Correlated Orienteering Problem and its Application to Persistent Monitoring Tasks

Jingjin Yu Mac Schwager Daniela Rus

***Abstract*—We propose the Correlated Orienteering Problem (COP) as a novel non-linear extension to the classic Orienteering Problem (OP). With the introduction of COP it becomes possible to model the planning of informative tours for the persistent monitoring of a spatiotemporal field with time-invariant spatial correlations using autonomous mobile robots, in which the robots are range- or time-constrained. Our focus in this paper is QCOP, a quadratic COP instantiation that looks at correlations between neighboring nodes in a node network. The main feature of QCOP is a quadratic utility function capturing the said spatial correlation. We solve QCOP using mixed integer quadratic programming (MIQP), with the resulting anytime algorithm capable of planning multiple disjoint tours that maximize the quadratic utility. In particular, our algorithm can quickly plan a near-optimal tour over a network with up to 150 nodes. Beside performing extensive simulation studies to verify the algorithm's correctness and characterize its performance, we also successfully applied QCOP to two realistic persistent monitoring tasks: *(i)* estimation over a synthetic spatiotemporal field, and *(ii)* estimating the temperature distribution in the state of Massachusetts.**

I. Introduction

Consider an application scenario in which unmanned aerial vehicles (UAVs) with on-broad cameras are dispatched to monitor traffic at intersections in a large city. Mobile aerial vehicles generally have limited fuel supply and therefore, limited flying time. At the same time, traffic events, such as congestion, tend to have strong spatial correlations, i.e., if the vehicle density at an intersection is high, the same is often true at nearby intersections. Therefore, blanket coverage of intersections following the road network's topological structure may offer little additional information. As aerial vehicles are not restricted to travel along roads, routes with carefully selected, not necessarily adjacent intersections can offer a better picture of the current traffic situation per unit of traveled distance. The following question then naturally arises: how to plan the best tours for the UAVs so that they can collect the maximum amount of information per flight? Such scenario (see Fig. 1) is far from unique: identical settings appear when one wants to deploy autonomous marine vehicles to collect samples for the detection of water pollution events such as oil spills, or event when a political candidate wants to maximize her reach given limited travel and time budgets.

J. Yu is with the Department of Computer Science, Rutgers University at New Brunswick. E-mail: jingjin.yu@cs.rutgers.edu. M. Schwager is with the Department of Aeronautics and Astronautics, Stanford University. E-mail: schwager@stanford.edu. D. Rus with the Computer Science and Artificial Intelligence Lab, the Massachusetts Institute of Technology. E-mail: rus@csail.mit.edu. This work was supported by ONR projects N00014-12-1-1000 and N00014-09-1-1051, and a startup fund from Rutgers University.

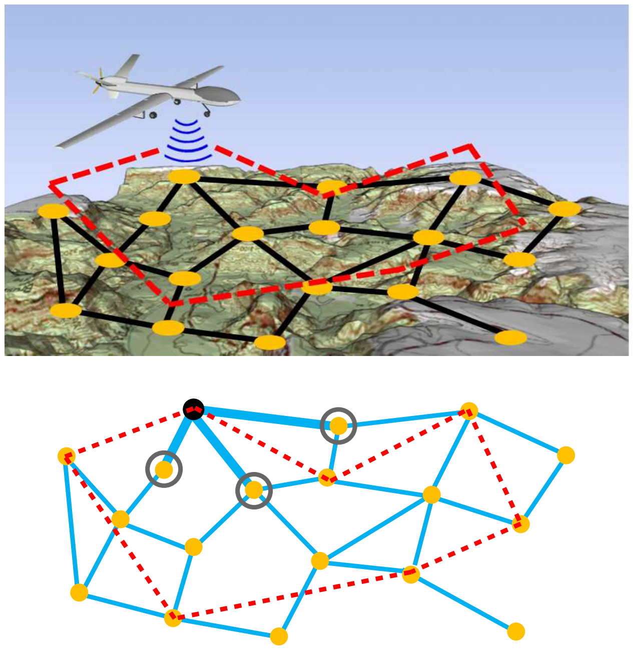


Fig. 1. [top] A surveillance scenario in which a range-limited UAV must cover a large area. [bottom] The abstracted node network and the tour (dashed lines) taken by the UAV. As a measurement is made from the UAV at a node (e.g., the black node), spatial correlation yields partial information about its neighbors (e.g., the three nearby, circled nodes). Following the given tour, much shorter than a traveling salesman (TSP) tour, the UAV can gather at least partial information of all nodes in the network.

In this paper, we propose the Correlated Orienteering Problem (COP) for capturing the above-mentioned scenarios, motivated by its potential application toward autonomous persistent surveillance and monitoring using mobile robots. COP has two defining elements: *(i)* the target domain may be represented as a network with spatial correlation between nearby nodes, and *(ii)* each mobile robot has limited travel *budget*, such as distance, that is often insufficient for blanket coverage. The goal, under these conditions, is to plan budget-limited tour(s) for the robot(s) to maximize the total information gain as measured by some *reward function*. Convexity or submodularity (of the information gain) over the underlying domain are not assumed. To demonstrate the modeling power of COP, we focus on Quadratic Correlated Orienteering Problem (QCOP) as a concrete instantiation of COP that only examines time-invariant spatial correlations between neighbors. Then, mixed integer quadratic programming (MIQP) models are developed for solving QCOP for both single and multiple robots. Our comprehensive evaluation suggests that COP and QCOP nicely capture spatial correlations and the *anytime* MIQP algorithm quickly produces approximate solutions to

QCOP for instances with about 150 nodes with good optimality guarantees. COP and QCOP are NP-hard problems.

Related work. A problem closely related to COP is the *Orienteering Problem* (OP) [1]. OP, as indicated by its name, has its origin from orienteering games [2], [3]. In such a game, rewards of uniform or varying sizes are spatially scattered. To collect a reward, a player must physically visit the location where the reward is placed to "pick it up". The goal for a player or a team of players is to plan the best path(s) to gather the maximum possible reward in the allocated time. Thus, OP can be viewed as a variation of both the Knapsack Problem (KP) [4] and the Traveling Salesman Problem (TSP) [5]. Research on OP has yielded effective algorithms for solving many versions of the problem, including the *Team Orienteering Problem* (TOP) [2], in which multiple tours must be planned.[1] For a detailed account of OP, see [1]. The key (and crucial) difference between OP and COP is that OP assumes that neighboring nodes have no correlation.

The information collection perspective of COP is tightly linked to informative path and policy planning problems. The literature on informative path and policy planning for persistent monitoring is fairly rich, covering theories, systems, algorithm designs, and applications. Fundamental limits as well as provably correct algorithms were established for a variety of persistent monitoring problems, for example in graph-based settings [6], pertaining to minimalism process design [7], for the monitoring of a continuously evolving scalar field [8], [9], or for stochastic data harvesting [10], [11], [12]. At the same time, comprehensive systems have been designed to address specific application domains, such as aerial [13], [14], [15], underwater [16], and multi-domain [17] applications. On work most closely related to ours, in [6], iterative TSP paths are planned to minimize the maximum latency across all nodes in a connected network. The authors show that the approach yields $O(\log n)$ approximation on optimality in which n is the number of nodes in the network. The problem of generating speed profiles for robots along predetermined cyclic paths for keeping bounded the uncertainty of a varying field is addressed in [9], in which the authors characterize appropriate policies for both single and multiple robots. In [18], decentralized adaptive controllers were designed to morph the initial closed paths of robots to focus on regions of high importance.

Sampling based planning methods (e.g., PRM, RRT, RRT* and their variations [19], [20], [21]) have also been applied to informative path planning problems. In [22], Rapidly-Exploring Random Graphs (RRG) are combined with *branch and bound* methods for planning the most informative path for a mobile robot. In [8], the authors tackle the problem of planning a cyclic trajectory for the estimation of a time-varying Gaussian Random Field, using a variation of RRT called Rapidly-Expanding Random Cycles (RRC).

Lastly, our problem, and OP in general, also has a *coverage* element. Coverage of a two-dimensional region has been extensively studied in robotics [23], [24], [25], [26], [27], as well as in purely geometric settings, for example, in [28], where the proposed algorithms compute the shortest closed routes for the continuous coverage of polygonal interiors under an infinite visibility sensing model. Coverage with limited sensing range was also addressed [29], [30], with recent progress on approximation algorithms [31].

Contribution. Our work brings the following contributions:

- We introduce COP as a novel non-linear extension to OP, to model and harness time-invariant spatial correlations frequently present in informative path and policy planning problems. In particular, our formulation addresses the challenge of planning information maximizing tours for single and multiple robots under a limited travel budget.
- We provide complete mixed integer quadratic programming (MIQP) models for solving QCOP, a quadratic COP instantiation. The MIQP models yield anytime algorithms that can effectively compute tour(s) over networks with tens to well over a hundred of nodes, for the (near-) optimal estimation of the underlying spatially correlated spatiotemporal fields. Our models and algorithms are shown to be effective over both synthesized benchmarks and real world data.

In comparison to the conference publication [32], the current paper provides a thorough treatment of COP, starting from a more general version of COP and the specialized QCOP instantiation. From the perspective of algorithmic solutions, we have developed an anytime version of the algorithm which provides a significant boost to computational speed, allowing much larger problems to be solved with good solution quality. In the simulation study, a much more comprehensive evaluation of the algorithmic performance as well as simulations on real temperature data are included. In particular, the superior performance of COP (QCOP) over OP is fully demonstrated.

