Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence

Schedule-Based Robotic Search for Multiple Residents in a Retirement Home Environment

Markus Schwenk and **Tiago Vaquero** and **Goldie Nejat**
Autonomous Systems and Biomechatronics Laboratory
Department of Mechanical and Industrial Engineering, University of Toronto
5 King's College Road, Toronto, ON, Canada M5S 3G8
markus.schwenk@mail.utoronto.ca, {tvaquero, nejat}@mie.utoronto.ca

Abstract

In this paper we address the planning problem of a robot searching for multiple residents in a retirement home in order to remind them of an upcoming multi-person recreational activity before a given deadline. We introduce a novel Multi-User Schedule Based (M-USB) Search approach which generates a high-level-plan to maximize the number of residents that are found within the given time frame. From the schedules of the residents, the layout of the retirement home environment as well as direct observations by the robot, we obtain spatio-temporal likelihood functions for the individual residents. The main contribution of our work is the development of a novel approach to compute a reward to find a search plan for the robot using: 1) the likelihood functions, 2) the availabilities of the residents, and 3) the order in which the residents should be found. Simulations were conducted on a floor of a real retirement home to compare our proposed M-USB Search approach to a Weighted Informed Walk and a Random Walk. Our results show that the proposed M-USB Search finds residents in a shorter amount of time by visiting fewer rooms when compared to the other approaches.

1 Introduction

The health and quality of life of older adults living in long-term care facilities can be improved by these individuals engaging in stimulating recreational activities such as playing games, playing musical instruments, doing crossword puzzles, or reading (Menec 2003). These types of activities can delay age-related health decline (Bath and Deeg 2005) and prevent social isolation (Findlay 2003), which could potentially decrease the risk of dementia in elder adults (Wilson et al. 2007). However, the lack of these activities in eldercare facilities (PriceWaterCoopers LLP 2001) exists due to a shortage of healthcare workers (Sharkey 2008), which could be aggravated in the near future due to the rapid growth of the elderly population (Centre for Health Workforce Studies 2006). Socially assistive robots have been shown to be a promising technology to assist the elderly and to support caregivers in eldercare facilities (Oida et al. 2011; McColl, Louie, and Nejat 2013).

Our research focuses on the development of socially assistive robots that can autonomously organize and facilitate group-based recreational activities for the elderly. In this paper, we address the planning problem of a robot searching for multiple residents in a retirement home environment in order to invite and remind them of an upcoming multi-person recreational activity. The robot's objective is to maximize the number of residents it finds in a given time frame before the activity starts. During the search, the robot has to consider that the residents have their own schedules which contain appointments in different rooms of the environment, during which the residents are not always available for interaction with the robot. Based on these schedules, the robot also considers the order in which the residents have to be found to avoid searching for unavailable people. We introduce a novel Multi-User Schedule Based (M-USB) Search method which plans the robot's search for a set of non-static residents in a known, structured environment within a given time frame based on the residents' daily schedules.

Robotic search for people in structured environments has been investigated in the literature for different scenarios. For example, in (Elinas, Hoey, and Little 2003), the robot HOMER was designed to deliver messages one at a time to a particular person in a workspace environment. The robot stored a likelihood function for each person's location and performed a best-first search. The search consisted of visiting the nearest location to the robot based on the likelihood function for a particular person. If the person was not found, the robot iteratively visited other rooms until either the person was found or all regions had been visited. For these scenarios, a person was assumed to be at a static location in the environment. The search for multiple static targets in an indoor environment has been addressed in (Lau, Huang, and Dissanayake 2005). A dynamic programming approach was used to plan the search on a topological ordered graph, using a probability distribution which models the probability of meeting one of the targets in a given room at a given time. In (Tipaldi and Arras 2011), a robot's ability to blend itself into the workflows of people within human office environments was addressed. The authors developed spatial affordance maps to learn spatio-temporal patterns of human activities. A Markov Decision Process (MDP) Model was used to generate a robot's path through the environment to maximize the probability of encountering a person.

Uniquely our work addresses the robotic search problem of finding a specific set of multiple moving residents in a retirement home setting considering their individual daily schedules. Such schedule-based multi-user search for non-static people has not yet been addressed in the literature. To address this problem, we obtain spatio-temporal likelihood functions for every resident of the retirement home. We propose an approach to generate these resident likelihood functions as a composition of: 1) the residents' schedules, 2) direct observations by the robot in the environment, and 3) the layout of the environment. To do this, a weighting is applied to the above sources of information based on their time-dependent certainties of predicting a person's location. The main contribution of our work is the development of an MDP planner that uses a novel approach to compute the reward to determine the robot's search plan for finding multiple residents using: 1) the resident likelihood functions, 2) the availabilities of the individual residents, and 3) the order in which the residents should be found.

