2010 IEEE International Conference on Robotics and Automation
Anchorage Convention District
May 3-8, 2010, Anchorage, Alaska, USA

Coordinated Multi-Robot Real-Time Exploration With Connectivity and Bandwidth Awareness

Yuanteng Pei, Matt W. Mutka, and Ning Xi

Abstract— **While there has been substantial progress for multi-robot exploration of an unknown area, little attention has been given to communication, especially bandwidth constraints in *time-sensitive* and bandwidth-consuming tasks such as search and surveillance. In such tasks, video/audio streams of a newly explored area should be sent back to the base station in a timely manner. To address this issue, we propose *Connectivity and Bandwidth Aware eXploration (CBAX)*, which is an efficient iteration based *real-time* exploration. CBAX divides the problem into frontier node placement, relay node placement with routing path selection, and the match of each robot with its target position. Moreover, we model bandwidth-constrained relay node placement into a new variant of the Steiner Minimum Tree problem and present our solution. While reducing the exploration time, CBAX maintains the network's connectivity and ensures the aggregated data flows are under the link capacity in transmission. Simulation shows that CBAX outperforms two recent exploration schemes *qualitatively* by demonstrating major improvement in terms of non-overflow transmission time and fully-connected transmission time. With enhanced communication quality, CBAX still reduces the exploration time, on average, by 40% and 15% respectively. In moderately dense scenarios, CBAX even decreases time by 50% and 25%.**

I. Introduction

Exploring an environment is one of the fundamental problems in mobile robotics. Recently, multi-robot exploration has received increased attention for its notable benefit of enhanced efficiency and robustness [1]. Many application domains of multi-robot exploration, such as surveillance, reconnaissance, search and rescue missions in dangerous areas require robot's bandwidth-consuming video/audio information from newly-explored area to be reliably and quickly sent back to an operator at the base station [2]. The reasons for this requirement are threefold. First, human operators often need to monitor the robot team's action and obtain the sensed information immediately. Second, as the current robotic sensing ability is not sufficient to accurately detect complex targets, such a process should be simultaneously augmented with human recognition [3]. Third, operators may even need to teleoperate robots promptly after target discovery or under exceptional circumstances. Thus, to augment limited robot intelligence by an operator can often be a necessary and effective way to monitor and control a robot team. Reliable and smooth communication is essential in such a process. Hence, such Multi-Robot *Real-Time* eXploration (MRRTX) is critical.

MRRTX can be described as follows: A team of n homogenous robots are sent from the *base station* (**BS**) to explore an unknown area. Each robot communicates with limited range by forming a multi-hop network with all robots and the BS. The question is: How to design a coordinated exploration strategy for the robot team to explore the area in a minimum amount of time under the following *two constraints*? i) The network is always connected when data is transmitted. ii) The aggregated consumed bandwidth of data flows from the frontier nodes cannot exceed the link bandwidth (capacity) when transmitting back to the BS. In fact, many types of real-time streaming data such as video/audio have a stream floor rate [4], or a minimum rate to make the stream workable. A link should always provide adequate bandwidth for each flow to meet the floor rate requirement. Without these two constraints, disconnected nodes can evolve and the aggregated data flows may exceed the link bandwidth, leading to considerable data loss and control disturbance.

To solve MRRTX, we present *Connectivity and Bandwidth Aware eXploration (CBAX)*. Differing from existing approaches, CBAX is an iterative exploration that divides the problem of the next round movement path generation and message routing into three subproblems, which are modeled and solved accordingly by variations of *algorithmic and graph-theoretic* problems, such as the set cover problem, Steiner tree problem, and the linear bottleneck assignment problem (LBAP). When we solve the bandwidth-constrained relay node placement, we model it as a new variation, or the one with the γ-*inflow* constraint, of the *Steiner Minimum Tree Problem with Minimum Number of Steiner Points and bounded edge length (SMT-MSP)* presented in [5,6]. CBAX not only reduces the exploration time compared with recent work [1,2], but also guarantees the adequacy of link bandwidth and enhances the communication *quality*. In Fig. 1, CBAX is illustrated by an exploration snapshot obtained from our simulation program with the *Garden* environment given in Section V.

II. Related Work

Communication in multi-robot systems has drawn increased attention. Novel research for maintaining connectivity of mobile networks from distributed control perspective is presented in [7]. However, it does not have bandwidth awareness or the specific goal of area exploration. While multi-robot exploration has been widely studied in [1,2,8],

This work is supported in part by NSF grants no. OCI-0753362 and CNS-0721441.

Yuanteng Pei and Matt W. Mutka are with the Department of Computer Science and Engineering, Michigan State University, East Lansing, MI 48824, USA {peiyuant,mutka}@cse.msu.edu

Ning Xi is with the Department of Electrical and Computer Engineering, Michigan State University, East Lansing, MI 48824, USA xin@egr.msu.edu

little prior work considers the bandwidth constraint and only some schemes assume limited communication range and attempt to maintain connectivity. Sensor-based random graph and segmentation based methods in [1,8] explore an area efficiently but provide only intermittent connectivity or assume robots are always in range.

III. System Model and Problem Formulation

Before presenting our solution, the system model and problem formulation are introduced as follows.

