Human-Robot Collaborative Topological Exploration for Search and Rescue Applications

Vijay Govindarajan, Subhrajit Bhattacharya and Vijay Kumar

Abstract We address the coordination between humans and robots in tasks that involve exploration and reconnaissance with applications to search and rescue. Specifically, we consider the problem of humans and robots cooperatively searching an indoor environment in a distributed manner where we assume that each robot is equipped with sensors that are able to locate targets of interest. Rather than have humans issue explicit commands to and guide robots, we allow humans to make decisions on their own and let the robots adapt to decisions taken by the human. The main contribution of this paper is a framework in which the robots in the team respond and adapt to the behavior of the human agents in the task of exploring and clearing an indoor environment. The central idea is the assignment of robots to homotopy classes that are complementary to the classes being pursued by human agents. By the virtue of the sparse topological representation of the agent trajectories, our algorithm lends itself naturally to a distributed implementation. The framework has three advantages: it (a) ensures that robots and humans pursue different homotopy classes; (b) requires very little communication between the humans and the robots; and (c) allows robots to adapt to human movement without having to model complex human decision-making behaviors. We demonstrate the effectiveness of the proposed algorithm through a distributed implementation on a ROS (Robot Operating System) platform.

Keywords Search and rescue · Path planning · Homotopy · Human-robot interaction

V. Govindarajan (✉)
Department of Electrical Engineering, University of Pennsylvania, Philadelphia, USA
e-mail: govvijay@seas.upenn.edu

S. Bhattacharya
Department of Mathematics, University of Pennsylvania, Philadelphia, USA
e-mail: subhrabh@math.upenn.edu

V. Kumar
Department of Mechanical Engineering and Applied Mechanics,
University of Pennsylvania, Philadelphia, USA
e-mail: kumar@seas.upenn.edu

N.-Y. Chong and Y.-J. Cho (eds.), *Distributed Autonomous Robotic Systems*,
Springer Tracts in Advanced Robotics 112, DOI 10.1007/978-4-431-55879-8_2

1 Introduction

We address the coordination between humans and robots in tasks that involve reconnaissance with applications to search and rescue. In these applications, robots may need to quickly and safely explore environments in collaboration with human counterparts. When confronted with two or more hallways, a human first responder may choose to explore one hallway, while his/her robot co-responder explores a different hallway. Similarly, in teams of multiple human and robot explorers, we want the exploration task to be naturally decomposed between the team members. At the same time, we want the human(s), who are better at interpreting the available information and at decision making, to decide what actions they want to take and let robots adapt accordingly.

We consider a setting where humans and robots are equipped with similar sensing, processing, and communication capabilities, so that robots and humans can be aware of each others' positions and robots can interpret human movements and intentions. The sensing capabilities of the agents are assumed to provide adequate range to detect anomalies (e.g., victims or intruders) in an environment. We assume that both humans and robots have access to blueprints of buildings and are thus aware of the major features in the environment. As a result, both humans and robots are able to localize themselves with respect to features in the buildings using onboard sensors such as laser scanners, cameras and IMUs, as well as GPS, if available. Finally, we assume that the human-worn computers are able to communicate with the robot co-workers and share their estimates of current location periodically.

Metric-based multi-robot coordinated exploration have been studied widely in the past [1–5]. Multi-robot coverage of environments are also fundamentally considered as metric problems relying inherently on a metric on the configuration space [6–10]. In addition, graph traversal approaches similar to the *traveling salesman* problem [11] have been explored in context of room-clearing [12] and pursuit-evasion problems [13]. Similar coverage problems can be formulated as a traversal on *Voronoi graph* or *topological map* [14, 15] of an environment.

However, in a problem setting as ours, it is likely that maps may not be perfect. Noise in the process dynamics and observations will induce errors in localization. Thus representations derived from metric information will require estimation techniques that yield estimates of states that are stochastic. Topological invariants such as homotopy, on the other hand, being robust towards environmental noise and measurement errors, are suitable for communication and coordination among the heterogeneous teams of humans and robots. Furthermore, our primary objective being quick information/intelligence gathering and clearing, it is not necessary that the agents visit every point in the environment (as done in graph traversal algorithms). Homotopy classes of trajectories form natural equivalence classes, within each of which the information available are similar. If two trajectories belong to the same homotopy class, then a single agent can perform the task of gathering the available information in that class, while diverting additional resources to other classes. While homotopy is directly related to visibility in most indoor environments (e.g. consisting mostly of hallways and corridors), even in presence of non-convex features within the class (e.g. a room by the corridor), traversing the class is sufficient to gather intelligence and information from the adjacent features as those (e.g., by glancing through the doors of the rooms) and does not need dedicated agents for each of those features. Another advantage of using homotopy classes as the primary means of decision making in coordinated reconnaissance and clearing is the simplicity of its representation, and thus the ease and efficiency in communication. Choosing *complementary homotopy classes* by the robot agents is achieved naturally and efficiently, and such choices can be easily adapted to change in the human actions. Such algorithmic simplicity is absent in graph traversal approaches.

There has been some recent research on using topological techniques in exploration of environments [16]. In this paper our fundamental approach is topological as well. We exploit topological features in the environment, namely the different *homo-*