Organization: The rest of the paper is organized as follows. In Section II, we formally introduce COP and QCOP. In Section III, we derive MIQP-based anytime algorithms for solving QCOP for single and multiple robots. In Sections IV, we perform extensive computational experiments to verify the correctness and evaluate the performance of our algorithmic solutions. We then illustrate how QCOP may be applied to solve realistic persistent monitoring tasks in Section V and conclude the paper in Section VI. Table I lists symbols that are frequently used in the paper.

II. Problem Statement

Due to spatial and temporal variations, spatiotemporal fields can be highly complex and dynamic. However, in applications involving large domains (e.g., terrains, road networks, forests, oceans, and so on), the underlying spatial structure often does not change quickly, limiting the variation of the field in the spatial domain. The observation motivates us to work with the premise that nearby nodes have mostly *time-invariant* spatial correlations[2], even though the overall field may change significantly over time. Assuming time-invariant spatial correlations and the field remains relatively static during a trip of the robot(s), the *Correlated Orienteering Problem* (COP) is

[1]Henceforth, we use *Orienteering Problem* (OP) as a blanket term to cover all additive/linear utility orienteering problems, which include TOP.

[2]Here, we use the broad meaning of correlation, which could be, but is not necessarily, the correlation of random variables.

TABLE I
LIST OF FREQUENTLY USED SYMBOLS AND THEIR INTERPRETATIONS.

$V = \{v_i\}$	Node or point of interest, $\|V\| = n$
$G = (V, E)$	Node network
$\mathbf{p}_i$	The two dimensional coordinate of v_i
r_i	Utility of knowing complete information about v_i
$\psi(v_i, t)$	Time-varying scalar field
A	$\{a_1, \ldots, a_m\}$, a set of mobile robots
v_{b_k}	Start and end node for robot a_k
c_k	Travel budget for robot a_k
Π	$\pi_1, \ldots, \pi_m$, a set of robot tours
$J(\Pi)$	The cost function for a given set of robot tours
w_{ij}	The weight measuring v_i's influence on v_j
x_i	Binary variable indicating whether v_i is on a tour
x_{ij}	Binary variable indicating whether v_j is visited immediately after v_i
x_{ijk}	Binary variable indicating whether v_j is visited immediately after v_i, by robot a_k
u_i	Integer variable, $2 \le u_i \le n$, the order of v_i in a tour path, if used
u_{ik}	Integer variable, $2 \le u_{ik} \le n$, the order of v_i in a tour path, if used, by robot a_k
d_{ij}	Travel cost from v_i to v_j, which may be asymmetric
α_{ij}, β_{ij}	Linear regression coefficients

stated. Then, we introduce a special COP with a quadratic cost function induced by independent, linear correlations between adjacent nodes. We call this instantiation of COP the *Quadratic Correlated Orienteering Problem* (QCOP). Below, COP and QCOP are formally defined.

A. Correlated Orienteering Problem

Let $V = \{v_1, \ldots, v_n\}$ be a set of spatially distributed nodes in some workspace $W \subset \mathbb{R}^2$. Each $v_i \in V$ is associated with coordinates $\mathbf{p}_i \in \mathbb{R}^2$. Let $\psi(\mathbf{p}, t), \mathbf{p} \in \mathbb{R}^2$ be a *time-varying scalar field* over W. The values on the nodes of V, with a slight abuse of notation, are denoted as $\psi(v_i, t), 1 \le i \le n$. We assume that the spatial relationship among the nodes of V, as determined by ψ, induces a directed graph G over V. More precisely, $G = (V, E)$ has an edge (v_j, v_i) (i.e., $\overrightarrow{v_j v_i}$) if and only if $\psi(v_i, t)$ is dependent on $\psi(v_j, t)$. That is, let $N_i = \{v_{i_1}, \ldots, v_{i_k}\}$ be the set of neighbors of v_i in G, with edges pointing to v_i, then for some time-invariant f_i,

$$\psi(v_i, t) = f_i(\psi(v_{i_1}, t), \ldots, \psi(v_{i_k}, t)). \quad (1)$$

There are $m \ge 1$ mobile robots, $A = \{a_1, \ldots, a_m\}$. Each robot follows the standard single integrator dynamics with constant magnitude on the control input. Each robot has a *base node* where it must start and end its tour. For a robot a_k, let its base be $v_{b_k} \in V$. Let $c : A \to \mathbb{R}^+, a_k \mapsto c_k$ represent the maximum distance (budget) the mobile robots can travel before they must return to their respective bases. Other than the single integrator dynamics and the distance budget constraints, these robots have no other motion constraints. In particular, a robot is not constrained to the implicitly defined graph G and can travel in a straight line between any $\mathbf{p}_i, \mathbf{p}_j \in W$ as permitted by the travel distance constraint.

Remark. We assume in this paper that the base nodes are fixed to have a more focused discussion. If there is flexibility in the base node, then techniques developed in [33] can be used to allow automatic selection of the optimal base node. $\triangle$

A mobile robot can measure $\psi(v_i, t)$ when the robot reaches $v_i \in V$. Let $\Pi = \{\pi_1, \ldots, \pi_m\}$ be a set of tours. Each $\pi_k \in \Pi$ is a cycle for a_k that goes through a set of nodes including v_{b_k}. The overall quality of Π is measured by some utility function $J : \{\Pi\} \to \mathbb{R}^+ \cup \{0\}$ that maps path sets to reals. We do not consider sensor measurement noises in the current paper.

A *Correlated Orienteering Problem* (COP) is specified by a 4-tuple $(V, \psi, \{v_{b_k}\}, J)$. The task is to find a policy Π that maximizes $J(\Pi)$. Note that ψ is fixed but generally unknown; it can only be measured (by mobile robots) at particular locations and time instances. An illustrative and qualitative example of what COP aims to achieve is given in Fig. 2.

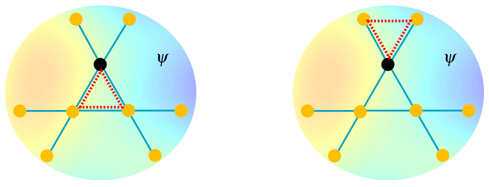


Fig. 2. Two tours with the same budget over the same node network and ψ. Here, all edges have unit length. Given a budget of 3 with the black node as the starting node, intuitively, with correlations of node values between nearby nodes, if we want to estimate ψ over all 9 nodes, the tour (dashed path) on the left is likely better because each of the 9 nodes is on the tour or adjacent to some node visited by the tour. COP aims to allow the planning of such a tour Π through the maximization of a properly defined $J(\Pi)$.

Remark. At a first glance, COP may appear to mimic a problem whose underlying process is a Markov Decision Process (MDP). In spite of some similarities (e.g., like in an MDP-based problem, in COP, the robots take actions to go to different physical states and the information to be collected follows some distribution), a key difference is that it is never beneficial to revisit a node in a COP instance but revisiting a state in an MDP problem can be rewarding. This is also a key source of computational difficulty associated with COP because dynamic programming techniques that are useful in solving MDP problems can no longer be applied. $\triangle$

B. Quadratic Correlated Orienteering Problem

Following the definition of COP, we now describe an instantiation of COP with a quadratic utility (reward) function. If tours of the m robots, Π, go through a node $v_i \in V$, then a utility of $r(v_i)$ is collected, defined according to the mapping $r : V \to \mathbb{R}^+, v_i \mapsto r_i$. The robots do not gain additional utility for revisiting v_i. To represent the total utility of QCOP, let $\{x_1, \ldots, x_n\}$ be n binary variables, with $x_i = 1$ if and only if v_i (i.e., $\mathbf{p}_i$) is on some $\pi \in \Pi$. To incorporate correlations among the nodes during the tour planning phase while also rendering the formulation more concrete, we let ψ (and therefore $\{f_i\}$) and J have the following instantiation. Let the weight function $w : E \to \mathbb{R}^+, (v_j, v_i) \mapsto w_{ji}$ represent the *effectiveness* of using $\psi(v_j, t)$ to estimate $\psi(v_i, t)$. One may view w_{ji} as representing the amount of information that $\psi(v_j, t)$ has about $\psi(v_i, t)$, independent of other nodes. Let $w_{ii} = 0$ by definition. The utility that can be collected over a node v_i is defined as

$$r_i \left(x_i + \sum_{\forall v_j, w_{ji} \neq 0} w_{ji} x_j (x_j - x_i) \right), \quad (2)$$

in which the quadratic term $x_j(x_j - x_i)$ is non-zero if and only if $x_j = 1$ and $x_i = 0$. Essentially, (2) states that correlations are only relevant for v_i if v_i is not directly visited by a robot. Obviously, for each $0 \le i \le n$, $\sum_{v_j \in N_i} w_{ji} \le 1$. Note that $w_{ij} = w_{ji}$ is not assumed. Over a set of tours for the robots, Π, the total utility to be maximized is

$$J(\Pi) = \sum_{i=1}^{n} \left(r_i x_i + \sum_{\forall v_j, w_{ji} \neq 0} r_i w_{ji} x_j (x_j - x_i) \right). \qquad (3)$$

We observe that the utility function (3) defines a natural quadratic extension to OP, which has a linear utility $\sum_{i=1}^{n} r_i x_i$. We denote this COP instantiation as the *Quadratic Correlated Orienteering Problem* (QCOP). By assuming independence among w_{ij}'s in formulating QCOP, we trade model fidelity for computational efficiency. Nevertheless, QCOP remains a computationally difficult problem.

