IEEE Transactions on Cybernetics. This is the author's version of an article that has been published. Changes were made to this version by the publisher prior to publication. (https://doi.org/10.1109/TCYB.2022.3166481)

A Multi-Robot Person Search System for Finding Multiple Dynamic Users in Human-Centered Environments

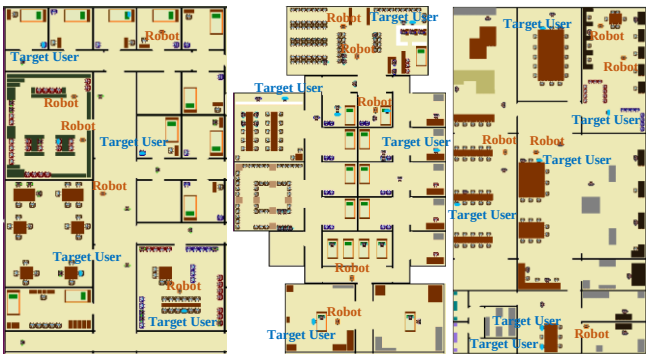
Sharaf C. Mohamed, Angus Fung, and Goldie Nejat, *Member, IEEE*

***Abstract*— Multi-robot coordination for finding multiple users in an environment can be used in numerous robotic applications including search and rescue, surveillance/monitoring, and activities of daily living assistance. Existing approaches have limited coordination between robots when generating team plans or do not consider user location probability within these plans. This results in long searches and robots potentially revisiting the same locations in succession. In this paper, we present a novel multi-robot person search system to generate search plans for multi-robot teams to find multiple dynamic users before a deadline. Our approach is unique in that it simultaneously considers the search actions of all robots and user location probabilities when generating team plans, where user location probabilities are represented as conditional spatial-temporal probability density functions. We model this multi-robot person search problem as a two-stage optimization problem to maximize the expected number of users found before the deadline. Stage 1 solves the action selection problem to determine a set of team actions, and the second stage solves the action allocation problem to distribute these actions amongst the robots. Namely, in stage 1, a novel conditional multi-period multi-knapsack problem is modeled as a min-flow graph solved sequentially by the Bellman-Ford shortest path algorithm. Stage 2 is a variant of the min-max multi-traveling salesperson problem which models the environment topology as a search region network and search times selected by the previous stage. This stage is solved by a novel fuzzy clustering method. Numerous experiments comparing our proposed method to other existing approaches with varying environment sizes, search durations, and number of users showed that our approach was able to find more target users before a defined deadline.**

***Index Terms*— Multi-Robot Coordination, Multiple Dynamic People, Search Within a Deadline, Multi-Period Multi-Knapsack Problem, Min-Max Multi-Traveling Salesperson Problem.**

I. Introduction

Robots need to search for people in various environments in order to provide assistance. Examples include discovering victims in disaster scenes [1]–[8]; finding lost persons in the wilderness [9]–[12]; surveilling urban areas [13]–[17]; and locating users in homes/offices to assist with daily tasks [18]–[22]. These searches are conducted by either a single robot [1], [10], [18]–[21] or a team of robots [2]–[7], [9], [11]–[17], [22]. The single robot search problem has been extensively studied, however, the number of search targets and search environment size are often too large for a single robot to find the users within a timeframe. This may result in loss of life, personal or property damage, or failure to provide assistance. For example, finding residents in long-term care, patients and staff in hospitals, employees in an office or victims trapped in a building; example scenarios presented in Fig. 1. Hence, there is a need to deploy multiple robots.



(a) (b) (c)

Fig. 1. Examples of typical scenarios of a floor in a: (a) long-term care home, (b) hospital, and (c) office building. Robots are orange and target people are cyan. Full sized images are included in the Supplementary Material A.

A number of existing multi-robot person search techniques have considered robots that plan independently, where individual robots do not consider other robots in their plans [9], [12], [14]. As plans are independent, multiple robots may end up searching: 1) the same location in close succession, even when it is unlikely for a search target to move to that location between these searches [9], [12], or 2) locations with low probabilities of containing a search target if robots are assigned to unique regions within the environment [14].

In contrast to planning independently, team coordination allows robots to be distributed within an environment visiting multiple locations with a high probability of containing a user. Coordination can either be weak or strong. Weak coordination planners have limited team coordination, as a robot only considers other robot actions in later stages of the planning process [6], [7], [9], [11], [15]–[17], [22]. To accommodate

Manuscript received July 16, 2021; revised Sept 17, 2021; accepted March 28, 2022. This work was supported by the Natural Sciences and Engineering Research Council of Canada (NSERC) and NSERC CREATE HeRO fellowship, the Ontario Centres of Excellence (OCE), AGE-WELL Inc., and the Canada Research Chairs program (CRC).

The authors are with the Autonomous Systems and Biomechatronics Laboratory, Department of Mechanical and Industrial Engineering, University of Toronto, Toronto M5S3G8 ON, Canada (e-mail: {sharaf.mohamed, angus.fung}@mail.utoronto.ca; nejat@mie.utoronto.ca).

Digital Object Identifier (DOI): see top of this page.

actions selected earlier in the process, the actions selected later are often restricted to either searching a location with a low user probability or incurring a large travel time.

Strong coordination approaches simultaneously consider all robots when planning [2]–[5], [13]. They reason about the environment layout but do not consider user location information in planning. Therefore, these approaches focus on optimizing the search for the entire environment (i.e., coverage). Many robotic search applications require a search deadline to ensure completion of the tasks at hand, including locating multiple passengers in airports to guide them to their gates [23]; finding survivors within a collapsed building [24]; and locating people in a multi-room environment to provide them with items, information, or reminders [18]–[20].

In our research, we consider robot search problems in human-centered environments that have a deadline and can benefit from prior user location probabilities. For example, our previous work solved the single robot search problem by uniquely predicting user locations conditioned on previous robot search actions [21]. Herein, we introduce a new problem requiring multiple robots to maximize the number of users found given the conditional spatial-temporal user location probabilities. Namely, we present a novel two-stage multi-robot person search system (2-MRPSS) for generating strongly coordinated search plans for a team of robots to maximize the expected number of dynamic users found within a deadline. Our approach uniquely generates team search plans that simultaneously consider the search actions of all robots and user location probabilities. We represent the search problem as a two-stage optimization problem, as there are no existing one-stage methods to solve this unique problem that requires a large solution space. First, we select team actions that maximize the number of users found by uniquely formulating and solving our novel conditional multi-period multi-knapsack problem (CMPKP). Then, we distribute the team actions amongst the robots to minimize the longest duration robot search plan by solving our novel min-max multi-robot search problem (min-max mRSP).

II. Related Work

Herein, we discuss existing multi-robot person search systems (MRPSS) with real-world robotics considerations. We adapt the taxonomy presented in [25] for multi-robot systems (MRS): 1) cooperation (cooperative or non-cooperative), 2) knowledge (aware or unaware of other robots), 3) coordination (strong, weak, or none), and 4) organization (distributed, weakly centralized, or strongly centralized).

MRPSS is a unique subset of MRS, where multiple robots search an environment for people. In general, the majority of existing MRPSS are strongly centralized, with only a handful of distributed approaches. When generating search plans, knowledge and coordination types are important to select robot search locations and times. MRPSS can be classified as follows: 1) unaware with no coordination [9], [12], [14], 2) aware with weak coordination [6], [7], [11], [15]–[17], [22], and 3) aware with strong coordination [2]–[5], [13].

A. Unaware Robot Searches with No Coordination

Unaware robot searches have no coordination between the robots. For example, in [9], UAVs searched for a single victim of a marine disaster using distress signals from victims' last known locations. They were distributed in the environment, and each UAV searched the location which maximized the probability of victim detection per unit of distance traveled. Simulated experiments showed UAV paths would cross frequently and converge causing redundant search actions.

In [12], a robot team searched for a moving victim lost in the wilderness. The victim's location was modeled using iso-probability curves which propagated outward from their last known location. Each robot searched along a unique curve as dictated by a central server. In [14], a team of stationary sentinels and UAVs detected trespassers in an outdoor urban environment. The environment was discretized into cells with equal arrival probability of a trespasser and a loss incurred per unit of time a trespasser was present. Upon sentinel detection of an intruder, a UAV searched within the area using a minimum time coverage approach. Simulated experiments showed performance deterioration for non-uniform loss functions as the sentinel areas had uneven workloads.

B. Aware Robot Searches with Weak Coordination

Weak coordination methods consider robots that are aware of each other's locations, but plan individually [25]. In [16], multiple UAVs were used for surveilling crime-prone areas. The environment was divided into grids with spatial-temporal incident probabilities. Scores were assigned based on when a grid was last searched and its incident detection history. Weak coordination was achieved by sequentially assigning robots to maximum goal locations and reducing the scores of grids within the vicinity. Simulation results showed that using incident detection history improved coverage. However, sequentially assigning robots may result in uneven workloads with some robots travelling large distances.

In [6], a team of UAVs patrolled an urban environment damaged by a disaster to provide assistance such as rescuing victims. Each UAV was assigned a patrolling area with little overlap between the areas. A Monte Carlo look-ahead tree was built independently by each UAV to solve a POMDP. Based on the joint set of trees, the team selected the joint set of actions which maximized information gain without any UAV exceeding a maximum damage threshold incurred from the disaster. Simulated results showed the approach outperformed randomly selecting actions, however, the limited amount of overlap between the patrolling areas led to uneven workloads.

In [17], a team of robots searched for unknown dynamic people. Target motion was modeled using Gaussian processes. A robot trajectory was obtained by either maximizing the expected number of detections (END) or maximizing the mutual information (MI) between target locations and measurements over a fixed horizon. Team trajectories were then selected using a greedy method with individual robot trajectories optimized sequentially and conditioned on the actions of previous robots. Simulations, conducted in a downtown city area, showed that the END objective tracked a

higher number of targets compared to the MI objective. However, generating robot trajectories sequentially, may result in suboptimal plans as the reduced solution space means the solutions will be more tractable but less optimal.