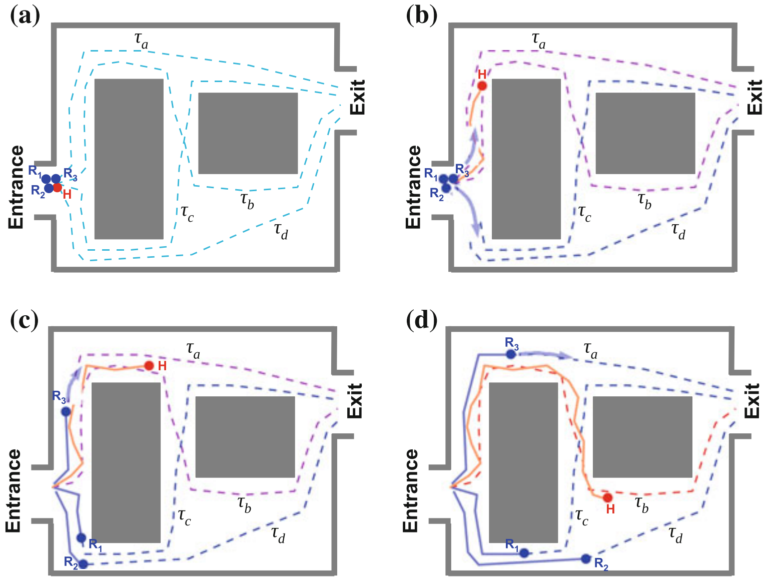

Fig. 1 A simple illustration of the idea behind the proposed algorithm involving a team of one human and three robots. Robots must respond to human action by choosing paths in homotopy classes complementary to those taken by humans to maximize exploration. The algorithm takes into account teams consisting of arbitrary number of humans and autonomous robots. In addition, robots can effectively adapt to erratic/unpredictable human behavior (not illustrated in this figure), where a human, after committing to a class, may turn back to choose a different homotopy class. **a** Four agents, three robots (R_1, R_2, R_3) and one human (H), enter an environment with 4 homotopy classes of paths (τ_a, τ_b, τ_c, τ_d) leading to the exit. The robots wait for the human to move first. **b** Based on human's initial trajectory (*solid curve*), the robots infer that H is taking the homotopy class of τ_a or τ_b. The homotopy classes of τ_c and τ_d are thus to be taken by robots. **c** R_3 tailgates the human (to pursue τ_a or τ_b—whichever not taken by H), while robots R_1 and R_2 commit to τ_c and τ_d. **d** H has committed to τ_b, and thus R_3 commits to homotopy class of τ_a

topy classes of trajectories, to guide our search and rescue missions. This topological reasoning is fundamental in deciding how the autonomous agents respond to human behaviors. Although we do use a metric on the space of trajectories in the workspace (*Hausdorff distance*), this is purely as an intermediate step towards classifying a human's trajectory into one or more of the homotopy classes.

Our algorithm design is inspired by the need to keep explicit human-robot communication (e.g., human commanding robots) at minimum in a time-critical mission such as search and rescue. The humans should have the freedom to choose actions based on their superior sensing ability (e.g., audio cues) and change actions without having to explicitly communicate intent to other agents. The robot agents should be able to adapt to the human actions and choose complementary tasks to maximize efficiency of coordinated survey and search. We illustrate the problem at hand using the scenario in Fig. 1 where there is one human and three autonomous robots. All the agents enter through a single entrance in the environment and need to clear the building and reach the exit. The figures illustrate how the robots take decisions and respond based on the human agent's actions. Our proposed algorithm is highly suited for a distributed implementation, requiring only limited inter-agent communication of coarse topological representation (h-signature) of their trajectories.

2 Background

In this section, we will review some preliminary definitions and algorithmic tools.

2.1 Homotopy Class of Trajectories

Suppose $W \subseteq \mathbb{R}^2$ is a simply-connected workspace for the agents with m counts of connected obstacles $O_1, O_2, \ldots, O_m \subseteq W$. Trajectories in an environment can be classified by their topologies into different homotopy classes, which arise from the presence of obstacles in an environment. We start by reviewing some of the standard definitions related to homotopy.

Definition 1 (*Homotopy classes of curves* [17]) Two curves $\gamma_1, \gamma_2 : [0, 1] \rightarrow (W - \mathscr{O})$ connecting the same start and end points, are homotopic (or belong to the same *homotopy class*) iff one can be continuously deformed into the other without intersecting any obstacle (see Fig. 2a. See [17, 18] for formal definitions).

For curves in 2-dimensional plane punctured by obstacles, computation of the homotopy class of a given curve can be performed in a relatively simple way [18–22]: We consider *representative points*, ζ_i, inside the ith obstacle O_i [17], and *parallel non-intersecting* rays, $r_1, r_2, \ldots, r_m$, emanating from the obstacles (Fig. 2b). If γ is a given curve whose homotopy class we are trying to identify, we construct

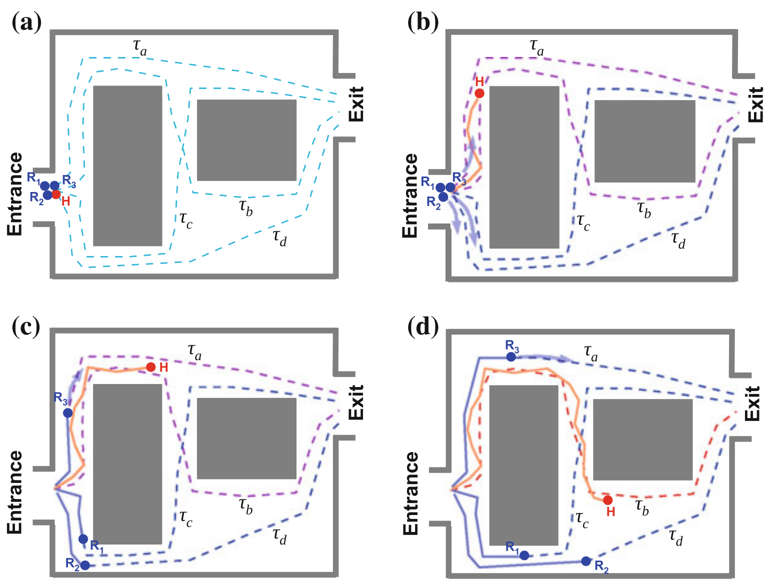

Fig. 2 Homotopy classes and their word representation. **a** γ_1 is homotopic to γ_2 since there is a continuous sequence of trajectories representing deformation of one into the other, but not to γ_3 since it cannot be continuously deformed into any of the other two. **b** ζ_i are representative points inside the obstacles, $O_1, O_2, \ldots, O_m$ (in that order), and r_i, $i = 1, \ldots, m$ are rays emanating from the respective points. The homotopy invariant of this curve γ is $h(\gamma) =$ "$r_2 r_3 r_5 r_6^{-1}$"