Lemma 1 *COP and QCOP are NP-hard.*

PROOF. We limit the case to a single robot ($m = 1$). First, we establish that OP is NP-hard (note that OP has been shown to be NP-hard [1]; a quick proof is provided here for completeness). For this purpose we reduce Euclidean TSP [34] to OP. An Euclidean TSP instance is fully specified by a set of nodes V in the plane with a real number c, with the decision problem being whether c is enough for finding a TSP tour through all $|V|$ nodes. We may reduce the Euclidean TSP instance to an OP instance via setting each node $v_i \in V$ to have a visiting reward of $r_i \equiv 1$. The OP decision problem is then simply the following: is a budget c enough for collecting a total reward of $|V|$? Clearly, a reward of $|V|$ is possible if and only if c is sufficient for finding a Euclidean TSP tour over V. Because Euclidean TSP is NP-hard [34], so is OP.

To show that COP and QCOP are NP-hard, we point out that OP is a special case of COP and QCOP in which the network has only nodes (i.e., the set V) and no connections between them. To reduce an OP instance to a COP instance, we may simply let the function set ψ to the empty set and ask the question: is there a tour with a distance of no more than c that yields a reward of $J = |V|$? To reduce an OP to a QCOP, we may equivalently set all weights $\{w_{ij} : i \neq j\}$ to be zero and ask the same question. □

Remark. For completeness, we point out that the utility given by (3) can be converted to have a linear form. To do this, for each (i, j) pair, define a binary variable z_{ij}. We then enforce $z_{ij} = 1$ if and only if $x_i(x_i - x_j) = 1$ by adding two constraints $z_{ij} \le x_i$ and $z_{ij} \le \frac{x_i - x_j + 1}{2}$ to the model. (3) is then updated to

$$J(\Pi) = \sum_{i=1}^{n} \left(r_i x_i + \sum_{\forall v_j, w_{ji} \neq 0} r_i w_{ji} z_{ji} \right). \qquad (4)$$

Having a linear utility effectively transforms our MIQP model into a mixed integer linear programming (MILP) model. We did not include result on this MILP model in this paper as simulation suggests that it does not demonstrate clear computational advantage over the MIQP model. △

C. Key Difference Between OP and COP

Whereas COP and QCOP models appear similar to OP, the quadratic cost causes COP to behave differently from OP with linear additive cost. In particular, COP encourages the spatial exploration of the underlying domain while OP is largely a one dimensional model. We illustrate the difference, intuitively, with the help of Fig. 3. For simplicity, a QCOP setup is used with weights w_{ji} set to be $1/|N_i|$. For example, if node i has three neighbors, then all w_{ji}'s are set to $1/3$. Assuming unit edge lengths and $r_i = 1$, for tours with maximum length constraint of 2-6, the objective function (3) induces the tour classes given in the top row of Fig. 3. Here, by *class*, we mean that there can be similarly shaped tours with the same $J(\Pi)$. These tours yield $J(\Pi)$ values of $4.0, 4.5, 5.7, 7.3$, and 9 (maximum possible), respectively. As a comparison, possible (classes of) optimal tours for OP with the same budgets are given in the bottom row of Fig. 3. For tour distance 2, either the red or the green tour are the same under OP. However, they have $J(\Pi)$ values of 4 and 3.18 under the QCOP model, respectively. This means that, on average, tours found with OP do not provide the spatial coverage to best harness correlation. Similar scenario happens for tour distance of 3, 5, and 6. More thorough comparative studies are provided in Section IV.

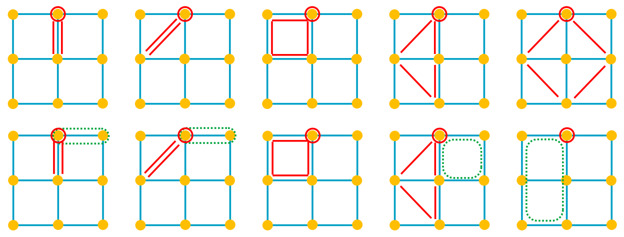


Fig. 3. [top row] Single robot tour classes under QCOP model with maximum distance constraint set to $2, 3, 4, 5$, and 6, in that order, from left to right. The circled node is the base node. [bottom row] Possible classes of tours under OP model with the same distance constraints from 2-6. Both solid (red) tours and dotted (green) tours are optimal solutions.

III. MIXED INTEGER QUADRATIC PROGRAMMING BASED ANYTIME ALGORITHMS FOR QUADRATIC COP

We develop a *mixed integer quadratic programming* (MIQP) model for solving QCOP with the quadratic objective function specified by (3). We start from the case of a single mobile robot, followed by the case of multiple robots. An algorithm outline then follows discussions on the MIQP approach's anytime property.

A. MIQP Model for a Single Robot

For a single robot ($m = 1$), we adapt the constraints for OP from [1], which yields a compact model. Whereas our description of the model is self-contained for completeness, more background knowledge on the development of linear OP models can be found in [1] and the references within.

Without loss of generality, let the robot start from v_1. Let x_{ij} be a binary variable with $x_{ij} = 1$ if and only if the robot visits v_j immediately after it visits v_i, which does not require v_i and v_j to be correlated. As it is never beneficial to revisit a node, the robot visits any node at most once (except the base node). The number of times that the robot enters (resp. leaves) a node i can be written as $\sum_{j=1,j\neq i}^{n} x_{ji}$ (resp. $\sum_{j=1,j\neq i}^{n} x_{ij}$); both quantities are at most one. The tour constraint requires $\sum_{j=1,j\neq i}^{n} x_{ji} = \sum_{j=1,j\neq i}^{n} x_{ij}$. Recall that x_i is the binary variable indicating whether v_i is visited. We have (note that $x_1 = 1$)

$$\sum_{j=1,j\neq i}^{n} x_{ij} = \sum_{j=1,j\neq i}^{n} x_{ji} = x_i \leq 1, \quad \forall 2 \leq i \leq n, \tag{5}$$

The constraints from (5) and $x_1 = 1$ ensure that the robot will take a tour starting and ending at v_1. They do not, however, prevent multiple disjoint tours from being created. To prevent this from happening, let $2 \leq u_i \leq n$ be *integer* variables for $2 \leq i \leq n$ and add the following constraints

$$u_i - u_j + 1 \leq (n-1)(1-x_{ij}), \quad 2 \leq i, j \leq n, i \neq j. \tag{6}$$

Since $u_i - u_j + 1 \leq n-1$ for all $2 \leq u_i, u_j \leq n$, (6) can only be violated if $x_{ij} = 1$. The condition $x_{ij} = 1$ only holds if v_j is visited immediately after v_i is. Setting u_i to be the order with which v_i is visited on the tour, if $x_{ij} = 1$, then $u_i - u_j + 1 = 0$, satisfying (6). On the other hand, if $v_{ij} = 1$ and there is another tour other than the one starting from v_1, then the RHS of (6) equals zero. We then must have $u_i < u_j$. However, this condition cannot hold for all consecutive pairs of nodes on a cycle. Thus, (6) allows a single cycle containing v_1.

The tour distance constraint can be enforced via

$$\sum_{i=1}^{n} \sum_{j=1,j\neq i}^{n} x_{ij} d_{ij} \leq c_1, \tag{7}$$

in which d_{ij} is the distance from v_i to v_j and c_1 is the tour distance constraint for the single robot. Note that the distance d_{ij} can be asymmetric. Moreover, it is straightforward to incorporate sensing cost at a node v_i by absorbing that cost into d_{ij} for all $j \neq i$. Alternatively, if the sensing cost is not compatible with the travel cost, an additional cost constraint can be added as well. Putting things together, we obtain a complete MIQP model for QCOP, summarized below.

$$\begin{aligned}
&\text{maximize } J(\Pi) = \sum_{i=1}^{n} \left(r_i x_i + \sum_{1 \leq j \leq n, w_{ji} \neq 0} r_i w_{ji} x_j (x_j - x_i) \right), \\
&\text{subject to} \\
&\sum_{j=1,j\neq i}^{n} x_{ij} = \sum_{j=1,j\neq i}^{n} x_{ji} = x_i \leq 1, 1 \leq i \leq n, x_1 = 1, \\
&u_i - u_j + 1 \leq (n-1)(1-x_{ij}), \quad 2 \leq i, j \leq n, i \neq j, \\
&\sum_{i=1}^{n} \sum_{j=1,j\neq i}^{n} x_{ij} d_{ij} \leq c_1.
\end{aligned} \tag{8}$$

B. MIQP Model for Multiple Robots

Extending a single tour to multiple tours is straightforward. To accommodate m robots, the variables $\{x_{ij}\}$ and $\{u_i\}$ are extended to x_{ijk} and u_{ik}, with $1 \leq k \leq m$ representing the labels of the robots. Then, (5) becomes (note that $x_{b_k} = 1$)

$$\sum_{j=1,j\neq i}^{n} x_{ijk} = \sum_{j=1,j\neq i}^{n} x_{jik} \leq 1, \quad \forall 1 \leq i \leq n, 1 \leq k \leq m, \tag{9}$$

$$\sum_{k=1}^{m} \sum_{j=1,j\neq i}^{n} x_{ijk} = \sum_{k=1}^{m} \sum_{j=1,j\neq i}^{n} x_{jik} = x_i \leq 1, \quad \forall 1 \leq i \leq n. \tag{10}$$

(5) splits into (9) and (10) because we need (9) to ensure that a node is used by at most one robot. With only (10) but not (9), it can happen that one robot enters a node while a different robot exits the same node, which should not happen.