A. System Model

Robot Model: Each robot has traveling, communication, and sensing capability. It scans the environment using cameras and acoustic sensors with sensing range $\boldsymbol{r_s}$ and communicates with other nodes with communication range $\boldsymbol{r_c}$ by a 802.11 radio, where $r_c > r_s$.

Environment and Exploration Model With a Base Station: The exploration begins with the operator selecting a targeting area, all of whose borders are reachable when robots form a straight line from BS. The area is modeled by the 2D occupancy map composed of grid cells. The cell size equals that of a robot. The states of a cell are *unvisited*, *exploring*, and *explored*. An explored cell can be either *free* or *obstructed*. An obstacle does not affect communication but blocks the sensing range. An unvisited cell denotes the area that has not been reached by the sensing range of robots. Arriving at their target positions, all nodes are synchronized and start simultaneously to sense the area and transmit streams back to BS for a fixed *sensing and transmitting interval* (**STI**). In STI, the unvisited grid cells within the range of r_s and not blocked by obstacles of a node are changed to *exploring* status. After STI, these cells change to *explored* status since the operator has observed and detected the targets in STI. The cells that are explored but have at least one unvisited neighbor cell are called *frontier cells* and denoted as $\boldsymbol{C_{frn}}$.

Coordinated exploration is executed in an iterative and synchronized way controlled by the BS. At iteration t, $\boldsymbol{Map^t}$ denotes the explored map and $\boldsymbol{T^t}$ gives the *total time* elapsed. Each robot i has a position $\boldsymbol{P_i^t}$ and is dynamically classified as a *relay node* (**RN**) or a *frontier node* (**FN**). A FN explores new area while a RN maintains connectivity for

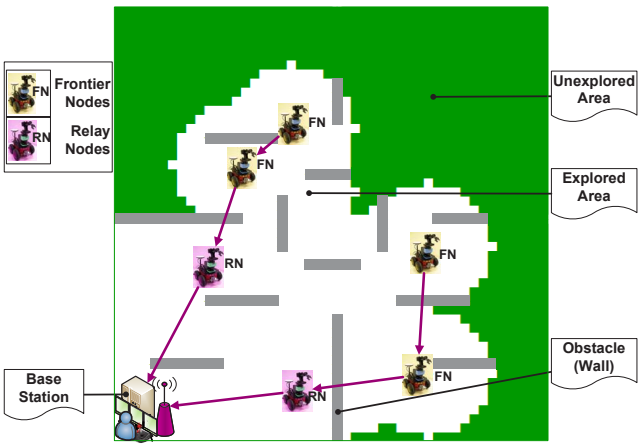


Fig. 1. Exploration snapshot of the *Garden* environment given in Section V. (Nodes are scaled larger for illustration purpose.)

FNs with BS. The *robot team size* is $\boldsymbol{n}$ with $\boldsymbol{n_{fn}}$ number of FNs and $\boldsymbol{n_{rn}}$ number of RNs. We define the *information gain* of node i in iteration t, or $\boldsymbol{IG_i^t}$, as the number of unvisited cells that can be explored with a new position P_i^t. A *position configuration* is the array of positions of the n robots. Specifically, $\boldsymbol{pCG^t}$, the position configuration at iteration t is defined as: $pCG^t = \{P_1^t, P_2^t, ...P_n^t\}$. The set of all possible pCG^t is denoted as $\boldsymbol{_P\mathcal{CG}^t}$. Each pCG^t is associated with a *total information gain*, or $\boldsymbol{sIG^t}$, which is the total number of non-overlapped unvisited cells that can be explored with a new position P_i^t for each node i. A *valid* position configuration is the one that satisfies the *Constraint I* defined below. Besides, $\boldsymbol{Path_i^t}$ is the migrating path of node i from iteration t-1 to t and $\boldsymbol{Path^t}$ is the set of $Path_i^t$ for each node i. Similar to [7], we define a *dynamic graph* $\boldsymbol{G(t)}$=$\{V, E(t)\}$ where V denotes the sets of vertices indexed by the set of robots with BS and $E(t)$ is the edge set representing the communication links of vertices and $(i,j) \in E(t)$ *iff* $dist(P_i^t, P_j^t) \leq r_c$. ($\boldsymbol{dist(i,j)}$ denotes the straight-line distance between i and j)

In addition, we define a *routing configuration* by the path array from all FNs back to BS where the streams traverse. Specifically, $\boldsymbol{rCG^t}$, the routing configuration at iteration t is defined as $rCG^t = \{\mathcal{RL}_1^t, \mathcal{RL}_2^t, ...\mathcal{RL}_n^t\}$, where $\boldsymbol{\mathcal{RL}_i^t}$ is the *route list*, or the sequence of nodes on the routing path from FN i to the BS in iteration t. A *valid* routing configuration is the one satisfying the *Constraint II*, or Equation 2 defined below. Combining both position and routing configurations, we define the *configuration* at iteration t as $\boldsymbol{CG^t}$= $\{pCG^t, rCG^t\}$. A *valid* configuration is the one that satisfies *Constraints I & II* and $\boldsymbol{\mathcal{CG}_V^t}$ denotes the set of all valid configurations in iteration t.