a *word* by tracing γ, and consecutively placing the letters of the rays that it crosses, with a superscript of '$+1$' (assumed implicitly) if the crossing is from left to right, and '-1' if the crossing is from right to left. Thus, for example, the word for γ in Fig. 2b will be "$r_2 r_3 r_4 r_4^{-1} r_5 r_6^{-1}$". We then *reduce* this word by canceling the same letters that appear consecutively but with opposite superscript signs. Thus, the word for γ in Fig. 2b can be reduced to "$r_2 r_3 r_5 r_6^{-1}$". This reduced word representation is a *homotopy invariant* for open curves (with fixed end points), γ, and we will write this as $h(\gamma)$ and call it the "h-signature of γ". The h-signature (reduced word) uniquely identifies the homotopy class of a curve. Note that if the end point of γ coincides with the start point of γ', then $h(\gamma \cup \gamma') = h(\gamma) \circ h(\gamma')$ (where '$\circ$' indicate concatenation of words).

2.2 *h-augmented Graph*

We use a discrete representation of the workspace, W, and construct a graph, G (with vertex set $\mathscr{V}(G)$ and edge set $\mathscr{E}(G)$), by placing a vertex in every accessible discrete cell (cells not intersecting with an obstacle) and by establishing an edge between

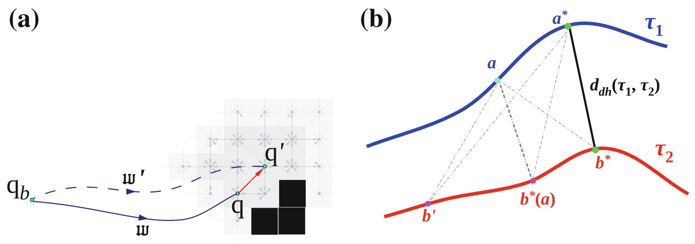

Fig. 3 Preliminaries: h-augmented graph and Hausdorff distance. **a** The h-augmented graph, G_h, is created from the discrete graph representation of the environment, G, so as to incorporate information about the homotopy class of trajectories. **b** The directed Hausdorff distance between τ_1 and τ_2 is determined as follows: Fix a point $a \in \tau_1$, and find its distance from τ_2. The directed Hausdorff distance is then the maximum of this value over all the possible points a on τ_1

the vertices of adjacent cells. While the graph, G, itself can be quite arbitrary, we used a uniform square discretization and an 8-connected graph representation of the environment in all our simulations for simplicity (Fig. 3a).

From such a graph, we construct an h-augmented graph, G_h, for keeping track of the homotopy class of the trajectories. The construction, in brief, is as follows: Vertices in this h-augmented graph, G_h, are of the form $(\mathrm{q}, \mathfrak{w})$ where $\mathrm{q} \in \mathscr{V}(G)$ is a position of an agent in the workspace (as a vertex in the discrete representation graph, $\mathscr{G}$) and $\mathfrak{w}$ is the *word* (i.e. the homotopy invariant) corresponding to the homotopy class of a trajectory leading up to q from a base vertex q_b (all h-signatures of trajectories are computed with respect to a fixed point, called the *base point*, the vertex at which is $\mathrm{q}_b \in \mathscr{V}(G)$). We write the tuple as $\mathrm{v} = (\mathrm{q}, \mathfrak{w}) \in \mathscr{V}(G_h)$, with $\mathrm{v}_b := (\mathrm{q}_b, \text{“ ”})$ being the vertex corresponding to the path of zero length. Thus the h-augmented graph encodes in its vertex set information about homotopy classes of paths, along with agent positions. The connectivity in the h-augmented graph is described as follows: For every directed edge $[\mathrm{q} \rightsquigarrow \mathrm{q}'] \in \mathscr{E}(G)$ (i.e., connecting q to q′ in $\mathscr{V}(G)$), and for every vertex of the form $\mathrm{v} = (\mathrm{q}, \mathfrak{w}) \in \mathscr{V}(G_h)$, there exists a vertex $(\mathrm{q}', \ \mathfrak{w} \circ h(\overrightarrow{\mathrm{q}\mathrm{q}'}))$ (where $\overrightarrow{\mathrm{q}\mathrm{q}'}$ is the line segment corresponding to the edge)—see Fig. 3a. Thus, starting from $(\mathrm{q}_b, \text{“ ”})$, this gives a recipe to construct the h-augmented graph, G_h, incrementally—a construction approach perfectly suited for any graph search algorithm (such as A* and Dijkstra's) involving *expansion* of vertices starting from an initial vertex, so as to find shortest paths leading up to a vertices of the form $(\mathrm{q}_g, *)$—i.e., the goal vertex, q_g, but reached via a specific homotopy class. The cost of an edge in G_h is chosen to be the same as the cost of the projected edge in G. That is, $c_{G_h}([(\mathrm{q}, \mathfrak{w}) \rightsquigarrow (\mathrm{q}', \mathfrak{w} \circ h(\overrightarrow{\mathrm{q}\mathrm{q}'})]) = c_G([\mathrm{q} \rightsquigarrow \mathrm{q}'])$ (where c_G and c_{G_h} represent the cost functions in the respective graphs). In our implementation we choose $c_G([\mathrm{q} \rightsquigarrow \mathrm{q}'])$ to be the Euclidean length of the line segment that constitutes the edge, $\overline{\mathrm{q}\mathrm{q}'}$. For more details, the reader can refer to similar construction appearing in recent works [23, 24].

2.3 Hausdorff Distance as a Metric on Space of Trajectories

The behavior of a human agent, by nature, can be highly unpredictable. Even if a human is presented with a clear set of trajectories to choose from, he/she may take a trajectory that deviates from the planned ones. In our problem it is critical that the robots quickly understand/estimate which homotopy class the human agents are potentially taking so that the robots can quickly follow the complementary classes. This is achieved by comparing the human's partial trajectory with a set of estimated/baseline candidate trajectories in different homotopy classes connecting the start and the goal locations. The h-signature by itself does not provide adequate information to evaluate distance between candidate trajectories. Rather, this comparison warrants a metric in the space of all trajectories in $(W - \mathscr{O})$. In particular, one can choose the *Hausdorff distance* [25], that is suitable for comparing any two subsets of a metric space.