2 M-USB Search

The Multi-User Schedule Based Search presented in this work is a planning procedure that provides a plan $\mathcal{P}^*$ for the robot to find as many people as possible from a given set of target people in a given order within a defined time frame. The retirement home environment is defined to consist of several different regions which represent the topology of the building (e.g., rooms and corridors). The plan $\mathcal{P}^*$ will include a sequence of actions which model the whole search process. The possible actions are: *drive* which lets the robot travel from one region to another; *rest* which lets the robot rest for a short period of time; and *search* in which the robot executes a low-level search procedure in a specific region (e.g., frontier exploration, random walk). We use backwards induction to compute $\mathcal{P}^*$ based on a Markov-Decision Process which models the search using the aforementioned actions. To obtain a reward for this MDP, we setup a likelihood function for each person which models the probability that the person is in a specific region at a given time of the day. The likelihood functions of the individual residents are combined to generate a reward which respects the target residents' availability constraints (obtained from their schedules) and the order in which the residents should be found.

This paper will focus on the computation of the plan $\mathcal{P}^*$. This plan is to be executed by a mobile socially assistive robot that can navigate the environment as well as detect and recognize individual residents. We assume a local-search routine already exists on the robot which allows the robot to search for people in a given region. Once a person has been detected, the robot has to compute a new plan (replanning).

Problem Setup

Environment. We model the environment as a set of regions $\mathbb{R} \in \mathbb{R}^E$ (e.g. rooms, corridors, common areas) in which the search takes place. For each region a room-class is assigned, e.g. "Common Room", "Bedroom", "Corridor", or "Dining Hall". The room-classes depend on the activities residents engage in when they are in these regions. Each region can be represented as a polygon. The edges of this polygon can be marked as "crossable" if there is no physical border at an edge (e.g. if the edge represents a doorway). We define $neighbours(\mathbb{R})$ to be all regions which share a crossable edge with $\mathbb{R}$, i.e., a person or a robot can walk from $\mathbb{R}$ to any region $\mathbb{R}_n \in neighbours(\mathbb{R})$ without entering a third region. For each person $p \in \mathbb{P}$, where $\mathbb{P}$ is the set of all residents who are living in the environment, we assign one region of the environment as solely that person's: his/her private room. Figure 1(a) shows an example environment.

Schedules and Availability of Residents. For each person $p \in \mathbb{P}$ we consider a schedule which defines all his/her appointments on a given day. An appointment has a start time and an end time, and is assigned to a region $\mathbb{R}$ in which it takes place. We model the availability of each person $p \in \mathbb{P}$ as function $\beta_p(t)$ such that $\beta_p(t) = 1$ if p is available at time t and $\beta_p(t) = 0$ otherwise.

Search Query. When the robot receives a query q, it is to find a set of residents $p \in \mathbb{P}_q \subseteq \mathbb{P}$ within a given deadline t_{max}. Query q also specifies the order in which the residents should be found.

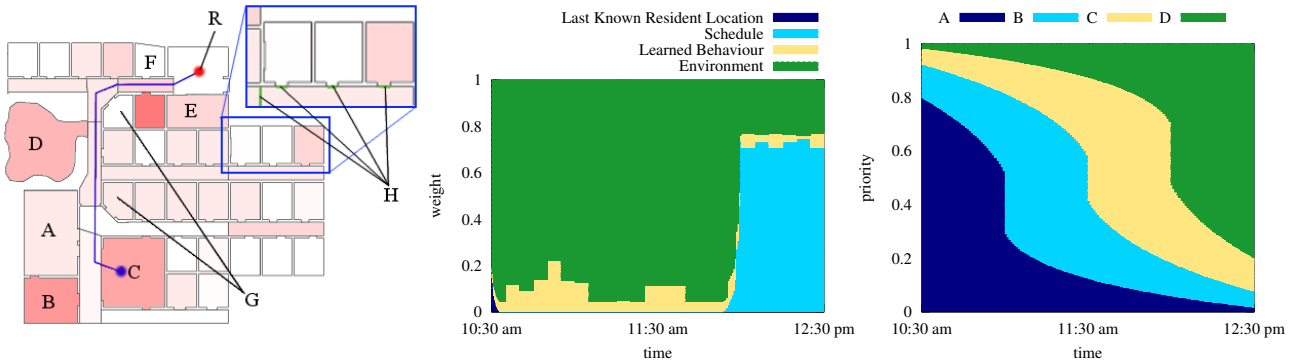
Setting up Resident Likelihood Functions

For each resident $p \in \mathbb{P}$ we can set up a likelihood function $\mathcal{L}(p, \mathbb{R}, t)$ which represents the probability that p is in region $\mathbb{R}$ at time t. This individual likelihood function is composed of four different likelihood functions $\mathcal{L}_k(p, \mathbb{R}, t)$, with $k \in \{s, lkrl, l, env\}$, which we will introduce in the following sections. As convention, we define $0 \leq \mathcal{L}_k \leq 1$ and $\sum_{\mathbb{R}} \mathcal{L}_k(p, \mathbb{R}, t) = 1$ for each person $p \in \mathbb{P}$ and each likelihood function $\mathcal{L}_k$. A value $\mathcal{L}_k(p, \mathbb{R}, t) = 1$ means that the person is in $\mathbb{R}$ at time t while $\mathcal{L}_k(p, \mathbb{R}, t) = 0$ indicates that the person cannot be in the region at this time.

