT_{SCP} + w_p OPT_{DOMtour} \quad (1)$$

Proof: The viewpoint set chosen by the optimal solution to Metric TVPP is a feasible solution to the corresponding SCP. Lemma 2 shows that the tour chosen is a domain tour. Hence their costs are lower bounded by OPT_{SCP} and $OPT_{DOMtour}$, respectively. ■

Also we givw the upper bound result for the cost of the algorithmic solution.

Lemma 4: The algorithmic solution cost $cost'$ is at most $w_v OPT_{SCP} \cdot (1 + 6D\frac{w_p}{w_v})O(\log m) + w_p OPT_{DOMtour} \cdot 1.5$.

Proof: We use the greedy SCP algorithm to get $\mathcal{V}'$. Its approximation ratio [18] implies:

$$cost'_{view} = w_v \cdot |\mathcal{V}'| \leq w_v \cdot OPT_{SCP} \cdot O(\log m) \quad (2)$$

We use Christofides' algorithm [5] to get a tour of all the centers in $\mathcal{V}''$. This tour is at most 1.5 times the cost of the optimal Metric TSP solution to connect the centers. Since we do not know the value of the latter, we will instead use the cost of another feasible Metric TSP solution to connect the centers. This solution is to first travel on the optimal domain tour and then detour to the centers (and back) if needed, as shown by the dotted path in Fig. 4. By the definition of the domain tour, for any center, there must exist a point on the tour within its $2D$ distance. This implies that the tour constructed in Step 3 of Algorithm 2 has length at most $1.5(OPT_{DOMtour} + |\mathcal{V}''| \cdot 4D)$. The $4D$ factor corresponds to traveling from such a point on the domain tour to the center and then traveling back.

Further detouring (if needed) from a center to any of its dependent viewpoints in $\mathcal{V}' \setminus \mathcal{V}''$, Step 4 of Algorithm 2, incurs again a traveling distance of at most $4D$. So the overall traveling cost of the algorithmic solution is upper bounded as follow:

$$\begin{aligned} cost'_{travel} &\leq w_p \cdot 1.5(OPT_{DOMtour} + 4D|\mathcal{V}''|) \\ &\quad + w_p \cdot 4D(|\mathcal{V}'| - |\mathcal{V}''|) \\ &\leq 1.5 w_p (OPT_{DOMtour} + 4D|\mathcal{V}'|). \end{aligned} \quad (3)$$

The factor 1.5 above is due to the approximation ratio of the Christofides' algorithm [5].

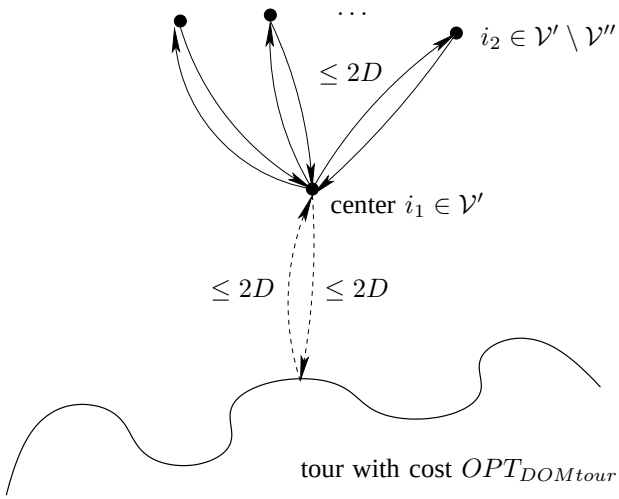

Fig. 4. A feasible Metric TVPP solution that takes the optimal domain tour and detours first to the centers, and then to rest of the planned viewpoints .

In conclusion, using both Eqs. 2,3, the total cost of the algorithmic solution is upper bounded as follows:

$$\begin{aligned} cost' &= cost'_{travel} + cost'_{view} \\ &\leq w_v OPT_{SCP} \cdot (1 + 6D\frac{w_p}{w_v})O(\log m) \\ &\quad + 1.5 w_p OPT_{DOMtour} \end{aligned} \quad (4)$$

■

With the upper and lower bounds of the optimal and algorithmic solution costs to Metric TVPP, we are ready to prove the approximation ratio result for Algorithm 2.

Theorem 5: The approximation ratio of Algorithm 1 is $O(\log m)$.

Proof: As mentioned before, the Metric TVPP solution by Algorithm 2 has no better cost than that by Algorithm 1. By Theorem 1, the best possible approximation algorithm has approximation ratio of $O(\log m)$, it suffices to prove that the approximation ratio of Algorithm 2 is $O(\log m)$.

Combining the lower bound for the optimal solution cost OPT_{TVPP}, Lemma 3, and the upper bound for the algorithmic solution cost $cost'$, Lemma 4, it is easy to show that the approximation ratio of Algorithm 2 is $\max\{1.5, (1 + 6D\frac{w_p}{w_v})O(\log m)\} = O(\log m)$. ■

VI. Conclusion

In this paper, we introduced a special case of Traveling VPP where the viewpoints (due to the point robot assumption) and surface patches are in the same metric space and the sensor is constrained by a certain visibility range. We proposed a two-level algorithm using the decoupled approach in [6]. It greedily solves VPP at the first step; and use Christofides' algorithm [5] to solve the Metric TSP to construct the tour of the viewpoints chosen at the first step. We showed that the approximation ratio of this algorithm is in the order of the logarithm of the number of surface patches, which matches (in the same order) the inapproximability result of Metric TVPP. The analysis was done by giving the approximation ratio of an alternative algorithm that has no better solution cost.