Wireless Communication Model: We analyze the wireless communication in area exploration with the following properties: i) The unit-disk communication model is adopted and two nodes i, j can communicate as long as $dist(i,j) \leq r_c$. ii) All links have a uniform *link capacity* $\boldsymbol{R_{capacity}}$ and a FN has a uniform *flow sending rate* $\boldsymbol{R_s}$ for its video/audio streams not less than their stream floor rate in STI, where a *flow* is the combined video and audio streams from a FN to BS. iii) Streams are sent in a single selected path rather than multiple paths. iv) Interference between nodes will be analyzed in future work.

Constraints I & II: In the STI of each iteration t, *I.* the robot team with BS's network $G(t)$ is connected. *II.* The aggregated consumed bandwidth cannot exceed the link capacity on each link of the path from FNs to BS, which is also expressed as:

$$R^e_{capaclity} \geq \sum_{f \in F_e} R^e_{consumed(f)} \tag{1}$$

Where $\boldsymbol{f}$ represents a flow and $\boldsymbol{F_e}$ denotes all the flows passing through link e. With the properties in our wireless communication model, the above equation is rewritten as:

$$R^e_{capaclity} \geq \gamma \cdot R_s \tag{2}$$

An integer $\boldsymbol{\gamma}$, or the *bandwidth ratio*, denotes the maximum

number of flows that can pass through link e without overflowing the link, e.g., $\gamma = 3$ when R_s=450KB/s with 11Mbps link; $\gamma = 5$ when R_s=128KB/s with 5Mbps link.

B. Problem Formulation

Based on the system model, the major problem is how to explore the whole area for a robot team with minimum time *under the communication constraints I & II*. Although the problem can be difficult to solve as a whole, our solution leverages a heuristic answer to a subproblem. The subproblem becomes more tractable but remains non-trivial, since it is further divided into variants of two NP-hard problems and a polynomially-solvable problem. This subproblem is to find a *local optimal configuration* in iteration t, or $\boldsymbol{CG^t_{lopt}}$, which has maximal total number of cells explored in unit migration time. Formally, the subproblem is: given pCG^{t-1}, find CG^t_{lopt} such that:

$$CG^t_{lopt} = \underset{CG^t \in \mathcal{CG}^t_V}{\operatorname{argmax}} \left(\frac{sIG^t}{MT^t}\right) \tag{3}$$

IV. Connectivity and Bandwidth Aware eXploration (CBAX)

A. CBAX Overview

Algorithm 1: General Procedure of CBAX

Input: pCG^{t-1}, Map^{t-1}, T^{t-1}, Robot team size n
Output: CG^t, $Path^t$, MT^t, T^t, Map^t

1 $n_{fn} \leftarrow n$ // "***tmp***" means "temporary" throughout this paper
2 **while** $n_{fn} > 0$ **do**
3 $pCG^t_{FNtmp} \leftarrow$ *placeFN*(Map^{t-1}, n_{fn})
4 $CG^t_{tmp}, n_{rn} \leftarrow$ *placeRN_RoutingPathSelection*(pCG^t_{FNtmp})
5 **if** $n_{fn} + n_{rn} \leq n$ **then**
6 **if** $n_{fn} + n_{rn} < n$ **then**
7 $pCG^t_{tmp} \leftarrow$ *placeRemainNode*(pCG^{t-1}, pCG^t_{tmp})
8 **break**
9 **else**
10 *reset*$(pCG^t_{FNtmp}, CG^t_{tmp}, n_{rn})$
11 $n_{fn} \leftarrow n_{fn} - 1$
12 $MT^t, T^t, Path^t, pCG^t \leftarrow$ *genPath*$(pCG^{t-1}, pCG^t_{tmp}, T^{t-1})$
13 $CG^t \leftarrow pCG^t, rCG^t$
14 $Map^t \leftarrow$ *search*(Map^{t-1}, CG^t)

Generally, CBAX attempts to enhance efficiency by maximizing the explored area by placing more robots in frontier areas, while reserving less robots as relay nodes. Additionally we endeavor to reduce $\boldsymbol{MT^t}$, or the *migration time* in iteration t, which is the bottleneck, or the longest moving time among all nodes. Clearly MT^t also directly impacts the exploration time. The problem of finding CG^t_{lopt} in Section III-B can be divided into largely three subproblems:

- Frontier node placement: Where to place n_{fn} number of FNs in C_{frn} to cover maximum amount of unexplored area? The distances from current positions to new ones are also considered and modeled in the utility function.
- Relay node placement with routing path selection: How to place minimum number of RNs in explored area to satisfy the *Constraint I,II* and select which paths to route flows from FNs to BS?
- Position assignment and path generation: How to assign each robot with its target position and generate movement paths to minimize the bottleneck time MT^t?

Alg. 1 illustrates the general procedure of CBAX for each iteration. Initially, n_{fn} is set as the team size because more FNs normally can cover more area. Given FN positions computed in line 3, RNs are placed accordingly with routing to relay FNs with BS and n_{rn} is decided. Line 5 checks whether required number of robots $n_{fn} + n_{rn}$ exceeds the fixed team size. If yes, the algorithm reduces n_{fn} and places FNs and RNs again until it finds the required number of robots can be satisfied. If $n_{fn}+n_{rn} < n$, then the remaining nodes may be able to move to frontier areas to explore extra areas where RNs can support the additional flows. In line 12, function genPath is called to assure the bottleneck distance is minimized when generating movement paths for each node to reach its target. The map will be updated after nodes arrive at their new positions and enter STI.