Schedule Analyzer. We model $\mathcal{L}_s(p, \mathbb{R}, t)$ using the schedule of resident p. Assuming that with a probability of $0 \leq p_a \leq 1$ the person participates in an appointment defined in the person's schedule, we set $\mathcal{L}_s(p, \mathbb{R}, t) = p_a$ for the time frame of the appointment for the region assigned to this particular appointment and $\mathcal{L}_s(p, \mathbb{R}, t) = \frac{(1-p_a)}{|\mathbb{R}^E|-1}$ for all other regions. For any time t_k between two appointments where the last appointment ends at time t_{k-1} and the next appointment starts at t_{k+1}, we define $\alpha_{k-1} = \frac{t_{k+1}-t_k}{t_{k+1}-t_{k-1}}$ and $\alpha_{k+1} = \frac{t_k-t_{k-1}}{t_{k+1}-t_{k-1}}$. If there is no next appointment, we set $\alpha_{k+1} = 0$ and $\alpha_{k-1} = 1$. If there is no previous appointment, the values become $\alpha_{k+1} = 1$ and $\alpha_{k-1} = 0$. We can then define the value of the likelihood function to be:

$$\begin{aligned}\mathcal{L}_s(p, \mathbb{R}, t_k) = \\ \alpha_{k-1} \cdot \sum_{\mathbb{R}' \in \mathbb{R}^E} \mathcal{L}_s(p, \mathbb{R}', t_{k-1}) \cdot p_m(\mathbb{R}', \mathbb{R}, p, t_k - t_{k-1}) + \\ \alpha_{k+1} \cdot \sum_{\mathbb{R}' \in \mathbb{R}^E} \mathcal{L}_s(p, \mathbb{R}', t_{k+1}) \cdot p_m(\mathbb{R}, \mathbb{R}', p, t_{k+1} - t_k)\end{aligned} \tag{1}$$

at time t_k. $p_m(\mathbb{R}_1, \mathbb{R}_2, p, \Delta t)$ is the probability that person p has moved from region $\mathbb{R}_1$ to region $\mathbb{R}_2$ in the time frame of



(a) Example Scenario and Simulation Environment Setup

(b) Weights for one Person

(c) A Priority Function

Figure 1: (a) The map of the simulated retirement home with the rooms: dining hall (A), games room (B), TV-room (C), garden (D), nurse station (E), family visit room (F) and shower rooms (G). All other rooms are private rooms and corridors. The shading of the regions indicates the current reward for each region (dark: high, light: low) of an example scenario. The current generated plan has the robot (R) driving to the TV-room (C) where it starts a local search. (H) shows the crossable edges (doors) in the scaled portion of the environment. (b) Weights for one person. At 12 pm the person has a one-hour appointment which gives the schedule a higher weight at this time. (c) Priority function for four residents who are all available in the considered time frame.

Δt. The value can be obtained from the person's speed and the distance between the two regions.

Last Known Resident Location. To be able to take into account when the robot last detected a person earlier that day who it is currently searching for, we set up a database which stores the time t_d^p and region $\mathbb{R}_d^p$ of the last detection of the person p. Using a simple motion model of the resident, this information can be used to generate $\mathcal{L}_{lkrl}(p, \mathbb{R}, t)$. As done for the Schedule Analyzer, we define the probability $p_m(\mathbb{R}_1, \mathbb{R}_2, p, \Delta t)$ that person p has moved from region $\mathbb{R}_1$ to region $\mathbb{R}_2$ in a time frame of Δt. We can then define:

$$\mathcal{L}_{lkrl}(p, \mathbb{R}, t) = p_m(\mathbb{R}_d^p, \mathbb{R}, p, t - t_d^p). \tag{2}$$

If the person has not been previously detected, we assign a uniform distribution $\mathcal{L}_{lkrl}(p, \mathbb{R}, t) = \frac{1}{|\mathbb{R}^E|}$.