The constraints on u_{ik} become (for all $1 \leq k \leq m$)

$$u_{ik} - u_{jk} + 1 \leq (n-1)(1-x_{ijk}), \quad i, j \neq b_k, i \neq j, 1 \leq i, j \leq n. \tag{11}$$

The traveled distance constraint, (7), becomes

$$\sum_{i=1}^{n} \sum_{j=1,j\neq i}^{n} x_{ijk} d_{ij} \leq c_k, \quad 1 \leq k \leq m. \tag{12}$$

Finally, the utility function (3) remains unchanged.

Remark. The above MIQP model does not allow two robots to start from the same base. We can accommodate such scenarios by modifyings (9) and (10) accordingly.

C. Anytime Property

An interesting and extremely useful property coming from the MIQP-based method is its *anytime* property. Integer programming solvers, using branch-and-bound or related algorithms, work with a polytope containing all feasible solutions to a (relaxed) continuous optimization problem. Roughly speaking, solvers break the polytope into increasingly smaller pieces and recursively check whether a piece may contain the optimal solution. At any given time, lower and upper bounds on the objective function are maintained for each active piece of the polytope, which allows the estimation of how optimal the current best solution (called the *incumbent*) is. The relative difference is the *optimality gap*. An optimal solution is found when the gap drops to zero. Under the context of QCOP, because there is a trivial feasible solution (i.e., a tour that does not go anywhere) yielding an already non-zero $J(\Pi)$, an anytime algorithm is obtained naturally. As QCOP is computationally challenging, having an anytime algorithm with known optimality gap is highly beneficial. Because it generally takes increasingly more time for an MIQP solver to find better and better solutions, having the option to stop early at a sub-optimal solution can save a significant amount of time. As we know conservatively how optimal the incumbent is at an arbitrary time instance during the execution process, we may stop running the algorithm and have the confidence that a desired level of optimality is achieved.

D. Algorithm Outline

Algorithm 1 outlines the MIQP-based anytime algorithm for solving QCOP. In lines 1–2, the MIQP model is set up and solved to obtain the desired set of tours for the robots. The robots then follow the planned tours and collect data as they pass over the nodes on these tours in line 3. The collected data $\{\psi'(v_i, t_s)\}$ is subsequently updated through correlation to yield the final estimated node values. Note that UpdateNodeEstimate$(\cdot)$ is only determined when QCOP is connected to a particular spatiotemporal field $\psi(\cdot,\cdot)$ (an example is given in Section V).

Algorithm 1: QCOP Estimation

```
Input  : $G = (V,E)$: node network, $|V| = n$
         $W = \{w_{ij}\}, 1 \le i,j \le n$: correlation weights
         $V_B = \{v_{b_1},\dots,v_{b_m}\}$: base nodes
         $C = \{c_1,\dots,c_m\}$: travel distance budgets
         $gap$: optimality tolerance
Output: $\psi'(v_i,t_s), 1 \le i \le n$: node estimation at time $t_s$

  %Compute robot tours
1 $M \leftarrow$ SetUpModel$(G,W,V_B,C)$ ;          %Set up model
2 $\{\pi_1,\dots,\pi_{b_m}\} \leftarrow$ SolveModel$(M,gap)$; %Compute tours

  %Run tours and collect data
3 $\{\psi'(v_i,t_s)\} \leftarrow$ CollectData$(\{\pi_1,\dots,\pi_{b_m}\})$

  %Estimate value at unvisited nodes
4 $\{\psi'(v_i,t_s)\} \leftarrow$ UpdateNodeEstimate$(\{\psi'(v_i,t_s)\})$

5 return $\{\psi'(v_i,t_s)\}$
```

IV. Computational Study: Effectiveness and Efficiency of MIQP-Based Algorithms for QCOP

In this section, we perform computational experiments to gauge the effectiveness and efficiency of the MIQP-based algorithm at solving QCOP. In particular, we highlight the comparative advantage of QCOP over OP. For this purpose, only lines 1–2 of Algorithm 1 are evaluated. We implemented both OP and QCOP models in JAVA interfacing with Gurobi [35]. All computations were performed on a commodity PC with an Intel Core-i7 5820 CPU under an 8GB JavaVM. All reported computation time is the wall-clock time.

A. *Effectiveness in Capturing Spatial Correlation*

We have briefly shown the expansive coverage induced by QCOP in Fig. 3. Here, additional examples are provided to further demonstrate this key property of QCOP, over both regular grids and irregular networks.

Regular grids. Under the same setup used for obtaining Fig. 3 (i.e., unit edge length, unit reward, and neighborhood-based weighting scheme), tours on a 4×4 grid are computed and shown in Fig. 4 for a single robot (top row) and for two robots (bottom row) under various travel budgets. For multiple robots, we let each robot have the same distance budget. Non-uniform budgets, however, are supported by our model by default. We note that, in general, the base nodes (circled) are manually selected to have them reasonably separated. As we mentioned, base node selection can be fully automated but is beyond the scope of this paper. We observe that the objective of QCOP appears to expand to assume a 2D structure whenever possible. From Fig. 4, we observe that QCOP causes the tours to be spread out over the two-dimensional domain.

Irregular node network In the second experiment, we employ the irregular node network from Fig. 1. The bounding rectangle of the network is set to be approximately 13 units by 8 units. For this network, weights (w_{ij}'s) are also computed based on the number of neighbors. Up to three robots were attempted with the longest running time being about 22 seconds. The results are given in Fig. 5. We again observe that QCOP ensures tour paths to spread out spatially, allowing them to effectively capture correlations among the nodes, as desired.

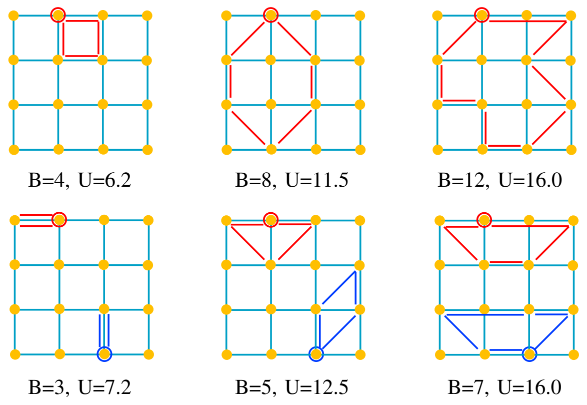


Fig. 4. [top] Single robot tours. [bottom] Two-robot tours. The numbers under each picture indicate the budget (B) per tour/robot and the total utility (U), respectively. Base nodes are marked with circles around the nodes.

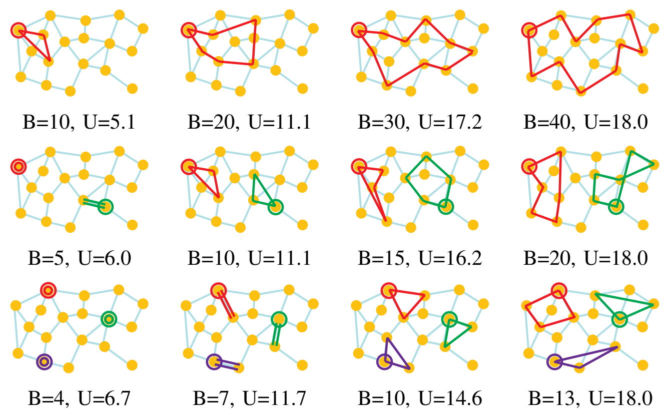


Fig. 5. Results from running MIQP models for QCOP on the irregular node network from Fig. 1. The numbers under each picture indicate the budget (B) per tour/robot and the total utility (U), respectively. Base nodes are marked with circles around the nodes.

B. *Comparison with Standard OP*

Fig. 3 and the associated discussion indicate that COP and QCOP produce tours more suitable for harnessing spatial correlations than OP does. We now establish this fact quantitatively, which is a key strength of QCOP (and COP).

Regular grids. We revisit regular grids, but on a larger scale, for comparing QCOP and OP. Although OP models are generally easier to solve computationally, for exact optimal solutions and a single robot, both QCOP and OP can only handle a grid size of 6×6 with a maximum of 2,500 seconds (wall-clock time). One set of exact solutions for QCOP and OP over a 5×5 grid is given in Fig. 6. Both were given a budget of 16. Similar to Fig. 3, the solution to QCOP covers the entire network more evenly. However, Fig. 6 shows more clearly that OP based solution does not optimize spatial coverage. The overall $J(\Pi)$s (for OP, after the linear cost based tour is computed, (3) is used to compute $J(\Pi)$) for QCOP and OP are 23.3 and 18.3, respectively.

