Transactions on Automation Science and Engineering Manuscript (This is the author's version of an article that has been published. Changes were made by the publisher prior to publication. (https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=8790973)

Person Finding: An Autonomous Robot Search Method for Finding Multiple Dynamic Users in Human-Centered Environments

Sharaf C. Mohamed, Sanjif Rajaratnam, Seung Tae Hong, and Goldie Nejat, *Member, IEEE*

***Abstract*— Robot search for multiple dynamic users within a multi-room environment is important for social robots to find and engage in various human-robot interaction scenarios with these users. In this paper, we present a novel autonomous person search technique for a robot finding a group of dynamic users before a deadline. The uniqueness of our approach is that unlike existing robot search methods, we consider activity information to predict where, when, and for how long a user will be in a specific room. This allows for the generation of search plans without any assumption on the frequency of user movements. We represent our search problem as an extension of the orienteering problem, which we define herein as the robot person search orienteering problem (PSOP). User activity information is represented as spatial-temporal user activity probability density functions (APDFs). We solve the PSOP using APDFs to generate a search plan to maximize the expected number of users found before the deadline. The solution of the PSOP is obtained in two steps. First, by solving a variant of the multiperiod knapsack problem to determine which rooms should be searched and for how long these rooms should be searched. Then we solve the traveling salesman problem to obtain the order in which to search these rooms. Experiments were conducted to validate the performance of our robot search method in finding different numbers of multiple dynamic users for varying environment sizes and search durations. We also compared our method with two coverage planners and a Markov decision process planner. On average, our planner found more users than the other planners for a variety of scenarios. Lastly, we performed experiments that introduced uncertainty into both the APDFs as well as during the search to validate the robustness of our overall approach.**

***Note to Practitioners*— The majority of current social robot applications either consider users being collocated with the robot in the same region or users being static within another region in the environment. However, several applications exist where users are dynamic within their environments and for which a robot needs to find them in order to provide assistance, for example, in office buildings, airports, museums, hospitals, and long-term care facilities. In general, these users are performing activities within these regions. We uniquely consider such activity information in order to model user location probabilities. We developed a robot search planner that uses these probabilities to find users of interest in multi-room environments. The planner is novel as it reasons about when and which regions to search and for how long, as well as if the same region needs to be searched multiple times as users can perform multiple activities during the search time frame in the same region or revisit a region to perform a new activity. We have integrated the search planner within a robot system architecture. The robot travels to each region and then uses a local planner to navigate to locations within the region. At each location, a person identification technique is used to identify the target users in order to engage in human-robot interactions. Experiments were performed for two search applications: 1) a simulated Blueberry robot finding multiple residents in a virtual representation of one of our collaborating long-term care facilities, and 2) the physical Blueberry robot finding multiple staff/students on a physical floor of a university building. For both experiments, plans were generated on the robot's onboard Lenovo Thinkpad X230 using the Robot Operating System (ROS) in Ubuntu. User activity data and maps used for the experiments in the care facility can be found on our website, here, under multi-user robot search. The physical Blueberry robot was also equipped with an ASUS Xtion IR depth camera, a Logitech pro c920 RGB camera, and a Hokuyo laser range finder for person identification and navigation in the environment. The results showed that our system was effective at finding multiple dynamic users under varying environment sizes and search durations. Our search planner also outperformed other planners and was robust to uncertainties in the user model. Future work will consider environments with multiple floors and crowded regions, planners which directly reason about environment dynamics, and local planners which reason about user location probabilities within regions.**

***Index Terms*— Social robots, search plans, multiple dynamic users, human-robot interaction, orienteering problem.**

I. Introduction

ROBOTS can be used to search for people in many different environments, such as inside buildings [1]–[18], and outdoor urban [19]–[23] and natural [24]–[26] settings. Applications in these environments range from search and rescue [1]–[5],[24]–[26], surveillance and monitoring [6]–[10],[19]–[23], to assisting users with daily tasks [11]–[18].

Social robots, in particular, have been developed to aid users through interactions in a variety of human-centered environments. For example, robots have been used as guides in museums [27] and airports [28]. Furthermore, robots have assisted guests in hotels with services such as item delivery [29]. In private homes, social robots have helped with meal

This work was supported by the Natural Sciences and Engineering Research Council of Canada (NSERC), Dr. Robot Inc., the Ontario Centres of Excellence (OCE), the Canadian Consortium on Neurodegeneration in Aging (CCNA), 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, sanjif.rajaratnam, rio.hong}@mail.utoronto.ca; nejat@mie.utoronto.ca).

preparation [12] and provided teleconferencing [13].

Social robots have also been used in various healthcare environments, including long-term care facilities and hospitals. In long-term care, they have provided assistance to older residents by guiding them to appointments [30], providing prompting assistance with eating [31], and giving reminders of upcoming activities [18]. They have also assisted with recreational activities, such as facilitating interactive reading groups [32], Bingo [33], and Trivia [34]. In hospital settings, social robots have guided patients and visitors [35].

The applications mentioned above can benefit from the robots being able to effectively search for a single person or multiple people of interest in the environment, in order to provide the necessary assistance via human-robot interaction (HRI). However, currently the majority of existing social robot applications assume that the users are initially collocated with the robots [27]–[35]. If they are not collocated, the robots are mainly searching for a single static user [11]–[13],[15].

A handful of methods have been developed to consider dynamic users [14],[16]–[18], using spatial-temporal probability density functions (PDFs). For example, in [14], a hidden Markov model was used to model a user's location. In [16], a user location PDF was modeled using a Bayesian network considering evidence observed in the environment during the search. In [17], a user location PDF was modeled using the frequency of visitation of the user to regions in the environment. In [18], user location PDFs were modeled using a combination of location frequency patterns, last known locations, and spatial-temporal information from user schedules. To the authors' knowledge, a person search technique has not yet considered both spatial-temporal user location probabilities and user activity probability information.

In this paper, we present a novel multi-person search planner to allow a robot to find, in real-time, multiple dynamic users in a multi-room environment. We uniquely model our problem as an extension of the orienteering problem (OP) [36], which we define herein as the person search orienteering problem (PSOP). Our approach is an extension of our unpublished workshop paper [37] which first introduced the concept of modeling this search problem as a combinatorial optimization problem. However, in the workshop paper, only a spatial-temporal user location PDF was used with the planner. The proposed PSOP extends this initial problem definition by introducing unique user activity probability density functions (APDFs), which model the dynamic users' behaviors considering activity regions, start times, and durations, to determine the probability of users remaining in a specific region or revisiting a region to perform an activity. As a result, the PSOP can be used to model users with varying levels of dynamic behaviors without assumptions on the frequency at which users move between regions.

Our novel multi-person search planner (MPSP) solves the PSOP in real-time to provide search plans which maximize the expected number of dynamic users found before a deadline. This planner uses a two-stage approach: 1) a variant of the multi-period knapsack problem [38] is first solved to determine which regions to search and the duration to search each region, and 2) the traveling salesman problem (TSP) [39] is solved to determine the order in which to search the regions.

II. Related Works

Herein, we discuss existing person search methods deployed by robots, and introduce orienteering problems which can be used to model our new person search problem.

A. Person Search by Robots

Existing person search approaches for robots can be categorized based on their specific application: 1) search and rescue, 2) surveillance and monitoring, and 3) task assistance.

The objective of time-critical search and rescue missions is to minimize the time to find victims. For example, in urban scenarios involving static victims and no prior location information, robots have used exploration techniques to maximize coverage in unknown environments [1]–[3]. These techniques include a team of robots executing graph traversal algorithms [1] and frontier exploration methods [2],[3]. When prior information was available, static victim discovery time was minimized using either dynamic programing [4], or a multi-agent partially observable Markov decision process (POMDP) solved with a discounted d-step lookahead planner [5]. In wilderness environments, a lost person's location PDF has been used, based on the victim's last known location and motion model [24]–[26]. In these searches, robots iteratively move to and search the location with the highest user location probability according to their location PDF.

In surveillance and monitoring problems, robots are used to protect an environment from malicious attackers. Given no prior attacker location information, teams of robots have performed searches which minimize the time to fully cover known environments [6],[19]–[21]. With prior information, lookahead techniques have been implemented to minimize uncertainty within the environment [22],[23]. Problems which require robots to both clear regions and block access from uncleared to cleared regions have been addressed using recursive blocking in a tree topology [7]; frontier exploration [8]; mixed integer programming [9]; and breath-first search [10].

In task assistance problems, robots search for users within indoor structured environments to assist them with specific tasks. The environments are generally divided into regions with each user being assigned a probability density for each region [11]–[18]. One-step lookahead techniques have been used to search the highest probability location when searching for static [11]–[13] or dynamic [14] users. A POMDP lookahead planner to search for a single dynamic user was also used [16]. Markov decision process (MDP) planners have also been designed for static [15] or dynamic [17],[18] users.

1) Summary and Challenges

The aforementioned person search methods can all be classified into the following categories: 1) full environment coverage planners [1],[2],[6]–[10],[19]–[21], 2) lookahead planners [3],[5],[11]–[14],[16],[22]–[26], and 3) MDP-based planners [4],[15],[17],[18].

When considering our robot search problem of looking for multiple dynamic users in a multi-room environment before a deadline, the aforementioned approaches have several limitations. In particular, full environment coverage techniques apply equal importance to searching each location within the environment as they do not use any prior user location information. Lookahead planners, on the other hand, use prior user location information to choose optimal search actions in the near future, however, they do not consider the entire search duration. This can generate plans in which search actions are performed that result in sub-optimal future actions.

MDP-based planners can generate optimal search plans while considering the full search duration, however, due to the Markovian assumption, the plans consider the probability of finding a user as a function of only the current state. Therefore, the history of when previous regions were searched is not available. As a result, a robot may end up searching a region, and then without considering the most likely time at which a user would have revisited the region, search the same region again.

B. Orienteering Problems (OPs)

The original orienteering problem considers a game where a player moves within a known environment to travel between pre-specified locations known as control points. A reward is available at each control point and travel time is known between each pair of control points [36]. Given a total amount of time that can be spent by a single player, the objective is to select an ordered subset of control points to be visited such that the total reward is maximized. OPs have recently been applied to robot task planning [40], robot surveillance [41], and robot building clearing [42] problems.

OPs that have extended the original problem can be categorized into problems that consider: 1) time-dependent rewards, 2) history-dependent rewards, 3) multiple visits to a point, or 4) reward acquisition time.

1) Orienteering Problems with Time-Dependent Rewards

OPs with time-dependent rewards can be categorized into ones that have: 1) rewards that change linearly over time [43],[44], 2) rewards that are discontinuous over time [45], or 3) rewards that are available only within a specific time window [46]–[49]. Although the rewards in these problems change over time, the rewards collected at each control point is independent of previously acquired rewards.

2) Orienteering Problems with History-Dependent Rewards

OPs with history-dependent rewards consider how the reward at a control point changes depending on previously visited control points. History-dependencies can be categorized as follows: 1) constraint on the minimum or maximum number of control points of a certain type (e.g., visiting at least one art exhibit in a museum, visiting at most two exhibits) [47],[48], 2) incompatibility between control points (e.g., visiting only one art exhibit) [50], or 3) non-binary history-dependence (e.g., visiting a second art exhibit having half the reward of the first exhibit) [51]. Although the above problems consider history, none allow for a control point to be visited more than once.

3) Orienteering Problems with Multiple Visits to a Point

A handful of problems have allowed for multiple visits to a control point [41],[49],[52]. For example, in [41] each control point had a time-dependent reward, and could be searched repeatedly. However, the rewards were not history-dependent. In [52], multiple vehicles visited blood banks to acquire donated blood, available within six hours of being donated. For a repeated visit to a location, any reward acquired within the past six hours could not be acquired again. In [49], a tourist trip-planner generated a plan with multiple visits by computing the reward as a function of the entire plan and starting time, as opposed to assigning a reward to each individual action in the plan. Although the techniques presented in both [49] and [52] considered the rewards for multiple visits to a control point, neither of these techniques reasoned about situations in which the reward received at a control point varies with the amount of time spent at that control point.