Learned Behaviour. A person's behaviour, which is not defined in the schedule (e.g., a person often takes a walk in the garden after lunch) but which has been learned by the robot based on its observations is also stored by the robot as $\mathcal{L}_l(p, \mathbb{R}, t)$. $\mathcal{L}_l(p, \mathbb{R}, t)$ is obtained by evaluating the frequency of the event "person p has been detected in region $\mathbb{R}$ at time t" and is initialized as a uniform distribution.

Environment. The topology of the environment can be used to generate $\mathcal{L}_{env}(p, \mathbb{R}, t)$. Assuming that a person will spend most of her/his spare time either in her/his private room or in the common rooms, we model this likelihood function in a way that it assigns a higher value to the common rooms and the person's private room than to the other regions.

Pre-computation. Since the schedules, the topology of the environment, and the learned behaviour remain the same once they are provided to the robot at the beginning of a day, $\mathcal{L}_s(p, \mathbb{R}, t)$, $\mathcal{L}_l(p, \mathbb{R}, t)$, and $\mathcal{L}_{env}(p, \mathbb{R}, t)$ can be computed before a search query is received. $\mathcal{L}_{lkrl}(p, \mathbb{R}, t)$ is computed dynamically when the query is received.

Combining the Likelihood Functions for one Person. The four likelihood functions $\mathcal{L}_k(p, \mathbb{R}, t)$ can be combined to generate $\mathcal{L}(p, \mathbb{R}, t)$. As the certainties with which the four likelihood functions can predict a resident's location differ (e.g., the Last Known Resident Location will have high uncertainty when the person has not been detected for several hours, and the Schedule Analyzer will have high certainty when the person has an appointment), a weighting function can be used. The certainty of one likelihood function $\mathcal{L}_k(p, \mathbb{R}, t)$ can be represented by its variance:

$$\mathrm{Var}(\mathcal{L}_k(p, \mathbb{R}, t)) = \frac{1}{|\mathbb{R}^E|} \cdot \sum_{\mathbb{R} \in \mathbb{R}^E} \left[\mathcal{L}_k(p, \mathbb{R}, t) - \frac{1}{|\mathbb{R}^E|} \right]^2. \tag{3}$$

For each likelihood function $\mathcal{L}_k$, we introduce the weight w_k at time t:

$$w_k(t) = \frac{\mathrm{Var}(\mathcal{L}_k(p, \mathbb{R}, t))}{\sum_k \mathrm{Var}(\mathcal{L}_k(p, \mathbb{R}, t))} \tag{4}$$

where $\sum_k w_k(t) = 1$. The final combined likelihood function is defined to be:

$$\mathcal{L}(p, \mathbb{R}, t) = \sum_k w_k(t) \cdot \mathcal{L}_k(p, \mathbb{R}, t). \tag{5}$$

Figure 1(b) shows an example of the four weights.

Modelling the Transition System

The objective is to find a sequence of actions the robot should execute in order to find as many persons as possible

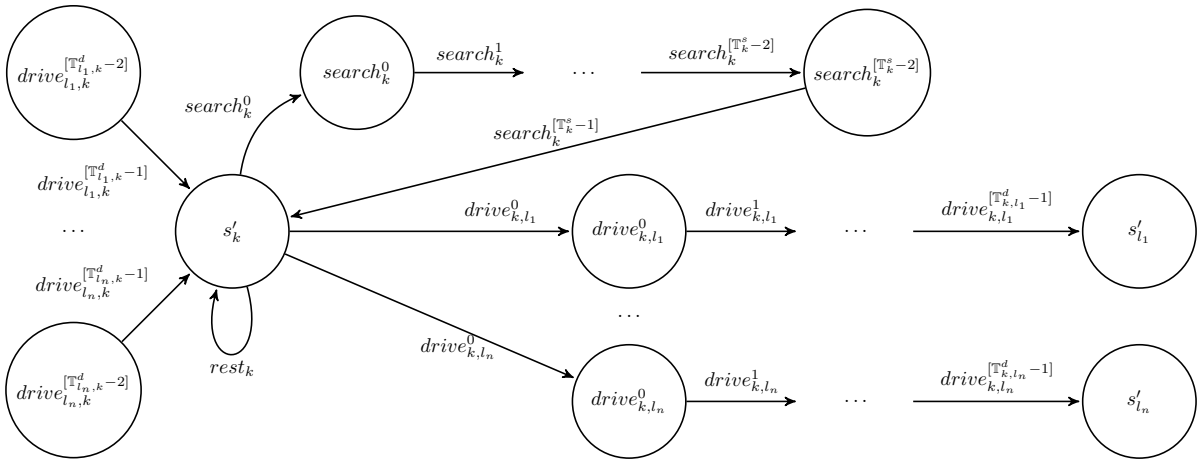


Figure 2: The transition system for an arbitrary region k with neighbours $l_1, \ldots, l_n$. Being in state s'_k, the robot can either *rest* one time step, start a full *search* sequence or a *drive* sequence to one of the neighbouring regions l_i with $i = 1, \ldots, n$ which leads to state s'_{l_i}. *Search* sequences and the *rest* action in region k lead to s'_k.