B. Frontier Robot Placement

Algorithm 2: Frontier Node Placement *placeFN()*

1 **Input:** Map^{t-1}, n_{fn} **Output:** pCG^t_{FNtemp}
2 $i \leftarrow n_{fn}$
3 $Map_{tmp} \leftarrow Map^{t-1}$
4 **while** $i > 0$ **do**
5 $q^* \leftarrow \operatorname{argmax}_{q \in C_{frn}} U(q)$
6 $pCG^t_{FNtemp}.add(q^*)$
7 $Map_{tmp} \leftarrow$ *updateLocalMap*(q^*, Map_{tmp})
8 $C_{frn} \leftarrow$ *updateFrontierPosSet*(Map_{tmp}, C_{frn})
9 $i \leftarrow i - 1$

Our utility function is defined as follows:

$$U(q) = IG^t_q \cdot e^{-\frac{d(q)}{\theta}}, \tag{4}$$

$$where \quad d(q) = \min_{i \in pCG^{t-1}} dist(q, i),$$

$$\theta = \max\{\theta_0 \cdot \delta, \theta_{th}\} \qquad (\delta = \frac{size(Map^{t-1})}{size(Map_{total})})$$

Where $\boldsymbol{\delta}$ is the *exploration ratio*, or explored area over total area. Empirically, θ_0 is set as 20 and θ_{th} (the threshold) is set as 12. When $\theta \to \infty$, distance is not considered and FNs are greedily placed as the positions that cover most. On the other hand, when $\theta \to 0$ the distance is weighed heavily so the frontier cell closest to the current positions pCG^{t-1} is selected. Thus, θ defined in equation 4 enhances performance with initial emphasis on coverage while focusing on migrating time in the end to avoid unnecessary fluctuating moves. This approach resembles the candidate evaluation method in [9] but shows disparities in setting θ dynamically to adjust the distance's weight and defining distance as the minimum from one point to a set of positions. Moreover, the problem can also be viewed as a variant of the well-known NP-hard *Set Cover* problem when $\theta \to \infty$. Therefore, our solution inherits its greedy approximation, along with dynamic cost modeling of migrating distance.

C. Relay Robot Placement with Routing Path Selection

Now that FNs are placed, we need to find i) whether the rest (n-n_{fn}) nodes are adequate to relay BS and FNs and how to place them; ii) If the nodes are abundant, then how to route the streams from each FN to BS. The solution to the problem depends on the value of bandwidth ratio γ defined in equation 2. Here, we focus on the small γ situations.

1) Bandwidth Sufficient Cases: When γ is sufficiently large, the first problem becomes a decision version of *Steiner Minimum Tree Problem with Minimum Number of Steiner Points and bounded edge length (SMT-MSP)* where FNs and BS are the terminal points and RNs are the Steiner points. We build the solution on the well-known approximation algorithm called *Steinerized Minimum Spanning Tree* for SMT-MSP with an approximation ratio of 4 [6]. Then the routing problem is easily solved by the breath-first search.

2) Bandwidth Constrained Cases: When γ is small and the bandwidth of aggregated flows may exceed link capacity, we model the problem as a decision version of SMT-MSP with the γ-*inflow* constraint. Specifically, each node can admit no more than γ flows from FNs (FNs may also relay flows). Alternatively, if we define a node as *saturated* when it carries γ flows and as *over-saturated* when it carries more than γ ones, the problem becomes SMT-MSP without *over-saturated* nodes when flows transmit from $n-1$ terminal points to one terminal point (BS). The problem is *as hard as* SMT-MSP (thus is also NP-hard) since if we can find a polynomial algorithm for it and we set $\gamma \to \infty$, SMT-MSP would be polynomially solvable.

The solution's central theme is to reduce unnecessary RNs by *aggregating flows* and the following definitions relevant to flow aggregation are introduced. First, on top of the dynamic graph $G(t)_{tmp}$ of FNs and BS, a *hybrid flow graph* $\boldsymbol{G_f(t)}=\{V, E(t) \cup \vec{E}(t)\}$ is defined where an undirected edge e denotes nodes in communication range while a directed edge $\vec{e}$ represents not only connected vertices, but also directed flow(s) from FNs to BS. Second, the *Vector of carried Flow Source* for each $\vec{e}$, or $\boldsymbol{V_{fs}^{\vec{e}}}$, denotes the vector of sources of the flows passing through this edge. For each node i, *Vector of Unrelayed carried Flow Source* ($\boldsymbol{V_{ufs}^{i}}$) gives the source vector of the unrelayed flows it carries. Third, $\boldsymbol{L_{cur}}$ gives the *current layer* of nodes and $\boldsymbol{L_{new}}$ gives the *new layer*. Fourth, a *cluster head* $\boldsymbol{CH}$ is defined as the node that aggregates flows in each layer and $\boldsymbol{V_{cur}^{ch}}$ denotes the cluster head vector of current layer nodes. Last but not least, $\boldsymbol{path_m(i,j)}$ denotes the shortest obstacle-aware migrating path from i to j and $\boldsymbol{N(i)}$ is node i's neighbor node set excluding L_{new} (j is a neighbor of i when $dist(i,j) \le r_c$).