4) Orienteering Problems with Reward Acquisition Time

The majority of problems have assumed that time is only spent traveling between control points, however some problems also considered time spent at the control points to acquire rewards. Reward acquisition time can be categorized as: 1) a constant acquisition time with a constant reward [47]–[49],[53], or 2) a variable acquisition time with a proportional reward [45],[54]. However, in both [45] and [54], the OP did not consider rewards that are dependent on the time of day and the control points could only be visited once.

5) Summary and Challenges

To-date, none of the aforementioned techniques have simultaneously considered all four of the OP extensions discussed above. However, to appropriately represent the person search problem for task assistance, it is necessary that the PSOP incorporates all these extensions. In the PSOP, control points represent searchable regions and rewards represent the expected number of target users found while searching a region. As the users are dynamic, a user may enter a region subsequent to the robot's search, and therefore, the robot may be required to search a region multiple times. Also due to dynamic user behavior, the probability of finding a user in a region changes with the time of day and is dependent on when the region was last searched. Furthermore, the users may have preferred locations within the regions, and as a result it may be desirable to only search portions of the region. Therefore, in our work we uniquely incorporate the combination of non-binary history-dependent rewards available within specific time windows, with variable reward acquisition times and multiple visits to a region.

III. Robot Person Search Problem (PSOP)

Our PSOP requires a mobile social robot to look for multiple dynamic users in a multi-room environment. Each search must be completed before a specified deadline. Upon finding a user, the robot initiates an assistive interaction with the user. In comparison with the original OP, our PSOP replaces: 1) control points with regions, 2) travel time between control points with

both travel time between regions and search time at the destination region, and 3) control point rewards with probabilities of finding users in the region. A complete list of symbols used herein is provided as supplementary material.

Environment: The search environment is composed of regions R (R_1, R_2, …, R_I). Physical boundaries such as walls define each region. Regions are designated as neighbors if they are physically accessible from one another, i.e., contain a mutual doorway. The time it takes the robot to traverse the shortest path between R_i and $R_{i'}$ is denoted as $t_i^{i'}$. Robot paths in the environment are obtained by successively moving to neighboring regions. Regions are categorized by the type of activities that can be performed in the region, e.g., eating occurs in the dining room.

Activities: An activity A_m denotes a task undertaken by a user. Examples of activity categories are: eating, meetings, watching television, reading, and taking a nap.

Users: The users in the shared environment are denoted as U (U_1, U_2, …, U_N). Throughout the day, users perform activities in regions for varying durations of time.

Social Robot: The social robot searches for target users by moving in the environment. It can navigate at a max speed of v m/s that is based on the average speed of the users who regularly occupy the environment. The robot can execute the following actions: *search*, *wait*, and *interact*. The *search* action requires the robot to move to a region and search locally within the region. The *wait* action is performed when the robot needs to stop, e.g., to allow a person to pass to avoid a collision or to prevent starting a search action when ahead of schedule. The *interact* action is performed once a target user is found.

Search query: The search query S identifies a set of target users U' (U'_1, U'_2, …, U'_Z) to be found by the robot between a time frame, t^{start} to t^{end}. Each time frame is composed of several discrete time periods T (T_1, T_2, …, T_Ω) of length t^{period}. A contiguous subset of T, $T_{j,k} = \{T_j, \dots, T_k\}$, is referred to as a time window. For example, for a time frame from $t^{start} = 10{:}00$ to $t^{end} = 10{:}15$, with $t^{period} = 5$ minutes, $T_1 = 10{:}00 - 10{:}05$, $T_2 = 10{:}05 - 10{:}10$, and $T_{1,2} = 10{:}00 - 10{:}10$.

Search Plan: A search plan SP is a sequence of *search* actions to be executed by the social robot in response to the search query S:

$$SP = \{a_1, \dots, a_H\}, \tag{1a}$$

$$a_h = a_{i(h),\omega(h)}, \tag{1b}$$

$$t(h) = t\left(a_{i(h),\omega(h)}\right) \in \{0, t^{unit}, 2t^{unit}, \dots, t^{max}\}, \tag{1c}$$

where $a_{i(h),\omega(h)}$, denotes the h^{th} *search* action of SP which is a search of region $R_{i(h)}$ during time period $T_{\omega(h)}$ for a duration of $t\left(a_{i(h),\omega(h)}\right)$. The possible search times within a region are discretized into increments of t^{unit} seconds, with a maximum of t^{max} seconds.

IV. User Location Model

The user location PDFs are used to generate the rewards needed in the PSOP. These PDFs are determined using activity probability density functions which specify the probability of an activity being performed by a user.

A. Activity Probability Density Function (APDF)

A user APDF is defined as:

$$P\left[\delta_{z,i,j,k}\right] = \frac{\mathrm{C}[\delta_{z,i,j,k}]}{\xi_z}, \tag{2}$$

where $\delta_{z,i,j,k}$ denotes the occurrence of an activity being performed by U'_z in R_i sharing a start time with T_j and an end time with T_k. $\mathrm{C}\left[\delta_{z,i,j,k}\right]$ is the number of incidences of $\delta_{z,i,j,k}$ observed in the environment. Namely, past user activity data can be obtained from observations of each U'_z in the environment for a duration of ξ_z days. These observations, $D(d_1, d_2, \dots, d_Y)$, are stored in a database, where each d_y is a tuple containing: an activity, a user, a region, a start time, and an end time. The APDF is then determined using the frequency in which $\delta_{z,i,j,k}$ occurs in D.

There are two special cases in which d_y may not have a start or end time that directly coincides with any T_ω, as seen in Fig. 1. In order to still consider these cases, d_y must be separated into a set of probabilistic observations that share start and end times with time periods, such that both the original observations and separated set of observations have the same probability of finding the user during all time periods.

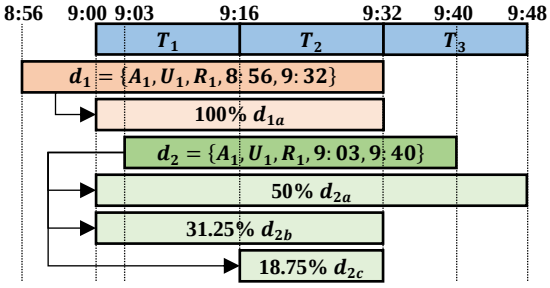


Fig. 1. Example of special cases (d_1 and d_2) where d_y does not share either a start or end time with any T_ω, and is separated into probabilistic observations.

Case 1: When the start or end times of d_y do not lie inside the search time frame, we truncate any segment of d_y that is before the first time period, T_1, or after the last time period, T_Ω. For example, in Fig. 1, d_1 is truncated to d_{1a}, resulting in an increase of $\mathrm{C}\left[\delta_{1,1,1,2}\right]$ by 1.

Case 2: When d_y lies within the search time frame but does not have the same start or end time with any T_ω, we consider the probability of the robot finding the user during searches in each time period. For example, for d_2 in Fig. 1, the probability of finding the user in T_1 is 81.25%, in T_2 is 100%, and in T_3 is 50%. We assign the lowest of these probabilities to a probabilistic observation corresponding to the smallest $T_{j,k}$ that contains all T_ω with non-zero probability, e.g. assigning 50% to d_{2a} corresponding to $T_{1,3}$. As a result, 50% of each T_ω probability is accounted for, leaving 31.25% in T_1, 50% in T_2, and 0% in T_3 to be accounted for. The process is repeated until all the probabilities have been accounted for, which results in 31.25% for d_{2b} corresponding to $T_{1,2}$, and 18.75% for d_{2c}

corresponding to $T_{2,2}$. The probabilities for d_{2a}, d_{2b}, and d_{2c} result in increases of $\mathrm{C}[\delta_{1,1,1,3}]$ by 0.5, $\mathrm{C}[\delta_{1,1,1,2}]$ by 0.3125, and $\mathrm{C}[\delta_{1,1,2,2}]$ by 0.1875, respectively.

B. User Location Probability Density Function

The user location PDF, denoted as $P[\bigcap_{\omega=j}^{k} \phi_{z,i,\omega}]$, determines the probability of U_z' being in R_i during $T_{j,k}$, where $\phi_{z,i,\omega}$ denotes U_z' being in R_i during time period T_ω. The APDF is used to determine $P[\bigcap_{\omega=j}^{k} \phi_{z,i,\omega}]$ as the probability that U_z' will perform an activity in R_i during a time window that contains $T_{j,k}$:

$$P\left[\bigcap_{\omega=j}^{k} \phi_{z,i,\omega}\right] = \sum_{r=1}^{j} \sum_{m=k}^{\Omega} P[\delta_{z,i,r,m}]. \quad (3)$$

V. Robot Search for Multiple Dynamic Users

The main objective of our proposed search approach is to generate a search plan for a given search query which maximizes the expected number of target users found.

As previously mentioned, we model our search problem as an extension of the orienteering problem defined as PSOP. The robot must select a subset of regions to search, a duration for how long to search these regions, and the order in which to search these regions. We divide the time frame into multiple time periods to account for dynamic user behaviors. Each region search must be completed within a time period. As opposed to searching every location in each visited region, we allow for a variable amount of search time to be spent at each region to account for varying user location preferences within the region. At each of the regions, the robot obtains a reward associated with the expected number of target users found based on the search duration and any previous searches of the same region. The multi-period aspect introduces a maximum search time within each time period, t^{period}, multiple visits to a region allowing for at most one visit per time period, region rewards that change across time periods, and conditional rewards based on previous searches of the same region in pervious time periods.

The objective of the PSOP is to maximize the total reward acquired during the search:

$$\begin{aligned} &\text{maximize} \quad W(a_1) + \sum_{h=2}^{H} W\left(a_h \middle| \bigcap_{x=1}^{h-1} a_x\right), \quad (4)\\ &\text{subject to} \quad \sum_{a_h \in SP_\omega} \left(t(h) + t_{i(h-1)}^{i(h)}\right) \le t^{period}, \forall \omega \in [1, \Omega], \end{aligned}$$

where $W(a_1)$ is the reward acquired by the robot performing the first search action in SP, $W\left(a_h \middle| \bigcap_{x=1}^{h-1} a_x\right)$ is the reward acquired from the robot performing search action a_h, considering all the prior planned search actions, and SP_ω is the set of search actions (search plan) executed in T_ω. The travel time, $t_{i(h-1)}^{i(h)}$, for the first search action, $h = 1$, is determined using R_0, which represents the starting region of the robot.

As the PSOP is an extension of the OP which is NP-hard [55], our problem is also NP-hard. Therefore, we approximate its optimal solution by solving two sub-problems: 1) the conditional multiperiod knapsack problem (CMPKP), which is a variant of the multiperiod knapsack problem (MPKP) [38], and 2) the traveling salesman problem (TSP) [39]. The CMPKP expands on the MPKP by using user APDFs when assigning rewards to a set of search actions for the same region in multiple time periods.

User location PDFs are first used to compute the reward for each search action. Using these rewards, the CMPKP generates an unordered plan, SP^{KP}, which specifies the time $t(a_{i,\omega})$ to spend searching each region R_i during every time period T_ω in order to maximize the total reward acquired. The TSP is then used to find a one-to-one mapping from the unordered CMPKP plan to the ordered multi-person search plan, $f: SP^{KP} \rightarrow SP$, which minimizes the total time required to complete the search in each respective time period. The interdependence between the KP and TSP requires the KP to allocate as much time as possible to searching, while still leaving enough time to travel between regions such that the TSP can generate a solution the robot can execute within the time frame. Too little search time results in the robot wasting time between actions, too much results in an infeasible plan. If the shortest sequence of performing the planned search actions exceeds t^{period}, the procedure is iterated to obtain a feasible search plan. During the execution of a search plan, if a target user is found, the rewards assigned to searching each region for that user are removed and the robot replans. An overview of our proposed PSOP approach is presented in Fig. 2.