Definition 2 (*Hausdorff distance* [25]) Consider the free space of the agents, $(W - \mathscr{O})$, equipped with the standard Euclidean metric. The Hausdorff distance, d_H, between two sets $A, B \subset (W - \mathscr{O})$ is then defined as

$$d_H(A, B) = \max(d_{dh}(A, B), d_{dh}(B, A)) \tag{1}$$

where, $d_{dh}(A, B) = \max_{a \in A} \min_{b \in B} \|a - b\|$ is the directed Hausdorff distance between sets A and B, and $\|a - b\|$ is the Euclidean distance between $a, b \in (W - \mathscr{O})$.

Hausdorff distance, as defined, is a valid metric (satisfying all the *axioms of a metric*) on the space of all subsets of $(W - \mathscr{O})$. In particular, this implies $d_H(A, B) = 0 \iff A = B$. We can thus use this metric to compare two trajectories, τ_1 and τ_2, when viewed as subsets of $(W - \mathscr{O})$. In particular, if τ_H is a trajectory that the human has traversed, and $\{\tau_1, \tau_2, \ldots\}$ are candidate trajectories in different homotopy classes estimated by the robots, then the values of $d_H(\tau_H, \tau_i)$ will help determine which homotopy class the human is following.

Figure 3b illustrates the computation of the directed Hausdorff distance, d_{dh}, for two trajectories τ_1 and τ_2 in an environment. The Hausdorff distance itself is a *symmetrized* version of that distance to satisfy the symmetry property of a metric.

3 Algorithm Design

As described earlier, the objective of this work is to develop a distributed algorithmic framework for autonomous agents in search and rescue operations consisting of a heterogeneous team of humans and robots, taking into account the unpredictability of human agents to efficiently explore an environment via complementary *homotopy* *classes of trajectories*. For simplicity, we assume that all trajectories have equal priority and that each robot and each human travel at the same speed, respectively. Priority and speed variation could potentially be considered, if needed, through modification of the cost function to prioritize most promising paths or agents.

As illustrated in Fig. 1, a key critical component in the algorithm design is the ability of the autonomous agents (robots) to identify the intent of the humans. In particular, the robots need to quickly narrow down the set of possible homotopy classes of paths that the humans are potentially taking, and thus follow the complementary classes. Furthermore, they need to monitor whether a human agent is altering his/her behavior (changing the homotopy class that he/she committed to), so that the robots can change their trajectories accordingly. The principal components involved in achieving these are (i) *Human path prediction*, (ii) *Robot path assignment* and (iii) *Human's path history truncation*. The following sub-sections describe these algorithmic components in details.

3.1 Human Path Prediction Algorithm

At the entrance to the environment, the robots compute a set of reference trajectories, $\mathscr{T} = \{\tau_1, \tau_2, \dots, \tau_p\}$, in p shortest homotopy classes that are to be tentatively pursued by the agents by performing A* search in the h-augmented graph. Ideally, we should choose all the non-looping and non-intersecting homotopy classes in the environment, but for most practical cases it is sufficient to choose $p = \max(P, n + m)$, where n is the number of human agents, m is the number of robot agents, and P is an upper bound on the number of homotopy classes that we compute (determined by our computation capability). We describe the path prediction algorithm for a single human. In the presence of multiple human agents, the algorithm is executed for each of them. Also, in a distributed implementation, the algorithm for predicting each human's path runs independently on each autonomous robot.

At the start, the set of potential paths that the human is following, denoted $\mathscr{S}_0$, is the entire of $\mathscr{T}$. The kth call of the path prediction algorithm computes $\mathscr{S}_k$, the set of potential paths that the human is following at the kth computation step, in a recursive manner. Suppose we computed $\mathscr{S}_k = \{\tau_{i_1}, \tau_{i_2}, \dots, \tau_{i_q}\} \subseteq \mathscr{T}$. The path prediction algorithm at the $(k+1)$th step takes the human's path history (say τ_H) and compares it with the reference trajectories in $\mathscr{T}$ (i.e., computes the Hausdorff distances from each τ_i) to determine the new set $\mathscr{S}_{k+1}$ of homotopy classes that the human is potentially following. The basic algorithm is as follows: Let the distances of the human's path history from the reference trajectories be $d_i := d_H(\tau_H, \tau_i),\ i = 1, 2, \dots, p$. These distances are normalized by the largest Hausdorff distance out of the most recent potential set of trajectories, $\mathscr{S}_k$, that the human was following to obtain a set of normalized distances: $\overline{d}_i = d_i / D,\ \forall i = 1, 2, \dots, p$, where $D = \max_{\tau_i \in \mathscr{S}_k} d_H(\tau_H, \tau_i)$. Based on these normalized distances, the objective is to determine if the human's trajectory is *close* to some of the trajectories in $\mathscr{T}$ and *far* from others. This decision ($(k+1)$th path prediction cycle) involves a two-step reasoning:

i. If $\min(\{\overline{d}_1, \overline{d}_2, \dots, \overline{d}_p\}) \geq \alpha$, where $\alpha \in [0, 1]$ is a parameter encoding the maximum uncertainty tolerated by the user, then the decision cannot be made yet—it is not yet clear what subset of $\mathscr{S}_k$ the human has narrowed down to. Thus $\mathscr{S}_{k+1}$ is not computed, and it is asserted that the $(k+1)$th path prediction cycle is still in progress with $\mathscr{S}_k$ being the set of possible homotopy classes that the human is still following. The robots waiting for the human to make the move keep waiting or continue to follow the same path as before.
ii. If however, $\min(\{\overline{d}_1, \overline{d}_2, \dots, \overline{d}_p\}) < \alpha$, we update the set of potential paths that the human is following to the set $\mathscr{S}_{k+1} := \{\tau_i \mid \overline{d}_i < \beta \cdot \min(\{\overline{d}_1, \overline{d}_2, \cdots, \overline{d}_p\})\}$, where $\beta \geq 1$ is another parameter. This simply implies that the set of potential homotopy classes that the human is following contains trajectories that are within a distance of at most β times the distance from the closest class. This provides a conservative buffer in the case of very similar paths.