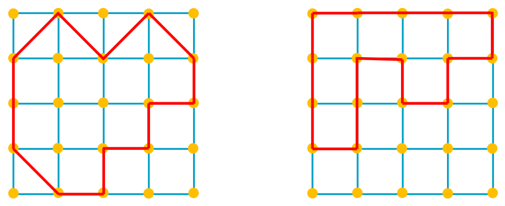
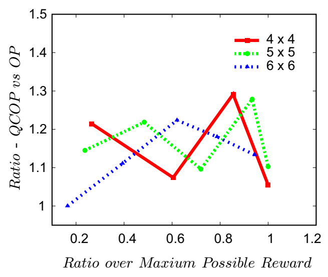


Fig. 6. Running QCOP (left) and OP (right) with the same budget of 16.

To provide a more complete picture, we plot the ratio of $J(\Pi)$ optimized by the QCOP model over that from the OP model, for grids with varying sizes, in Fig. 7. In the first (left) figure, grids of sizes 4×4 to 6×6 are considered, over which QCOP and OP can both compute optimal solutions under 2,500 seconds. In our computational experiment, for each fixed grid size and budget, two objective values, one for QCOP and one for OP, are produced. Denoting these two numbers as J_{COP} and J_{OP}, and assuming the grid has N nodes, then a point in the figure is computed as

$$\left(\frac{\max\{J_{COP},J_{OP}\}}{N}, \frac{J_{COP}}{J_{OP}}\right). \tag{13}$$

As an example, for the tours from Fig. 6, $N = 25$, $J_{COP} = 23.3$, $J_{OP} = 18.3$. These yield a data point $(0.93, 1.27)$. We group the data points by the grid size N to form lines. As we can see, QCOP produced tours that always outperform these given by the OP model. For this set of tests, QCOP shows an advantage around 15% with a peak difference of 29%.



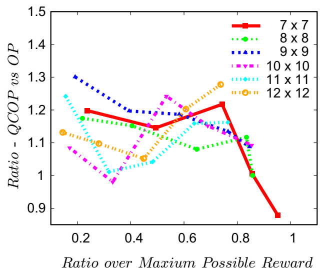


Fig. 7. Two sets of performance comparison of QCOP versus OP. [left] Exact computation over grids of size 4×4, 5×5, and 6×6. [right] QCOP with 20% gap versus OP 5% gap on larger networks.

The second (right) figure is constructed similarly over grids of sizes 7×7 to 12×12. These data points are sub-optimal solutions given the difficulty of QCOP and OP. For QCOP, a gap of 20% is used. Because OP is relatively easier to solve, we set a gap of 5% which puts QCOP at a disadvantage. Nevertheless, we again observe that in all but one cases, QCOP produces better tours for harnessing correlation. This set of tests yields an average advantage of 12% for QCOP with a peak of 30%. In practice, with additional computation time or better computing resources (MIQP solvers can be readily clustered), the advantage could be further boosted.

Irregular node network Over the irregular network from Fig. 1, Fig. 8 is produced following the same procedure used for generating Fig. 7. More budgets are incorporated in addition to those listed in Fig. 5. This simulation provides a data point for QCOP versus OP over both irregular networks and for multiple robots.

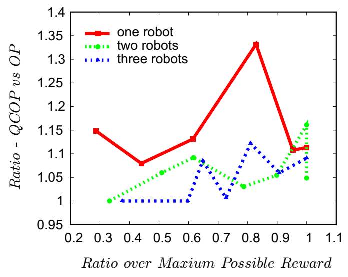


Fig. 8. Performance comparison of QCOP versus OP over the irregular network given in Fig. 1 for 1-3 robots.

We make three observations regarding Fig. 7 and Fig. 8. First, for very small and very large budgets, QCOP and OP are expected to produce similar results. For small budgets, not enough distance is present for providing substantial spatial coverage. On the other hand, large budgets saturate the utility gain. Second, for multiple robots, because robots are already spread out on the network, the advantage of QCOP becomes less obvious as compared to the single robot case. Third and most importantly, regardless of the network type, QCOP, even in the case of finding near-optimal solutions, consistently outperforms OP with an advantage of about 14% on average. The set of experiments therefore fully demonstrates that COP and QCOP achieve the goal of picking up the non-linear reward (utility) that OP is unable to collect.

C. Computational Performance

In this subsection, the computational performance of Algorithm 1 is put to test. A 2,500-second time limit is set for a single problem; the majority of runs took much less time. We emphasize that computational experiments, carried out on a commodity personal computer, demonstrate that our approach is fully capable of handling decent sized problems with about 150 nodes. Because MIQP solvers can be easily distributed across machines due to the inherent structure of branch-and-bound methods, larger instances can be readily tackled using multiple machines with longer computation time.

Single robot, regular grid For a single robot over regular grids, the MIQP model can handle networks with 36 nodes when an (exact) optimal solution is required, with all computation completed within 200 seconds (Table II; B, U, and T refer to distance budget, utility, and computation time, respectively). Some but not all cases attempted on the 7×7 grid can be solved within the time limit. When the anytime property is employed with a maximum 20% tolerance on optimality, regular network with 144 nodes can be solved (Table III). Because of the 20% maximum tolerance, as the utility is getting close to the maximum possible, more budget does not lead to increased utility. This happened once in Table III (the last two entries of the first row), as the utility falls within 20% of the maximum possible 49.

Because QCOP is NP-hard, it is unclear how to asymptotically characterize the computational advantage of the anytime

TABLE II
PERFORMANCE, ONE ROBOT, REGULAR GRIDS, EXACT SOLUTION

Grid		Trial # 1	2	3	4	5
4×4	B/U	3.2/4.3	6.4/9.7	9.6/13.7	12.8/16.0	16.0/16.0
	T(s)	0.03	0.34	0.29	0.05	0.04
5×5	B/U	4.0/5.9	8.0/12.1	12.0/18.0	16.0/23.3	20.0/25.0
	T(s)	0.02	2.4	4.4	0.94	0.3
6×6	B/U	4.8/5.92	9.6/14.2	14.4/22.3	19.2/28.3	24.0/34.0
	T(s)	0.05	27.2	13.4	97.6	189

TABLE III
PERFORMANCE, ONE ROBOT, REGULAR GRIDS, 20% GAP

Grid		Trial # 1	2	3	4	5
7×7	B/U	8.4/11.6	16.8/24.3	25.2/36.4	33.6/41.9	42.0/41.0
	T(s)	16.5	20.5	6.5	0.05	0.7
8×8	B/U	9.6/14.0	19.2/25.9	28.8/41.5	38.4/53.4	48.0/54.9
	T(s)	6.3	54.8	35.9	15.9	3.2
9×9	B/U	10.8/15.5	21.6/31.9	32.4/47.2	43.2/62.0	54.0/68.5
	T(s)	28.3	440	574	55.9	59.1
10×10	B/U	12.0/17.1	24.0/33.3	36.0/53.4	48.0/68.9	60.0/85.4
	T(s)	125	361	86.5	145	70.9
11×11	B/U	13.2/18.8	26.4/38.5	39.6/58.3	52.8/77.3	66/92.6
	T(s)	208	139	548	789	1874
12×12	B/U	14.7/20.8	19.3/40.3	44.9/64.7	58.7/87.4	73.3/106.2
	T(s)	90.3	718	728	808	553

property. Fig. 9 provides some intuition. Over the 6×6 grid and various budget-gap combinations, the required computation time for solving the resulting models is plotted. We observe here speedups from several folds to several hundreds folds, which are typical in our computational experiments.

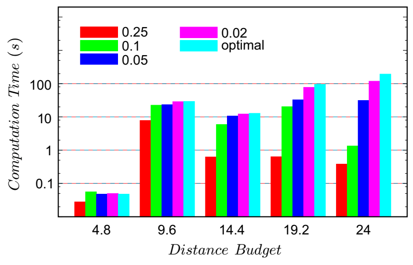


Fig. 9. Computation time for various budgets and optimality tolerance (gap) levels over the 6×6 grid.

Single robot, perturbed network As regular grids have particular structures, the performance on them may not represent typical scenarios. To ensure a fair evaluation of the MIQP model for QCOP, a more general set of examples is created. Starting from a random grid, several sources of significant perturbation are introduced:

1) For the eight-neighborhood of each node, edges are randomly selected to be present with 0.5 probability;
2) The locations of the nodes are perturbed up to ±30%;
3) The utility (i.e., r_i) of a node is perturbed up to ±25%.

After these modifications, the resulting network has a highly random structure. Fig. 10 provides an example with 28 nodes. The network is not necessarily connected.

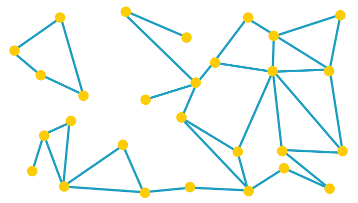


Fig. 10. A perturbed network with 28 nodes.

For comparison, exact computation was done over perturbed grids from 4×4 to 6×6. Given the randomized nature, each data point is collected as the average over 10 sequential randomized runs. The result, including standard deviations, summarized in Table IV, shows very similar performance characteristics seen in Table II. The standard deviations on computation times show that the difficulty of the randomized problems is highly variable.