In [22], two weakly coordinated multi-robot approaches for social robots were proposed for searching and tracking a single person in urban environments to assist them using reinforcement learning or a particle filter. Both methods maintained and updated the belief of the person's location. Each robot planned independently using a 1-step lookahead that selected the next goal location based on a person location probability map and the robots' combined observations. Results showed that the particle filter method was able to get robots closer to the person's real position. The robots performed better when they communicated with each other.

In [11], a team of UAVs searched for lost persons in the wilderness. A victim's location was modeled using iso-probability curves, which propagated outwards from their last known location. Each robot was assigned to search along an iso-probability curve. Weak coordination was achieved via a post-hoc step of assigning UAVs to non-intersecting sections of the iso-probability curves to remove redundant actions. These assignments were limited to the previously assigned iso-probability curves. Simulation results showed that the proposed method found more people than coverage methods.

In [7], a multi-robot approach was proposed for search and delivery of medical supply tasks to victims. The tasks were allocated using the Hungarian method to minimize the time required to attend to all victims. The disaster area was partitioned into Voronoi cells with each cell assigned to a robot. A frontier-based exploration method was used to track the centroid of the Voronoi cell. 2D simulations showed that the approach was able to complete a mission scenario. However, partitioning the environment to be uniquely assigned to each robot may lead to uneven workloads.

C. Aware Robot Searches with Strong Coordination

Strong coordination approaches concurrently generate all plans for the aware robots [2]–[5], [13]. In [2], robots searched for static victims in an unknown damaged building. The team plan was based on multi-robot graph traversal algorithms such as breadth-first search (BFS) and abortable Dijkstra's algorithm with Hungarian method (ADAHM). Simulated experiments showed that BFS found the victims the fastest for the large and small environments, and the ADAHM was the fastest in medium environments which had fewer paths between regions. In [3], a team of semi-autonomous robots and human operators searched an unknown disaster scene for trapped static victims. A MAXQ hierarchical controller assigned robots to subscenes, determined when robots should perform victim identification, and when an operator needed to intervene. Simulated experiments showed the robots found 99% of victims with an average coverage of 93.4%.

In [4], a team of robots searched a damaged building for stationary survivors. The team consisted of robots for finding survivor locations without prior information, and robots for rescuing survivors. Robot plans were arbitrarily generated such that a unique robot was assigned to each survivor location. Then, the plans were optimized by exhaustively removing a location from a plan if adding it to another plan would decrease the time to perform the action. In [5], the aforementioned approach was modified to remove locations from a robot plan if another robot could search the location before the victim's condition was critical and removing that location would allow the robot to search a currently unsearched location. Simulations showed the modification increased the number of victims rescued.

In [13], a robot team performed graph clearing to pursue adversarial attackers. Namely, the team searched the environment to find existing attackers while also stopping future attackers from entering the searched environment. Graph clearing used frontier exploration where robots were either guards or followers. The frontiers were contained within the joint field of view of the guards. As the frontier expanded, the followers were added to the set of guards to maintain the required field of view. Simulated experiments demonstrated six robots could clear a hospital wing with seventeen regions.

D. Summary and Challenges

The unaware MRPSS with no coordination consisted of robots in a team having independent search plans which result in suboptimal use of resources. For example, robots may search a location repeatedly without considering the probability of the targets moving to that location between the searches [9], [12] or the workload may be unevenly distributed between robots [14]. Both cases can result in robots being allocated to locations with a low user probability.

Aware MRPSS with weak coordination consisted of 1) assigning robots to areas of the environment with minimal overlap [6], [7], 2) selecting a combination of independently generated plans [15], 3) generating robot plans sequentially [16], [17], [22], or 4) performing a post-hoc step on independent plans to remove redundant actions [11]. Similar to the unaware case, uneven workload may result between the robots and combining independent plans without considering the compatibility between the plans may also result in redundant actions. While the sequential and post-hoc coordination approaches reduce uneven workload and redundancy, actions selected later in the planning process must accommodate for actions already assigned. As a result, a later action is often restricted to either searching a nearby low probability location or incurring a large travel time to search a faraway high probability location.

Aware MRPSS with strong coordination simultaneously generate plans for the entire team [2]–[5], [13]. These coverage and graph clearing problems solve a different problem than our proposed problem as they require the robots to minimize the time to explore or clear an environment, as opposed to maximizing the number of dynamic users found. Additionally, the coverage techniques have focused only on finding static users [2]–[5] and have not yet considered dynamic users in the environment. Namely, they do not search regions that have already been searched, but people are dynamic and can return to already visited regions. The graph

clearing approach in [13] focused on assigning robots to block regions even when there is a low probability that a user may enter them. Therefore, both the coverage and clearing approaches plan consider the environment layout to maximize the number of regions searched, and do not consider user location probabilities. However, reasoning about user location probabilities is important as there is no need to search areas where there is little to no probability of people being there.

We introduce the new problem of multiple robots finding multiple dynamic users in a human-centered environment before a deadline. Due to the aforementioned limitations, the existing methods cannot be directly applied to our novel problem. Thus, we have developed a novel aware 2-MRPSS with strong coordination. Our approach reasons about user location information, represented by conditional spatial-temporal probability density functions (PDFs) to determine the expected number of users found by each search action in the team plan. Therefore, the team search actions have a high probability of finding the users within the search duration.

III. The Multi-Robot Person Search Problem

The multi-robot person search problem is defined herein as a team of robots cooperating to search an environment to find dynamic people within a specified time frame. Examples of typical scenarios are presented in Fig. 1.

Environment: The environment is an indoor area occupied by the users daily and consists of a set of regions $R = \{R_1, \dots, R_I\}$, with a total of I regions.

Target Users: The target users $U' = \{U'_1, \dots, U'_Z\}$ are those who need to be found during the search.

Time Frame, Periods and Windows: The time frame specifies the start time, t^{start}, and end time, t^{end}, of the search. It is divided into Ω discrete time periods $T = (T_1, \dots, T_\Omega)$ of equal length t^{period}. A time window is a contiguous subset of time periods $T_{j,k} = (T_j, \dots, T_k)$.

Robot Team: The robot team consists of B robots $\mathbb{R} = \{\mathbb{R}_1, \dots, \mathbb{R}_B\}$. The robots perform search actions, $a_{i,\omega,t}$, consisting of searching in R_i during T_ω for a duration of t. We discretize the problem by considering durations as a multiple of t^{unit}. $t_i^{i'}$ denotes the time to move between R_i and $R_{i'}$. At the start of the search, each robot $\mathbb{R}_b$ has an initial region $R_0^{(b)}$.

Planning Duration: The planning duration, t^{plan}, is the amount of time allotted for generating the team plan.

Search Query: The search query, S, specifies the target users, the time frame, the team, and the planning duration.

Team Plan: The team plan, TP, is generated to maximize the number of target users found given S. TP specifies a search plan, $SP_\omega^{(b)}$, for each robot $\mathbb{R}_b$ during each time period T_ω consisting of an ordered set of actions:

$$TP = \left(\left\{SP_1^{(1)}, \dots, SP_1^{(B)}\right\}, \dots, \left\{SP_\Omega^{(1)}, \dots, SP_\Omega^{(B)}\right\}\right), \quad (1a)$$

$$SP_\omega^{(b)} = \left(a_{\omega,1}^{(b)}, \dots, a_{\omega,H(b)}^{(b)}\right), \quad (1b)$$

where $a_{\omega,h}^{(b)}$ denotes the h^{th} search action of $SP_\omega^{(b)}$ which occurs at $R_{\omega,h}^{(b)}$ during T_ω for a duration of $t_{\omega,h}^{(b)}$. We define $t\left(SP_\omega^{(b)}\right)$ as the time required to perform $SP_\omega^{(b)}$. Multiple robots can work together to search R_i during T_ω.

Team Action: A team action, $a_{i,\omega}^*$, is the culmination of search actions performed in R_i during T_ω. The time associated with a team action, $t\left(a_{i,\omega}^*\right)$, is determined as the total time of all the search actions performed in R_i during T_ω:

$$t\left(a_{i,\omega}^*\right) = \sum_{\mathrm{a}_{k,h}^{(\mathrm{b})} \in \mathbb{S}_{i,\omega}} \left(t_{k,h}^{(b)}\right), \quad (2a)$$

$$\mathbb{S}_{i,\omega} = \left\{\mathrm{a}_{k,h}^{(\mathrm{b})} \middle| \mathrm{a}_{k,h}^{(\mathrm{b})} \in TP, R_{k,h}^{(b)} = R_i, T_k = T_\omega\right\}. \quad (2b)$$

A complete list of all the symbols used in this paper is provided in the Supplementary Material B.

IV. Proposed Multi-Robot Person Search System

Our proposed 2-MRPSS generates a team plan, TP, that maximizes the number of target users found within a time frame, as indicated by the team total reward, $W(TP)$. Namely, TP is generated such that it maximizes $W(TP)$ without any robot exceeding the allowable search time in any time period:

$$\begin{aligned} \underset{TP}{\text{maximize}} \quad & W(TP), \quad (3) \\ \text{subject to} \quad & t\left(SP_\omega^{(b)}\right) \le t^{period}, \forall SP_\omega^{(b)} \in TP, \\ \text{where} \quad & t\left(SP_\omega^{(b)}\right) = \sum_{a_{\omega,h}^{(b)} \in SP_\omega^{(b)}} \left(t_{\omega,h}^{(b)} + t_{R_{\omega,h-1}^{(b)}}^{R_{\omega,h}^{(b)}}\right). \end{aligned}$$

The time spent by a robot $t\left(SP_\omega^{(b)}\right)$ is the summation of time spent both searching and moving to its allocated regions.