In implementation, following the $(k+1)$th path prediction step, the human broadcasts the h-signatures of the paths in set $\mathscr{S}_{k+1}$ only that are being followed by the human, rather than the full set of vertices describing the predicted path itself. This gives a compact communication protocol purely based on topological information rather than denser metric information. Thus the communication burden is minimized for each human, allowing for more effective and efficient coordination between humans and robots.

Figure 4 shows a simple example of the path prediction algorithm and how the Hausdorff distance is used as a metric to select the set $\mathscr{S}_{k+1}$ of paths for the human's potential trajectory.

3.2 Robot Assignment Algorithm

At the very beginning (start/entrance to the environment), the robots wait for all the humans to make first moves until the humans have a narrowed-down set of possible homotopy classes (i.e., the number of elements in $\mathscr{S}_1$, for each human, has gone below the number of elements in $\mathscr{S}_0 = \mathscr{T}$). Then the robots coordinate among themselves to determine the h-signature of the path (or a set of h-signatures) that each robot should follow. In particular, the cost of a path in a given homotopy class is used to prioritize the assignment of robots to expedite clearing of the environment. This assignment process is run every time a new cycle of path prediction returns a new set of possible homotopy classes that a human is following.

Suppose for the jth human the k_jth cycle of path prediction algorithm returned a new $\mathscr{S}_{k_j}$. The robot assignment algorithm works by first determining the shortest p paths for each robot in the environment, along with the associated path costs. Following which there are two stages in the assignment algorithm:

i. *Choose complementary homotopy classes:* The h-signatures for paths that are not in the set of potential paths that any of the humans are following (i.e. not in $\mathscr{S}_{k_j}$ of any of the humans) are prioritized first—unassigned homotopy classes

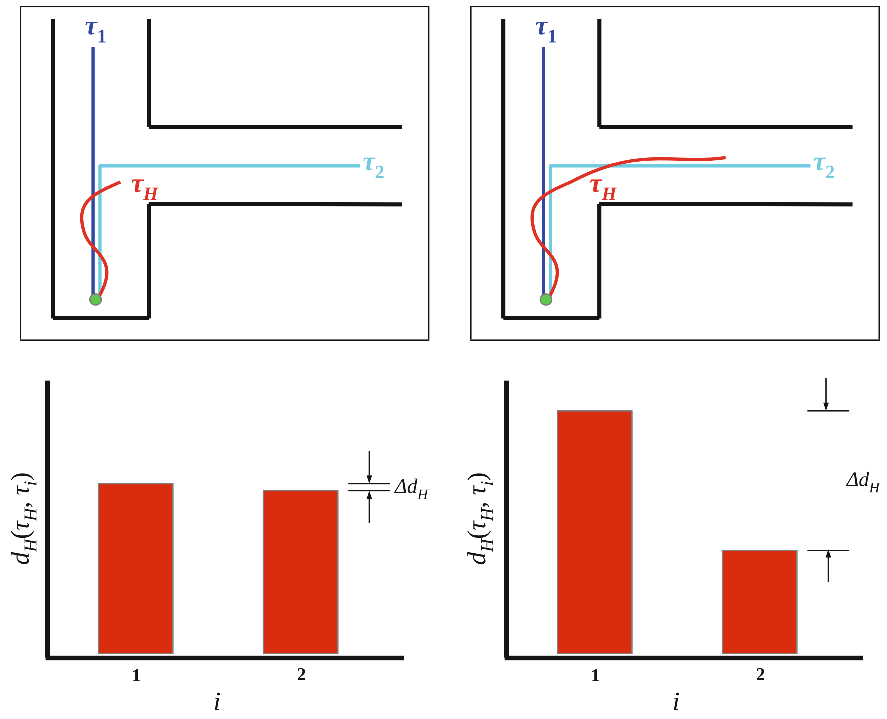

Fig. 4 Initially (*left column*), the human's path τ_H (in *red*) is close to either of τ_1 and τ_2 (i.e. the set of potential paths that the human is following is $\mathscr{S}_k = \{\tau_1, \tau_2\}$). As the human travels further (*right column*), the human's path, τ_H, gets closer to τ_2. In this case, the human path prediction algorithm would update the set of potential homotopy classes of paths to $\mathscr{S}_{k+1} = \{\tau_2\}$. Note how Δd_H (difference between d_1 and d_2) increases indicating a clear demarcation between the distances from the two reference trajectories

get assigned to the robot with the shortest path cost for that homotopy class. This behavior is illustrated by robots R_1 and R_2 in Fig. 1c.

ii. *Tailgate humans with more than one homotopy class in the possible set of homotopy classes:* Once all classes not in any of the human's $\mathscr{S}_{k_j}$ have been assigned to robots, then the remaining robots are assigned to follow human agents with excess elements in their set of possible homotopy classes (the jth human's $\mathscr{S}_{k_j}$). This is the behavior of robot R_3 in Fig. 1c.

This path assignment algorithm is also executed again for groups of tailgating robots every time the human which they are following passes through a junction/branching point (i.e., the path prediction algorithm returns a new set $\mathscr{S}_{k+1}$).

3.3 Human's Path History Truncation for Robustness to Unpredictable Actions

This algorithmic component is necessary to incorporate sudden changes in human behavior that contradict the decision made in a prior path prediction step. In case a human turns back and goes past an earlier junction/fork point, and starts following a different homotopy class, the clear demarcation between the Hausdorff distances from the trajectories in $\mathscr{S}$ and those from the other, as was illustrated in Fig. 4, will fade—triggering the 'path history truncation' procedure. In order to figure out which new homotopy class of trajectory the human has taken up, we need to *chop off* the part of the human's path history involving the "U-turn" from the earlier homotopy class, and replace it with a path in the same homotopy class. Consequently, the human path prediction algorithm (say the $(k+1)$th cycle) will be able to identify the new homotopy class the human is following, and compute $\mathscr{S}_{k+1}$ accordingly.