TABLE IV
PERFORMANCE, ONE ROBOT, PERTURBED GRIDS, EXACT SOLUTION

Grid		Trial # 1	2	3	4	5
4	B	3.2	6.4	9.6	12.8	16
$\times$	U	4.7 ± 0.8	10.6 ± 1.0	15.0 ± 0.9	16.1 ± 0.5	16.0 ± 0.5
4	T(s)	0.01 ± 0.01	0.21 ± 0.05	0.28 ± 0.22	0.03 ± 0.02	0.03 ± 0.02
5	B	4	8	12	16	20
$\times$	U	6.0 ± 0.6	12.5 ± 0.9	19.5 ± 1.5	24.3 ± 0.8	25.2 ± 0.7
5	T(s)	0.03 ± 0.02	3.3 ± 2.1	5.2 ± 4.6	3.0 ± 3.2	0.2 ± 0.1
6	B	4.8	9.6	14.4	19.2	24
$\times$	U	7.5 ± 1.1	15.7 ± 1.5	25.9 ± 1.5	33.0 ± 1.8	35.7 ± 0.9
6	T(s)	0.2 ± 0.1	137 ± 120	217 ± 281	122 ± 273	57.7 ± 89.7

Results on larger networks with up to about 100 nodes, with each data point an average over 5 runs, are listed in Table V, computed using a sub-optimality tolerance of 20%. Since standard deviations do not add additional insights, they are omitted from this table. One of the five instances per data entry did not complete within 2500 seconds (the entry is marked with a † in Table V). Scaling on networks with over 100 nodes under a randomized setting proves to be challenging–it appears that, although infrequently, some very hard instances get created due to the randomization process.

Multiple robots Table VI lists the computational results on grids for two robots and with up to 25 nodes; one instance is not solved under the time limit. Because the MIQP model for multiple robots requires one extra set of variables for each extra robot, computing exact solutions for multiple robots quickly becomes more difficult as the number of robots increases. For three robots (we omit the limited detail here), an exact solution for 25 nodes is already difficult to compute within 2500 seconds for any budget. As expected, engaging the anytime property allows us to push further in the multi-robot case as well (Table VII and Table VIII). Using a 20% tolerance on optimality, we were able to solve all instances on

TABLE V
PERFORMANCE, ONE ROBOT, PERTURBED GRIDS, EXACT SOLUTION

Grid		Trial #				
		1	2	3	4	5
5×6	B/U	4/5.8	8/13.2	12/20.0	16/24.9	20/26.4
	T(s)	0.07	21.0	8.1	1.2	0.4
6×8	B/U	4.8/7.5	9.6/15.9	14.4/24.6	19.2/32.6	24/38.8
	T(s)	0.6	557	266	28.5	8.9
7×10	B/U	10/16.4	20/34.7	30/52.2	40/59.6	50/61.3
	T(s)	981	612	60.0	13.9	8.7
8×12	B/U	12/21.5[†]	24/42.6	36/61.6	48/77.0	60/83.4
	T(s)	889	1304	630	249	37.1

TABLE VI
PERFORMANCE, TWO ROBOTS, REGULAR GRIDS, EXACT SOLUTION

Grid		Trial #			
		1	2	3	4
4×4	B/U	3.2/8.5	4.8/11.8	6.4/16.0	8.0/16.0
	T(s)	0.01	7.2	0.5	0.02
5×5	B/U	4.0/11.8	6.0/16.8	8.0/–	10.0/25.0
	T(s)	16.7	2022	> 2500	14.8

the 6×6 network and half of the instances on the 7×7 grid. For three robots, a 20% gap allows us to solve all instances on the 5×5 grid and half of the instances on the 6×6 grid.

TABLE VII
PERFORMANCE, TWO ROBOTS, REGULAR GRIDS, 20% GAP

Grid		Trial #			
		1	2	3	4
4×4	B/U	3.2/8.5	4.8/11.3	6.4/13.5	8.0/16.0
	T(s)	0.02	2.5	0.02	0.01
5×5	B/U	4.0/11.8	6.0/16.8	8.0/20.8	10.0/21.3
	T(s)	8.0	76.0	0.4	0.2
6×6	B/U	4.8/11.8	7.2/20.2	9.6/26.2	12.0/30.3
	T(s)	188	1005	1004	0.9
7×7	B/U	5.6/–	8.4/–	11.2/31.3	14/39.2
	T(s)	> 2500	> 2500	951	1844

TABLE VIII
PERFORMANCE, THREE ROBOTS, REGULAR GRIDS, 20% GAP

Grid		Trial #			
		1	2	3	4
4×4	B/U	2.1/10.7	3.2/12.1	4.3/13.8	5.3/14.1
	T(s)	0.01	0.02	0.21	0.04
5×5	B/U	2.7/11.0	4.0/17.3	5.3/19.8	6.7/21.9
	T(s)	0.02	131	1992	2.3
6×6	B/U	3.2/12.5	4.8/–	6.4/–	8.0/30.0
	T(s)	0.04	> 2500	> 2500	36.0

V. APPLICATIONS TOWARD PERSISTENT MONITORING

In this section, we demonstrate how QCOP and our algorithm may be applied to realistic persistent monitoring problems. Because our focus here is on estimation quality, we apply the exact algorithm on a single mobile robot. To be extensive, we include one simulation experiment performed over a synthetic spatiotemporal field and one simulation experiment using real temperature data from 14 weather stations in the state of Massachusetts. When it comes to applying QCOP to persistent monitoring tasks, we must first obtain $\{w_{ij}\}$ from historical data collected over all nodes, which is a *learning* problem. Recall that $N_i := \{v_j \mid (v_i, v_j) \in E\}$ is the neighbor set of $v_i \in V$. We apply a linear regression model over ψ, i.e.,

$$\psi(v_i,t) = \alpha_{0i} + \sum_{v_j \in N_i} \alpha_{ji}\psi(v_j,t), \tag{14}$$

in which α_{0i} and α_{ji}'s are coefficients to be computed from T sets of data. (14) corresponds to models such as the Gaussian Process (GP). To map these coefficients to QCOP, for each w_{ji}, we compute two sets of such coefficients. The first set of coefficients α'_{ji}'s are computed assuming all of N_i are visited; the second set, α''_{ji}'s, are computed assuming v_j is v_i's only visited neighbor. We then compute the weight w_{ji} via

$$w_{ji} = \frac{\alpha'_{ji} + \alpha''_{ji}}{\sum_{k\in N_i}(\alpha'_{ki} + \alpha''_{ki})}. \tag{15}$$

(15) was chosen to balance between the impact of single neighbors and the impact of the entire neighborhood. With $\{w_{ij}\}$, for a given travel budget, a utility maximizing tour can be computed. Assuming the robot collects exact values at time t_s from nodes it visits, the values on nodes that are not visited by the robot are estimated as follows. Let v_i be such a node and let N'_i be its neighbor set such that a node $v_j \in N'_i$ has either measured or estimated value (at time t_s). The historical data for v_i and nodes in N'_i from $1 \le t \le T$ are then used to perform multiple linear regression according to

$$\psi(v_i,t) = \beta_{0i} + \sum_{v_j \in N'_i} \beta_{ji}\psi(v_j,t). \tag{16}$$

The obtained β_{0i} and β_{ji}'s can then be used to compute the estimated $\psi'(v_i,t_s)$ using (16). The process is repeated until all nodes are covered. This iterative process defines the function UpdateNodeEstimate$(\cdot)$ in Algorithm 1.

A. Measuring a Time-Varying Scalar Field

The first application-driven simulation verifies the effectiveness of (15) in connecting scalar fields to QCOP. Our experiments are performed over a synthetic scalar field generated by three two-dimensional Gaussian distributions. These distributions have fixed centers but varying intensities and covariance matrices over time; we fix the centers to ensure that the spatial correlations are relatively time-invariant. The field is simulated for 201 time steps; the snapshots of the field at time steps $0, 50, 100, 150$, and 200 are provided in Fig. 11. The node network used here is a 5×5 randomized grid (similar to Fig. 10 but without edge randomization) scaled to the dimensions of the support of the scalar field. For each fixed travel budget, 100 random 5×5 node networks are generated. In each randomly generated network, the nodes of the network are given equal importance (i.e., unit utility). To estimate α'_{ij}'s

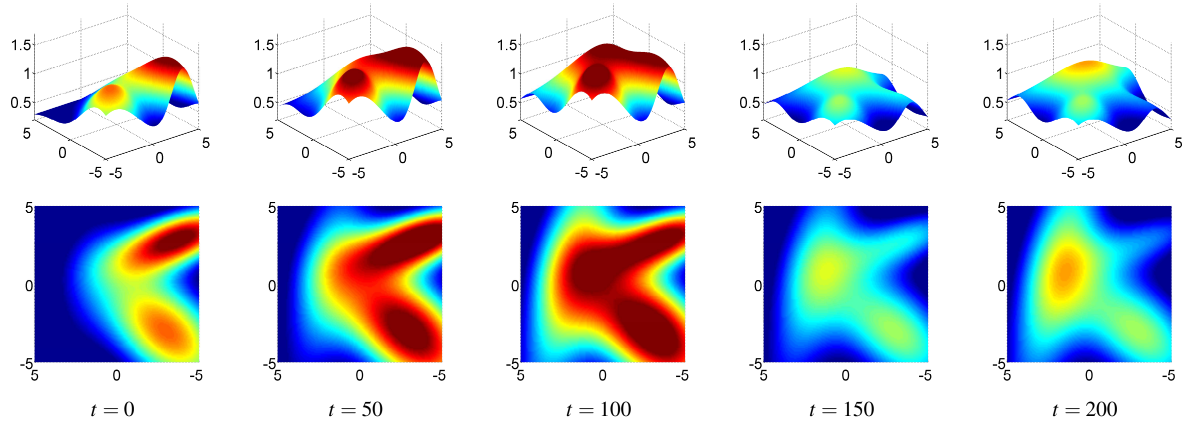


Fig. 11. Snapshots of a synthetic scalar field at time steps $0, 50, 100, 150$, and 200, from left to right, respectively. [top row] Three-dimensional views. [bottom row] Two-dimensional heat-map views.