For future work, we are interested in generalizing Metric TVPP to its unknown case, where the surface patches to cover are not given in advance but generated online, and designing *competitive* algorithms. Competitive algorithms for online problems, where the inputs are iteratively given in an online fashion, are polynomial algorithms with guaranteed performance w.r.t. the offline optimal cost, the optimal cost to an offline problem where all the inputs are given in advance [2]. This corresponds to the robot exploration scenarios where the knowledge of the environment is (partially) unavailable.

Appendix: Inapproximability and L-Reduction

Although different NP-complete optimization problems are considered at the same complexity level in terms of solving for the optimal solutions, they can have different difficulty levels in obtaining approximation solutions. This concept is quantified by the *inapproximability* or the *hardness of approximation*, which refers to a problem specific constant or a function of the input size as the lower bound on the approximation ratio of any polynomial algorithm, assuming some generally-believed complexity class relations, for example $P \neq NP$. For example, the inapproximability result for SCP says that the optimal solution to SCP cannot be approximated within its $(1 - o(1)) \ln n$ ratio unless NP admits quasi-polynomial algorithms [7].

We use the simplest form of a common technique, called *L-reduction*, to establish the inapproximability result for Metric TVPP, which works similarly as the reduction method [9] used in proving NP-completeness results. Rather than covering the theory, we refer to [13] for a detailed coverage on L-reduction. An intuitive explanation of the form we use is as follows. Suppose the problem of interest is P_1 and we know the inapproximability result for problem P_2. If, given an arbitrary instance of P_2, we can construct in polynomial time an instance of P_1 such that the optimal solution costs of the two instances are the same, and for any solution to the P_2 instance, we can construct in polynomial time a solution to the P_1 instance with the same cost, we can claim that the inapproximability result of P_2 extends to P_1. Otherwise, if there exists an algorithm to P_1 with a better approximation ratio, we can recover a better approximation solution to any P_2 instance by first solving the constructed P_1 instance using this better algorithm. This is contradictory to the inapproximability result we know for P_1. Note that if the inapproximability result depends on the input size, for example that for SCP is $\log n$ where n is the number of elements in the universe, we will have to introduce the size difference between the constructed P_1 instance and the P_2 instance to P_1's inapproximability result. However, this is not the case for our reduction given in Section III where the given SCP instance and the constructed Metric TVPP instance have the same number of surface patches.

References

[1] P. Blaer and P. Allen. View planning for automated site modeling. In *Proc. of IEEE International Conference on Robotics and Automation*, pages 2621– 2626, Orlando, USA, May 2006.

[2] A. Borodin and R. El-Yaniv. *Online Computation and Competitive Analysis*. Cambridge University Press, 1998.

[3] K. Bowyer and C. Dyer. Aspect graphs: an introduction and survey of recent results. *International Journal of Imaging Systems and Technology*, 2:315–328, 1990.

[4] W. Chin and S. Ntafos. Watchman routes in simple polygons. *Discrete and Computational Geometry*, 6(1):9–31, 1991.

[5] N. Christofides. Worst-case analysis of a new heuristic for the travelling salesman problem. Technical report 338, Grad School of Indistrial Administration, Carnegie Mellon University, 1976.

[6] T. Danner and L. Kavraki. Randomized planning for short inspection paths. In *Proc. of IEEE International Conference on Robotics and Automation*, pages 971 – 976, 2002.

[7] U. Feige. A threshold of $\ln n$ for approximating set cover. *Journal of the ACM*, 45(4):634 – 652, July 1998.

[8] S. Fekete, R. Klein, and A. Nuchter. Online searching with an autonomous robot. In *Proc. of Workshop on Algorithmic Foundation of Robotics*, pages 350–365, 2004.

[9] M. Garey and 1979. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W.H. Freeman & Co., New York, 1979.

[10] H. Gonzalez-Banos and J. Latombe. A randomized art-gallery algorithm for sensoer placement. In *ACM Symposium on Computational Geometry*, pages 232 – 240, 2001.

[11] V. Isler, S. Kannan, and K. Daniilidis. Local exploration: online algorithms and a probabilistic framework. In *Proc. of IEEE International Conference on Robotics and Automation*, pages 1913 – 1920, 2003.

[12] L. Kavraki, P. Svestka, J. Latombe, and M. Overmars. Probabilistic roadmaps for path planning in high-dimensional configuration spaces. *IEEE Transactions on Robotics and Automation*, 12(4):556–580, 1996.

[13] B. Korte and J. Vygen. *Combinatorial Optimization*. Springer-Verlag, 2000.

[14] J. Latombe. *Robot Motion Planning*. Kluwer Academic Publishers, 1991.

[15] W. Scott, G. Roth, and J. Rivest. View planning with a registration component. In *Proc. of the 3D Imaging and Modeling Conference (3DIM)*, pages 127–134, Québec, Canada, May 28 - June 1 2001.

[16] W. Scott, G. Roth, and J. Rivest. View planning for automated three-dimensional object reconstruction and inspection. *ACM Computing Surveys*, 35(1):64–96, March 2003.

[17] X. Tan. Approximation algorithm for the watchman route and zookeeper's problems. *Discrete Applied Mathematics*, 136(2-3):363–376, 2004.

[18] V. Vazirani. *Approximation algorithms*. Spinger, 2001.

[19] P. Wang, R. Krishnamurti, and K. Gupta. View planning problem with combined view and traveling costs: Problem formulation, hardness of approximation, and approximation algorithms. Technical Report TR2006-17, Simon Fraser University, Burnaby, B.C., Canada, May 2006. A verson also submitted to ICRA 2007.