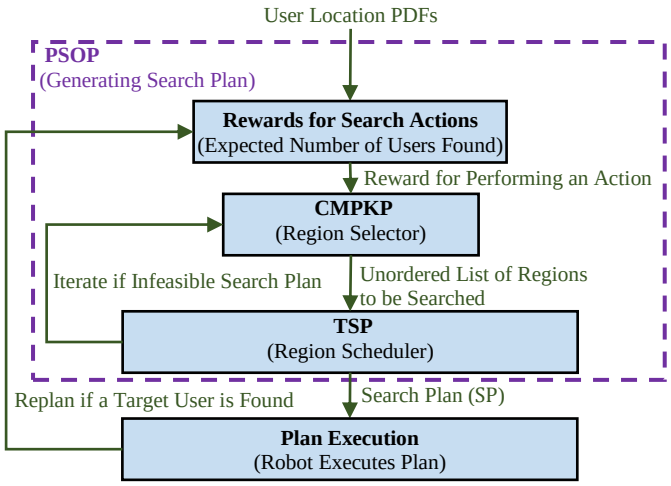


Fig. 2. Search plan generation using the proposed MPSP to solve the PSOP.

We apply our aforementioned approach, as even small instances of the PSOP cannot be solved optimally. For example, considering a scenario with 6 users, 6 private rooms, 4 common rooms, 3 time periods, 4 search durations within each region (e.g. 12, 24, 36, 48 seconds), and a 10 minute time frame: the total number of solutions to explore is lower bounded by the total number of permutations of all ten rooms in all time periods, i.e., $(10!\,4^{10})^3 = 5.5\text{x}10^{37}$. However, as previously mentioned the PSOP can be decomposed into the CMPKP and TSP sub-problems. Both sub-problems can be solved optimally in a feasible time as discussed in the following subsections. Even though solving the sub-problems results in regions being selected by approximating the travel time, this approximation can produce near-optimal results for the PSOP as the best solutions are likely to spend significantly more time searching within regions than traveling between

regions. Similar sub-problem decompositions for first selecting control points and then ordering the control points have previously been successfully applied to OPs [56],[57].

A. *Rewards for Search Actions*

The MPSP assigns a reward for a search action $a_{i,\omega}$ being implemented in R_i during T_ω for a duration $t(a_{i,\omega})$. This reward represents the expected number of target users found when performing $a_{i,\omega}$. This reward is defined as $W(a_{i,\omega})$ if the planner has not selected to search R_i during any previous time period. In order to determine $W(a_{i,\omega})$, the planner requires the probability, $P[\phi_{z,i,\omega}]$, of each target user U'_z being in region R_i, and the probability, $P[\theta_{z,i,\omega}|\phi_{z,i,\omega}]$, of the robot finding each target user given the user is in the region. $\theta_{z,i,\omega}$ denotes the robot finding U'_z when searching R_i during T_ω for a duration of $t(a_{i,\omega})$. $P[\phi_{z,i,\omega}]$ is provided from the APDF, Eq. (3). $P[\theta_{z,i,\omega}|\phi_{z,i,\omega}]$ is provided by a local planner used by the robot to search within regions. An example local planner, used in the experiments for this paper, could divide a region into cells and assign a probability to finding a user in each cell as a function of the time period. The resulting probability provided to the planner would correspond to the sum of probabilities of the searched cells, based on the local plan generated for the specified search duration provided. $W(a_{i,\omega})$ is then determined as follows:

$$W(a_{i,\omega}) = \sum_{z\in U'} P[\theta_{z,i,\omega}|\phi_{z,i,\omega}]P[\phi_{z,i,\omega}]. \quad (5)$$

If the planner has already selected to search R_i in previous time periods, the reward assigned to $a_{i,\omega}$ is defined as:

$$W(a_{i,\omega}|\cap_{k=1}^{\omega-1} a_{i,k}) = W(a_{i,\omega}) - \sum_{z\in U'}\sum_{j=1}^{\omega-1} P\left[\cup_{k=j}^{\omega-1}\theta_k, \theta_\omega|\cap_{k=j}^{\omega}\phi_k\right]_{z,i} P\left[\cap_{k=j}^{\omega}\phi_k\right]_{z,i}, \quad (6)$$

where the notation $[\cdot]_{z,i}$ indicates that the variables in the brackets apply to U'_z and R_i. $P\left[\cup_{k=j}^{\omega-1}\theta_k, \theta_\omega|\cap_{k=j}^{\omega}\phi_k\right]_{z,i}$ is the probability of the robot finding U'_z when searching R_i during a previous time window $T_{j,\omega-1}$ and during the current time period T_ω, given U'_z stays within R_i for the entire duration of $T_{j,\omega}$. We subtract this probability to ensure rewards are not assigned twice for finding the same target user. We obtain $P\left[\cup_{k=j}^{\omega-1}\theta_k, \theta_\omega|\cap_{k=j}^{\omega}\phi_k\right]_{z,i}$ from a local search planner. In Section VI, we provide an example of a local search technique that can be used, including the probabilities it provides.

B. *Conditional Multiperiod Knapsack Problem*

The CMPKP uses the search action rewards to select an unordered set of search actions which maximizes the total acquired reward while ensuring each search action can be performed within its allocated time period:

$$\text{maximize } \sum_{i\in R}\left(W(a_{i,1}) + \sum_{\omega=2}^{\Omega} W(a_{i,\omega}|\cap_{k=1}^{\omega-1} a_{i,k})\right), \quad (7)$$
$$\text{subject to } \sum_{i\in R}\left(t(a_{i,\omega}) + \gamma_{i,\omega} t^{move}\right) \le t^{period}, \forall\omega \in [1,\Omega],$$

where $\gamma_{i,\omega}$ indicates if a search occurs within R_i during T_ω, i.e., $\gamma_{i,\omega} = 0$ if duration $t(a_{i,\omega}) = 0$, and $\gamma_{i,\omega} = 1$ otherwise, and t^{move} is a constant estimated travel time between the regions as the order in which the regions are visited is unknown.

We represent the CMPKP as a minimum flow graph, Fig. 3. Within this graph, all feasible amounts of elapsed search time per time period, defined as $Q = \{Q_1, \dots, Q_\Omega\}$, are enumerated. Each node N_Q^i in the graph represents a decision node in which the time to search R_i, denoted as $\tau_i = \{t(a_{i,1}), \dots, t(a_{i,\Omega})\}$, is determined given that Q has already been allocated to search regions R_1 to R_{i-1}. Nodes $N_Q^{I'+1}$ represent terminal nodes. I' represents the number of rooms in which a target user may be present. A pair of nodes is connected by an edge $E_{\tau_i}^i$ which corresponds to one of the possible search time decisions for R_i.

Each edge weight is set to the negative sum of the rewards corresponding to the search actions that would be performed to transition between its two connecting nodes:

$$W(E_{\tau_i}^i) = -W(a_{i,1}) - \sum_{\omega=2}^{\Omega} W(a_{i,\omega}|\cap_{k=1}^{\omega-1} a_{i,k}). \quad (8)$$

We assign negative edge weights such that minimizing the sum of the edge weights when traversing the graph will result in maximizing the reward. To solve the minimum flow graph, we use the Bellman-Ford shortest path algorithm [58] from the starting node $N_{0,\dots,0}^1$ to any terminal node.

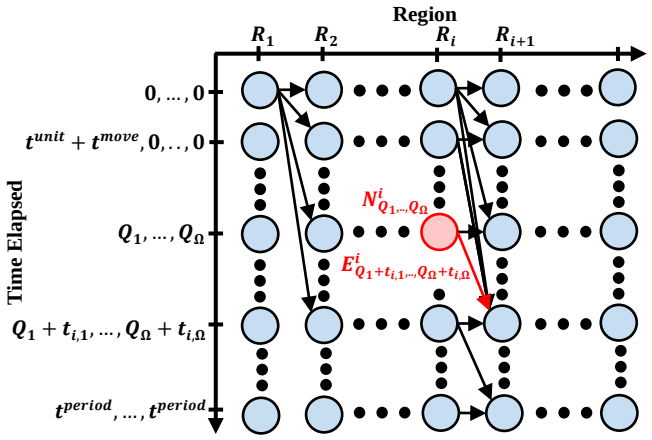


Fig. 3. Minimum flow graph for the CMPKP where $t_{i,\omega}$ represents $t(a_{i,\omega})$.

The CMPKP can become practically infeasible to solve for a large number of time periods, Ω, or for a large value of $\frac{t^{period}}{t^{unit}}$, since the time complexity of the Bellman-Ford algorithm is the number of nodes, $|N|$, multiplied by the number of edges, $|E|$:

$$O(|N||E|) = O\left(\left(\frac{t^{period}}{t^{unit}}\right)^{2\Omega} I'^2 \left(\frac{t^{max}}{t^{unit}}\right)^{\Omega}\right). \quad (9)$$

Therefore, we can approximate its solution for such scenarios using the iterative MPSP (I-MPSP) which solves each time period sequentially, as shown in Fig. 4. Hereafter, we refer to the non-iterative MPSP discussed above as the complete time frame MPSP (C-MPSP), while MPSP will be used to refer to the planner class which contains both the C-MPSP and I-MPSP solutions.

For the I-MPSP, we represent the CMPKP as Ω minimum flow graphs, denoted as $G_1, \dots, G_\Omega$. The decision nodes $N_{Q_\omega}^{i,\omega}$ correspond to selecting the search time in T_ω for R_i given that

Q_ω has already been assigned to search R_1 to R_{i-1}. The edges $E^{i,\omega}_{t(a_{i,\omega})}$ represent the decision to search R_i for $t(a_{i,\omega})$ in T_ω. Each edge in graph G_ω has a weight corresponding to the negative reward of the search action that would be performed. This reward is dependent on the edges selected for the same region in the previous graphs G_1 to $G_{\omega-1}$:

$$W\left(E^{i,\omega}_{t(a_{i,\omega})}\middle|E^{i,\omega-1}_{t(a_{i,\omega-1})},\dots,E^{i,1}_{t(a_{i,1})}\right) = -W\left(a_{i,\omega}\middle|\bigcap_{k=1}^{\omega-1} a_{i,k}\right). \quad (10)$$

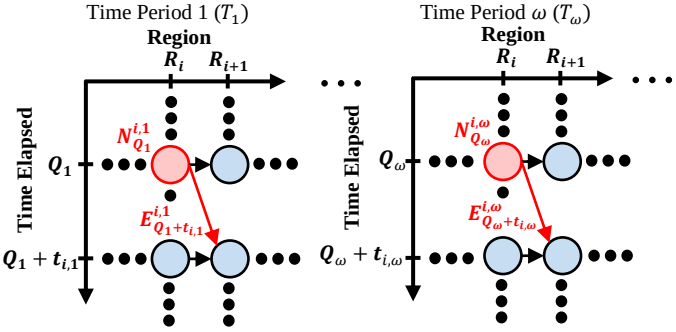


Fig. 4. Sequential approach to solve the CMPKP where $t_{i,\omega}$ represents $t(a_{i,\omega})$.

The search plan, SP^{KP}, is generated by sequentially solving each minimum flow graph using the Bellman Ford algorithm as discussed above. The time complexity of the approximation is:

$$O(\Omega|N||E|) = O\left(\Omega\left(\frac{t^{period}}{t^{unit}}\right)^2 I'^2\left(\frac{t^{max}}{t^{unit}}\right)\right). \quad (11)$$

Hence a solution can be obtained even for large values of Ω and $\frac{t^{period}}{t^{unit}}$.