Additionally, we define $\boldsymbol{V_{cf}^{i}}$ as the *Vector of Covered Flow* of i, in which each element is a linked list that begins with the node ID $j \in N(i) \cup i$ and continues with sources from V_{ufs}^{j}. Note that V_{cf}^{i} depicts the potential flows that may be carried where V_{ufs}^{i} describes the flows that have already been carried. A node $j \in N(i) \cup i$ is *fully covered* by node i when *all* of j's flows can be aggregated by i. $\boldsymbol{V_{fcn}^{i}}$ denotes the vector of i's *Fully Covered Nodes* ($j \in V_{fcn}^{i}$ *iff* $j \in N(i) \cup i \wedge V_{ufs}^{j} \subseteq V_{cf}^{i}$). Also, $|V_{fs}^{\vec{e}}|$, $|V_{ufs}^{i}|$, $|V_{cf}^{i}|$, and $|V_{fcn}^{i}|$ give the cardinality. Note that $|V_{cf}^{i}| \le \gamma$ and $|V_{cf}^{i}|$ only counts the source nodes. Each node must first carry its own flow if it is in L_{cur}, or its parent's flow if it is a RN in L_{new}. Then $|V_{fcn}^{i}|-1$ denotes the number of extra RNs can be saved with such aggregation using the remaining capacity.

Flow aggregation: To minimize $|RN|$, flow aggregation is executed in an order with maximal $|V_{fcn}^{i}|$ first. The problem of finding maximal $|V_{fcn}^{i}|$ can be solved optimally by greedily covering minimal $|V_{ufs}^{i}|$ node first. For example, in Fig. 2, new layer RN node n can maximally cover two extra nodes, or save two RNs immediately otherwise needed by node 3 and 4. n can aggregate all flows from them: n's $V_{fcn}^{n} = \{3,4\}$ with the remaining capacity 9-4=5 after it obtains its parent node 1's four flows.

Algorithm 3: Bandwidth-Constrained Relay Node Placement With Routing Path Selection *ConstrainedSteinerTree_Routing()*

1 **Input:** pCG_{FNtmp}^{t} **Output:** CG_{tmp}^{t}, n_{rn}
2 $L_{cur} \leftarrow pCG_{FNtmp}^{t}$, $L_{new} \leftarrow \emptyset$ *// FNs as current layer*
3 **foreach** $i \in L_{cur}$ **do** *initialize*(V_{ufs}^{i}), *setCoveredFlow_Node*(i)
4 **while** $\exists$ *node* $i(i \in L_{cur} \wedge dist(i,BS) > r_c)$ **do**
5 $L_{cur}^{tmp} \leftarrow L_{cur}$
6 **while** $L_{cur}^{tmp} \neq \emptyset$ **do**
7 Select cluster head i_{ch} with max $|V_{fcn}^{i}|$
8 Update $V_{cur}^{ch}, L_{cur}^{tmp}$; *aggrF_UpdateRoute* (i_{ch})
9 *placeRN* (i_{ch}) *// Generate* L_{new}
10 **foreach** $i \in L_{cur}^{tmp}$ **do** *setCoveredFlow_Node*(i)
11 $L_{new}^{tmp} \leftarrow L_{new}$
12 **foreach** $i \in L_{new}^{tmp}$ **do** *setCoveredFlow_Node*
13 **while** $L_{new}^{tmp} \neq \emptyset$ **do**
14 Select first node to aggr. flow node $i^{\dagger}$ with max $|V_{fcn}^{i}|$
15 **if** $|V_{ufs}^{i^{\dagger}.parent}| = 0$ **then** L_{new}.*remove*$(i^{\dagger})$ *// Filtered*
16 L_{new}^{tmp}.*remove*$(i^{\dagger})$; *aggrF_UpdateRoute* $(i^{\dagger})$
17 **foreach** $i \in L_{new}^{tmp}$ **do** *setCoveredFlow_Node*(i)
18 CG_{RNtmp}^{t}.*add*(L_{new}); $L_{cur} \leftarrow L_{new}$, $L_{new} \leftarrow \emptyset$
19 $n_{rn} \leftarrow |CG_{RNtmp}^{t}|$, $pCG_{tmp}^{t} \leftarrow pCG_{FNtmp} + pCG_{RNtmp}$
20 **foreach** i **do** rCG_{tmp}^{t}.*add*$(\mathcal{RL}_{i}^{t})$
21 *placeRN* (i): *// Subfunction begins*
22 **if** *dist(i,BS)*$> r_c$ **then** *// Place RN if needed*
23 $RN_{tmp} \leftarrow$ at p $(p|dist(p,i) = r_c)$ on $line(i,BS)$
24 **if** $P_{RN_{tmp}}$=*obstructed* **then**
25 $path_m(i,BS) \leftarrow$ A* search(*i*,BS)
26 $RN_{tmp} \leftarrow$ at $(p|dist(p,i) = r_c)$ on $path_m(i,BS)$
27 RN_{tmp}.parent$\leftarrow i$, L_{new}.add(RN_{tmp}) *// Set parent*

Major strategy with illustration: RN placement and routing path selection are solved jointly in a layer-based approach, which is also described in Alg. 3 and illustrated in an example in Fig. 3. First, FNs pCG_{FNtmp}^{t} are set as L_{cur} and covered nodes/flows are computed for each node $i \in L_{cur}$ based on V_{ufs}^{i}, which is initialized as itself being the only element since a node carries its own flow (line 3). With L_{cur}, the algorithm is to compute L_{new} as new RN until all current layer nodes are in range of BS, which is the terminating condition gives in line 4.