and α''_{ij}'s for computing the weights, data from the first fifty time steps were used ($T = 50$). For running the model to obtain a utility maximizing tour, the second diagonal node from the top-left corner was used as the base node. The resulting tour is then used to obtain $\psi'(v_i, t_s)$ for $t_s = 100, 150$, and 200, according to (16). We define the *quality* of $\psi'(v_i, t_s)$ as

$$\frac{\sum_{t_s \in \{100,150,200\}} (\psi(v_i, t_s) - |\psi'(v_i, t_s) - \psi(v_i, t_s)|)}{\sum_{t_s \in \{100,150,200\}} \psi(v_i, t_s)}. \tag{17}$$

To compare to our results, we also exhaustively search through the network for a tour starting and ending at the same base node that minimizes the same quality defined by (17) under the same travel budget. This experiment was limited to travel budgets 6 and 8, corresponding to tours containing up to five nodes. While our model can produce tours with many more nodes, for comparing the result, we have to exhaustively search through all tours starting from the base node to find the best one, which becomes very costly as the number of nodes is over 5. The quality score obtained this way is denoted as the "actual quality". The result comparing the approaches is given in Table IX. Using the given metric, the average quality error was less than one, meaning that it was not more than the error incurred by omitting a single node. In roughly 30% of the cases, the tour found using our method was identical to the one found using exhaustive tour search.

TABLE IX
MODEL FIDELITY OVER A SYNTHETIC SCALAR FIELD

Travel Budget	Model Quality	Actual quality	Relative error
6.0	7.16	7.64	0.48
8.0	8.46	9.38	0.92

As a secondary and more intuitive measure of the quality of our method, we put a regular 6×6 node network fitted over the same field (Fig. 11) and run the MIQP model such that we just have enough budget to obtain a full utility of 36. We let the start node be the second node from the left on the first row. From the output we can then estimate values for all nodes that are not visited on the tour. We plot the much sparser survey data over the same space for time steps $100, 150$, and 200 as shown in Fig. 12. Comparing these figures with the corresponding ones from Fig. 11, we observe that our models provide reasonable estimates of the entire synthetic scalar field without the need to visit all the nodes.

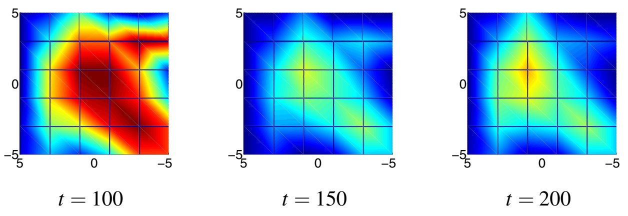


Fig. 12. Estimated scalar field using Algorithm 1.

B. Temperature Scalar Field

Our second simulation works with real temperature data retrieved from National Oceanographic Data Center[3]. The data consists of monthly average temperatures taken at 14 locations in the state of Massachusetts (see Fig. 13) over a 24 month period. Using methodology similar to the synthetic scalar field example, we use the first year's data as training data (i.e., $T = 12$) and then run our algorithm to sample and estimate temperature for the next year for every three months (a total of $12/3 = 4$ sets of temperatures). Node 10 (Boston) is selected as the base. The ground truth for these four sets is plotted in the top row of Fig. 14.

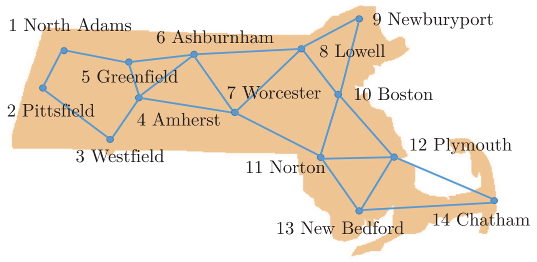


Fig. 13. Node network of 14 weather stations in Massachusetts.

[3]`http://www.nodc.noaa.gov/General/temperature.html`

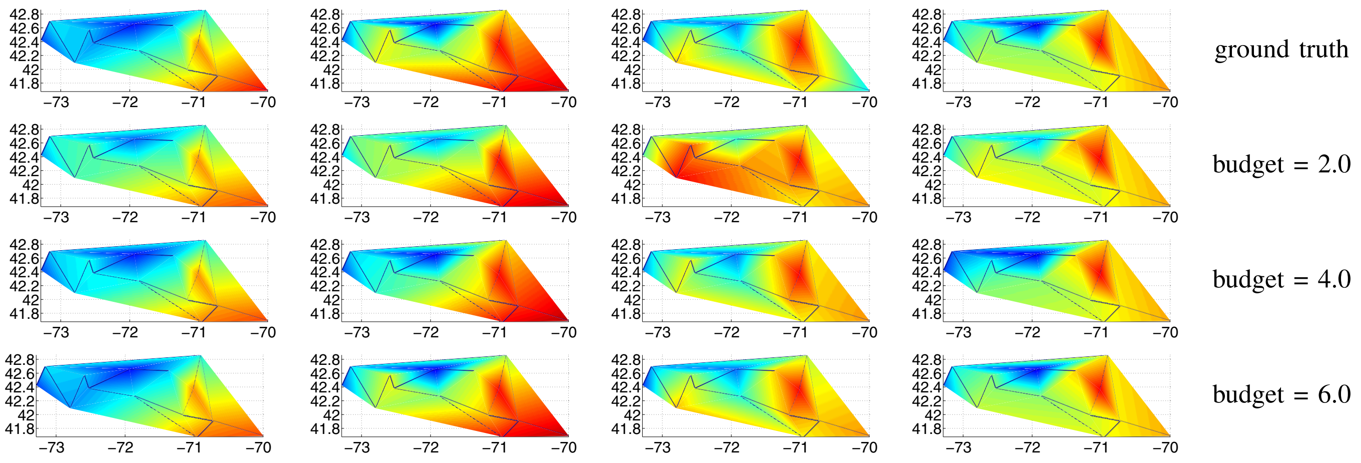


Fig. 14. Heat maps of Massachusetts in four different seasons of a year. Note that the color-scaling for each figure is different horizontally but the color-scaling used for each column of figures is the same. [top row] Interpolations using temperature data at 14 weather stations. [rows 2-4] Interpolations using estimated data with various budgets.

For constructing the budget, since the area of Massachusetts is relatively small, we treat it as flat and use longitudes and latitudes to measure the distance between the nodes. We run our algorithm using varying budgets from 2.0 to 6.0. The results are plotted using heat maps in Fig. 14. Although the figures are visualizations of discrete data through interpolation, via similarity, we observe that additional budget allows better sampling and estimation quality. Quantitatively, since temperature itself is a good metric, we measure the quality of the estimation by summing up the absolute difference between actual and estimated values at each node. Then, the total error is averaged over the number of nodes. The outcome is listed in Table X. At a budget of 3.0, which is enough to visit half of the nodes, the estimation quality is already fairly good with an average error of 0.27°C. The error goes to less than 0.1°C with a budget of 6.0. On the other hand, visiting all nodes requires a budget of roughly 8.5.

TABLE X
TEMPERATURE ESTIMATION ERROR WITH RESPECT TO BUDGET

Travel Budget	2.0	3.0	4.0	5.0	6.0
Average Error (°C)	0.46	0.27	0.22	0.15	0.08

VI. CONCLUSION AND FUTURE WORK

We introduced COP (and QCOP) as a generalization to OP to address the problem of planning tours for surveying a spatially correlated field that also changes over time. Our computational experiments show that the MIQP-based anytime algorithms for QCOP are effective in capturing correlations among nearby nodes, indicating that QCOP and the associated MIQP models are applicable to persistent monitoring tasks in which the mobile robots have limited range.

There are many promising directions along which the current work may be advanced; we mention two here. First, instead of looking at only immediate correlations as we did with QCOP, it may be beneficial look at correlations of nodes that are further apart in the node network, i.e., nodes that are in the 2- or 3-neighborhood (a k-neighborhood of a node v contains all nodes on paths from v with at most k edges; e.g., a 2-neighborhood contains v's neighbors and v's neighbors' neighbors). This extension is non-trivial because the inclusion of additional nodes poses new computational challenges. We expect, however, the gain in estimation accuracy will degrade quickly as the neighborhood expands. Therefore, a 2-neighborhood is perhaps all that is needed. Second, for QCOP, the current weight selection criterion for applying the model in practice is somewhat ad-hoc and has room for improvement. Through a more systematic approach, perhaps via statistical methods, we hope to derive more principled procedures for selecting the weights for the MIQP models to further improve the modeling power and applicability.

Furthermore, this paper only begins to address the problem of using correlations in informative path and policy planning in a discrete fashion. The dual problem to this estimation problem is a learning one: how can we learn the correlations among the nodes for applying the methods from this paper? How can we learn with limited number of mobile robots? What about concurrent learning and estimation?