A. 2-MRPSS Framework

The problem of generating a team plan is NP-hard, as is even generating a single robot search plan [21]. Therefore, it is computationally impractical to solve the entire problem together due to the large number of possible team plans, $I^{\left(\Omega \times B \times \frac{t^{period}}{t^{unit}}\right)}$. For example, for a problem size of $I = 30$ regions, $\Omega = 3$ time periods, $B = 3$ robots, $t^{period} = 900s$, and $t^{unit} = 15s$, the total number of possible team plans is 4.4×10^{697}. Our 2-MRPSS solves the problem through a two-stage approach: 1) first selecting team search actions, and 2) assigning these team actions among the robots to generate a team plan. As there is no available one-stage approach to solve our problem, in our two-stage approach, the system may not know the actual travel time to arrive at a specific region (the search ordering of these regions is not determined until the second stage). Instead, a constant travel time is estimated. However, in a realistic scenario: 1) successive regions are expected to have low travel times, 2) the estimated travel time will be close to the actual through iteration, and 3) optimal solutions will spend more time searching than traveling. Thus, it is expected that a two-stage approach will have the same optimal solutions as a hypothetical one-stage approach. In fact, the exponential reduction in solution space means that for real-time applications, the quality of a two-stage approach will be better. The 2-MRPSS framework is presented in Fig. 2.

Prior to the search, user location data is acquired during several days while the user performs daily activities. This data is used to determine user location PDFs, representing the

probability of a user being in a region R_i during a time window $T_{j,k}$. Based on the PDFs, rewards for the team actions are assigned. Then, a set of unallocated team search actions, UA, is selected to maximize the reward acquired by solving a new variant of the multi-knapsack problem (mKP) [26], which we define herein as the conditional multi-period multi-knapsack problem (CMPMKP). The CMPMKP, stage 1 of our approach, is modeled by a min-flow graph. In stage 2, the actions in UA are allocated to the robots such that the longest robot plan in each time period is minimized. For this allocation, we have developed an extension of the min-max multi-traveling salesperson problem (min-max mTSP) [27], which we define as the min-max multi-robot search problem (min-max mRSP). The min-max mRSP is modeled by a search region network. If the resulting TP is infeasible, i.e., it cannot be executed within the allotted time, the procedure is iterated to determine UA using a larger travel time estimate. During the plan execution, if a target user is found, replanning generates a plan optimized for the remaining target users.

The below subsections present the detailed procedure of our 2-MRPSS framework highlighting our novel contributions.

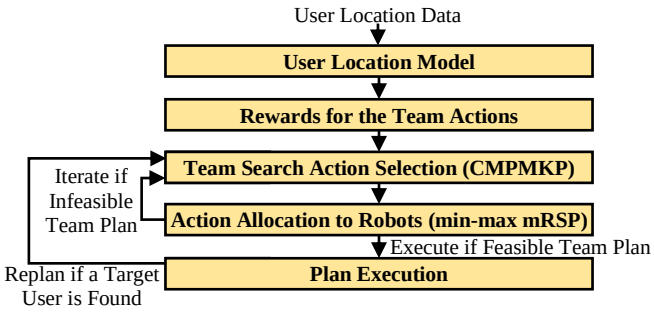


Fig. 2. 2-MRPSS framework.

B. User Location Model

The user location model is extended from [21] for a team of robots. Herein, we remove the assumption that the start and end time of data acquired on the users must align with the start and end of time periods, and by integrating over the time periods. We capture the conditional dependence of user location probabilities in sets of non-contiguous time periods i.e., a user in R_1 for T_1 and T_3, but not during T_2. The model formulation is further discussed in Supplementary Material C.

C. Rewards for the Team Actions

The reward for UA is denoted as $W(UA)$ and is based on the expected number of target users found when executing UA:

$$W(UA) = \sum_{z=1}^{Z} P(\theta_z^{UA}), \tag{4}$$

where $P(\theta_z^{UA})$ is the probability of finding a target user U_z' when executing UA. $P(\theta_z^{UA})$ can be expressed in terms of $\theta_{z,i}^{UA}$ which indicates the occurrence of finding U_z in R_i when executing UA. Namely, the probability of finding U_z' is equal to the probability of finding U_z' in any region:

$$P(\theta_z^{UA}) = P\left(\bigcup_{i=1}^{I} \theta_{z,i}^{UA}\right). \tag{5}$$

Combining Eqs. (5) and (6), the reward is:

$$W(UA) = \sum_{z=1}^{Z} \left(P\left(\bigcup_{i=1}^{I} \theta_{z,i}^{UA}\right)\right), \tag{6}$$

and is updated to assume that a user will never be found in two regions during the full execution of a team plan, and as a result $\theta_{z,i}^{UA}$ and $\theta_{z,i'}^{UA}$ $\forall i' \neq i$ are treated as mutually exclusive, allowing the union to be computed as a summation:

$$W(UA) = \sum_{z=1}^{Z}\left(\sum_{i=1}^{I} P\left(\theta_{z,i}^{UA}\right)\right). \tag{7}$$

The revised reward captures that replanning occurs when a user is found, creating a new plan optimized for the remaining users. $P\left(\theta_{z,i}^{UA}\right)$ is determined by a local planner that generates a plan for searching within a specific region given the time to search the region in all time periods, $\left\{t\left(a_{i,k}^{*}\right)\forall k \in [1,\omega]\right\}$. Our proposed 2-MRPSS can incorporate any local search planner that can provide $P\left(\theta_{z,i}^{UA}\right)$. This paper presents a multi-robot multi-period coverage planner that uses a grid-based technique to generate a team plan to search within a region. The technique divides the region into cells and assigns each robot a set of cells to search in each time period. Details of this local search method are present in the Supplementary Material D.

D. Team Search Action Selection

Based on the rewards for the team actions, a set of search actions is determined. Selecting the search actions for each individual robot is infeasible due to the large number of combinations; e.g., for $I = 30$ regions, $t^{period} = 900s$, $B = 3$ robots, and $t^{unit} = 15s$, there is a total of $\left(\frac{t^{period}}{t^{unit}}\right)^{\mathrm{IB}} = \left(\frac{900}{15}\right)^{(30)(3)} \approx 1.1 \times 10^{160}$ combinations of unordered search plans. Instead, the team search action selection determines the team actions in UA to maximize the total reward acquired:

$$\underset{t\left(a_{i,\omega}^{*}\right)\in UA}{\text{maximize}} \; W(UA) = \sum_{i=1}^{I}\left(\sum_{z=1}^{Z} P\left(\theta_{z,i}^{UA}\right)\right), \tag{8}$$

$$\text{subject to } \sum_{i=1}^{I}\left(t\left(a_{i,\omega}^{*}\right) + \gamma_{i,\omega} t^{move}\right) \leq B t^{period}, \forall \omega \in [1,\Omega],$$

$$\text{where } \quad \gamma_{i,\omega} = \begin{cases} 0, \; if \; t\left(a_{i,\omega}^{*}\right) = 0 \\ 1, \; if \; t\left(a_{i,\omega}^{*}\right) > 0 \end{cases}, \forall \omega \in [1,\Omega], \forall i \in [1,\mathrm{I}].$$

As UA is unordered, the actual travel time required to perform the set of actions is estimated to be t^{move}. $\gamma_{i,\omega}$ represents the occurrence of the team searching R_i during T_ω. t^{move} is only added if the region is searched. Bt^{period} represents the search time available for the B robots in the team to perform all the selected team search actions $a_{i,\omega}^{*}$ in each time period T_ω.

Selecting the team search actions allows the team to share actions as needed between the robots. As the team can share actions, it is expected that a team plan TP can be generated from UA such that each robot completes its search plan in each time period within the allotted time t^{period}. The iteration step in Fig. 2 is incorporated to check the feasibility of the team plan TP using Eq. (3), and if the plan is infeasible, UA is replanned with a larger travel time estimate t^{move} in Eq. (8).

To optimally solve Eq. (8), we have developed a new CMPMKP solver extended from the single robot method we presented in [21]. It uses a min-flow graph to consider a discrete combination of team search actions given that both t^{move} and $t\left(a_{i,\omega}^{*}\right)$ must be multiples of t^{unit}. t^{unit} is introduced to reduce the infinite set of continuous actions to a

finite set of discrete actions. The time elapsed in each time period, Bt^{period}, is the cumulative maximum allowable search time for the B robot team. This allows the approach to select a set of team search actions in which each robot will spend approximately t^{period} searching within each time period. For problems with large search durations, solving the CMPMKP is time-consuming, and a time-efficient approximation is to solve the min-flow graph in each time period sequentially [21]. For the multi-robot search, we use a sequential approach as the time allotted in each time period, Bt^{period}, can be large. There are Ω min-flow graphs, one for each time period, denoted as $(G_1, \dots, G_\Omega)$. Each graph is solved sequentially starting at G_1.

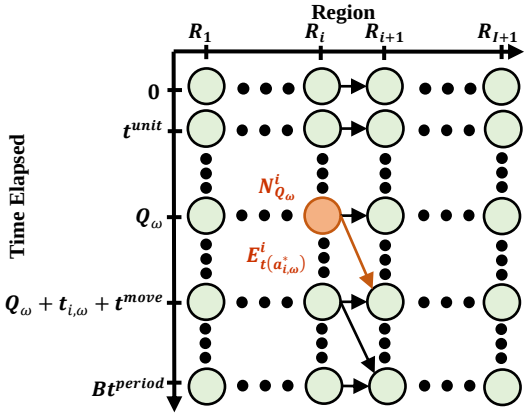


Fig. 3. Sequential min-flow graph G_ω for the CMPMKP.