Computing L_{new} is executed in two steps: i) Clustering nodes and aggregating flows in L_{cur} with CH selection and

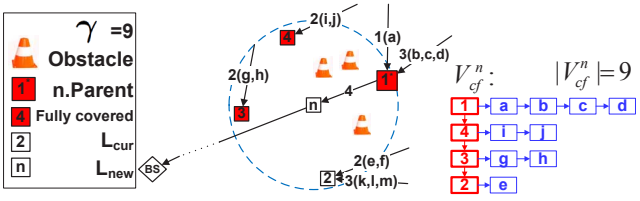
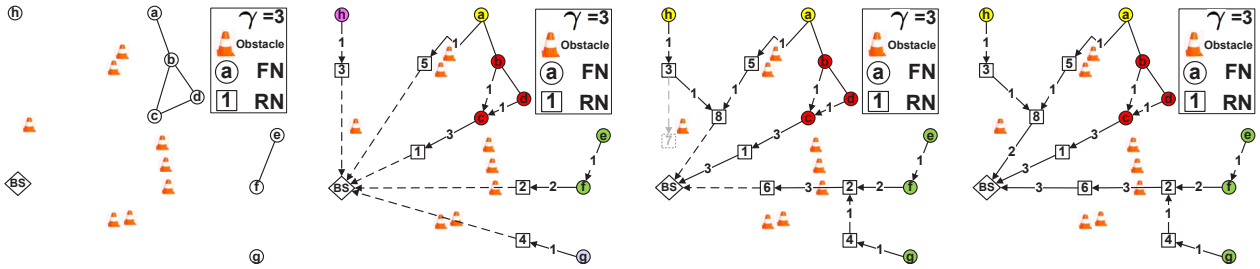


Fig. 2. Aggregate flows to nodes with maximal number of fully covered neighbors first to conserve RNs.



(a) With FNs and BS only. (b) Placing 1st layer RNs (c) Placing 2nd layer RNs (d) Final placement

Fig. 3. Example of bandwidth-constrained relay node placement and routing. (γ=3) Edge weight denotes $|V_{fs}^{\vec{e}}|$

RN placement. ii) Aggregating flows into L_{new} with L_{new} RN filtering, which are given in line 4-10, 11-18 respectively.

In the first step, we repeatedly select cluster heads CH in L_{cur} and inserts them into V_{cur}^{ch} to aggregate flow in the cluster in the order of "maximal $|V_{fcn}^{i}|$ first" (Choose minimum distance to BS if $|V_{fcn}^{i}|$ are equal since the shorter distance indicates fewer needed RNs). Moreover, we place RNs when needed to form L_{new} in the paths from V_{cur}^{ch} to the BS. When the RN is placed at an obstructed cell, *A* search* [10] is adopted to compute the obstacle-aware shortest path from one end (starting point) to the other end (goal) of the edge (line 25 in Alg. 3) and the RN is placed again on the new path. Note that i_{ch} is set as the new RN's *parent*. Function *aggrF_UpdateRoute* (i) is called when a node i aggregates flows by moving elements in neighbor node j's V_{ufs}^{j} to V_{ufs}^{i} following i's covered flow V_{cf}^{i}. Moreover, source s's $\mathcal{RL}_{s}^{t}$ is updated with the new RN appended. For instance, in Fig. 3(a), FN c is chosen as cluster head since it is the closest to BS among the ones with maximal $|V_{fcn}^{i}| = 3$. V_{ufs}^{c} is updated as {c,b,d} and c is added in $\mathcal{RL}_{b}^{t}$ and $\mathcal{RL}_{d}^{t}$ for sources b and d.

In the second step, CBAX executes flow aggregation with the same order of "maximal $|V_{fcn}^{i}|$ first" (Choose maximum size of covered flows if $|V_{fcn}^{i}|$ are equal) for L_{new} and attempts to remove unnecessary nodes by filtering the new layer nodes whose parent's flows have all been carried by others. To illustrate in Fig. 3(c), new layer RN_8 is first to aggregate flows with $|V_{fcn}^{RN\_8}| = 2$ and then RN_7 has $|V_{ufs}^{RN\_3}| = 0$ and is consequently filtered.

A layer placement ends when L_{new}^{tmp} is empty (line 13). Then L_{new} becomes L_{cur} and a new round starts. To conclude, the flows, denoted by V_{ufs}^{i} for node i , are aggregated in CH in L_{cur} and then in L_{new} with best endeavor to reduce the number of RNs.