in $\mathbb{P}_q$ within the given deadline t_{max}. We model the search as a Markov Decision Process. We discretize time using time steps of duration Δt which is the execution time for each individual action. The M-USB Search is modelled to consist of three possible action sequences the robot can perform in each region: 1) $rest$ in which the robot rests for one time step, 2) $search$ in which the robot performs a local search within the region, and 3) $drive$ in which the robot drives to one of the neighbouring regions. Since the time it takes to perform a local search within a region depends on the geometry of the region, we introduce $\mathbb{T}_k^s$ to represent the number of time steps Δt a search within region $\mathbb{R}_k$ takes. For region $\mathbb{R}_l \in neighbours(\mathbb{R}_k)$, we define $\mathbb{T}_{k,l}^d$ as being the number of time steps the transition between $\mathbb{R}_k$ and $\mathbb{R}_l$ takes.

The states $s \in \mathbb{S}$ of the MDP represent the states of the robot, which depend on the region the robot is in and the current action sequence it is performing. We define the following sets of states $\mathbb{S}_k^s$ and $\mathbb{S}_k^d$ to contain all states during the $search$ and $drive$ sequences for each region $\mathbb{R}_k \in \mathbb{R}^E$:

1. $\mathbb{S}_k^s = \{search_k^t\}$ with $0 \leq t < \mathbb{T}_k^s - 1$, and
2. $\mathbb{S}_k^d = \bigcup_{\mathbb{R}_l} \left\{drive_{k,l}^t\right\}$ with $0 \leq t < \mathbb{T}_{k,l}^d - 1$, $\mathbb{R}_l \in neighbours(\mathbb{R}_k)$.

In addition to the aforementioned states within the $search$ and $drive$ sequences, we define the state s'_k for each region $\mathbb{R}_k$ as the robot's state: 1) after a region has been entered by any drive sequence, 2) after the rest action has been executed, 3) after the full search sequence of $\mathbb{R}_k$ has been executed, or 4) when the robot is creating a new plan when being in $\mathbb{R}_k$. The set of all possible states for region $\mathbb{R}_k$ is defined to be:

$$\mathbb{S}_k = \{s'_k\} \cup \mathbb{S}_k^s \cup \mathbb{S}_k^d. \tag{6}$$

We define the following robot actions α for each region $\mathbb{R}_k \in \mathbb{R}^E$:

1. $\mathbb{A}_k^r = \{rest_k\}$,
2. $\mathbb{A}_k^s = \{search_k^t\}$ with $0 \leq t < \mathbb{T}_k^s$, and
3. $\mathbb{A}_k^d = \bigcup_{\mathbb{R}_l} \left\{drive_{k,l}^t\right\}$ with $0 \leq t < \mathbb{T}_{k,l}^d$, $\mathbb{R}_l \in neighbours(\mathbb{R}_k)$.

For each region $\mathbb{R}_k$, the set of all possible actions is:

$$\mathbb{A}_k = \mathbb{A}_k^r \cup \mathbb{A}_k^s \cup \mathbb{A}_k^d. \tag{7}$$

Transitions between the states describe how a robot state changes when it performs a particular action. In particular, the successor $succ(\alpha)$ of action α is defined as the state which follows α. The overall transition system is shown in Figure 2.

For each action a time-dependent reward $R(\alpha, t)$ is assigned. This reward is evaluated to compute the plan $\mathcal{P}^*$ and depends on the region in which the action is performed. Therefore, we define a region to each action: $\mathbb{R} = region(\alpha)$. For each action α in $\mathbb{A}_k^r$ and $\mathbb{A}_k^s$, we define $region(\alpha) = \mathbb{R}_k$. For each pair of neighbouring regions $\mathbb{R}_k$ and $\mathbb{R}_l$, we define $\mathbb{T}_{k,l}^{d,cross}$ to be the number of time steps after which the region $\mathbb{R}_l$ is entered during a drive sequence from $\mathbb{R}_k$ to $\mathbb{R}_l$. We then define $region(\alpha) = \mathbb{R}_k$ if $t < \mathbb{T}_{k,l}^{d,cross}$ and $region(\alpha) = \mathbb{R}_l$ if $t \geq \mathbb{T}_{k,l}^{d,cross}$ for each drive action α.