The sequentially min-flow graph G_ω used to solve the CMPMKP is presented in Fig. 3. The *y*-axis represents a time elapsed in the time period, Q_ω and the *x*-axis represents a region R_i. At each node $N^i_{Q_\omega}$, a decision is made for how much time the team will spend searching R_i in T_ω. This considers that Q_ω time has already been allocated to $\{R_1, \dots, R_{i-1}\}$ by G_ω, and $\left(t(a^*_{i,1}), \dots, t(a^*_{i,\omega-1})\right)$ search times have already been allocated to R_i in $(T_1, \dots, T_{\omega-1})$ by $(G_1, \dots, G_{\omega-1})$. Each possible decision at a node is represented by an edge $E^i_{t(a^*_{i,\omega})}$ indicating a transition from $N^i_{Q_\omega}$ to $N^i_{Q_\omega + t(a^*_{i,\omega}) + \gamma_{i,\omega} t^{move}}$ based on the selected team search time $t(a^*_{i,\omega})$.

Each edge cost, $W\left(E^{i,\omega}_{t(a^*_{i,\omega})}\right)$, is computed based on the negative expected number of users found when performing the action, $UA_{i,\omega}$, corresponding to searching R_i for time $t(a^*_{i,\omega})$:

$$W\left(E^{i,\omega}_{t(a^*_{i,\omega})}\right) = -\sum_{z=1}^{Z} P\left(\bigcup_{\omega'=1}^{\omega} \theta_{z,i}^{UA_{i,\omega'}}\right) + \sum_{z=1}^{Z} P\left(\bigcup_{\omega'=1}^{\omega-1} \theta_{z,i}^{UA_{i,\omega'}}\right). \quad (9)$$

UA_ω denotes the team actions selected during T_ω, and is determined to be the set of actions, $\{UA_{1,\omega}, \dots, UA_{I,\omega}\}$ that form the minimum cost path across G_ω as selected using the Bellman-Ford algorithm [28]. UA is then the union of the actions in each time period $(UA_1, \dots, UA_\Omega)$.

E. Action Allocation to Robots

UA is used to generate a TP that satisfies Eq. (3). Namely, we allocate the team search actions during each time period T_ω, denoted as UA_ω, among the robots to generate TP_ω, which minimizes the duration of the longest robot plan during T_ω:

$$\min_{TP_\omega} \max_{b} \quad \sum_{a^{(b)}_{\omega,h} \in TP_\omega} \left(t^{(b)}_{\omega,h} + t^{R^{(b)}_{\omega,h}}_{R^{(b)}_{\omega,h-1}} \right), \quad (10)$$

$$\text{subject to } \sum_{a^{(b)}_{k,h} \in \mathbb{S}_{i,\omega}} \left(t^{(b)}_{k,h}\right) = t\left(a^*_{i,\omega}\right), i \in [1, \mathrm{I}],$$

$$\text{where} \quad \mathbb{S}_{i,\omega} = \left\{ \mathrm{a}^{(\mathrm{b})}_{k,h} \middle| \mathrm{a}^{(\mathrm{b})}_{k,h} \in TP, R^{(b)}_{k,h} = R_i, T_k = T_\omega \right\}.$$

The above min-max objective is used as all search plans will be feasible within the allotted time if the longest duration plan is feasible. Namely, if any mapping $f: UA_\omega \to TP_\omega$ generates a feasible team plan, the min-max TP_ω will also be feasible.

To address the problem in Eq. (10), we extend the min-max mTSP [27] and introduce our new min-max mRSP. The min-max mTSP considers the problem of a team visiting multiple regions while minimizing the longest duration robot plan [27]. This only considers closed tours, which arbitrarily start and end at one of the regions to be visited, as well as a single robot visiting each region. The duration of a closed tour is invariant to the selected start/end region. However, we need to consider the starting regions of the robots which may not coincide with the regions in UA_ω. As a result, there may be additional time required for a robot to move to the first search region. There is also no need for a robot to return to its starting location, and it may be beneficial for multiple robots to work together in searching a single region. Therefore, our min-max mRSP includes the additional considerations of robots: 1) starting regions, 2) not returning to their starting regions, and 3) working together to complete a single team search action. We model the min-max mRSP using a search region network, where the distances between regions are stored in a distance matrix, and the search times for the regions are stored in a vector. To solve this min-max mRSP, we present a novel fuzzy clustering team search action allocator. Fuzzy clustering was used as it is the only clustering method that allows for region sharing between robots. Hence, multiple robots can collaboratively search a single region. This differs from existing approaches that require only one assigned robot to search a specific region without the aid of other robots.

1) Fuzzy Clustering Team Search Action Allocator

Fuzzy clustering [29] is a clustering method in which a team action can be shared between multiple robots. Our fuzzy clustering approach uses expectation maximization (EM) [30]. In each time period, T_ω, a set of fuzzy clusters, $FC_\omega = \{FC_{\omega,1}, \dots, FC_{\omega,F}\}$ is considered. Each cluster is represented as $FC_{\omega,f} = \{\mathbb{R}_{\omega,f}, \rho_{1,\omega,f}, \dots, \rho_{I,\omega,f}\}$, where f is the cluster's unique ID, $\mathbb{R}_{\omega,f}$ is the robot assigned to the cluster, and each $\rho_{i,\omega,f}$ is the ownership of $FC_{\omega,f}$ over action $a^*_{i,\omega} \in UA$. The set of parameters $\mathbb{R}_{\omega,f}, \forall f \in [1, F]$ and $\rho_{i,\omega,f}, \forall i \in [1, I]$ are determined to minimize the highest cluster cost while the robot team completes the team actions specified in UA_ω:

$$\min_{FC_\omega} \max_{f} \quad \Psi\left(FC_{\omega,f}\right), \quad (11)$$

$$\text{subject to } \sum_{f \in [1,F]} \left(\rho_{i,\omega,f}\right) = 1, i \in [1, \mathrm{I}],$$

where $\Psi\left(FC_{\omega,f}\right)$ is the cost of cluster $FC_{\omega,f}$, representing the

amount of time the robot $\mathbb{R}_{\omega,f}$ would take to perform the set of actions associated with $FC_{\omega,f}$. The constraint in Eq. (11) guarantees all actions in UA_ω are completed by the team.

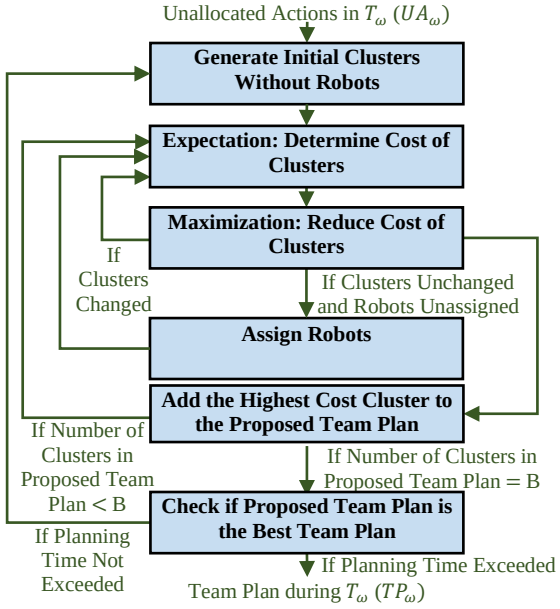


Fig. 4. Architecture of fuzzy clustering solver for min-max mRSP.

Our overall fuzzy clustering approach is presented in Fig. 4. Starting with UA_ω, we generate initial clusters without robots, i.e. $\mathbb{R}_{\omega,f}$ = NULL. In the expectation step the cost of each cluster, $\Psi(FC_{\omega,f})$, is evaluated. Next, the maximization step updates the values of $\rho_{i,\omega,f}$ to reduce the highest cost cluster. Then the expectation step is repeated. If no beneficial change is made by the maximization step, the first EM phase is completed, and we assign robots to the clusters and the EM phase is repeated with the assigned robots. After the second EM phase, we add the highest cost cluster to the proposed team plan as a robot search plan. If the team plan has fewer than B robot search plans, the EM phase with robots is repeated to minimize the maximum cost of the remaining clusters. Once the proposed team plan has B robot search plans, we check if it is the best team plan generated thus far with minimum longest duration search plan, Eq. (10). If planning time remains, the entire approach is repeated to generate multiple proposed team plans. If not, the best proposed team plan is output as TP_ω.

We first perform an EM phase on clusters with the robots unassigned to distribute actions without being restricted by robot locations. Then, robots are assigned for the second EM phase so the clusters can optimize for the robot locations. Moreover, by iteratively adding the highest cost cluster, the EM can further improve all cluster costs. The details of each module in our clustering architecture are discussed below.

a. Generate Initial Clusters Without Robots

Using the unallocated actions UA_ω, an initial set of $F = B$ clusters FC_ω is generated such that the joint ownership of all the clusters accounts for all actions in UA_ω, Eq. (11). The approach used to determine the initial clusters is K-means++, which fully assigns each action $a^*_{i,\omega}$ with search duration $t(a^*_{i,\omega}) > 0$ to a single cluster, $FC_{\omega,f}$, by setting the corresponding $\rho_{i,\omega,f}$ to 1 [31]. $\mathbb{R}_{\omega,f}$ is set to NULL for all clusters, indicating that a robot is not assigned. Once the initial cluster FC_ω is generated, the expectation is calculated.

b. Expectation: Determine Cost of Clusters

To compute the cost of a cluster $\Psi(FC_{\omega,f})$ we first determine the order in which the actions will be performed, referred to as the cluster plan $CP_{\omega,f} = \left(a^{(f),CP}_{\omega,1}, \dots, a^{(f),CP}_{\omega,M^{CP}_f}\right)$. $a^{(f),CP}_{\omega,g}$ denotes the g^{th} search action of the cluster plan with search region $R^{(f),CP}_g$, time period T_ω, and search duration $t\left(a^{(f),CP}_{\omega,1}\right)$. $CP_{\omega,f}$ must complete all actions with both non-zero ownership and search duration in $FC_{\omega,f}$. If $a^*_{i,\omega}$ is mapped to $a^{(f),CP}_{\omega,g}$, then $t\left(a^{(f),CP}_{\omega,g}\right) = \rho_{i,\omega,f} t(a^*_{i,\omega})$ and $R^{(f),CP}_g = R_i$. To determine $CP_{\omega,f}$, we solve the TSP [32]. TSP_1, considers when the cluster does not have a robot assigned and $\mathbb{R}_{\omega,f} = NULL$. TSP_2, considers when $\mathbb{R}_{\omega,f} \neq NULL$.

