Reactive Semantic Planning in Unexplored Semantic Environments Using Deep Perceptual Feedback

Vasileios Vasilopoulos, Georgios Pavlakos, Sean L. Bowman, J. Diego Caporale,
Kostas Daniilidis, George J. Pappas, Daniel E. Koditschek

***Abstract*—This paper presents a reactive planning system that enriches the topological representation of an environment with a tightly integrated semantic representation, achieved by incorporating and exploiting advances in deep perceptual learning and probabilistic semantic reasoning. Our architecture combines object detection with semantic SLAM, affording robust, reactive logical as well as geometric planning in unexplored environments. Moreover, by incorporating a human mesh estimation algorithm, our system is capable of reacting and responding in real time to semantically labeled human motions and gestures. New formal results allow tracking of suitably non-adversarial moving targets, while maintaining the same collision avoidance guarantees. We suggest the empirical utility of the proposed control architecture with a numerical study including comparisons with a state-of-the-art dynamic replanning algorithm, and physical implementation on both a wheeled and legged platform in different settings with both geometric and semantic goals.**

***Index Terms*—Reactive and Sensor-Based Planning, Motion and Path Planning, Semantic Scene Understanding**

I. INTRODUCTION

A. Motivation and Prior Work

NAVIGATION is a fundamentally topological problem [1] reducible to purely reactive (i.e., closed loop state feedback based) solution, given perfect prior knowledge of the environment [2]. For geometrically simple environments, "doubly reactive" methods that reconstruct the local obstacle field on the fly [3], [4], or operate with no need for such reconstruction at all [5], can guarantee collision free convergence to a designated goal with no need for further prior information. However, imperfectly known environments presenting densely cluttered or non-convex obstacles have heretofore required incremental versions of random sampling-based tree construction [6] whose probabilistic completeness can be slow to be realized in practice, especially when confronting settings with narrow passages [7].

Manuscript received: February 24, 2020; Revised May 4, 2020; Accepted June 4, 2020.

This paper was recommended for publication by Editor Prof. Tamim Asfour upon evaluation of the Associate Editor and Reviewers' comments. This work was supported by AFRL grant FA865015D1845 (subcontract 669737-1), AFOSR grant FA9550-19-1-0265 (Assured Autonomy in Contested Environments), and ONR grant #N00014-16-1-2817, a Vannevar Bush Fellowship held by the last author, sponsored by the Basic Research Office of the Assistant Secretary of Defense for Research and Engineering. The authors thank Diedra Krieger for assistance with video recording.

The authors are with the General Robotics, Automation, Sensing and Perception (GRASP) Lab, University of Pennsylvania, Philadelphia, PA 19104. (e-mail: vvasilo@seas.upenn.edu; pavlakos@seas.upenn.edu; seanbow@seas.upenn.edu; jdcap@seas.upenn.edu; kostas@seas.upenn.edu; pappasg@seas.upenn.edu; kod@seas.upenn.edu)

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

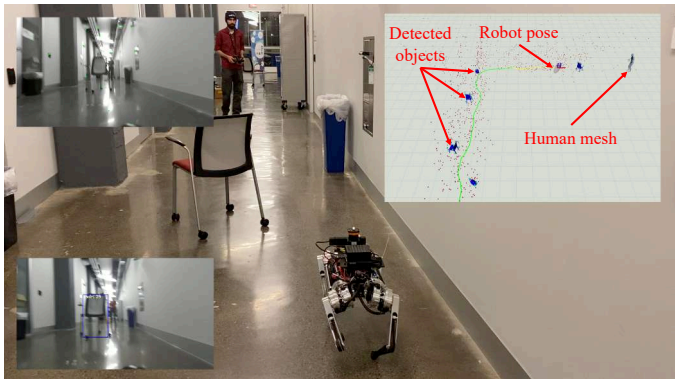

Fig. 1: Ghost Spirit [8] following a human, while avoiding some familiar and some novel obstacles in a previously unexplored environment. Familiar obstacles are recognized and localized using visually detected semantic keypoints (bottom left inset) [9], combined with geometric features (top left inset) [10] and avoided by a local deformation of space (Fig. 3) that brings them within the scope of a doubly reactive navigation algorithm [5]. Novel obstacles are detected by LIDAR and assumed to be convex, thus falling within the scope of [5]. Formal guarantees are summarized in Theorems 1 and 2 of Section IV, and experimental settings are summarized in Fig. 7.