D. Position Assignment And Path Generation

Algorithm 4: LBAP Based Position Assignment and Path Generation with Collision Avoidance *genPath()*

1 **Input:** $pCG^{t-1}, pCG^{t}, T^{t-1}$ **Output:** $MT^{t}, T^{t}, Path^{t}$
2 Construct complete bipartite graph G_{cb}= $(pCG^{t-1} + pCG^{t}, E)$
// update edge weight as obstacle-aware ones
3 **foreach** $e \in E_{G_{cb}}$ **do** $e \leftarrow$ A* search(e)
// Call bipartite graph and perfect matching based LBAP() in [11]
4 $MT^{t}, Path^{t} \leftarrow graphBasedLBAP(G_{cb})$
5 $MT^{t}, Path^{t} \leftarrow avoidCollision(MT^{t}, Path^{t})$
6 $T^{t} \leftarrow T^{t-1} + MT^{t}$ *// Algorithm ends*

With current positions pCG^{t-1} from the execution of last iteration of n robots and the computed n next positions pCG^{t}, there is a problem with how to assign each robot to go to which newly computed position, so that MT^{t}, which is decided by the bottleneck longest distance of all pairs, is minimum. We formally model this problem as the *linear bottleneck assignment problem (LBAP)*. Mathematically, LBAP is:

$$MT^{t} = \min_{x_{ij}\in\{0,1\}} \max_{i,j} c_{ij}x_{ij}, \tag{5}$$

$$\text{subject to } \sum_{i=1}^{n} x_{ij} = 1, \quad j = 1, ..., n,$$

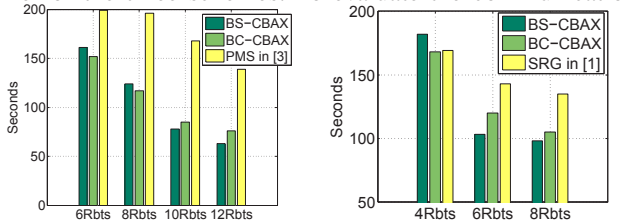
$$\sum_{j=1}^{n} x_{ij} = 1, \quad i = 1, ..., n,$$

$$x_{ij} \in \{0, 1\}, \quad i, j = 1, ..., n$$

Where $\{c_{ij}\}_{n\times n}$ is the cost matrix and c_{ij} is the cost as the length of shortest obstacle-aware path by A* from node i to position j. Besides, $\{x_{ij}\}_{n\times n}$ is the resulting binary matrix where x_{ij}=1 *iff* the node i is assigned to position j. We adopt the algorithm in [11] with expected running time O(n^2), which generates the *optimal* result.

V. Performance Evaluation

We evaluate CBAX's performance using a multi-robot simulator in C++ and compare CBAX with two recent schemes that consider limited communication range: i) A distributed "Sensor-based Random Graph" (**SRG**) scheme in [1], which is more efficient but does not guarantee connectivity; ii) A centralized "Possible Moves Sampling" (**PMS**) approach in [2] that assures connectivity but is less efficient. The metrics for comparison include exploration efficiency and communication quality. We use the total exploration time as the primary criteria for efficiency. If not stated otherwise, the time is measured from a clustered start at the left-bottom corner to a state with over 95% of explored area. Also, the constant STI in each iteration is not included in the measured time for the three schemes. To evaluate the communication



(a) Exp. time: CBAX vs. PMS in [2] with 6,8,10,12 of robots in the *open* environment. (b) Exp. time: CBAX vs. SRG in [1] with 4,6,8 robots in the *garden* environment.

Fig. 4. Simulation results on exploration time.

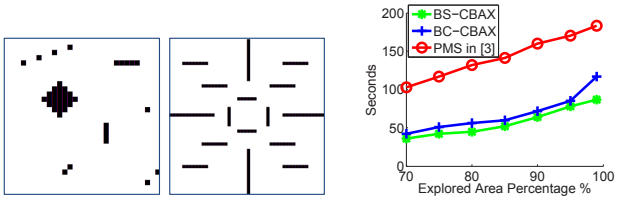
quality, the *Non-Overflow Transmission Time ratio* (**NO_TT**), or the percentage of transmission time when over-saturated flows do not occur divided by the total transmission time, is used. Besides, the percentage of iterations in which all robots maintain connected with the BS and within the robot team, or *fully-Connected Time ratio* with the Base Station (**CT_BS**) and within the Robot Team (**CT_RT**), are also measured. Two environments, the *Open* and *Garden* showed in Fig. 5(a), are used to compare the schemes.

The parameters are set to make the comparisons fair. We obtain the results and parameters from SRG's sample video in and simulate CBAX using the same set of parameters (with proper scaling) and the same *Garden* environment. While teams with only 4,6,8 robots are tested in SRG, we simulate PMS and compare with CBAX on 6-12 robots to obtain results for more complex topologies using the *Open* environment. Specifically, the grid cell and robot size are 1m, the robot's moving velocity is 1m/s, and the accelerating/decelerating time is ignored. Map sizes for the *Open* and *Garden* environments are 45m×45m and 48m×48m and (r_c, r_s) are (10m,7m) and (21m,7m) respectively. All experiments are run five times to obtain the average. For CBAX, two relay placement strategies are evaluated: **BS-CBAX**: CBAX with Bandwidth-Sufficient relay placement; **BC-CBAX**: CBAX with Bandwidth-Constrained relay placement (γ=3).