Suppose, for a human, the last path prediction cycle returned $\mathscr{S}_k$. The path truncation algorithm seeks to isolate only the most recent path history for the human so that the path prediction algorithm only uses the most recent, freshest human path data and ignores the convoluted past path behavior. This is achieved by looking at the minimum distance from points on the human's path to reference trajectories in $\mathscr{S}_{k-1}$. Consider a point u' earlier on the human's trajectory (see Fig. 5). The minimum distance of this point from any trajectory of homotopy classes that the human was potentially following before taking the U-turn, $d_{min}(u', \tau_i),\ \forall \tau_i \in \mathscr{S}_k$, can be expected to be low. While the distances from the other homotopy classes, $d_{min}(u', \tau_i),\ \forall \tau_i \in (\mathscr{S}_{k-1} - \mathscr{S}_k)$, can be expected to be high. However, if we consider a point u'' later on the human's trajectory (after the U-turn), this will just be the reverse—$d_{min}(u'', \tau_i),\ \forall \tau_i \in \mathscr{S}_k$ will be high, while $d_{min}(u'', \tau_i),\ \forall \tau_i \in (\mathscr{S}_{k-1} - \mathscr{S}_k)$ will be low. This observation is key in determining the truncation point. In particular, we choose the truncation point to be a point u_{trunc} on the human's trajectory at which the the average of the distances $d_{min}(u_{trunc}, \tau_i),\ \forall \tau_i \in \mathscr{S}_k$ becomes equal to the average of the distances $d_{min}(u_{trunc}, \tau_i),\ \forall \tau_i \in (\mathscr{S}_{k-1} - \mathscr{S}_k)$. After truncation, the human's trajectory is updated by replacing the part before truncation with the shortest path leading to u_{trunc} but in the same homotopy class as the truncated part of the trajectory. This approach will be effective as long as a human is not perpetually indecisive switching between classes forever.

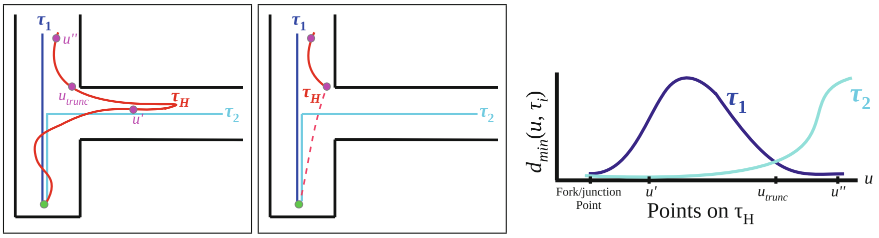

Fig. 5 Identifying the point on the human's trajectory at which to truncate it

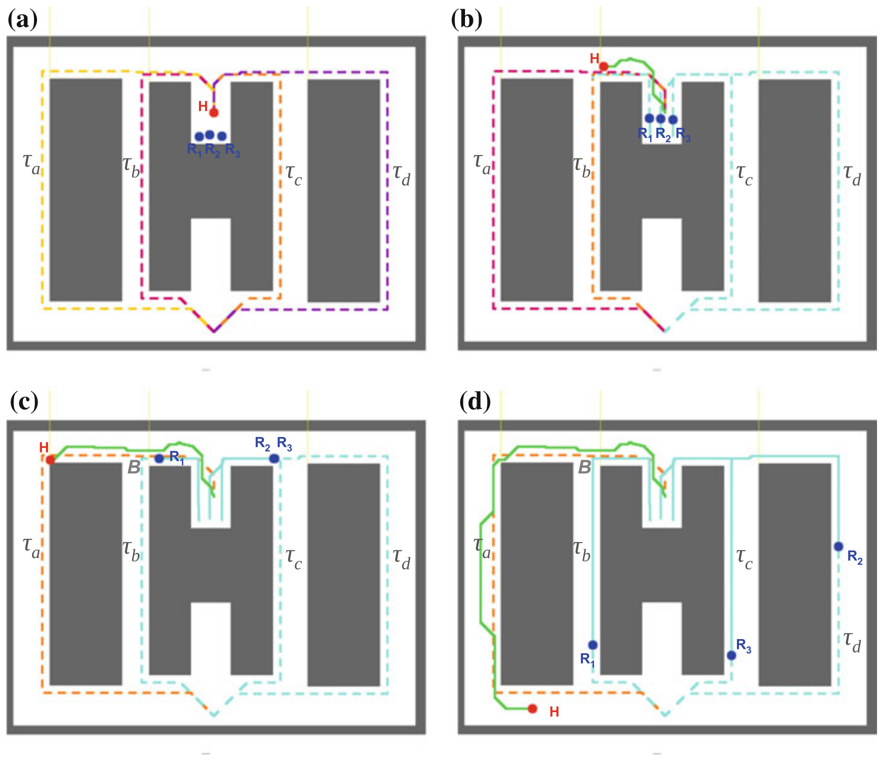

Fig. 6 Collaborative topological exploration in complementary homotopy classes demonstrating how autonomous agents respond to human actions by choosing complementary homotopy classes. **a** Three robots (R_1, R_2, R_3, in *blue*) and one human (H, in *red*) start at the *upper* "cup" of the H-shaped obstacle. They find 4 homotopy classes leading to the goal at the *bottom* of the environment. Optimal trajectories in the different homotopy classes are shown in different colors for easy visualization. At this point the potential set of possible homotopy classes that the human can take is $\mathscr{S}_0 = \{\tau_a, \tau_b, \tau_c, \tau_d\}$. **b** As the human moves forward, its set of possible homotopy classes is narrowed down to $\mathscr{S}_1 = \{\tau_a, \tau_b\}$. The corresponding trajectories are shown in *red* and *orange*. It is determined that homotopy classes of τ_c and τ_d have not been taken up by the human. Thus, homotopy classes of τ_c and τ_d are assigned to robots (R_2 and R_3) with priority. Remaining robot (R_1) is assigned to tailgate human, H, since $\mathscr{S}_1$ contains more than one elements. **c** After crossing a branching point, B, the human commits to homotopy class of τ_a (*orange dotted curve*). So now $\mathscr{S}_2 = \{\tau_a\}$ contains a single element. R_1 thus will choose the complementary class τ_b. Robots R_2, R_3 continues as usual. **d** A final frame showing that the humans and robots followed complementary homotopy classes to reach the goal