Monolithic end-to-end learning approaches to navigation – whether supporting metric [12] or topological [13] representations of the environment – suffer from the familiar problems of overfitting to specific settings or conditions. More modular data driven methods that separate the recruitment of learned visual representation to support learned control policies achieve greater generalization [14], but even carefully modularized approaches that handcraft the interaction of learned topological plans with learned reactive motor control in a physically informed framework [15] cannot bake into their architectures crucial properties that afford guaranteed policies.

Unlike the problem of safe navigation in a completely known environment, the setting where the obstacles are not initially known and are incrementally revealed online has so far received little theoretical interest. Some few notable exceptions include considerations of optimality in unknown spaces [16], online modifications to temporal logic specifications [17] or deep learning algorithms [18] that assure safety against obstacles, or the use of trajectory optimization along with offline computed reachable sets [19] for online policy adaptations. However, none of these advances (and, to the best of our knowledge, no work prior to [11]) has achieved simultaneous guarantees of obstacle avoidance and convergence. In this paper, we extend these guarantees to the setting of (non-adversarial) moving targets, while also affording semantic specifications - capabilities that have not been heretofore available, even in settings with simple obstacles.

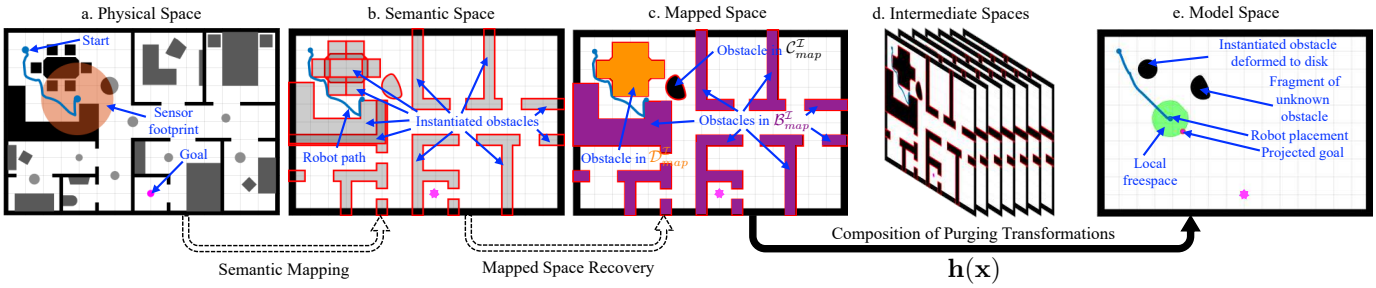

Fig. 2: Snapshot Illustration of Key Ideas, following [11]: The robot moves in the physical space, in an environment with known exterior boundaries (walls), toward a goal (pink) discovering along the way (black) both familiar objects of known geometry but unknown location (dark grey) and unknown obstacles (light grey), with an onboard sensor of limited range (orange disk). As in [11], these obstacles are processed by the perceptual pipeline (Fig. 4) and stored permanently in the semantic space if they have familiar geometry, or temporarily, with just the corresponding sensed fragments, if they are unknown. The consolidated obstacles (formed by overlapping catalogued obstacles from the semantic space), along with the perceptually encountered components of the unknown obstacles, are again stored in the mapped space. A change of coordinates, $\mathbf{h}$, entailing an online computation greatly streamlined relative to its counterpart in [11] deforms the mapped space to yield a geometrically simple but topologically equivalent model space. This new change of coordinates defines a vector field on the model space, which is transformed in realtime through the diffeomorphism to generate the input in the physical space.

B. Summary of Contributions

1) Architectural Contributions: In [11], we introduced a Deep Vision based object recognition system [9] as an "oracle" for informing a doubly reactive motion planner [5], [20], incorporating a Semantic SLAM engine [10] to integrate observations and semantic labels over time. Here, we extend this architecture in two different ways (see Fig. 4). First, new formal advances, replacing the computationally expensive triangulation on the fly [11] with convex decompositions of obstacles as described below, streamline the reactive computation, enabling robust online and onboard implementation (perceptual updates at 4Hz; reactive planning updates at 30Hz), affording tight realtime integration of the Semantic SLAM engine. Second, we incorporate a separate deep neural net that captures a wire mesh representation of encountered humans [21], enabling our reactive module to track and respond in realtime to semantically labeled human motions and gestures.