Attractivities of Actions. Assuming that during a search the probability that a person who is in the searched region is detected is different from the probability that a person is detected while the robot is just driving between two regions or resting in one region, we define an attractivity $\delta(\alpha)$ with $0 \leq \delta(\alpha) \leq 1$ for each action α. This attractivity depends on the geometry of the region $\mathbb{R} = region(\alpha)$. We define the attractivity $\delta(\alpha_s) = \mathcal{A}(region(\alpha_s))^{-1}$ for each search action α_s with $\mathcal{A}(region(\alpha_s))$ being the area of $region(\alpha_s)$. We define $\delta(\alpha_r) = a \cdot \delta(\alpha_s)$ with $0 < a \leq 1$ for each rest action and $\delta(\alpha_d) = b \cdot \delta(\alpha_s)$ with $0 < b \leq 1$ for each drive action. If the robot detects residents with higher probability while driving then $b > a$ holds; $a > b$ holds otherwise.

Modelling the Order and Availability of Residents

The order in which the residents should be found is given and has been obtained from the persons' schedules to avoid searching for unavailable residents. The robot should try to keep this order if possible. However, if the robot can maximize the number of people found by changing the order, it can also do so. To search for the residents $p \in \mathbb{P}_q$ in the given order we introduce a priority function $\pi_p(t)$ with $0 \leq \pi_p(t) \leq 1$ for each person $p \in \mathbb{P}_q$ and apply the following constraints:

$$\sum_{p \in \mathbb{P}_q} \pi_p(t) \cdot \beta_p(t) = 1 \; \forall t \tag{8}$$

and

$$\int_{t_0}^{t_{max}} \pi_p(t) \cdot \beta_p(t) \, dt = \frac{t_{max} - t_0}{|\mathbb{P}_q|} \;\; \forall p. \tag{9}$$

We model $\pi_p(t)$ to provide a high priority in the time interval assigned to the resident p based on the given order. However, to allow the robot to search for other residents $p' \in \mathbb{P}_q$ during this time interval, we allow $\pi_{p'}(t) \neq 0$ when $\beta_{p'}(t) = 1$. Figure 1(c) shows such a priority function for four people A, B, C, and D to be searched in this order.

Finding $\mathcal{P}^*$

In order to find the set of residents within the given deadline, we define a reward for each action of the MDP model of the search. The reward is based on the resident likelihood functions and the availabilities of the residents in $\mathbb{P}_q$, the aforementioned priority functions and the attractivities:

$$R(\alpha, t) = \delta(\alpha) \cdot \sum_{p \in \mathbb{P}_q} \pi_p(t) \cdot \beta_p(t) \cdot \mathcal{L}(p, region(\alpha), t). \tag{10}$$

Since a deadline is given, the search evolves to be a finite horizon MDP which can be solved using simple backwards induction (Tipaldi and Arras 2011). In particular, the utility $U_t(s)$ is evaluated for each possible state s at time t using the Bellman equation:

$$U_t(s) = \max_{\alpha} [R(\alpha, t) + \gamma \cdot U_{t+1}(succ(\alpha))] \tag{11}$$

where α is any action that can be taken from s and γ is a factor with $0 \leq \gamma \leq 1$ which provides a weighting for the relation between the importance of rewards which are earned in the near and in the far future. A policy $\Pi_t(s)$ can be identified which assigns the action α which has to be taken in order to maximize the collected reward given t and s:

$$\Pi_t(s) = \arg\max_{\alpha} [R(\alpha, t) + \gamma \cdot U_{t+1}(succ(\alpha))]. \tag{12}$$

For the last time step we initialize the utilities to be 0 for each state. A plan can be obtained given both a policy and a state s_0, where s_0 is the initial state the robot is in when planning. We set $s_0 = s'_k$ with $\mathbb{R}_k$ being the region the robot is currently in. To avoid endless search loops, we use a greedy approach for the plan generation by introducing a number of new regions k for the robot to search prior to the robot searching a region again. We set the rewards for a search action α to be zero when $region(\alpha)$ will be contained within the next k regions that will be searched when taking action α.

3 Simulated Experiments

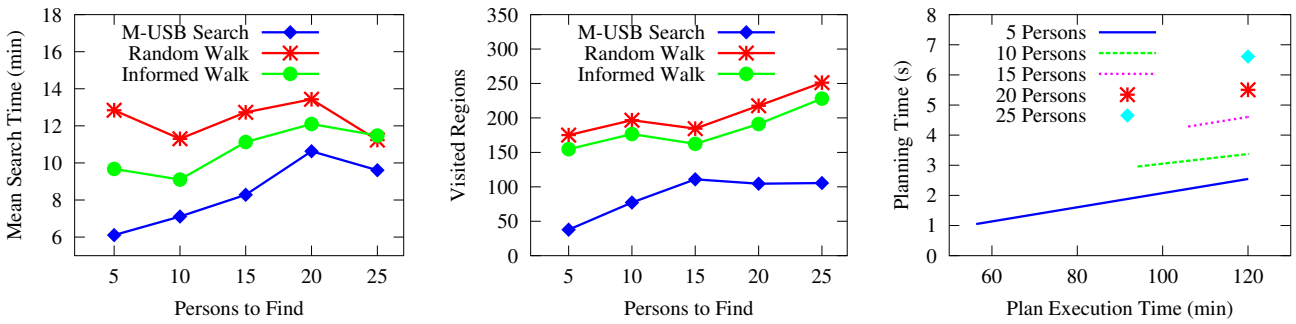
Simulation Setup

To test the performance of the M-USB Search, we use a simulator we have developed to simulate a robot in a realistic retirement home environment. The simulation was executed on a Ubuntu machine with an AMD A10-5700 Processor and 12GB RAM.