$\boldsymbol{TSP_1}$**:** During the first EM phase, $\mathbb{R}_{\omega,f} = NULL$, a closed tour is considered with arbitrary start/end region, $R^{(f),CP}_1$. The objective is to generate $CP_{\omega,f}$ with minimum total travel time:

$$\Psi(FC_{\omega,f}) = \min_{CP_{\omega,f}} \sum_{g=2}^{M^{CP}_f} t^{R^{(f),CP}_g}_{R^{(f),CP}_{g-1}} + t^{R^{(f),CP}_1}_{R^{(f),CP}_{M^{CP}_f}}. \quad (12)$$

$\boldsymbol{TSP_2}$: During the second EM phase, $\mathbb{R}_{\omega,f} = \mathbb{R}_b$, the start region is the robot's initial region during T_ω, denoted as $R^{(b)}_{\omega,0}$, and the robot is not required to return to $R^{(b)}_{\omega,0}$. The objective is again to generate $CP_{\omega,f}$ while minimizing total travel time:

$$\Psi(FC_{\omega,f}) = \min_{CP_{\omega,f}} t^{R^{(f),CP}_1}_{R^{(b)}_{\omega,0}} + \sum_{g=2}^{M(f)} t^{R^{(f),CP}_g}_{R^{(f),CP}_{g-1}}. \quad (13)$$

For both TSP cases, a Lin-Kernighan heuristic (LKH) approach [33] is used to generate the solution in real-time. After solving the TSP, we have the cost of each cluster $\Psi(FC_{\omega,f})$ and the current value of the objective in Eq. (12). To minimize the objective, we perform the maximization step.

c. Maximization: Reduce Cost of Clusters

To reduce the cluster with the highest cost $\overline{FC}_\omega$, we aim to transfer ownership to another cluster FC'_ω. To achieve this, we first attempt to transfer ownership from $\overline{FC}_\omega$ to the cluster closest to it, $F\dot{C}_\omega$, where cluster distance is defined by their closest pair of actions. If this transfer results in an increase to the objective in Eq. (11), then the transfer is not performed, and we group $\overline{FC}_\omega$ and $F\dot{C}_\omega$ into a group called the close set, CS. The remaining clusters are grouped into the far set, FS. After creating CS and FS, we iteratively attempt to transfer ownership from CS to FS using the procedure in Fig. 5.

Step 1: Create Close and Far Sets of Clusters

Initially, we create a close set of clusters, CS, containing only the cluster with the highest cost $\overline{FC}_\omega$ and a far set of clusters, FS, containing the rest of the clusters in FC_ω.

Step 2: Transfer Action Ownership from Close Set to Far Set

To transfer action ownership from *CS* to *FS*, the closest pair of clusters between the two sets are selected, $\{FC_{\omega,f} \in CS, FC_{\omega,f'} \in FS\}$. The distance $CT_f^{f'}$ between $FC_{\omega,f}$ and $FC_{\omega,f'}$ is determined based on their closest pair of actions with non-zero search times, $a_{\omega,g}^{(f),CP}$ and $a_{\omega,g'}^{(f'),CP}$. The distance between $a_{\omega,g}^{(f),CP}$ and $a_{\omega,g'}^{(f'),CP}$ is the travel time for moving between their corresponding regions.

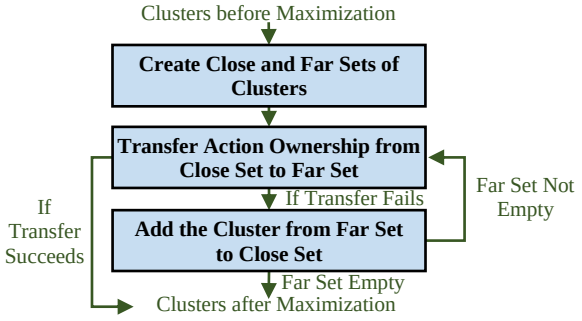


Fig. 5. Flow chart of maximization step.

We aim to transfer ownership $\rho_{i,\omega,f}$ of $a_{\omega,g}^{(f),CP}$ from $FC_{\omega,f}$ to $FC_{\omega,f'}$. Similar to when solving the CMPMKP, we require that all search actions $a_{\omega,1}^{(f),CP}$ have a duration $t\left(a_{\omega,1}^{(f),CP}\right)$ that is a multiple of t^{unit}. Therefore, an amount of ownership of $\rho_{i,\omega,f}$, denoted as $\rho_{i,\omega}^{+}$, can only be transferred in multiples of $\frac{t^{unit}}{t\left(a_{i,\omega}^{*}\right)}$. After a transfer, the resulting ownerships of $FC_{\omega,f}$ and $FC_{\omega,f'}$ over action $a_{i,\omega}^{*}$ are:

$$\rho_{i,\omega,f}^{+} = \rho_{i,\omega,f} - \rho_{i,\omega}^{+}, \tag{14a}$$

$$\rho_{i,\omega,f'}^{+} = \rho_{i,\omega,f'} + \rho_{i,\omega}^{+}. \tag{14b}$$

The new costs are $\Psi^{+}\left(FC_{\omega,f}\right)$ and $\Psi^{+}\left(FC_{\omega,f'}\right)$, respectively. We select the largest $\rho_{i,\omega}^{+}$ such that $\Psi^{+}\left(FC_{\omega,f'}\right)$ does not exceed the worst cluster cost $\Psi\left(\overline{FC_f}\right)$. Also, the new receiving cluster cost $\Psi^{+}\left(FC_{\omega,f'}\right)$ must not exceed the original giving cluster cost $\Psi\left(FC_{\omega,f}\right)$ if on the next iteration of the maximization step $FC_{\omega,f'}$ will receive an action before $FC_{\omega,f}$:

$$\begin{aligned} &\text{maximize } \rho_{i,\omega}^{+}, \\ &\text{subject to } \Psi^{+}\left(FC_{\omega,f'}\right) < \Psi\left(\overline{FC_f}\right), \\ &\qquad \Psi^{+}\left(FC_{\omega,f'}\right) < \Psi\left(FC_{\omega,f}\right), if\ \rho_{i,\omega,f}^{+} = 0\ , a_{i,\omega}^{*} = \widetilde{a_f}, \end{aligned} \tag{15}$$

where $\widetilde{a_f}$ is the action in $FC_{\omega,f}$ closest to CS. If the maximum value that satisfies Eq. (15) is $\rho_{i,\omega}^{+} = 0$, then a transfer is not made and we add $FC_{\omega,f'}$ to the close set.

Step 3: Add the Cluster from the Far Set to the Close Set

If no transfer is performed, then $FC_{\omega,f'}$ is moved from FS to CS. We let $\widetilde{FC_\omega} = \left(\widetilde{FC_{\omega,1}}, \dots, \widetilde{FC_{\omega,F}}\right)$ indicate the clusters in the order they are added to CS. The costs of the clusters in $\widetilde{FC_\omega}$ are denoted as $\widetilde{\Psi_\omega} = \left(\widetilde{\Psi_{\omega,1}}, .., \widetilde{\Psi_{\omega,F}}\right)$. The objective in iterating between steps 2 and 3 is to minimize $\widetilde{\Psi_\omega}$:

$$\begin{aligned} &\underset{\rho_{i,\omega,f},\forall i\in[1,I]}{\text{minimize}}\ \widetilde{\Psi_\omega}, \\ &\text{where } \left(\widetilde{\Psi'_{\omega,1}}, .., \widetilde{\Psi'_{\omega,F}}\right) < \left(\widetilde{\Psi_{\omega,1}}, .., \widetilde{\Psi_{\omega,F}}\right)\ if\ \exists f \in [1,F] \\ &\qquad s.t. \widetilde{\Psi'_{\omega,f}} < \widetilde{\Psi_{\omega,f}}, \widetilde{\Psi'_{\omega,f'}} = \Psi_{\omega,f'} \forall f' \in [1, f-1]. \end{aligned} \tag{16}$$

Namely, an improvement is only made if, on the next iteration of the maximization step, the cluster with reduced duration will be considered for an action transfer before the cluster with increased duration. Note that the sequence $\widetilde{FC_\omega}$ only changes if a cluster transfers all ownership over its action $\widetilde{a_f}$ closest to CS. Therefore, the constraints in Eq. (15) of step 2 ensure all transfers result in a decrease of $\widetilde{\Psi_\omega}$. Over several iterations of the EM, this approach will decrease the highest cluster cost while minimizing the increased cost incurred by other clusters.

The attempt of transferring ownership is repeated until a transfer is made or the far set is empty. In the former case, we update FC_ω accordingly and repeat the expectation step, as explained above in Section IV.E.1.b. In the latter case, we assign robots to clusters, as discussed in Section IV.E.1.d. If the robots are already assigned, we add the highest cost cluster to the proposed team plan, detailed below in Section IV.E.1.e.

d. Assign Robots

After the EM steps are completed, robots are assigned to the clusters by solving the linear bottleneck assignment problem (LBAP) [34]. The objective of the LBAP is to determine the optimal complete bipartite matching (CBM) which minimizes the maximum cluster cost, Eq. (11). A CBM is a matching which assigns exactly one robot to each cluster.

To find the optimal CBM we consider a subset $\mathbb{S}_\omega(W^{Max})$ of all robot-cluster pairs, $\mathbb{R} \times FC_\omega$, with a cost less than W^{Max}. The cost of a robot-cluster pair for robot $\mathbb{R}_b$ and cluster $FC_{\omega,f}$ is determined by solving TSP_2 for $FC_{\omega,f}$ with $\mathbb{R}_{\omega,f} = \mathbb{R}_b$. If $\mathbb{S}_\omega(W^{Max})$ contains a CBM, a feasible plan can be formed from the pairs in $\mathbb{S}_\omega(W^{Max})$ as each robot can be uniquely assigned to each cluster. To determine if $\mathbb{S}_\omega(W^{Max})$ contains a CBM, a maximum bipartite matching (MBM) is solved using the Hopcroft-Karp algorithm [35] to determine the maximum number of robots that can be uniquely assigned to a cluster.