C. Traveling Salesman Problem

The CMPKP provides an unordered set of search actions per time period. However, as the solution to the PSOP requires an ordered set of search actions, we must find a mapping $f: SP^{KP} \rightarrow SP$ that will provide an optimal ordering that minimizes the travel time within each time period:

$$\text{minimize} \quad \sum_{a_h \in SP_\omega}\left(t^{i(h)}_{i(h-1)}\right), \forall\omega \in [1,\Omega]. \quad (12)$$

The ordering for each time period is solved iteratively. The last region of the previous time period is used as the starting region for the next time period. The environments we consider for the task assistance application, such as a floor of an office building or long-term care facility, can be represented using tree topologies (i.e., they consist of long hallways with rooms branching off of the hallways). Given a tree topology, in order to obtain the optimal shortest tour that visits all the selected regions, a depth first search is performed [59]. Namely, the region farthest from the starting region is searched last. This algorithm for solving the TSP has a linear time complexity. For environments with non-tree topologies, a depth first search does not solve the TSP, in which case we recommend using the Lin-Kernighan heuristic to solve the TSP. This technique has been shown to have a run-time which scales linearly with the number of nodes and has generated optimal solutions for several TSP problems of sizes up to 10,000 nodes [60].

D. Feasible Search Plans

Initially, we use a greedy approach which selects t^{move} to be equal to t^{unit}. Upon obtaining the search plan, SP, if the time it takes to execute all the search actions in the same time period exceeds the time allotted to that time period, we replan by increasing t^{move} by t^{unit}, until the condition below is satisfied:

$$\sum_{a_h \in SP_\omega}\left(t(h) + t^{i(h)}_{i(h-1)}\right) \leq t^{period}, \forall\omega \in [1,\Omega]. \quad (13)$$

When the feasible search plan is executed, once a target user U'_z is found, the plan no longer consists of an optimized sequence of search actions to find the remaining users. Therefore, replanning occurs.

E. Replanning

To replan, a new plan is generated such that it only searches for the remaining users during the time that remains in each time period. The rewards for the search actions in the new plan are conditioned on all search actions already performed by the robot and no longer consider the users that have already been found. As a result, the CMPKP only considers edges in the minimum flow graphs that match the time already spent searching regions in previous time periods and are at least equal to time spent searching regions in the current time period.

Replanning can also be used to deal with unexpected changes in the environment (e.g., a closed door) in which two regions which were previously neighboring are no longer physically accessible to one another [61]. In this case the robot's stored map of the environment can be updated to indicate which regions are no longer neighbors. Any pair of regions with a shortest path affected by this change has a new shortest path computed using the updated environment information, and the travel time between the pair of regions is also updated. Using the updated environment and travel times, a new search plan is generated for the time remaining in each time period.

VI. Implementation

We integrated the MPSP with a local search planner and a person identification method on a mobile robot platform to validate its performance within a multi-room environment.

A. The Socially Assistive Robot Blueberry

The robot used in our experiments is the socially assistive robot Blueberry, Fig. 5. Blueberry navigates using its differential drive base and has a number of sensors, including a Hokuyo laser range finder, an Xtion IR depth camera, a Logitech pro c920 RGB camera, and optical wheel encoders. The robot navigates an environment with an average speed of 0.8m/s. It can also interact with users using its synthesized voice and animated face. We have developed a system architecture, Fig. 6, to integrate our planner and a local search planner with the Blueberry robot. Our architecture is implemented within the robot operating system (ROS) framework on Ubuntu.

The MPSP initiates the search planning process based on a

search query request. It generates a search plan for the robot to implement. Once the robot enters a region, the local planner is used to search the region for a specified time. The local planner sends navigation goals to the robot's navigation system and user identification goals to the person identification module. The person identification module uses RGB and depth data from the cameras to detect and recognize target users. Once a target user is recognized, the planner performs replanning to find the remaining target users.

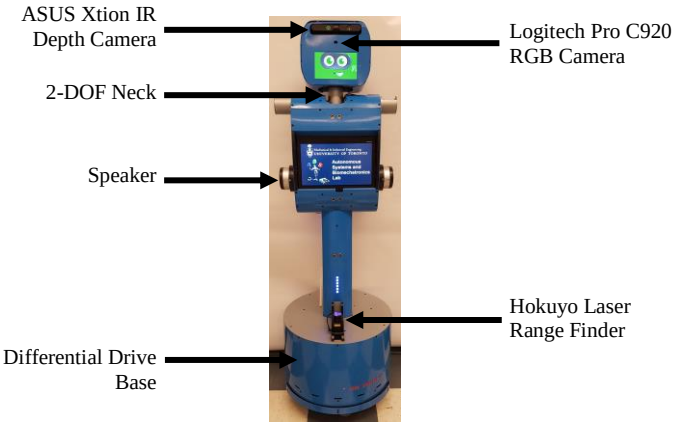


Fig. 5. The socially assistive robot Blueberry.

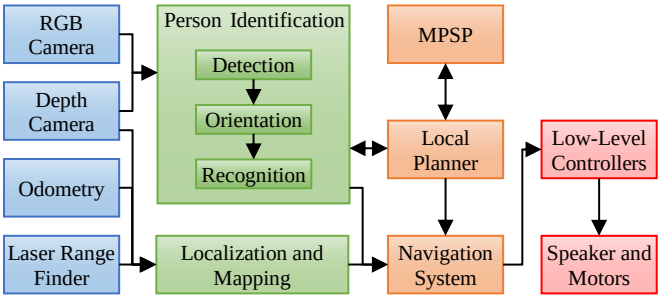


Fig. 6. Robot system architecture.

1) Localization and Mapping

A 2.5D grid map of the environment is generated for localization and navigation purposes using both laser scans from the laser range finder and 3D point clouds from the depth camera. The latter provides obstacle information in the height range of the robot. To generate the map, Blueberry navigates the environment of interest while using this sensory information as input into the Gmapping simultaneous localization and mapping (SLAM) technique [62].

B. Local Planner

Our planner is not dependent on a specific local planner and can be integrated with any local planner that can provide both $P[\theta_\omega|\phi_\omega]_{z,i}$ and $P\left[\cup_{k=j}^{\omega-1}\theta_k, \theta_\omega|\cap_{k=j}^{\omega}\phi_k\right]_{z,i}$, as needed in Eqs. (5) and (6), respectively. For the experiments presented herein, the local planner divides the regions into cells, and assigns an equal probability to finding a user in any cell. To generate a search plan, we used a TSP local planner that uses dynamic programming to plan a minimum time tour of each region [63]. The tour consists of locations in the region from which cells are searched. This approach is used as the robot does not have a priori information regarding user locations within regions, and therefore the local planner can only optimize the order of cells to search by minimizing travel time between subsequent cells. The first time a region is searched, the robot begins the tour by searching the closest cell to the door and then following the tour order. If the robot needs to perform a subsequent search of a region, the tour is continued from the last visited cell. The local planner assumes each cell can be searched in t^{cell}, to account for the time needed to travel between locations and perform person identification. For the TSP local plan, $P[\theta_\omega|\phi_\omega]_{z,i}$ is determined as the percentage of R_i that can be searched in $t(a_{i,\omega})$:

$$P[\theta_\omega|\phi_\omega]_{z,i} = \frac{t(a_{i,\omega})}{t^{cell}\beta_i}, \tag{14}$$

where β_i denotes the number of cells in R_i. $P\left[\cup_{k=j}^{\omega-1}\theta_k, \theta_\omega|\cap_{k=j}^{\omega}\phi_k\right]_{z,i}$ is determined as the percentage of R_i that has already been searched in $T_{j,\omega-1}$ and is being searched again in T_ω:

$$P\left[\cup_{k=j}^{\omega-1}\theta_k, \theta_\omega|\cap_{k=j}^{\omega}\phi_k\right]_{z,i} = max\left(0, \frac{t(a_{i,\omega})}{t^{cell}\beta_i} - max\left(1 - \frac{\sum_{k=j}^{\omega-1} t(a_{i,k})}{t^{cell}\beta_i}, 0\right)\right). \tag{15}$$

If the time spent searching R_i during $T_{j,\omega-1}$ is more than enough time to search every cell in R_i, then any cells searched during T_ω will have already been previously searched in $T_{j,\omega-1}$. If the time spent searching R_i during both $T_{j,\omega-1}$ and T_ω is not enough to search the entire region, then the cells that were not searched during $T_{j,\omega-1}$ are then searched during T_ω.

C. Person Identification

Once the robot navigates to a location within a region, it searches the corresponding cell for target users. The robot obtains RGB and depth images using multiple head orientations to obtain full coverage of the cell. The person identification technique is then performed in four stages: 1) person detection, 2) orientation recognition, 3) person recognition, and 4) person identification contingency strategy.

1) Person Detection

To perform real-time person detection, we adapt the template matching technique presented in [64]. We conduct 2D head and shoulder silhouette template matching to find the highest correlation between templates of different sizes and each segment from a depth image, where segments are generated using a sliding window approach. We iterate over the matches in descending order of correlation, merging any matches within a defined distance (i.e., 100 pixels). We then use a support vector machine (SVM) classifier with a radial basis function kernel to classify each template match as a person, based on a feature set. This set includes the following features for each match: correlation, number of merged matches, average correlation across merged matches, theoretical head area, ratio between actual and theoretical head radii, ratio between theoretical head and actual body areas, and position of actual head relative to body. The actual head radius is obtained by finding the distance from the center of the head

to the nearest edge in the depth image. The theoretical head radius and area are based on the depth of the head in the depth image [64]. The actual body area is obtained by finding contours in the depth image [65], approximating polygons that fit these contours [66], and then finding the area of the smallest rectangle that encloses each polygon.

The SVM classifier was trained using a dataset of 1,700 images of people at different distances and orientations, as well as 300 images without people. The people in the dataset were not the same as those who participated in the experiments presented below.

2) Orientation Recognition

Once a person has been detected, his/her head orientation is determined by using both front facing and profile Haar cascades [67]. The head orientation is approximated to be front (detected by front facing cascade), front-left/front-right (detected by front facing and respective profile cascades), or left/right (detected by only the respective profile cascade).

3) Person Recognition

Once the head orientation is known, the person is identified using a deep convolutional neural network (DCNN) [68]. The DCNN was trained using a database of 100 RGB face images of each target user, who the robot needs to find, in different orientations. During the person recognition stage, a person is recognized if a confidence level above 70% is achieved.

4) Person Identification Contingency Strategy

In the cases where orientation or person recognition fails, a person identification contingency strategy is used by the robot, Table I. Namely, if either orientation or person recognition fails, Blueberry performs the corresponding action 1, which consists of the robot changing both its orientation and position with respect to the person in order to rerun the respective recognition. During these actions the robot maintains a social distance of approximately 1.5m from the person [69]. If action 1 is implemented for both cases, and recognition is not achieved, the robot verbally requests the person to identify himself/herself by asking "Hi, what is your name?".

TABLE I. PERSON IDENTIFICATION CONTINGENCY PLAN

Failures	Action 1	Action 2
Orientation Recognition	Move in increments of 120° to new locations approximately 1.5m from the person and rerun orientation recognition	Verbal confirmation
Person Recognition	Move in increments of 30° to new locations about the person until approximately 1.5m in front of the person and rerun person recognition	

VII. Search Experiments

We have conducted experiments in which: 1) a simulated Blueberry robot searches for multiple residents in virtual long-term care environments, and 2) the physical Blueberry robot searches for multiple employees on a floor of a real university building. The overall objective of these experiments was to validate the robustness of our person search method in finding people of interest under varying conditions. As mentioned above, the experiments used a TSP local planner to generate search plans within regions.

A. Long-term Care Environments

Our first set of experiments focused on a simulated Blueberry robot searching for multiple residents in a virtual long-term care facility. The environments used were modeled and scaled based on the layout of one of our partner long-term care facilities. We developed a custom simulator in C++ using the OpenGL library. The simulator interfaces with the search planner providing a search query and receiving a search plan. A grid-world environment was created in which the robot and users moved in cardinal directions. To move between multiple locations the robot and users followed shortest path trajectories.