Simulation Environment. We created a map of a floor in a retirement home with 25 residents. The map consists of the residents' private rooms, two common rooms (TV-Room and Games Room), one Dining Hall, two Shower rooms, one Nurse Station, one Room for Family visits, and an outdoor Garden. All residents have their own unique schedules for the day. These schedules contain three meal times, breakfast (8 am-9 am), lunch (12 pm-1 pm), and dinner (6 pm-7 pm) during which the residents are available for the robot to interact with them. In addition, each schedule includes one 1-hour activity during which the residents are also available for interaction (e.g., walk and reading) and 2 to 4 appointments during which they must not be disturbed (e.g., doctor's visit). In his/her spare time, each resident visits random rooms at random times. A probability of $p_{miss} = 0.1$ is given for the residents not participating in their scheduled activities and behaving as if they have spare time. The map used for these experiments is shown in Figure 1(a).

Performance Comparison. We compare the performance of our M-USB Search to both a *Weighted Informed Walk* and a *Random Walk* approach for the problem of a robot finding a group of residents within a deadline in the retirement home setting in order to remind them of an upcoming group-based recreational activity. The robot uses a speed of $v = 0.6\,\mathrm{m/s}$ and can detect all people in a range of $r = 1.8\,\mathrm{m}$ with respect to the robot. The investigated search algorithms are:

1. **Random Walk.** The robot chooses a random room in the map, drives to this room, and starts a local search in the room. This is repeated until all target residents are found or until the deadline is reached.
2. **Weighted Informed Walk.** Similar to the Random Walk, the Weighted Informed Walk algorithm picks a random room, drives there and starts a search in this room. However, a higher weighting is given to a resident's private room and common rooms. Namely, a weighting technique is applied to identify the importance of the regions accordingly to their room-classes. The algorithm also considers the last k regions it has searched and does not search them again before $k = 4$ other regions have also been searched.
3. **M-USB Search.** The proposed M-USB Search is used with a time discretization of $\Delta t = 10\,\mathrm{s}$. The schedule analyzer uses $\Delta t = 30\,\mathrm{s}$ and the database in which the robot saves the learned behaviour operates with $\Delta t = 300\,\mathrm{s}$. The attractivities for the actions are $\delta(\alpha_d) = 0.9 \cdot \delta(\alpha_s)$ and $\delta(\alpha_r) = 0.7 \cdot \delta(\alpha_s)$. For Eqs. (11) and (12) $\gamma = 0.99$ is used. For the greedy M-USB Search the reward of an action α is set to zero when the room $region(\alpha)$ is contained in the next 4 searched regions.



(a) Mean Search Times (b) Number of Visited Regions (c) Planning Time

Figure 3: Comparison Results: (a) mean search time per person, (b) number of visited regions during the search and (c) planning time needed to compute a plan when receiving a query.

Local Search in a Region. As our focus in this paper is on the high-level search to regions, for this comparison all aforementioned search approaches used the same random walk local search approach when they were searching within the region. The search time in the individual rooms was set to one second per squared meter.

The Search Queries. Each search approach was tested with ten different search queries q, which consisted of a robot finding $N = |\mathbb{P}_q|$ residents with $N = 5, 10, 15, 20, 25$ during two separate times of the day. The start times were chosen such that the robot searched for residents in a time frame encompassing cases where residents had appointments, activities and spare time. For all searches, the robot started in the Games Room and had a deadline of 2 hours to complete a search. Each query was repeated 5 times per search approach. The time t_0 indicates the time when the query was received.

Search Performance and Runtime

The performance metrics for the comparison are the mean search time per person and the number of visited regions during the search procedure. We also measure the pre-computation time T_p needed at software start-up to create the MDP model and load the learned behaviour, and set up the three likelihood functions $\mathcal{L}_s$, $\mathcal{L}_l$, and $\mathcal{L}_{env}$. This time is independent of the number of residents to be found, since $\mathcal{L}_s$, $\mathcal{L}_l$, and $\mathcal{L}_{env}$ are computed for every person. We also measure the planning times T_N^d for the single plans which are computed when a query is received. N is the number of residents to be found and d is the duration of the resulting plan which is the time we are looking into the future. For the initial plan this time is two hours. We include plans generated with re-planning the search when a person has been found.