To determine the optimal cost for W^{Max}, we initialize W^{Max} to the median cost of all robot-cluster pairs and perform a binary search where the allowable costs for W^{Max} are the costs of any of the robot-cluster pairs. We select the minimum W^{Max} with subset $\mathbb{S}_\omega(W^{Max})$ that can form a CBM. The robot-cluster pair with cost equal to the minimum W^{Max} is the robot-cluster pair that minimizes the maximum cluster cost. Therefore, the robot-cluster pair is the optimal assignment. This robot-cluster assignment is selected and held constant, and the LBAP is solved with the remaining robots and clusters to minimize the remaining cluster costs.

Once all robots are assigned to a unique cluster, the EM steps are repeated to farther minimize the maximum cluster cost. The highest cost cluster is then added to the team plan.

e. Add the Highest Cost Cluster to the Proposed Team Plan

The highest cost cluster, $FC_{\omega,f} = \overline{FC_f}$, is added to the team plan by assigning the cluster's actions to robot search plan $SP_\omega^{(b)}$, where $\mathbb{R}_b = \mathbb{R}_{\omega,f}$. Namely, we solve TSP_2 using $FC_{\omega,f}$

to generate cluster plan $CP_{\omega,f}$ and then set $SP_{\omega}^{(b)} = CP_{\omega,f}$.

To assign the remaining robot plans, we remove $\mathbb{R}_b$ from the list of robots and subtract the region search times in $SP_{\omega}^{(b)}$ from the region search times in UA_{ω}, then continue the allocation from the expectation step in Fig. 4 with robots assigned.

Once B search plans are added to the proposed team plan it is complete and we check if it is the best team plan.

f. Check Proposed Team Plan is the Best Team Plan

If planning time remains after generating a proposed TP_{ω} for each time period T_{ω} sequentially, the search action allocation process is repeated to propose an alternative team plan. If no planning time remains, TP is generated by selecting the proposed TP_{ω} in each time period with minimum time for the longest robot search plan, as per Eq. (10). TP is the output of the *Action Allocation to Robots* module in Fig. 2.

F. Iterate if Infeasible Team Plan

As the team search action selection for UA occurs separately from the action allocation which generates TP, the feasibility of TP must be verified. For TP to be feasible, it must satisfy the constraint in Eq. (3). Namely, the plan for each robot in each time period, $SP_{\omega}^{(b)}$, must be completed within the allotted time t^{period}. If TP is not feasible, both the team search action selection and action allocation to robots, Fig. 2, are repeated with the estimated travel time t^{move} increased by t^{unit}.

At each iteration, TP is modified by truncating all actions that cannot be executed within the allotted time t^{period}. For example, if $t^{period} = 900s$ and the plan for $\mathbb{R}_1$ takes $990s$, with the last action taking $180s$, then this action would be truncated to $90s$. Of all the truncated plans, the one which minimizes the objective in Eq. (3) is executed by the robots.

G. Plan Execution

For the team plan to be executed, each robot performs the actions specified in their respective search plans. Namely, robot $\mathbb{R}_b$ starts in region $R_0^{(b)}$ and performs search plans $SP_1^{(b)}, \dots, SP_{\Omega}^{(b)}$ sequentially. When $\mathbb{R}_b$ performs search action $a_{\omega,h}^{(b)}$ it travels to $R_{\omega,h}^{(b)}$ and then spends $t_{\omega,h}^{(b)}$ to search $R_{\omega,h}^{(b)}$ during T_{ω}. If the robot arrives at $R_{\omega,h}^{(b)}$ before the start of T_{ω}, it waits for the start of T_{ω} before searching. To determine how to search within a region, each robot follows the local grid-based search planner discussed in Supplementary Material D. During the search, if a target user is found, replanning occurs to generate a new team plan for finding the remaining users.

H. Replanning

The replanning repeats the planning approach in Fig. 2, with the following modifications. First, to account for all actions executed by the robots prior to replanning, the reward for the edges in the min-flow graphs in Eq. (9) are updated when generating UA to account for the already completed actions:

$$W\left(E_{t(a_{i,\omega}^*)}^{i,\omega}\right) = -\sum_{z=1}^{Z} P\left(\cup_{\omega'=1}^{\omega} \theta_{z,i}^{TA_{i,\omega'}}\right) + \sum_{z=1}^{Z} P\left(\cup_{\omega'=1}^{\omega-1} \theta_{z,i}^{TA_{i,\omega'}}\right) + \sum_{z=1}^{Z} P\left(\theta_{z,i}^{EA_{i,\omega}}\right), \quad (17)$$

where $TA_{i,\omega'}$ is a set of actions that combine the newly planned action $UA_{i,\omega'}$ and the actions already executed prior to replanning $EA_{i,\omega'}$. As such, the team search times $tq(a_{i,\omega}^*)$ in $TA_{i,\omega'}$ are equal to the sum of the team search times $t(a_{i,\omega}^*)$ in $UA_{i,\omega'}$ and $q(a_{i,\omega}^*)$ in $EA_{i,\omega'}$, i.e., $tq(a_{i,\omega}^*) = t(a_{i,\omega}^*) + q(a_{i,\omega}^*)$. The time constraint in Eq. (3) is also updated to account for the time already expended in each time period:

$$t\left(SP_{\omega}^{(b)}\right) \leq t^{period} - Q_{\omega}, \forall SP_{\omega}^{(b)} \in TP, \quad (18)$$

where Q_{ω} is the time spent during T_{ω} prior to replanning.

The time and space complexity for the proposed team action selection and action allocation approaches, and the overall 2-MRPSS method are provided in Supplementary Material E.

V. Implementation Scenarios

We have tested our 2-MRPSS in various multi-robot search scenarios. One application of interest is deploying multiple robots in long-term care homes to find multiple dynamic elderly residents in order for the robots to engage in social human-robot interaction with them. The increasingly high number of long-term care residents relative to care staff can result in staff burnout [36]. Therefore, robots can assist staff by providing residents with reminders of upcoming activities [20], teleconferencing with family and friends [21], and facilitation of recreational activities [37]. To provide such assistance, the robots must first find the residents within the environments.

Our 2-MRPSS approach for finding residents in long-term care considers the locations in which the residents perform various activities of daily living including eating, meeting with family and friends, watching television, reading, and taking a nap. Some activities are performed in a resident's own private room such as taking a nap. The private rooms are not accessed by other residents. Other activities are performed in shared rooms such as eating in the dining hall, watching television in the recreational room, reading in the garden, and meeting with family and friends in the lobby. The shared rooms are accessed by all residents. Some rooms are off-limits to the residents such as the kitchen and robot charging station while some are off-limits to the robots such as the nurses' station.

Environment Layouts: The environment layouts considered are informed by our long-term care partners. They consist of 26 private rooms, one for each resident, and several shared rooms. The number of shared rooms ranges from 4 to 16. The total number of all types of rooms is 30, 33, 36, 39, and 42 rooms. The rooms are represented by regions and divided into cells, with the largest regions containing 20 cells, resulting in $t_{max} = 440s$, as $t^{cell} = 22s$ is required for a robot to search each cell. t^{unit} was set equal to t^{cell}, and t^{move} was initialized to 0. An example environment layout is in Fig. 6, with shared room and private room configurations as in Fig. 7.

Target Users: A subset of residents was selected to be found by all robots. During each search, the dynamic residents visited various regions, sometimes visiting a region more than once. The number of target users was 1, 5, 10, 15, and 20 residents.

Time Frame: The searches occurred between 10:00am and 6:00pm, with a search duration of 15, 30, 45, 60, or 75 minutes.

Time Period: Each search used $\Omega = 3$ time periods.
Robot Team: Multiple robots executed each team plan. The robots moved at a speed of 0.8m/s. When searching a cell, each robot was able to identify any resident in the cell. The number of robots used were 1, 3, 5, 7, and 9 robots.
Planning Duration: Each search used $t^{plan} = 10$ms planning duration.
Dataset: Each search used *Y*=30 days of simulated data for the location models. The data set can be found on our website.
Search queries: All combinations of the above parameter values were considered to test our MRPSS: *environment size* = {30, 33, 36, 39, 42} shared rooms, *search duration* = {15, 30, 45, 60, 75} minutes, *number of target users* = {1, 5, 10, 15, 20}, and *number of robots*= {1, 3, 5, 7, 9}. Each combination was repeated for a *search start time* = {10:00, 12:00, 14:00, 16:00, 18:00} on a 24-hour clock. This resulted in 3,125 trials.

A representative video of our 2-MRPSS method using a 3-member robot team to search a 30 room environment for a 15 minute duration to find 5 target users is presented in here.

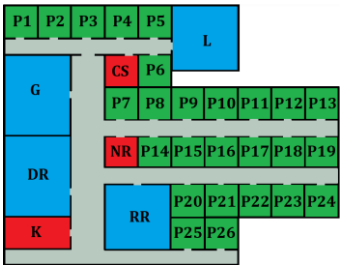
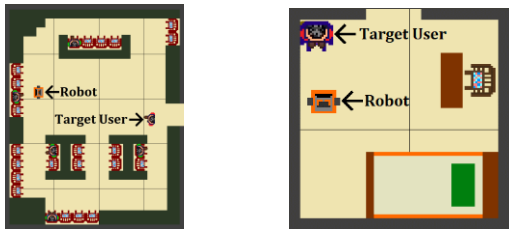


Fig. 6. Environment layout with 30 searchable rooms. Shared rooms (blue) include: lobby (L) and recreational room (RR) - 8m x 8m; and dining room (DR) and garden (G) - 8m x 10m. Private rooms (green) are represented by P - 4m x 4m. Unsearchable rooms (red) include: kitchen (K) - 4m x 8m; and charging station (CS) and nurses' station (NR) - 4m x 4m.