Activities: Different activity sets were used to test the performance of our search technique when looking for users with varying activity preferences. Each activity set consisted of such activities as take a nap, read, listen to music, play games, watch T.V., and eat, being performed at varying times of the day and in specific regions of the environment, Table II. In Table II, the time of day column refers to times during which the activities can be performed, however users only perform the activity for a duration between 15 and 60 minutes in a single region. Activities with longer durations are possible if a user performs the same activity consecutively. Similarly, a single activity can be performed for an extended duration in multiple regions. The activity sets were generated using the daily schedules of residents at our partner care facility.

Environments: We used multiple environments consisting of a combination of common rooms (shared by multiple users) and 26 private rooms (for individual users), representing regions in which the users could be found. The regions were divided into cells of size 2m by 2m based on the sensing range of the robot. The largest regions in the environment were the 8m by 10m Garden and Dining Room, which contained 20 cells, resulting in $t_{max} = 240s$. The environments increased in size based on the number of rooms they contained, e.g. 30, 33, 36, 39, and 42 rooms. An example environment layout for 33 rooms is shown in Fig. 7. Fig. 8 shows example layouts of both a common and a private room within the environment.

TABLE II. ACTIVITY SETS INDICATING ACTIVITY AVAILABILITY AND LOCATION

	Activity Set 1		Activity Set 2		Activity Set 3		Activity Set 4		Activity Set 5	
	Time of Day (hr)	Region	Time of Day (hr)	Region	Time of Day (hr)	Region	Time of Day (hr)	Region	Time of Day (hr)	Region
Take a Nap	7-10,13-16,19-21	PR,RR	7-13	PR,RR	7-10,13-16,19-21	PR			7-21	All
Read	7-9	PR,L,G,RR	13-21	PR,L	8-10,12-14,16-18	G, RR	7-21	G	7-21	All
Listen to Music	10-12,16-18	G,RR	9-12,14-18,20-21	G	10-12,14-16,18-20	G, RR	9-11,13-15	L	7-21	All
Play Games	7-8,9-12,13-21	DR,L,RR	7-12,16-21	RR	7-9,14-16,19-21	RR, L	7-8,10-12,19-21	G	7-21	All
Watch T.V.	7-21	PR,RR	7-21	PR	7-8,9-12,13-17,18-21	PR, DR, RR	7-21	RR	7-21	All
Eat	8-9,12-13,17-18	DR	8-9,12-13,17-18	DR	8-9,12-13,17-18	DR	8-9,12-13,17-18	DR	7-21	All

*Region Classifications: Private Room (PR), Dining Room (DR), Lobby (L), Garden (G), and Recreational Rooms (RR).

Users: There was a unique set of $N = 26$ residents for every combination of activity set and environment. Each user had a unique set of spatial-temporal activity preferences (STAP). For each activity, the STAP contained a preference for the activity, a minimum and maximum duration between 15 and 60 minutes, and a preference for each allowable region. Throughout the day, upon completing an activity, a new activity, duration, and region were selected based on the relative preferences for the allowable activities at the time.

For example, from 12:00 to 1:00, one of the user's STAP from activity set 1 (Table II) had a preference of 20% for watching T.V. and 80% for eating; a watching T.V. duration of 30 to 60 minutes with a preference of 40% for their private room and 60% for the recreational room; and an eating duration of 30 to 45 minutes with a 100% preference for the dining room. Given this STAP, the user chose watching T.V. in their private room from 12:00-12:30 and eating in the dining room from 12:30 to 1:00.

User activity data and maps used for the simulated experiments in the care facility can be found on our website, here, under multi-user robot search. Using the aforementioned data, the user location probabilities for the experiments presented herein were obtained using ξ_z = 30 days of observation data.

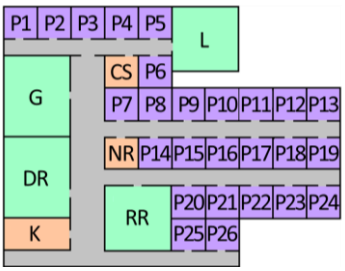
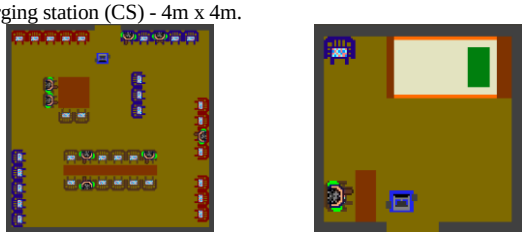


Fig. 7. Long-term care environment layout: hallway (H); private room (PR) - 4m x 4m; garden (G) and dining room (DR) - 8m x 10m; recreational room (RR) and lobby (L) - 8m x 8m; kitchen (K) - 4m x 8m; and nurses' station (NR) and charging station (CS) - 4m x 4m.



(a) (b)

Fig. 8. Example layout of (a) recreational room, and (b) private room.

Simulated Blueberry Robot: The simulated blueberry robot moved to regions and searched within regions specified in the search plan. The robot speed was 0.8m/s to match the real robot. An RGB-D sensor was simulated to have the same field of view as the ASUS Xtion camera used on the real robot. A total time of 12s was assigned to searching each 2 by 2 cell, as $t^{cell} = 12s$. Experiments were conducted with the robot first being able to identify all users in all searched cells, and then with uncertainty introduced while searching a cell. t^{unit} was selected, as a multiple of t^{cell}, $t^{unit} = \mu t^{cell}$, such that our C-MPSP, Eq. (9), could plan within 1 second. For a fair comparison, the same t^{unit} was used for all planners, including during replanning.

Search query: Each search started between 10:00am and 6:00pm, and had a duration of 15, 21, 30, 39, or 45 minutes. During these searches, it can be expected that users will perform 1-3 different activities and may commence a new activity in an already searched region. In this case, the robot may have to return to search that region again in order to find the user. The time frame for each search was divided into $\Omega = 3$ time periods.

A total of 31,250 search trials were conducted. Each search trial considered a unique combination of: *environment size* = {30, 33, 36, 39, 42} rooms, *number of target users* = {1, 5, 10, 15, 20}, *search duration* = {15, 21, 30, 39, 45} minutes, *activity set* = {1, 2, 3, 4, 5}, and *search start time* = {10:00, 12:00, 14:00, 16:00, 18:00} on a 24 hour clock. Each combination was repeated 10 times.

1) C-MPSP and I-MPSP Performance

The performance of each planner was evaluated using the success rate of each trial:

$$success\ rate = \frac{\#\ of\ users\ found}{\#of\ target\ users}. \quad (16)$$

The mean success rate across trials for the varying environment sizes, search durations, and number of target users are presented in Fig. 9. The results indicated that the C-MPSP and I-MPSP technique performed similarly. This suggest that I-MPSP is a good approximation of C-MPSP and can be used to generate plans for large environment sizes, search durations, number of target users, cells per region, and time periods, as the I-MPSP computation time scales linearly with these variables.

As expected, the mean success rate achieved by our planner was inversely proportional to the environment size and directly proportional to the search duration. Furthermore, as our approach was designed to search for multiple users, the mean success rate was robust to the number of target users, achieving a similar success rate across a varying number of target users.

2) Comparative Study

We conducted a comparison study of the performance of our planners with respect to a full environment coverage (FEC) planner, a common room coverage (CRC) planner, and an MDP planner inspired by [18]. We chose an MDP planner for comparison as the existing person search planners for dynamic users in task assistance scenarios are MDP-based, e.g. [17],[18]. Both coverage planners generate plans to maximize the number of cells searched within each time period. FEC considers the entire environment, while CRC only considers common rooms. Coverage was solved optimally using a modified TSP that considered all subsets of regions as well as all varying search durations within each region. The MDP planner uses actions of either moving to a region, searching within a region for a number of time steps, or waiting in a region. The reward for performing each action depends on the expected number of target users found without considering previously searched regions, Eq. (5). The states represent the robot's current action and number of time steps remaining to

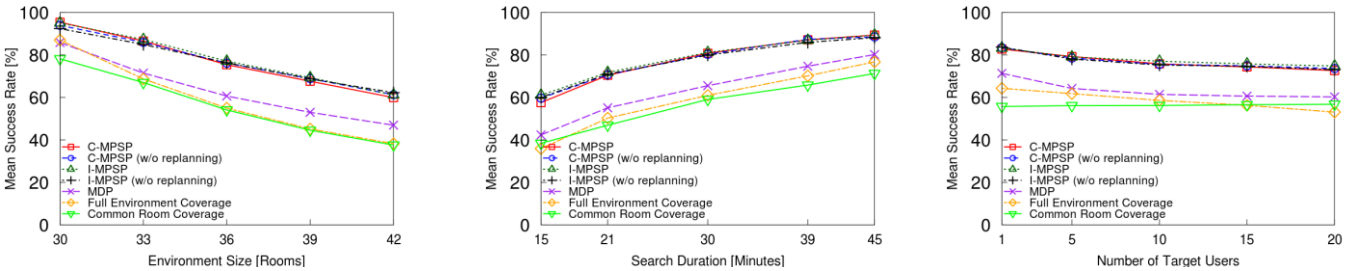


Fig. 9. Mean success rates of each planner across independent performance variables

complete the action. The full details of the MDP planner are presented in Appendix A and the coverage planners in Appendix B. For a fair comparison, as the state-of-the-art MDP planners do not use replanning, we compared the MDP planner to our planner with and without replanning.

a. Performance Comparison of Search Techniques

The overall mean success rates were 77.4% for our C-MPSP, 77.6% for our C-MPSP without replanning, 78.2% for our I-MPSP, 76.7% for our I-MPSP without replanning, 63.6% for the MDP planner, 58.8% for the FEC planner, and 56.3% for the CRC planner.

As shown in Fig. 9, there is a clear performance advantage to using the C-MPSP and I-MPSP for any parameterization of the problem. In particular, the C-MPSP and I-MPSP, both with and without replanning, performed better than the MDP and coverage planners, including for search scenarios with larger environments or shorter search durations.

Kruskal-Wallis H tests with Bonferroni corrections were conducted to determine if the differences in the mean success rates for our planners in comparison to the MDP and coverage planners, both overall and for any parameterization, were statistically significant. Post-hoc two-tailed Dunn's Multiple Comparison tests with a Bonferroni correction were used for pairwise comparisons. A significance level of $\alpha_{KW,revised} = 0.003$ for the Kruskal-Wallis tests and $\alpha_{D,revised} = 0.00025$ for Dunn's were used to avoid Type 1 errors across multiple comparisons. The Bonferroni correction was calculated given an original $\alpha = 0.05$ divided by 16 Kruskal-Wallis tests (i.e., overall mean, and mean across independent performance variables) and 12 Dunn's tests per Kruskal-Wallis test (i.e., all pairs between our 4 planners and the 3 other planners). The results of both tests showed that both overall and for each of the parameterizations statistically significant difference exist between our planners, with and without replanning, and the MDP and coverage planners. The Kruskal-Wallis tests showed differences in the overall mean success rates across the planners, $\chi^2(6) = 16145, p < 0.0001$, and for all the parameterizations, with the minimum $\chi^2(6) = 2511, p < 0.0001$.

b. Comparison of Search Techniques with User Models

To investigate the effects of using the APDF, we compare the C-MPSP with replanning to the MDP planner as the latter also considers a user model. We provide a detailed example of the rewards obtained for the following search query using activity set 3: 10 target users, 36 regions, 30 minute time frame, and starting at 14:00. Table III shows the summation over all the target users for the probability of a target user performing an activity within a region during a time window, i.e., $\sum_{z \in U'} P[\delta_{z,i,j,k}]$.