Results and Discussion

The comparison results are presented in Figure 3. Figure 3(a) shows that our M-USB Search is the fastest search approach, while the Random Walk is the slowest. The results show that the use of the persons' schedules, learned behaviours as well as the topology of the environment, and the attempt to keep the search order in the proposed approach lowers the mean search times. The Weighted Informed Walk has better results than the Random Walk due to the consideration of the layout of the environment but requires more time to find the residents than the M-USB Search. The Random Walk does not use any prior knowledge and therefore is the slowest approach. Figure 3(b) shows that the number of visited regions during the search is also the lowest for the M-USB Search compared to the two other approaches. The measured mean pre-computation time during system start-up for our approach was $T_p = 40.82$ s. The planning time needed when a query was received can be seen in Figure 3(c). It is linear for the number of residents for which the plan has to be generated since $\mathcal{L}_{lkrl}$ and the reward have to be computed N times. The time is also linear for the plan execution time since the reward and the policy have to be computed for every time step during backwards induction. In general, the pre-computation times are very short (e.g. we measure a mean of 7 s for $N = 25$ and $d = 120$).

4 Conclusion

In this paper we address the problem where a robot is searching for multiple non-static residents within a retirement home environment. We have developed the M-USB Search planning procedure which generates a high-level-plan to maximize the number of residents that are found within a given time frame. We obtain spatio-temporal likelihood functions for the individual residents using the schedules of the residents, the layout of the retirement home environment as well as direct observations by the robot. The M-USB search method uses a novel approach to compute the reward to determine the robot's search plan for finding multiple persons. We have compared our M-USB search method to a Weighted Informed Walk search and a Random Walk search for the proposed problem. Our results showed that the M-USB Search can find the residents in a shorter amount of time by visiting a fewer number of rooms.

Acknowledgments

This research has been funded by a Natural Sciences and Engineering Research Council of Canada (NSERC) Collaborative Research and Development Grant and by Dr Robot Inc. The first author has been funded by the German Academic Exchange Service (DAAD) PROMOS program. The authors would like to thank Prof. Kai O. Arras for helping the first author to undertake his studies at the University of Toronto.

References

Bath, P. A., and Deeg, D. 2005. Social engagement and health outcomes among older people: introduction to a special section. *European Journal of Ageing* 2(1):24–30.

Centre for Health Workforce Studies. 2006. The Impact of the Aging Population on the Health Workforce in the United States: Summary of Key Findings.

Elinas, P.; Hoey, J.; and Little, J. J. 2003. HOMER: Human Oriented MEssenger Robot. *AAAI Spring Symposium on Human Interaction with Autonomous Systems in Complex Environments, Stanford CA*.

Findlay, R. A. 2003. Interventions to reduce social isolation amongst older people: where is the evidence? *Ageing and Society* 23(5):647–658.

Lau, H.; Huang, S.; and Dissanayake, G. 2005. Optimal search for multiple targets in a built environment. In *IROS*, 3740–3745. IEEE.

McColl, D.; Louie, W.-Y. G.; and Nejat, G. 2013. Brian 2.1: A Socially Assistive Robot for the Elderly and Cognitively Impaired. *IEEE Robot. Automat. Mag.* 20(1):74–83.

Menec, V. H. 2003. The relation between everyday activities and successful aging: A 6-year longitudinal study. *The Journals of Gerontology Series B: Psychological Sciences and Social Sciences* 58(2):S74–S82.

Oida, Y.; Kanoh, M.; Inagaki, M.; Konagaya, Y.; and Kimura, K. 2011. Development of a Robot-Assisted Activity Program for Elderly People Incorporating Reading Aloud and Arithmetic Calculation. In *Asian Perspectives and Evidence on Health Promotion and Education*. Springer. 67–77.

PriceWaterCoopers LLP. 2001. Report of a Study to Review Levels of Service and Responses to Need in a Sample of Ontario Long Term Care Facilities and Selected Comparators.

Sharkey, S. 2008. People caring for people impacting the quality of life and care of residents of long-term care homes: a report of the independent review of staffing and care standards for long-term care homes in Ontario.

Tipaldi, G. D., and Arras, K. O. 2011. I want my coffee hot! Learning to find people under spatio-temporal constraints. In *ICRA*, 1217–1222. IEEE.

Wilson, R. S.; Krueger, K. R.; Arnold, S. E.; Schneider, J. A.; Kelly, J. F.; Barnes, L. L.; Tang, Y.; and Bennett, D. A. 2007. Loneliness and Risk of Alzheimer Disease. *Arch Gen Psychiatry* 64(2):234–240.