Exploration efficiency: Compared with PMS, BS-CBAX and BC-CBAX reduce explore time on average by 40.9% and 39.7% respectively as shown in Fig. 4(a). The improvement is more significant when team size increases. With 12 robots, the improvement are 54.7% and 45.3%. The major reason is that CBAX expends nodes promptly and systematically places nodes to reduce relay robots while in PMS, nodes are expended slowly especially when number of nodes increase. In addition, Fig. 5(b) shows that CBAX is more efficient than PMS with different percentages of explored area. Compared with SRG, CBAX remains the more efficient approach, even though SRG is a distributed and efficient scheme maintaining neither connectivity nor bandwidth adequacy. BS-CBAX and BC-CBAX decrease explore time on average by 15.9% and 13.0% respectively as shown in Fig. 4(b). With 8 robots, the improvement are 27.4% and 22.2%.

Communication quality: Compared with PMS that only guarantees connectivity, CBAX futher assures the real-time data flow of being under link capacity. In PMS, overflow transmissions occur in 21.5% of the iterations on average, in which 31.5% of data is lost. While CBAX always maintains connectivity and ensures no overflow occur, SRG with 4 robots[1] shows that only in 20.7% of iterations all nodes are (through multi-hops) connected to BS and 48.3% of iterations all nodes are connected within the robot team. Besides, in 79.3% of iterations overflow transmission occurs and 83.7% of data is lost in such overflow sessions. Fig. 6 also depicts the results.

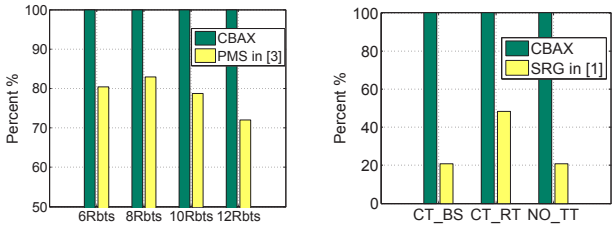
[1]Video available online is only with 4 robots. However, the results with 6/8 robots are similar since SRG is not aware of connectivity and bandwidth.



(a) The *Open* and *Garden* environments where black parts denote obstacles.

(b) Exp. time: CBAX vs. PMS in [2] with varying percentages of explored area (10 Robots).

Fig. 5. Environments and exploration time.



(a) **NO_TT**: CBAX vs. PMS in [2] with 6,8,10,12 of robots in the *open* environment.

(b) **CT_BS**, **CT_RT**, and **NO_TT**: CBAX vs. SRG in [1] with 4 robots in the *garden* environment.

Fig. 6. Simulation results on communication quality.

VI. Conclusion and Future Work

Mobility and communication are two critical and interdependent topics and we believe that jointly considering the two aspects is essential in many research fields. From the mobile robotics viewpoint, we leverage algorithmic and graph-theoretic tools to systematically solve the connectivity and bandwidth-aware multi-robot exploration problem. CBAX achieves both improved efficiency in exploration time and enhanced quality for communication. From the multi-hop mobile network perspective, CBAX demonstrates how a pragmatic coordinated mobility model can ensure the network connectivity and enhance its quality of service. Future work includes: i) Implementing CBAX in a real-robot test bed; ii) Considering heterogeneous communication ranges and data rates along with the impact of interference.

References

[1] A. Franchi, L. Freda, G. Oriolo, and M. Vendittelli, "The sensor-based random graph method for cooperative robot exploration," *Mechatronics, IEEE/ASME Transactions on*, vol. 14, no. 2, April 2009.

[2] M. Rooker and A. Birk, "Multi-robot exploration under the constraints of wireless networking," *Control Engineering Practice*, 2007.

[3] H. Sugiyama, T. Tsujioka, and M. Murata, "Coordination of rescue robots for real-time exploration over disaster areas," in *Object Oriented Real-Time Distributed Computing, IEEE ISORC 2008.*

[4] Z. Cen, M. Mutka, Y. Liu, A. Goradia, and N. Xi, "Qos management of supermedia enhanced teleoperation via overlay networks," in *IROS 2005*.

[5] X. Cheng, D. Du, L. Wang, and B. Xu, "Relay sensor placement in wireless sensor networks," *Wireless Networks*, vol. 14, no. 3, 2008.

[6] D. Du and X. Hu, *Steiner Tree Problems In Computer Communication Networks*. World Scientific Publishing Co., 2008, pp. 177–193.

[7] M. Zavlanos and G. Pappas, "Distributed connectivity control of mobile networks," *Robotics, IEEE Transactions on*, 2008.

[8] K. Wurm, C. Stachniss, and W. Burgard, "Coordinated multi-robot exploration using a segmentation of the environment," in *IROS 2008*.

[9] H. Gonzales and J. Latombe, "Navigation strategies for exploring indoor environments," *International Journal of Robotics Research*.

[10] S. Russell and P. Norvig, *Artificial Intelligence: A Modern Approach, 2nd Edition*. Prentice-Hall, 2003, pp. 97–101.

[11] U. Pferschy, "The random linear bottleneck assignment problem," *RAIRO-Operations Research*, vol. 30, no. 2, 1996.