The plans generated by each planner for this search query are presented in Table IV. In the table, $\frac{t(a_{i,\omega})}{t^{cell}}$ represents the number of cells searched within a region. For the MDP planner a reward, $W(a_{i,\omega})$, is assigned to a current search action without considering previous search actions. Although the MDP planner does not consider previous search actions, for comparison purposes we show the reward, $W^*(a_{i,\omega} | \cap_{k=1}^{\omega-1} a_{i,k})$, that our C-MPSP would have assigned to the search actions chosen by the MDP planner.

TABLE III. SUMMATION OF TARGET USERS' ACTIVITY PROBABILITIES

Private Room1 (PR1)			Private Room3 (PR3)			Garden (G)		
T_1	T_2	T_3	T_1	T_2	T_3	T_1	T_2	T_3
0.11			0.01			0.28		
0.07			0.10			0.13		
	0.02			0.01			0.68	
0.08	0.00	0.01	0.16	0.00	0.01	0.18	0.03	0.61
Dining Room (DR)			**Recreational Room (RR)**			**Lobby (L)**		
T_1	T_2	T_3	T_1	T_2	T_3	T_1	T_2	T_3
0.29			1.38			0.93		
0.11			0.92			0.62		
	0.64			0.83			0.79	
0.19	0.04	0.66	0.88	0.03	0.79	0.69	0.04	0.01

TABLE IV. GENERATED C-MPSP AND MDP PLANS

	C-MPSP Plan w/ replanning			MDP Plan			
T_ω	R_i	$\frac{t(i,\omega)}{t^{cell}}$	$W(a_{i,\omega} \mid \cap_{k=1}^{\omega-1} a_{i,k})$	R_i	$\frac{t(i,\omega)}{t^{cell}}$	$W(a_{i,\omega})$	$W^*(a_{i,\omega} \mid \cap_{k=1}^{\omega-1} a_{i,k})$
T_1	PR3	4	0.27	PR3	4	0.27	0.27
	RR	16	3.18	L	16	2.24	2.24
	L	16	2.24	PR1	2	0.13	0.13
				RR	16	3.18	3.18
T_2	DR	20	1.08	G	7	0.39	0.39
	G	20	1.12	RR	16	3.16	0.86
				L	16	2.38	0.83
T_3	RR	16	1.62	L	16	2.49	0.77
	DR	6	0.20	PR1	1	0.04	0.04
	L	16	1.56	G	10	0.78	0.78
				RR	16	3.00	0.79
Total		**114**	**11.27**		**120**	**18.06**	**10.28**

As can be seen in Table IV, the C-MPSP plan received a smaller total reward than the MDP plan. However, the mean success rate when executing the above MDP plan was determined to be 17% lower than that of the C-MPSP plan. This is due to the MDP planner not considering previous search

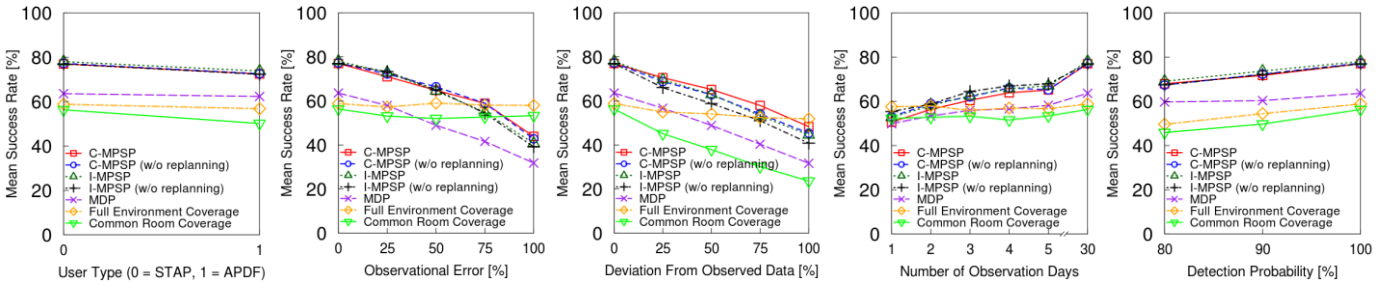


Fig. 10. Mean success rate of each planner across independent uncertainty variables

actions. For example, when the MDP planner chose to search all 16 cells in the recreational room during T_2, it assigned a reward of 3.16. However, considering it had previously chosen to search all 16 cells in the same room during T_1, the reward assigned by our C-MPSP would have only been 0.86. This reward does not include the summations of the target users' probabilities of performing an activity within the recreational room during $T_{1,2}$ and $T_{1,3}$, i.e., 0.92 and 1.38 from Table III, as they have already been accounted for during T_1. On the other hand, the MDP planner double counts these summations, both in the reward for the search action during T_1 and the search action during T_2.

3) Introducing Uncertainty

We conducted an additional set of experiments where we introduced uncertainty into the APDF as well as during the search to investigate the impact on the planners. The uncertainties introduced were: 1) misalignment of user activities with time periods, 2) observational errors during data collection, 3) deviation of user behaviors from their observed data, 4) varying number of observation days, and 5) introduction of detection errors during the search.

The first uncertainty experiment, Experiment #1, investigated how the APDF models the user activity choices, namely how well the APDF handles the special case discussed in Section IV.A. Under normal conditions, users selected activities based on their STAP, and as a result the activities may not have aligned with the time periods (i.e., activities may have changed during a time period). As a result, we compared users who selected activities regularly based on their STAP, and users who selected activities based on their APDF (i.e., only changed their activities at the start of a time period). Activity misalignment was measured by iterating over each time period and considering all activities that occurred within that time period. The measurement was the percentage of the time period during which the activity was not being performed, normalized by the percentage of the activity duration that occurred within the time period.

Experiment #2 examined observational errors in the data set, which varied from 0%, 25%, 50%, 75%, to 100% (fully uniform distribution). These errors were modeled by considering each observation in the data set and selecting to change the region with a probability equal to the error percentage. If a region was changed, the new region was randomly selected from all regions with a probability proportional to the area of each region.

Experiment #3 examined the effect of users deviating from their STAP, varying from 0% to 100%. This deviation was modeled by having users select each next region from all regions with a probability equal to the deviation, otherwise, their next region was selected from their STAP. When selecting from all regions in the environment, their next region was selected with a probability proportional to the area of each region.

Experiment #4 investigated varying the number of observation days. The cases considered were 1, 2, 3, 4, 5 and 30 days. 1 to 5 represented practical amount of time to spend acquiring user data, where as 30 days was selected to provide enough data to act as a ground truth.

Experiment #5 introduced varying detection rates during the search from 80%, 90%, to 100%. When searching a cell containing a user, the probability of identifying the user was equal to the detection rate.

For each of the uncertainty experiments, 2025 search trials were conducted. Each search trial considered a unique combination of: *environment size* = {30, 36, 42} rooms, *number of target users* = {1, 10, 20}, *search duration* = {15, 30, 45} minutes, *activity set* = {1, 3, 5}, and *search start time* = {10:00, 12:00, 14:00, 16:00, 18:00} on a 24 hour clock. Each combination was repeated 5 times. The time frame was divided into $\Omega = 3$ time periods. The mean success rates for each of the 7 planners across the uncertainty variables are shown in Fig. 10.

Kruskal-Wallis and a two-tailed Dunn's test, both with Bonferroni correction, were conducted to determine if the differences in the mean success rates for our planners in comparison to the MDP and coverage planners, for any parameterization, were statistically significant. A significance level of $\alpha_{KW,revised} = 0.0019$ and $\alpha_{D,revised} = 0.00016$ were used based on 26 Kruskal-Wallis tests and 12 Dunn's tests per Kruskal-Wallis test. For all tests, the Kruskal-Wallis test showed a statistically significant difference among the planners for the mean success rates across all parameterizations, with the minimum $\chi^2(6) = 59, p < 0.0001$. For the majority of cases, the Dunn's test showed a statistically significant difference in the mean success rates of our planners, and the MDP and coverage planners. The exceptions were I-MPSP without replanning versus: i) FEC for 2 observation days; and ii) CRC for 75% observational error. Furthermore, I-MPSP with replanning and C-MPSP without replanning versus: i) FEC for 2 observation days, 75% observational error, and 75% deviation from observed data; and ii) CRC for 1 observation day. Additionally, C-MPSP

with replanning versus: i) FEC for 75% observation error and 2 observation days; and ii) CRC for 1 observation day.

For Experiment #1, the STAP activity misalignment ranged from 6 to 18, whereas the APDF misalignment was only 2 to 6. The results demonstrated that the misalignment had little impact on the performance of our planners which directly reasoned about the APDF. Experiment #2 showed that our planners continued to outperform the MDP planner at all levels of observation error and only performed worse than the coverage planners at 75% or more observational error. Similar to the second experiment, Experiment #3 showed our planners consistently outperformed the MDP planner, and only performed worse than the FEC technique at more than 75% deviation. Even at 100% deviation from observed data, we observed the C-MPSP remained within 5% of the FEC technique. This indicated that the coverage aspect of our planners was robust to the users' behaviors as it assigned a large component of time to searching within regions in comparison to the time spent searching between regions. This property suggests, as previously stated, that selecting regions while only considering an average travel cost was an effective technique for generating near-optimal plans. Experiment #4 indicated that 3 or more observation days were required to outperform the coverage approaches. Also, as expected, Experiment #5 demonstrated that the success rate of all techniques increased proportional with the detection rate.

From the above experiments, we can see that the APDF accurately models the user activity choices, namely when handling the special case of activities in misalignment with time periods. We can also see that our planners outperformed the MDP planner under all conditions and the coverage planners for the majority of the conditions, in large part due to reasoning about conditional rewards provided by the APDF. Furthermore, we can see that our planners are robust to errors in the APDF. One thing to note is that when the observational errors and behavior deviations increased to above 75%, or there were less than 2 observation days, the FEC technique performed better.

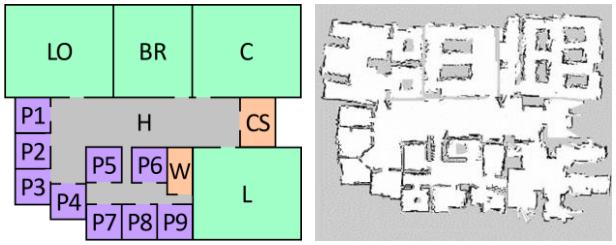
B. *University Office Building*

Our second set of experiments consisted of the Blueberry robot searching a multi-room floor of a university building for varying groups of dynamic users. Four different search trials were conducted with varying numbers of target users, search start times, and search durations.

Activities: The following set of activities were used: meetings, independent work, laboratory work, reading, socializing, and eating.

Environment: The environment consisted of nine private offices and four shared rooms- a cafeteria, boardroom, lounge, and research laboratory. The layout of the environment is presented in Fig. 11a. The 2.5D map of this environment generated using Gmapping is presented in Fig. 11b. The size of each cell in the region was 2m by 2m, and the search duration per cell was defined to be $t^{cell} = 24$s, and $t^{unit} = t^{cell}$ (i.e., $\mu = 1$). The largest regions in the environment were 8m by 10m, which contain 20 cells, resulting in $t^{max} = 480s$.

Users: The number of target users for Blueberry to find in the four trials was 1, 3, 6, and 9, respectively. All trials were conducted with $N = 9$ users sharing the environment. User schedules were obtained from the occupants of the floor for $\xi_z = 30$ days.



(a) (b)

Fig. 11. (a) Layout of the Office floor: hallway (H); private office (P) - 4m x 4m; boardroom (BR) – 8m x 8m; cafeteria (C), lounge (LO), and research laboratory (L) - 8m x 10m; charging station (CS) - 6m x 4m; and washroom (W) – 3m x 5m; and (b) 2.5D map.