(a) (b)

Fig. 7. Example layout of rooms: (a) shared garden, and (b) private room.

VI. Search Experiments

We conducted simulated experiments for our long-term care implementation problem to investigate the performance of our proposed method with respect to: 1) the team plan duration of our action allocation to robots, and 2) the number of target users found using our overall 2-MRPSS.

A. Experiment #1: Team Plan Duration Time

The purpose of the *Action Allocation to Robots* module discussed in Section IV.E is to solve the min-max mRSP in order to allocate a set of unallocated actions (UA) such that the overall team plan duration is minimized. To generate UA, as discussed in Section IV.D, we must select parameter t^{move} to estimate the robot travel time between regions by considering the impact of t^{move} on UA. Namely, we note that a larger t^{move} results in fewer actions in UA and that it is easier to optimally allocate a small number of actions due to the small number of permutations. As the number of actions increases, the number of permutations, and thus complexity of the allocation problem, also increases. Therefore, we have selected $t^{move} = 0$, as the resulting UA from solving the CMPMKP will have a large number of actions. This allows us to test the capabilities of our clustering-based action allocator in a hard, high complexity instance of the allocation problem.

Note that the solution to the min-max mRSP is a team plan for a single time period, and the overall team plan is generated by sequentially solving the min-max mRSP in each time period. Thus, we consider the amount of time it takes the team to perform the plan in each time period, $t(TP_\omega)$, determined by the longest robot search plan in that time period:

$$t(TP_\omega) = \max_{b\in[1,B]} t(SP_\omega^b). \quad (19)$$

The mean team plan duration for a single search is the average of $t(TP_\omega), \forall \omega \in [1, \Omega]$. To avoid biasing the results towards scenarios with longer search durations, which will have longer team plan durations, we introduce the mean maximum search time (MMST). The MMST considers the average $t(TP_\omega)$ as a percentage of the search time in a time period:

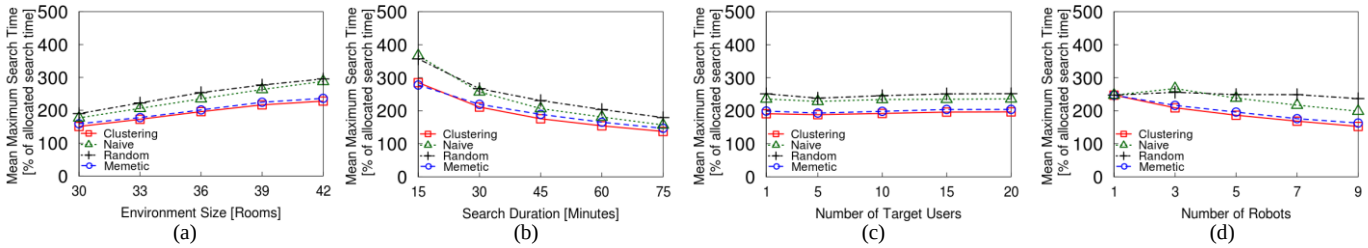
$$\text{MMST} = \frac{\sum_{\omega=1}^{\Omega} t(TP_\omega)}{\Omega t^{period}}. \quad (20)$$

As we used $t^{move} = 0$, an expected underestimate of the travel time, all values of MMST will be above 100%.

Herein, we determine and compare the MMST of our clustering method for action allocation against three potential alternatives methods: 1) naïve [21], 2) random, and 3) memetic [38]. The details of these alternative methods are presented in the Supplementary Material F. We also provided a validation study in the Supplementary Material G for the selected planning duration of $t^{plan} = 10$ms to demonstrate how all four allocation methods gain no additional performance benefits from a planning duration above 10ms.

1) Results

The MMST across various environment sizes, search durations, number of target users, and number of robots for our comparison are presented in Fig. 8. Each point represents



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

Fig. 8. Mean maximum search time across (a) environment size, (b) search duration, (c) number of target user, and (d) number of robots.

the mean of 625 trials. Results show that our clustering action allocator outperformed the alternative allocators across all scenarios with the exceptions of a 15 minute search duration, where it was comparable with the memetic method, and a single robot search, where all methods performed the same.

For all combination of scenarios, 3,125 trials, the MMST was 193% for our clustering method, 200% for the memetic, 234% for the naive, and 248% for the random methods. Namely, the team plans generated by our clustering approach were at least 7% faster than the alternatives. In general, our clustering approach had a statistically lower mean maximum search time than the alternatives as it: 1) considered all robot locations simultaneously, and 2) had multiple robots cooperate in searching a single region. As the naïve and random methods considered each robot sequentially, they had inefficient travel times. Although the memetic method did not have travel time inefficiencies, the inability to assign multiple robots to a single region resulted in an unbalanced workload. For example, in one scenario, the memetic approach assigned all the cells in both RR and DR to a single robot resulting in a long team plan duration of 1,803s. For this scenario, our clustering approach assigned another robot to assist in searching RR, resulting in a shorter overall team plan duration of 1,618s.

A non-parametric Kruskal-Wallis test (α=0.05) conducted for all 3,125 trials found that there was a statistically significant difference in the MMST between the four action allocators, $\chi^2(3) = 921, p < 0.0001$. A post-hoc Dunn's test (α=0.05) with a Bonferroni correction (α=0.0167) showed a statistically significant difference between the clustering and the alternatives: naïve allocators ($Z(3125) = 17.7, p < 0.0001$), the random allocators ($Z(3125) = 27.2, p < 0.0001$), and the memetic ($Z(3125) = 4.70, p < 0.0001$).

B. *Experiment #2: Number of Users Found*

We investigated the mean success rate of finding target users using our proposed aware 2-MRPSS with strong coordination to solve the multi-robot search problem within a deadline. To the authors' knowledge there are currently no existing approaches for solving this specific multi-robot search problem. However, we were able to adapt both the segmented (unaware with no coordination) [14] and sequential (aware with weak coordination) [22] planners from the literature by combining them with our two-stage approach of first selecting team search actions and then generating robot plans. The segmented planner assigns a unique area for each robot to search, while the sequential planner generates a plan sequentially for each robot. We compared our 2-MRPSS approach to these methods in order to investigate the effect that strong coordination has on maximizing the number of dynamic users found within a deadline. The development details of both these planners are in Supplementary Material H.

1) *Results*

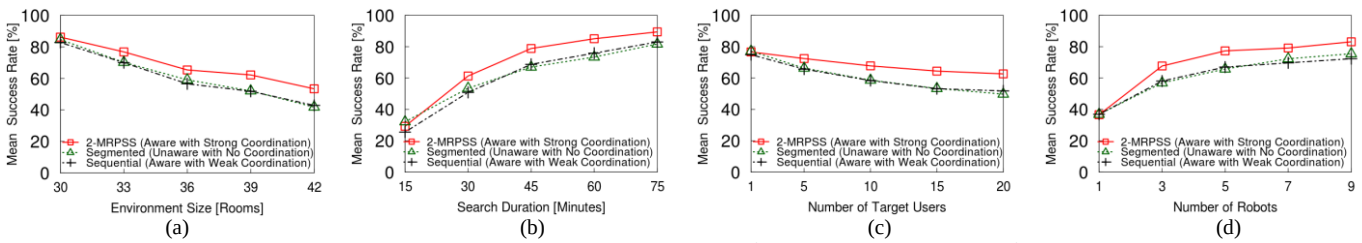
The mean success rates for all three methods across various environment sizes, search durations, number of target users, and number of robots are presented in Fig. 9. Each point represents the mean of 625 trials. The results show that our 2-MRPSS approach outperformed the alternatives with the exceptions of a 15 minute search duration, one target user, and one robot. For these latter scenarios, the plans were all similar as the majority of cases had only one action per robot or all actions were assigned to a single robot.

The overall mean success rates across all scenarios (3,125 trials) in Fig. 9 is 61% for both the segmented and sequential approaches, and 72% for our 2-MRPSS approach. Namely, our approach searched several high probability locations within the search duration, while the segmented approach generally had at least one robot assigned to a search area with low user probability and the sequential approach required robots planning later in the process to accommodate existing plans. For example, in a scenario with 3 robots searching for 15 users, our 2-MRPSS approach assigned the team to search 4 shared rooms as these had the highest user probabilities, resulting in 13 users being found. However, in the segmented approach, 2 robots searched 2 shared rooms where they found 8 users, and the third robot searched 4 private rooms where no users were found. In the sequential approach, 2 robots searched 2 shared rooms and a portion of another shared room. Therefore, the third robot searched the remainder of the shared room as well as 2 nearby private rooms in order to have enough time to travel between rooms. 10 users were found in the shared rooms and none in the private rooms.

A Kruskal-Wallis test ($\alpha = 0.05$) showed a statistically significant difference between the overall means, $\chi^2(3) = 118, p < 0.0001$. A post-hoc Dunn's test (α=0.05) with a Bonferroni correction (α=0.025) determined statistically significant differences between our 2-MRPSS approach and the 1) segmented method, $Z(3125) = 9.07, p < 0.0001$, and 2) sequential method, $Z(3125) = 9.69, p < 0.0001$.

VII. Conclusions

In this paper, we present a novel multi-robot person search system for finding multiple dynamic users before a deadline in human-centered environments considering user data. Our aware 2-MRPSS approach with strong coordination simultaneously considers all robot locations as well as user location probability density functions when generating a team plan. This allows our approach to minimize the team plan duration such that it can search high probability user locations



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

Fig. 9. Mean success rate across (a) environment size, (b) search duration, (c) number of target user, and (d) number of robots.

within the deadline. Comparison experiments validate that our approach has a lower mean maximum search time and is able to find more targets before a deadline when compared to other alternative methods. Our future work will consist of 1) incorporating online evidence, similar to [19] within the user location model from direct robot observations as they search the environment, and 2) integrating our 2-MRPSS method within a multi-robot control architecture with complementary perception and human-robot interaction modules for deploying a team of physical robots in partner long-term care centers.