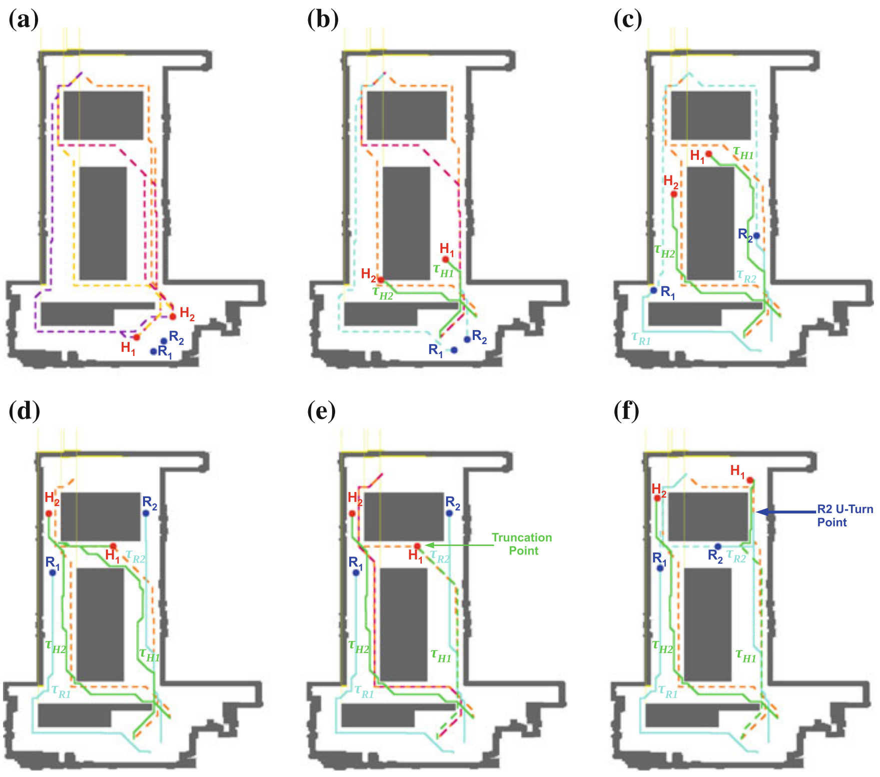

Fig. 7 An example of the human-robot coordinated exploration in an indoor search and rescue scenario, with two robots, two humans and a demonstration of the path truncation algorithm. **a** Two robots (R_1, R_2, in *blue*) and two humans (H_1, H_2, in *red*) start near the *bottom* of the map. They find 4 homotopy classes leading to the goal at the *top* of the map. As in the example with one human, the four shortest paths in different homotopy classes for H_1 and H_2 are displayed. **b** As H_1 and H_2 travel away from the initial junction point, R_1 responds by planning a path in a complementary homotopy class while R_2 tailgates the humans. The planned paths for R_1 and R_2 are shown in *cyan* color. **c** At this point, R_1 and R_2 have already started moving towards the goal. H_1 and H_2 are closer to the goal and appear to only be following one path, respectively, so R_2 goes from tailgating the human to planning a path in the remaining complementary homotopy class. **d** H_1 has turned back from the path it was following previously. H_1 is now traveling towards R_2. The indecisive behavior results in less clarity regarding the human's path behavior. This triggers the *path truncation algorithm* to be executed, so that any future predictions will only focus on the most recent human path data. **e** H_1's path was truncated at the labeled truncated point, eliminating the "U-turn" points from being used as data in the path prediction algorithm. The path before the truncation point is replaced by the shortest path in the same homotopy class as the part of the path that was chopped off (*green dashed curve*). **f** In response to the update in H_1's predicted paths after the path truncation was completed, R_2 makes a "U-turn" to take the path abandoned by H_1—the path to the *left* of the uppermost obstacle. Essentially, H_1 and R_2 have switched places. From this point onwards, all agents travel along these planned paths to the goal point

4 Results

Implementation: The described algorithm was implemented in ROS (Robot Operating System) with human agents simulated through mouse-driven user interface controlled by the authors and autonomous robots navigating using the proposed algorithm. Dynamics or kinematics of the agents were not simulated; however, our implementation is completely distributed, with the agents communicating using h-signatures as compact representations of trajectories. The environment was provided to ROS as a bitmap, with automated identification of connected components of obstacles and placement of representative points. In order to avoid multiplicity of homotopy classes created by small obstacles/noise, a minimum size threshold was placed on the obstacles on which to place *representative points*. Additionally, the obstacles in the bitmap were inflated by the radius of robot to enable collision avoidance and modeling of robots as points in the inflated obstacle map. For the path prediction algorithm we chose the parameters $\alpha = 0.5$ and $\beta = 1.50$ based on experimentation on a benchmark environment. In practice and for simplicity, the path prediction algorithm for each human was implemented on the human agent itself (its processor thread). The predicted paths were communicated to the other agents by reporting the h-signatures of the predicted paths.

Figure 6 shows how three robots and one human split up the process of exploring an environment in four different homotopy classes. Figure 7 demonstrates our algorithm in a more complex indoor environment with two humans and two robots. The example also illustrates the path truncation algorithm. For each of the results, the figure captions describe the algorithm in action. The algorithm was also tested in more complex environments—these results can be viewed at http://www.eecs.berkeley.edu/~govvijay/DARS14.html.