Search Query: The search start times were: 5pm for 15 minutes (1 target user), 3pm for 15 minutes (3 target users), 11am for 30 minutes (6 target users), and 9am for 30 minutes (9 target users), for each trial respectively. All trials were conducted with $\Omega = 3$ periods per time frame.

1) *Results*

The results for C-MPSP predicting user locations, and person identification identifying a located user, are both presented in Table V. Example behaviors performed by the Blueberry robot while conducting the search are shown in Fig 12. A video showing the robot implementing the search plan for trial #4 is presented here on our YouTube channel.

(a) (b) (c)

Fig. 12. Robot behaviors: (a) moving to a region, (b) searching within a region, and (c) performing person identification of a localized target user.

TABLE V. SEARCH RESULTS

	Trial 1	Trial 2	Trial 3	Trial 4
Search Start Time	5pm	3pm	11am	9am
Target Users to Find	1	3	6	9
Search Duration (mins)	15	15	30	30
Target Users Located	100% (1)	100% (3)	100% (6)	100% (9)
Target Users Identified	100% (1)	100% (3)	83.3% (5)	100% (9)

During all trials, the planner was able to predict the locations of each of the target users. Furthermore, during the first, second, and fourth search trials, 100% of users were also identified. However, during the third search trial, when the robot located a target user in his private office, the user was not detected using the template matching technique due to his poufy hairstyle that day, which made it difficult to match the head contour. Later in the search, the same target user was located in the cafeteria, however, the template matching approach was still unable to detect the user.

There were other cases where the C-MPSP was able to predict the location of target users who were not identified the

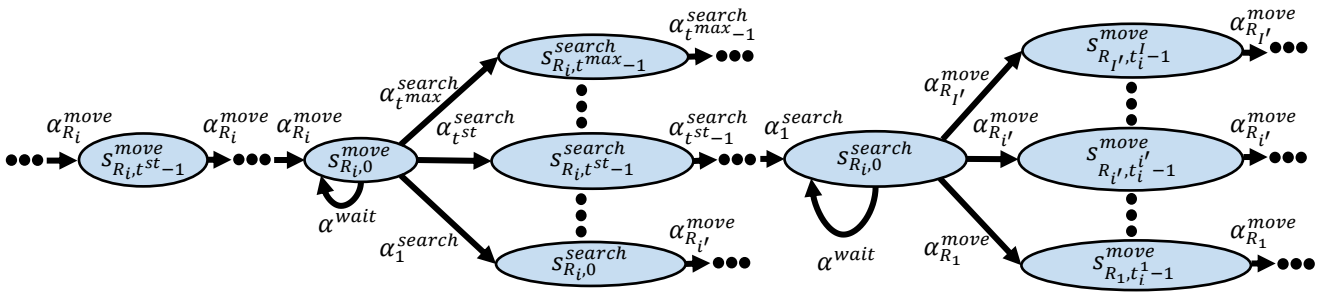


Fig. A1. Finite state machine for the MDP planner.

first time by the person identification module. For example, during trial #3, the robot located a target user looking down while reading a book in the lounge. The user was detected, but could not be recognized as his head was tilted too far down. The C-MPSP was able to later locate the same user in the boardroom and person identification was able to successfully recognize this user.

VIII. Conclusions and Future Work

This paper presents a novel person search orienteering problem, PSOP, and a multi-person search planner, MPSP, which solved this problem by generating a plan to find multiple dynamic users in a multi-room indoor environment. User activity probability density functions are used to predict the probability of a person remaining in a region or revisiting the region again during a search time frame. Experiments conducted showed that both the complete time frame, C-MPSP, and iterative, I-MPSP, were robust to finding varying numbers of users during different time frames and in environments of different sizes. Comparisons with an MDP planner as well as coverage planners showed that our planner was able to find a larger number of target users under the different conditions. Additional experiments verified that the C-MPSP and I-MPSP techniques were robust to both uncertainty in the user APDFs as well as uncertainty introduced during the search. Furthermore, the results also validated the integration of the MPSP within a robot architecture for real-world robot search applications. Our future work consists of extending our MPSP to search for dynamic users in crowded environments. We also plan to scale our problem to include cases with large environments that have multiple floors and perform experiments incorporating local planners that reason about user location probabilities. Additionally, we will consider modeling the uncertainty in environment dynamics using a probabilistic model considering the probability of two adjacent regions being neighbors, similar to previous approaches for modeling environment dynamics as presented in [70]. This model can be directly used during the planning phase.

Appendix A- MDP Planner

The MDP planner discretizes the time frame of the search into time steps, each of length t^{cell}. During each time step, the robot performs an action which allows it to transition into a new state from the current state. The set of actions is selected to maximize the expected number of target users found.

A. Actions

At each time step, the MDP planner selects an action to take, $\alpha \in \{\alpha_{R_i}^{move}, \alpha_{t^{st}}^{search}, \alpha^{wait}\}$, where: 1) $\alpha_{R_i}^{move}$ is move toward region R_i; 2) $\alpha_{t^{st}}^{search}$ is search the current region for the next t^{st} time steps; and 3) α^{wait} is do nothing.

B. States

At the start of each time step, the robot is in a particular state, $s \in \{s_0, s_{R_i,t^{st}}^{move}, s_{t^{st}}^{search}\}$, where: 1) s_0 is the initial state; 2) $s_{R_i,t^{st}}^{move}$ is moving to region R_i with t^{st} time steps remaining before arrival; and 3) $s_{t^{st}}^{search}$ is searching the current region for t^{st} time steps.

C. Finite State Machine

The FSM used by the MDP planner is shown in Fig. A1. To account for the dynamic behavior of a user, the MDP allows for a region to be searched multiple times during the search, however, only once within a single time period. The FSM also does not allow for a search action, $\alpha_{t^{st}}^{search}$, to be selected when there is less than t^{st} time steps remaining in the current time period.

D. Plan Generation

The MDP planner generates a plan using backwards induction. The approach works by traversing the FSM in reverse. Beginning at the last time step, a total reward, $V(s)$, is assigned to each state to indicate the maximum number of target users that can be found over the remainder of the search. $V(s)$ is determined by considering the expected number of target users found by each immediate action, given by $W(s,\alpha)$, and the total reward for transitioning into the next state $s'(\alpha)$ by performing α:

$$V(s) = \max_{\alpha} \left(W(s,\alpha) + V\left(s'(\alpha)\right) \right). \quad \text{(A1)}$$

The reward $W(s,\alpha)$ is 0 if the action is to move or wait. If the action is to search, the reward is computed in the same manner as $W(a_{i,\omega})$, Eq. (5), where R_i is the current region, T_ω is determined using the current time step, and $t(a_{i,\omega}) = t^{st}t^{cell}$. The MDP plan is then generated by starting at the initial state and taking the sequence of actions that maximizes the expected number of target users found over the duration of the search:

$$\Pi(s) = \arg\max_{\alpha} \left(W(s,\alpha) + V\left(s'(\alpha)\right) \right). \quad \text{(A2)}$$

APPENDIX B – COVERAGE PLANNERS

The coverage planners plan a sequence of search actions to maximize the total number of unique cells visited within the time frame of the search. The full environment coverage (FEC) planner considers all private and common rooms the target users may occupy (e.g., if user 10 is not in the search query private room 10 is not searched):

$$\begin{aligned} &\text{maximize} && \textstyle\sum_{a_h \in SP}\big(t(h)\big), && \text{(B1)}\\ &\text{subject to} && \textstyle\sum_{a_h \in SP}\big(t(h) + t_{i(h-1)}^{i(h)}\big) \le t^{end} - t^{start}, \\ & && \textstyle\sum_{\omega=1}^{\Omega} t(i,\omega) < t^{cell}\beta_i, \forall i \in [1, I']. \end{aligned}$$

The common room coverage (CRC) planner considers all common rooms, denoted as I'':

$$\begin{aligned} &\text{maximize} && \textstyle\sum_{a_h \in SP}\big(t(h)\big), && \text{(B2)}\\ &\text{subject to} && \textstyle\sum_{a_h \in SP}\big(t(h) + t_{i(h-1)}^{i(h)}\big) \le t^{end} - t^{start}, \\ & && \textstyle\sum_{\omega=1}^{\Omega} t(i,\omega) < t^{cell}\beta_i, \forall i \in [1, I'']. \end{aligned}$$

Repeated full searches of the environment are performed until there is not enough time to search the entire environment. When this happens, the robot returns to the first region searched and generates a new plan with the remaining search time based on the above formulations.

A plan is generated for optimizing the search objective in the coverage planners by considering every connected sub-tree in the environment. As the optimal coverage solution cannot pass by a region without searching it, the optimal solution must exist within one of these sub-trees. The TSP is solved for each of these sub-trees, using the technique discussed in Section V.C, and a plan is generated for each sub-tree where every region in the subtree is fully searched. If the resulting plan is not feasible, the plan is modified to contain the largest partial search of the region to be visited last, such that the plan becomes feasible, if possible. Of all the feasible plans generated, the optimal plan is the plan with the largest number of cells searched.

ACKNOWLEDGEMENTS

The authors would like to thank our partner long-term care facility and our experiment participants.

REFERENCES

[1] Ł. Białek, J. Szklarski, M. 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.

[2] 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.

[3] P. Dames and V. Kumar, "Autonomous localization of an unknown number of targets without data association using teams of mobile sensors," *IEEE Trans. Autom. Sci. Eng.*, vol. 12, no. 3, pp. 850–864, 2015.

[4] H. Lau, S. Huang, and G. Dissanayake, "Optimal search for multiple targets in a built environment," *IEEE/RSJ Int. Conf. on Intell. Robots and Syst.*, pp. 3740–3745, 2005.

[5] G. Hollinger, S. Singh, and J. Djugash, "Efficient multi-robot search for a moving target," *Int. J. Robot. Res.*, vol. 28, no. 2, pp. 201–219, 2009.

[6] R. Patel, P. Agharkar, and F. Bullo, "Robotic surveillance and Markov chains with minimal weighted Kemeny constant," *IEEE Trans. Autom. Contr.*, vol. 60, no. 12, pp. 3156–3167, 2015.

[7] A. Kolling and S. Carpin, "Pursuit-evasion on trees by robot teams," *IEEE Trans. Robot.*, vol. 26, no. 1, pp. 32–47, 2010.

[8] 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.

[9] J. Thunberg and P. Ögren, "A mixed integer linear programming approach to pursuit evasion problems with optional connectivity constraints," *Auton. Robots*, vol. 31, no. 4, p. 333–343, 2011.

[10] N. M. Stiffler and J. M. O'Kane, "A complete algorithm for visibility-based pursuit-evasion with multiple pursuers," *IEEE Int. Conf. on Robot. Autom.*, pp. 1660–1667, 2014.

[11] P. Elinas, J. Hoey, and J.J. Little, "HOMER: Human oriented messenger robot," in *AAAI Spring Symp. Human Inter. with Auton. Syst. in Compl. Env.*, pp. 45–51, 2003.

[12] P. Bovbel and G. Nejat, "Casper: An assistive kitchen robot to promote aging in place," *J. Med. Devices*, vol. 8, no. 3, pp. 1–2, 2014.

[13] M. Volkhardt and H. M. Gross, "Finding people in home environments with a mobile robot," *Eur. Conf. Mobile Robots*, pp. 282–287, 2013.

[14] A. Bayoumi, P. Karkowski, and M. Bennewitz, "Speeding up person finding using hidden Markov models," *Robot. Auton. Syst.*, vol. 115, no. 1, pp. 40–48, 2019.

[15] S. A. Mehdi and K. Berns, "Behavior-based Search of Human by an Autonomous Indoor Mobile Robot in Simulation," *Universal Access Inform. Soc.*, vol. 13, no. 1, pp. 45–58, 2014.

[16] 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.

[17] 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.

[18] 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.

[19] 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.