REFERENCES

[1] B. Doroodgar, Y. Liu, and G. Nejat, "A Learning-Based Semi-Autonomous Controller for Robotic Exploration of Unknown Disaster Scenes While Searching for Victims," IEEE Trans. Cybern., vol. 44, no. 12, pp. 2719–2732, 2014.

[2] Ł. Białek, J. Szklarski, M. Borkowska, and M. Gnatowski, "Reasoning with four-valued logic in multi-robotic search-and-rescue problem," Challenges in Autom. Robot. and Meas. Techn., vol. 440, no. 1, pp. 483–499, 2016.

[3] Y. Liu and G. Nejat, "Multirobot Cooperative Learning for Semiautonomous Control in Urban Search and Rescue Applications," J. Field Robot., vol. 33, no. 4, pp. 512–536, 2016.

[4] W. Zhao, Q. Meng, and P. Chung, "A Heuristic Distributed Task Allocation Method for Multivehicle Multitask Problems and Its Application to Search and Rescue Scenario," IEEE Trans. Cybern., vol. 46, no. 4, pp. 902–915, 2016.

[5] J. Turner et al., "Distributed Task Rescheduling With Time Constraints for the Optimization of Total Task Allocations in a Multirobot System," IEEE Trans. Cybern., vol. 48, no. 9, pp. 2583–2597, 2018.

[6] S. Chen et al., "Decentralized Patrolling Under Constraints in Dynamic Environments," IEEE Trans. Cybern., vol. 46, no. 12, pp. 3364–3376, 2016.

[7] D. Yanguas-Rojas, G.A. Cardona, J. Ramirez-Rugeles, E. Mojica-Nava, "Victims search, identification, & evacuation with heterogeneous robot networks for search and rescue," IEEE Conf. on Auto. Control, pp. 1–6, 2017.

[8] A. Fung, L. Y. Wang, K. Zhang, G. Nejat, and B. Benhabib, "Using Deep Learning to Find Victims in Unknown Cluttered Urban Search and Rescue Environments," *Curr. Robot. Rep.*, vol. 1, no. 3, pp. 105–115, Sep. 2020.

[9] H. Xiao, R. Cui, and D. Xu, "A Sampling-Based Bayesian Approach for Cooperative Multiagent Online Search with Resource Constraints," IEEE Trans. Cybern., vol. 48, no. 6, pp. 1773–1785, 2018.

[10] L. Lin and M. Goodrich, "Hierarchical Heuristic Search Using a Gaussian Mixture Model for UAV Coverage Planning," IEEE Trans. Cybern., vol. 44, no. 12, pp. 2532–2544, 2014.

[12] Z. Kashino, G. Nejat, and B. Benhabib, "Multi-UAV based Autonomous Wilderness Search and Rescue using Target Iso-Probability Curves," Int. Conf. Unmanned Aircr. Systems, pp. 636–643, 2019.

[12] A. Macwan, J. Vilela, G. Nejat, and B. Benhabib, "A Multirobot Path-Planning Strategy for Autonomous Wilderness Search and Rescue," IEEE Trans. Cybern., vol. 45, no. 9, pp. 1784–1797, 2015.

[13] J. Durham, A. Franchi, and F. Bullo, "Distributed pursuit-evasion without mapping or global localization via local frontiers," Auton. Robots, vol. 32, no. 1, pp. 81–95, 2012.

[14] N. Basilico, T.H. Chung, and S. Carpin, "Distributed online patrolling with multi-agent teams of sentinels and searchers," Distrib. Auton. Robot. Syst., pp. 3–16, 2016.

[15] P. B. Sujit and D. Ghose, "Self assessment-based decision making for multiagent cooperative search," IEEE Trans. Autom. Sci. Eng., vol. 8, no. 4, pp. 705–719, 2011.

[16] S. Lee and Y. Kim, "Cooperative Reactive Persistent Surveillance Algorithm Using Multiple UAVs Considering Incident Arrivals," Int. Federation of Autom. Control, vol. 50, no. 1, pp. 2347–2352, 2017.

[17] P. Dames, P. Tokekar, and V. Kumar, "Detecting, localizing, and tracking an unknown number of moving targets using a team of mobile robots," Int. J. Robot. Res., vol. 36, no. 13–14, pp. 1540–1553, 2017.

[18] G. D. Tipaldi and K. O. Arras, "I want my coffee hot! Learning to find people under spatio-temporal constraints," IEEE Int. Conf. Robot. Autom., pp. 1217–1222, 2011.

[19] S. Lin and G. Nejat, "Robot Evidence Based Search for a Dynamic User in an Indoor Environment," ASME Int. Design Eng. Technical Conf. & Comp. and Inf. in Eng. Conf., pp. 1–8, 2018.

[20] M. Schwenk, T. S. Vaquero, G. Nejat, and K. O. Arras, "Schedule-based robotic search for multiple residents in a retirement home environment," AAAI Conf. Artificial Intell., pp. 2571–2577, 2014.

[21] S. C. Mohamed, S. Rajaratnam, S. T. Hong, and G. Nejat, "Person Finding: An Autonomous Robot Search Method for Finding Multiple Dynamic Users in Human-Centered Environments," IEEE Trans. Autom. Sci. Eng., vol. 17, no. 1, pp. 433–449, 2020.

[22] A. Goldhoorn, A. Garrell, R. Alquézar, and A. Sanfeliu, "Searching and tracking people with cooperative mobile robots," Auton. Robots, vol. 42, no. 4, pp. 739–759, 2018.

[23] R. Triebel et al., "SPENCER: A socially aware service robot for passenger guidance and help in busy airports," Field & Service Robot., pp. 607–622, 2016.

[24] S. F. Ochoa and R. Santos, "Human-centric wireless sensor networks to improve information availability during urban search and rescue activities," Inf. Fusion, vol. 22, no. 1, pp. 71–84, 2015.

[25] A. Farinelli, L. Iocchi, and D. Nardi, "Multirobot systems: a classification focused on coordination," IEEE Trans. Syst. Man Cybern. Part B Cybern., vol. 34, no. 5, pp. 2015–2028, 2004.

[26] B. H. Faaland, "Technical note—The multiperiod knapsack problem," Oper. Res., vol. 29, no. 3, pp. 612–616, 1981.

[27] P. M. França, M. Gendreau, G. Laporte, and F. M. Müller, "The m-Traveling Salesman Problem with Minmax Objective," Transp. Sci., vol. 29, no. 3, pp. 267–275, 1995.

[28] R. Bellman, "On a Routing Problem," Quart. Appl. Math., vol. 16, no. 1, pp. 87–90, 1958.

[29] D. E. Gustafson and W. C. Kessel, "Fuzzy clustering with a fuzzy covariance matrix," IEEE Conf. Decision and Control including the 17th Symp. Adaptive Processes, pp. 761–766, 1978.

[30] F. Dellaert, "The Expectation Maximization Algorithm," Georgia Inst. of Technol. Tech. Rep., pp. 1–7, 2002.

[31] D. Arthur and S. Vassilvitskii, "k-means++: The advantages of careful seeding," Stanford Tech. Rep., pp. 1–11, 2006.

[32] J. Kruskal, "On the shortest spanning subtree of a graph and the traveling salesman problem," Amer. Math. Soc., vol. 7, no. 1, pp. 48–50, 1956.

[33] K. Helsgaun, "An effective implementation of the Lin–Kernighan traveling salesman heuristic," Eur. J. Oper. Res., vol. 126, no. 1, pp. 106–130, 2000.

[34] R. E. Burkard and U. Derigs, "The Linear Bottleneck Assignment Problem," Assignment and Matching Problems: Solution Methods with FORTRAN-Programs, vol. 184, no. 1, pp. 16–24, 1980.

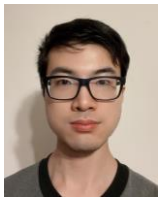
[35] J. Hopcroft and R. Karp, "An n^5/2 algorithm for maximum matchings in bipartite graphs," SIAM J. Comput., vol. 2, no. 4, pp.225–231, 1973.

[36] G. S. Rai, "Burnout Among Long-Term Care Staff", Admin. in Social Work, vol. 34, no. 3, pp. 225–240, 2010.

[37] J. Li, W. G. Louie, S. Mohamed, F. Despond, and G. Nejat, "A user-study with Tangy the bingo facilitating robot and long-term care residents," Int. Symp. on Robot., Intell. Sensors, pp. 1–7, 2016.

[38] Y. Wang et al., "Memetic algorithm based on sequential variable neighborhood descent for the minmax multiple traveling salesman problem," Comp. Ind. Eng., vol. 106, no. 1, pp. 105–122, 2017.

Sharaf C. Mohamed completed his Ph.D. in 2021 in the Department of Mechanical & Industrial Engineering (MIE) at the University of Toronto (UofT). He was a member of the Autonomous Systems and Biomechatronics Laboratory (ASBLab). His research interests include multi-robot coordination, human-robot interaction, and autonomous robotics. He received his B.A.Sc. in Electrical & Computer Engineering at UofT.



Angus Fung is a Ph.D. candidate in MIE at the University of Toronto (UofT). He is a member of the Autonomous Systems and Biomechatronics Laboratory (ASBLab). His research interests include robotics and computer vision. He received his B.A.Sc. in Engineering Science (Robotics) at UofT.

Goldie Nejat (S'03-M'06) is the Canada Research Chair in Robots for Society and a Professor in MIE at UofT. She is the Founder/Director of the ASBLab. She is also an Adjunct Scientist at both KITE in the Toronto Rehabilitation Institute, and the Rotman Institute at Baycrest Health Sciences. Her research interests include intelligent assistive/service robots, human-robot interactions, robot learning and autonomous control. She received her B.A.Sc. and Ph.D. degrees in Mechanical Engineering at UofT.