5 Conclusion

A human-robot coordinated exploration problem in context of search and rescue operations is addressed in this paper. The autonomous robots intelligently choose actions to complement the actions of the human agents. In particular, the idea of *complementary homotopy classes* of trajectories help the autonomous agents choose trajectories for fast and efficient exploration. The proposed algorithm consists of prediction of the homotopy classes of the human agents' paths, assignment of complementary paths to the robots, and a truncation algorithm for increased robustness to the indecisive/uncertain behavior of human agents. We demonstrated the practical applicability of the algorithm through ROS simulations with distributed implementation. In the near future, we plan to conduct extensive experiments on real robots and explore the optimal selection of parameters α and β in the path prediction algorithm for an arbitrary environment. Development of additional interfaces for fast and easy communication of intent between the agents is under progress.

Acknowledgments We gratefully acknowledge the support of Army Research Laboratory grant number W911NF-10-2-0016, Air Force Office of Scientific Research grant number FA9550-10-1-0567, and Office of Naval Research grant number N00014-09-1-103. The first author would also like to thank the Rachleff Scholars Program.

References

1. Stachniss, C.: Exploration and mapping with mobile robots. Ph.D. thesis, University of Freiburg, Freiburg, Germany, Apr 2006
2. Bhattacharya, S., Michael, N., Kumar, V.: Distributed coverage and exploration in unknown non-convex environments. In: *Proceedings of 10th International Symposium on Distributed Autonomous Robotics Systems*. Springer, 1–3 Nov 2010
3. Hazon, N., Mieli, F., Kaminka, G.A.: Towards robust on-line multi-robot coverage. In: Proceedings 2006 IEEE International Conference on Robotics and Automation, 2006. ICRA 2006, pp. 1710–1715, May 2006
4. Rekleitis, I., New, A.P., Rankin, E.S., Choset, H.: Efficient boustrophedon multi-robot coverage: an algorithmic approach. Ann. Math. Artif. Intell. **52**(2–4), 109–142 (2008)
5. Zheng, X., Koenig, S., Kempe, D., Jain, S.: Multirobot forest coverage for weighted and unweighted terrain. IEEE Trans. Robot. **26**(6), 1018–1031 (2010)
6. Bhattacharya, S., Ghrist, R., Kumar, V.: Multi-robot coverage and exploration on Riemannian manifolds with boundary. Int. J. Robot. Res. **33**(1), 113–137 (2014). doi:10.1177/0278364913507324
7. Lloyd, S.P.: Least squares quantization in PCM. IEEE Trans. Inf. Theory **28**, 129–137 (1982)
8. Cortez, J., Martinez, S., Bullo, F.: Spatially-distributed coverage optimization and control with limited-range interactions. ESIAM: Control Optim. Calc. Var. **11**, 691–719 (2005)
9. Cortez, J., Martinez, S., Karatas, T., Bullo, F.: Coverage control for mobile sensing networks. IEEE Trans. Robot. and Automat. **20**(2), 243–255 (2004)
10. Bullo, F., Cortés, J., Martínez, S.: Distributed Control of Robotic Networks: A Mathematical Approach to Motion Coordination Algorithms. Applied Mathematics Series. Princeton University Press, Princeton (2009)
11. Sariel-Talay, S., Balch, T.R., Erdogan, N.: Multiple traveling robot problem: a solution based on dynamic task selection and robust execution. IEEE/ASME Trans. Mechatron. **14**(2), 198–206 (2009). April
12. Carlin, A., Ayers, J., Rousseau, J., Schurr, N.: Agent-based coordination of human-multirobot teams in complex environments. In: Proceedings of the 9th International Conference on Autonomous Agents and Multiagent Systems: Industry Track, AAMAS'10, pp. 1747–1754, Richland, SC, International Foundation for Autonomous Agents and Multiagent Systems (2010)
13. Kehagias, A., Hollinger, G., Singh, S.: A graph search algorithm for indoor pursuit/evasion. Math. Comput. Modell. **50**(910), 1305–1317 (2009)
14. Choset, H., Burdick, J.: Sensor-based exploration: the hierarchical generalized Voronoi graph. Int. J. Robot. Res. **19**(2), 96–125 (2000)
15. Tully, S., Kantor, G., Choset, H.: A unified Bayesian framework for global localization and SLAM in hybrid metric/topological maps. Int. J. Robot. Res. (2012)
16. Kim, S., Bhattacharya, S., Ghrist, R., Kumar, V.: Topological exploration of unknown and partially known environments. In: 2013 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 3851–3858, Nov 2013
17. Bhattacharya, S., Likhachev, M., Kumar, V.: Topological constraints in search-based robot path planning. Auton. Robots 1–18, (2012). doi:10.1007/s10514-012-9304-1
18. Hatcher, A.: Algebraic Topology. Cambridge University Press, Cambridge (2001)

19. Grigoriev, D., Slissenko, A.: Polytime algorithm for the shortest path in a homotopy class amidst semi-algebraic obstacles in the plane. In: ISSAC'98: Proceedings of the 1998 international symposium on symbolic and algebraic computation, pp. 17–24, New York, NY, USA. ACM (1998)
20. Hershberger, J., Snoeyink, J.: Computing minimum length paths of a given homotopy class. Comput. Geom. Theory Appl **4**, 331–342 (1991)
21. Tovar, B., Cohen, F., LaValle, S.M.: Sensor beams, obstacles, and possible paths. In: Workshop on the Algorithmic Foundations of Robotics (2008)
22. Narayanan, V., Vernaza, P., Likhachev, M., LaValle, S.M.: Planning under topological constraints using beam-graphs. In: 2013 IEEE International Conference on Robotics and Automation (ICRA), pp. 431–437. IEEE (2013)
23. Kim, S., Bhattacharya, S., Heidarsson, H., Sukhatme, G., Kumar, V.: A topological approach to using cables to separate and manipulate sets of objects. In: Proceedings of the Robotics: Science and System (RSS), Syndey, Australia, 24–28 June 2013
24. Kim, S., Bhattacharya, S., Kumar, V.: Path planning for a tethered mobile robot. In: Proceedings of IEEE International Conference on Robotics and Automation (ICRA) (2014) (Accepted. To appear)
25. Gromov, M., Lafontaine, J., Pansu, P.: Metric Structures for Riemannian and Non-Riemannian Spaces. Progress in Mathematics. Birkhäuser, Basel (1999)