REFERENCES

[1] P. Vansteenwegen, W. Souffriau, and D. V. Oudheusden, "The orienteering problem: A survey," *European Journal of Operational Research*, vol. 209, pp. 1–10, 2011.
[2] I. Chao, B. Golden, and E. Wasil, "Theory and methodology—the team orienteering problem," *European Journal of Operational Research*, vol. 88, pp. 464–474, 1996.
[3] ——, "Theory and methodology - a fast and effective heuristic for the orienteering problem," *European Journal of Operational Research*, vol. 88, pp. 475–489, 1996.
[4] R. M. Karp, "Reducibility among combinatorial problems," in *Complexity of Computer Computations*, R. E. Miller and J. W. Thatcher, Eds. New York: Plenum, 1972, pp. 85–103.
[5] G. Laporte, "The traveling salesman problem: An overview of exact and approximate algorithms," *European Journal of Operational Research*, vol. 59, pp. 231–247, 1992.
[6] S. Alamdari, E. Fata, and S. L. Smith, "Persistent monitoring in discrete environments: Minimizing the maximum weighted latency between observations," *International Journal of Robotics Research*, 2012, to appear.

[7] E. Arvelo, E. Kim, and N. C. Martins, “Memoryless control design for persistent surveillance under safety constraints,” September 2012, available at http://arxiv.org/abs/1209.5805.

[8] X. Lan and M. Schwager, “Planning periodic persistent monitoring trajectories for sensing robots in Gaussian random fields,” in *Proc. of the IEEE International Conference on Robotics and Automation (ICRA 13)*, May 2013, pp. 2407–2412.

[9] S. L. Smith, M. Schwager, and D. Rus, “Persistent robotic tasks: Monitoring and sweeping in changing environments,” *IEEE Transactions on Robotics*, vol. 28, no. 2, pp. 410–426, April 2012.

[10] J. L. Ny, M. A. Dahleh, E. Feron, and E. Frazzoli, “Continuous path planning for a data harvesting mobile server,” in *47th IEEE Conference on Decision and Control*, 2008, pp. 1489–1494.

[11] C. G. Cassandras, X. Lin, and X. Ding, “An optimal control approach to the multi-agent persistent monitoring problem,” *IEEE Transactions on Automatic Control*, vol. 58, no. 4, pp. 947–961, April 2013.

[12] J. Yu, S. Karaman, and D. Rus, “Persistent monitoring of events with stochastic arrivals at multiple stations,” in *Proceedings IEEE International Conference on Robotics & Automation*, 2014, pp. 5758–5765, preliminary extended version available at http://arxiv.org/abs/1309.6041.

[13] A. Girard, A. Howell, and J. Hedrick, “Border patrol and surveillance missions using multiple unmanned air vehicles,” in *Proc. 43rd IEEE Conference on Decision and Control*, 2004, pp. 620–625.

[14] N. Nigam and I. Kroo, “Persistent surveillance using multiple unmanned air vehicles,” in *Proc. IEEE Aerospace Conference*, 2008, pp. 1–14.

[15] N. Michael, E. Stump, and K. Mohta, “Persistent surveillance with a team of MAVs,” in *Proceedings IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2011, pp. 2708–2714.

[16] R. N. Smith, M. Schwager, S. L. Smith, B. H. Jones, D. Rus, and G. S. Sukhatme, “Persistent ocean monitoring with underwater gliders: Adapting sampling resolution,” *Journal of Field Robotics*, vol. 28, no. 5, pp. 714–741, Sep-Oct 2011.

[17] B. Grocholsky, J. Keller, V. Kumar, and G. Pappas, “Cooperative air and ground surveillance,” *IEEE Robotics and Automation Magazine*, vol. 13, no. 3, pp. 16–25, Sep 2006.

[18] D. E. Soltero, M. Schwager, and D. Rus, “Generating informative paths for persistent sensing in unknown environments,” in *Proc. of the International Conference on Intelligent Robots and Systems (IROS 12)*, October 2012, pp. 2172–2179.

[19] L. E. Kavraki, P. Svestka, J.-C. Latombe, and M. H. Overmars, “Probabilistic roadmaps for path planning in high-dimensional configuration spaces,” *IEEE Transactions on Robotics & Automation*, vol. 12, no. 4, pp. 566–580, Jun. 1996.

[20] S. M. LaValle, “Rapidly-exploring random trees: A new tool for path planning,” Iowa State University, Tech. Rep., Oct 1998, computer Science Department TR 98-11.

[21] S. Karaman and E. Frazzoli, “Sampling-based algorithms for optimal motion planning,” *International Journal of Robotics Research*, vol. 30, no. 7, pp. 846–894, June 2011. [Online]. Available: http://ares.lids.mit.edu/papers/Karaman.Frazzoli.IJRR11.pdf

[22] G. A. Hollinger and G. S. Sukhatme, “Sampling-based robotic information gathering algorithms,” *International Journal of Robotics Research*, vol. 33, no. 9, pp. 1271–1287, 2014.

[23] H. Choset, “Coverage of known spaces: The boustrophedon cellular decomposition,” *Autonomous Robots*, vol. 9, pp. 247–253, 2000.

[24] ——, “Coverage for robotics—a survey of recent results,” *Annals of Mathematics and Artificial Intelligence*, vol. 31, pp. 113 – 126, 2001.

[25] Y. Gabriely and E. Rimon, “Competitive on-line coverage of grid environments by a mobile robot,” *Computational Geometry*, vol. 24, no. 3, pp. 197–224, 2003.

[26] P. Fazli, A. Davoodi, and A. K. Mackworth, “Multi-robot repeated area coverage,” *Autonomous robots*, vol. 34, no. 4, pp. 251–276, 2013.

[27] C. Franco, D. M. Stipanović, G. López-Nicolás, C. Sagüés, and S. Llorente, “Persistent coverage control for a team of agents with collision avoidance,” *European Journal of Control*, vol. 22, pp. 30–45, 2015.

[28] W.-P. Chin and S. Ntafos, “Optimum watchman routes,” *Information Processing Letters*, vol. 28, pp. 39–44, 1988.

[29] P. Hokayem, D. Stipanovic, and M. Spong, “On persistent coverage control,” in *Proc. 46th IEEE Conference on Decision and Control*, 2008, pp. 6130–6135.

[30] S. Ntafos, “Watchman routes under limited visibility,” *Computational Geometry.*, vol. 1, pp. 149–170, 1991.

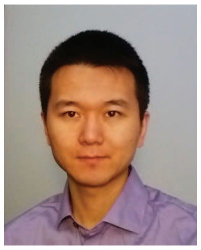
[31] J. Faigl, “Approximate solution of the multiple watchman routes problem with restricted visibility range,” *Neural Networks, IEEE Transactions on*, vol. 21, no. 10, pp. 1668–1679, 2010.

[32] J. Yu, M. Schwager, and D. Rus, “Correlated orienteering problem and its application to informative path planning for persistent monitoring tasks,” in *Proceedings IEEE/RSJ International Conference on Intelligent Robots & Systems*, 2014.

[33] J. Yu, J. Aslam, S. Karaman, and D. Rus, “Anytime planning of optimal schedules for a mobile sensing robot,” in *Proceedings IEEE/RSJ International Conference on Intelligent Robots & Systems*, 2015.

[34] C. H. Papadimitriou, “The euclidean travelling salesman problem is np-complete,” *Theoretical Computer Science*, vol. 4, no. 3, pp. 237–244, 1977.

[35] Gurobi Optimization Inc., “Gurobi optimizer reference manual,” 2014. [Online]. Available: http://www.gurobi.com



Jingjin Yu is an assistant professor with the Department of Computer Science at Rutgers University. He received his B.S. degree from USTC in China in 1998. He holds M.S. degrees in Chemistry (Univ. Chicago, 2000), Mathematics (Univ. Illinois at Chicago, 2001), and Computer Science (Univ. Illinois at Urbana-Champaign, 2010). He obtained his Ph.D. degree in Electrical and Computer Engineering from the University of Illinois at Urbana-Champaign in 2013, where he briefly stayed as a postdoctoral researcher. He was a postdoctoral researcher at the Massachusetts Institute of Technology from 2013 to 2015 with a joint appointment at Boston University from 2013 to 2014. He is broadly interested in the areas of robotics and control, focusing on computational complexity issues and designing efficient algorithms with provable guarantees.

Mac Schwager is an assistant professor with the Aeronautics and Astronautics Department at Stanford University. He obtained his BS degree in 2000 from Stanford University, his MS degree from MIT in 2005, and his PhD degree from MIT in 2009. He was a postdoctoral researcher working jointly in the GRASP lab at the University of Pennsylvania and CSAIL at MIT from 2010 to 2012, and was an assistant professor at Boston University from 2012 to 2015. His research interests are in distributed algorithms for control, perception, and learning in groups of robots and animals. He received the NSF CAREER award in 2014.

Daniela Rus is the Andrew (1956) and Erna Viterbi Professor of Electrical Engineering and Computer Science and Director of the Computer Science and Artificial Intelligence Laboratory (CSAIL) at MIT. Rus research interests are in robotics, mobile computing, and data science. Rus is a Class of 2002 MacArthur Fellow, a fellow of ACM, AAAI and IEEE, and a member of the National Academy of Engineering. She earned her PhD in Computer Science from Cornell University. Prior to joining MIT, Rus was a professor in the Computer Science Department at Dartmouth College.