[20] J. J. Acevedo, B. C. Arrue, I. Maza, and A. Ollero, "Cooperative large area surveillance with a team of aerial mobile robots for long endurance missions," *J. Intell. Robot. Syst.*, vol. 70, no. 1, pp. 329–345, 2013.

[21] J. Keller, D. Thakur, M. Likhachev, J. Gallier, V. Kumar, "Coordinated path planning for fixed-wing UAS conducting persistent surveillance missions," *IEEE Trans. Autom. Sci. Eng.*, vol. 14, no. 1, pp. 17–24, 2017.

[22] A. Wallar, E. Plaku, and D. A. Sofge, "Reactive motion planning for unmanned aerial surveillance of risk-sensitive areas," *IEEE Trans. Autom. Sci. Eng.*, vol. 12, no. 3, pp. 969–980, 2015.

[23] 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.

[24] M. A. Goodrich, B. S. Morse, D. Gerhardt, J. L. Cooper, and M. Quigley, "Supporting wilderness search and rescue using a camera-equipped mini UAV," *J. Field Robot.*, vol. 25, no. 1, pp. 89–110, 2008.

[25] B. Lavis, T. Furukawa, and H. F. D. Whyte, "Dynamic space reconfiguration for Bayesian search and tracking with moving targets," *Auton. Robots*, vol. 24, no. 4, pp. 387–399, 2008.

[26] 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.

[27] M. G. Rashed, R. Suzuki, A. Lam, Y. Kobayashi, and Y. Kuno, "Toward museum guide robots proactively initiating interaction with humans," *10th Annual ACM/IEEE Int. Conf. Human-Robot Interaction Extended Abstracts*, pp. 1–2, 2015.

[28] R. Triebel, K. Arras, R. Alami, L. Beyer, S. Breuers, R. Chatila, M. Chetouani, D. Cremers, V. Evers, M. Fiore, H. Hung, O. A. I. Ramírez, M. Joosse, H. Khambhaita, T. Kucner, B. Leibe, A. J. Lilienthal, T. Linder, M. Lohse, M. Magnusson, B. Okal, L. Palmieri, U. Rafi, M. van Rooij, and L. Zhang, "SPENCER: A socially aware service robot for passenger guidance and help in busy airports," *Field and Service Robot.*, pp. 607–622, 2016.

[29] E. Zalama, J. G. García-Bermejo, S. Marcos, S. Domínguez, R. Feliz, R. Pinillos, and J. López, "Sacarino, a service robot in a hotel environment," *ROBOT2013: 1st Iberian Robot. Conf.*, pp. 3–14, 2014.

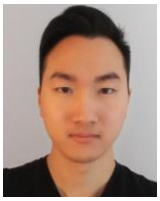
[30] M. Montemerlo, J. Pineau, N. Roy, and S. Thrun, "Experiences with a mobile robotic guide for the elderly," *18th Nat. Conf. on Artificial Intell.*, pp. 587–592, 2002.

[31] D. McColl, W. G. Louie, and G. Nejat, "Brian 2.1: A socially assistive robot for the elderly and cognitively impaired," *IEEE Robot. Autom. Mag.*,

vol. 20, no. 1, pp. 74–83, 2013.
[32] W. Moyle, M. Cooke, E. Beattie, C. Jones, B. Klein, G. Cook, and C. Gray, "Exploring the effect of companion robots on emotional expression in older adults with dementia: A pilot randomized controlled trial," *J. Gerontological Nursing*, vol. 39, no. 5, pp. 46–53, 2013.
[33] 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.
[34] C. Thompson, S. Mohamed, W. G. Louie, J. Chen He, J. Li, G. Nejat, "The robot Tangy facilitating Trivia games: A team-based user-study with long-term care residents," *Int. Symp. on Robot., Intell. Sensors*, pp.173–178, 2017.
[35] M. K. Hasan, A. S. M. Hoque, and T. Szecsi, "Application of a plug-and-play guidance module for hospital robots," *Int. Conf. Ind. Eng. and Operations Manage.*, pp. 1–6, 2013.
[36] T. Tsiligrides, "Heuristic Methods Applied to Orienteering," *J. Oper. Res. Soc.*, vol. 35, no. 9, pp. 797-809, 1984.
[37] S. Mohamed and G. Nejat, "Autonomous search by a socially assistive robot in a residential care environment for multiple elderly users using group activity preferences," *26th Int. Conf. Automat. Planning and Scheduling Workshop on Planning and Robot.*, pp. 58–66, 2016.
[38] B. H. Faaland, "Technical note—The multiperiod knapsack problem," *Oper. Res.*, vol. 29, no. 3, pp. 612–616, 1981.
[39] J. B. 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.
[40] K. E. C. Booth, T. T. Tran, G. Nejat, and J. C. Beck, "Mixed-Integer and Constraint Programming Techniques for Mobile Robot Task Planning," *IEEE Robot. Automat. Lett.*, vol. 1, no. 1, pp. 500–507, 2016.
[41] S. Bernardini, M. Fox, and D. Long, "Combining temporal planning with probabilistic reasoning for autonomous surveillance missions," *Auton. Robots*, vol. 41, no. 1, pp. 181–203, 2017.
[42] M. Morin, M. P. Castro, K. E. C. Booth, T. T. Tran, C. Liu, and J. C. Beck, "Intruder alert! Optimization models for solving the mobile robot graph-clear problem," *Constraints*, vol. 23, no. 3, pp. 335–354, 2018.
[43] M. Avci and M. G. Avci, "A GRASP with iterated local search for the traveling repairman problem with profits," *Comp. Ind. Eng.*, vol. 113, no. 1, pp. 323–332, 2017.
[44] E. J. Lodree, D. Carter, and E. Barbee, "The Donation Collections Routing Problem," *Int. Conf. Dynamics of Disasters*, pp. 159–189, 2016.
[45] W. Zheng, Z. Liao, and J. Qin, "Using a four-step heuristic algorithm to design personalized day tour route within a tourist attraction," *Tourism Manage.*, vol. 62, pp. 335–349, 2017.
[46] M. Mann, B. Zion, D. Rubinstein, R. Linker, and I. Shmulevich, "The Orienteering Problem with Time Windows Applied to Robotic Melon Harvesting," *J. Optimization Theory Appl.*, vol. 168, no. 1, pp. 246–267, 2016.
[47] C. Zhang, H. Liang, and K. Wang, "Trip Recommendation Meets Real-World Constraints: POI Availability, Diversity, and Traveling Time Uncertainty," *ACM Trans. Inform. Syst.*, vol. 35, no. 1, pp. 1–28, 2016.
[48] K. Sylejmani, J. Dorn, and N. Musliu, "Planning the trip itinerary for tourist groups," *Inform. Technol. Tourism*, vol. 17, no. 3, pp. 275–314, 2017.
[49] W. Y. Kwon, M. Kim, and I. H. Suh, "Probabilistic tourist trip-planning with time-dependent human and environmental factors," *IEEE Int. Conf. Big Data and Smart Computing*, pp. 505–508, 2016.
[50] P. J. Palomo-Martínez, M. Angélica Salazar-Aguilar, G. Laporte, and A. Langevin, "A hybrid variable neighborhood search for the Orienteering Problem with mandatory visits and exclusionary constraints," *Comput. Oper. Res.*, vol. 78, pp. 408–419, 2017.
[51] S. Rossi, F. Barile, C. Galdi, and L. Russo, "Artworks Sequences Recommendations for Groups in Museums," *Int. Conf. Signal-Image Technol. Internet-Based Syst.*, pp. 455–462, 2016.
[52] A. Mobasher, A. Ekici, and O. Ö. Özener, "Coordinating collection and appointment scheduling operations at the blood donation sites," *Comp. Ind. Eng.*, vol. 87, pp. 260–266, 2015.
[53] S. E. Butt and D. M. Ryan, "An optimal solution procedure for the multiple tour maximum collection problem using column generation," *Comput. Oper. Res.*, vol. 26, no. 4, pp. 427–441, 1999.
[54] G. Erdogan and G. Laporte, "The Orienteering Problem with Variable Profits," *Networks*, vol. 61, pp. 104–116, 2013.
[55] B. L. Golden, L. Levy, and R. Vohra, "The orienteering problem," *Naval Res. Logistics*, vol. 34, no. 3, pp. 307–318, 1987.
[56] A. Beham, J. Fechter, M. Kommenda, S. Wagner, S. M. Winkler, and M. Affenzeller, "Optimization Strategies for Integrated Knapsack and Traveling Salesman Problems," *Comp. Aided Syst. Theory*, pp. 359–366, 2015.
[57] P. Bolzoni and S. Helmer, "Hybrid Best-First Greedy Search for Orienteering with Category Constraints," *Advances in Spatial and Temporal Databases*, pp. 24–42, 2017.
[58] R. Bellman, "On a Routing Problem," *Quart. Appl. Math.*, vol. 16, no. 1, pp. 87–90, 1958.
[59] N. Christofides, "Worst-case analysis of a new heuristic for the travelling salesman problem.," *Symp. Algorithms Complex.*, vol. 1, no. 1, pp. 1–5, 1976.
[60] K. Helsgaun, "An effective implementation of the Lin–Kernighan traveling salesman heuristic," *Eur. J. Oper. Res.*, vol. 126, no. 1, pp. 106–130, 2000.
[61] S. Oßwald, M. Bennewitz, W. Burgard, and C. Stachniss, "Speeding-Up Robot Exploration by Exploiting Background Information," *IEEE Robotics and Automation Letters*, pp. 716 – 723, 2016.
[62] G. Grisetti, C. Stachniss, and W. Burgard, "Improved techniques for grid mapping with Rao-Blackwellized particle filters," *IEEE Trans. Robot.*, vol. 23, no. 1, pp. 34–46, 2007.
[63] R. Bellman, "Dynamic programming treatment of the travelling salesman problem". *J. Assoc. Comput. Machinery*, vol. 9, no. 1, pp. 61–63, 1962.
[64] L. Xia, C. C. Chen, and J. K. Aggarwal, "Human detection using depth information by Kinect," *Comput. Vis. Pattern Recognition Workshops*, pp. 15–22, 2011.
[65] S. Suzuki and K. Abe, "Topological structural analysis of digitized binary images by border following," *Comput. Vis. Graph. Image Process.*, vol. 30, no. 1, pp. 32–46, 1985.
[66] D. H. Douglas and T. K. Peucker, "Algorithms for the reduction of the number of points required to represent a digitized line or its caricature," *Canadian Cartographer*, vol. 10, no. 2, pp. 112-122, 1973.
[67] R. Lienhart and J. Maydt, "An extended set of Haar-like features for rapid object detection," *Int. Conf. Image Process.*, vol. 1, no. 1, pp. 900–903, 2002.
[68] B. Amos, B. Ludwiczuk, and M. Satyanarayanan, "Openface: A general-purpose face recognition library with mobile applications," *CMU School of Comp. Sci.*, pp. 1–20 2016.
[69] M. L. Walters, K. Dautenhahn, K. L. Koay, C. Kaouri, R. Boekhorst, C. Nehaniv, I. Werry, and D. Lee, "Close encounters: Spatial distances between people and a robot of mechanistic appearance," *IEEE-RAS Int. Conf. Humanoid Robots*, pp. 450–455, 2005.
[70] J. M. Santos, T. Krajník, and T. Duckett, "Spatio-temporal exploration strategies for long-term autonomy of mobile robots", *Robotics and Autonomous Systems*, pp. 116-126, 2017.

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

Sanjif Rajaratnam was an M.A.Sc. student in the Department of Mechanical & Industrial Engineering at UofT and a member of the ASBLab. His research interests include robotics and autonomous systems. He received his B.A.Sc. in Mechanical Engineering from the University of Waterloo.



Seung Tae Hong was a B.A.Sc. student in the Division of Engineering Science, Biomedical Systems option, at UofT and a member of the ASBLab. His research interests include biomechatronics and human-robot interaction.

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