2) Theoretical Contributions: We introduce a new change of coordinates, replacing the (potentially combinatorially growing) triangulation on the fly of [11] with a fixed convex decomposition [22] for each catalogued obstacle and revisit the prior hybrid dynamics convergence result [11] to once again guarantee obstacle free geometric convergence. However, this streamlined computation, enabling full realtime integration of the Semantic SLAM engine, now allows us to react logically as well as geometrically in unexplored environments. In turn, realtime semantics combined with human recognition capability motivates the proof of new rigorous guarantees for the robots to track suitably non-adversarial (see Definition 4) moving targets, while maintaining collision avoidance guarantees.

3) Empirical Contributions: We suggest the utility of the proposed architecture with a numerical study including comparisons with a state-of-the-art dynamic replanning algorithm [23], and physical implementation on both a wheeled and legged platform in highly varied environments (cluttered outdoor and indoor spaces including sunlight-flooded floors as well as featureless hallways). Targets are robustly followed up to speeds amenable to the perceptual pipeline's tracking rate. Importantly, the semantic capabilities of our pipeline are exploited to introduce more complex task logic (e.g., track a given target unless encountering a specific human gesture).

C. Organization of the Paper and Supplementary Material

After stating the problem, summarizing our solution and introducing technical notation in Section II, Section III describes the diffeomorphism between the mapped and model spaces, and Section IV includes our main formal results. Section V and Section VI continue with our numerical and experimental studies, and Section VII concludes with ideas for future work. The supplementary video submission includes our empirical studies; we also include pointers to open-source software implementations, for both the MATLAB simulation package[1], and the ROS-based controller[2], in C++ and Python.

II. PROBLEM FORMULATION AND APPROACH

A. Problem Formulation

As in [11], [20], we consider a robot with radius r, centered at $\mathbf{x} \in \mathbb{R}^2$, navigating a compact, polygonal, potentially non-convex workspace $\mathcal{W} \subset \mathbb{R}^2$, with known boundary $\partial\mathcal{W}$, towards a target $\mathbf{x}_d \in \mathcal{W}$. The robot is assumed to possess a sensor with fixed range R, for recognizing "familiar" objects and estimating distance to nearby obstacles[3]. We define the *enclosing workspace*, as the convex hull of the closure of the workspace $\mathcal{W}$, i.e., $\mathcal{W}_e := \left\{\mathbf{x} \in \mathbb{R}^2 \,|\, \mathbf{x} \in \text{Conv}(\overline{\mathcal{W}})\right\}$.

The workspace is cluttered by a finite but unknown number of disjoint obstacles, denoted by $\tilde{\mathcal{O}} := \{\tilde{O}_1, \tilde{O}_2, \ldots\}$, which might also include non-convex "intrusions" of the boundary of the physical workspace $\mathcal{W}$ into $\mathcal{W}_e$. As in [5], [11], we define the *freespace* $\mathcal{F}$ as the set of collision-free placements for the closed ball $\overline{\mathsf{B}(\mathbf{x}, r)}$ centered at $\mathbf{x}$ with radius r, and the *enclosing freespace*, $\mathcal{F}_e$, as $\mathcal{F}_e := \left\{\mathbf{x} \in \mathbb{R}^2 \,|\, \mathbf{x} \in \text{Conv}(\overline{\mathcal{F}})\right\}$.

Although none of the positions of any obstacles in $\tilde{\mathcal{O}}$ are à-priori known, a subset $\tilde{\mathcal{P}} := \{\tilde{P}_i\}_{i \in \mathcal{N}_\mathcal{P}} \subseteq \tilde{\mathcal{O}}$ of these obstacles, indexed by $\mathcal{N}_\mathcal{P} := \{1, \ldots, N_P\} \subset \mathbb{N}$, is assumed to be "familiar" in the sense of having a known, readily recognizable polygonal geometry, that the robot can instantly identify and localize. The remaining obstacles in $\tilde{\mathcal{C}} := \tilde{\mathcal{O}} \backslash \tilde{\mathcal{P}}$, indexed by $\mathcal{N}_\mathcal{C} := \{1, \ldots, N_C\} \subset \mathbb{N}$, are assumed to be strongly convex

[1]https://github.com/KodlabPenn/semnav_matlab

[2]https://github.com/KodlabPenn/semnav

[3]For our hardware implementation, this idealized sensor is reduced to a combination of a LIDAR for distance measurements to obstacles and a monocular camera for object recognition and pose identification.

according to [5, Assumption 2], but are otherwise completely unknown to the robot.

To simplify the notation, we dilate each obstacle by r, and assume that the robot operates in the freespace $\mathcal{F}$. We denote the set of dilated obstacles derived from $\tilde{\mathcal{O}}, \tilde{\mathcal{P}}$ and $\tilde{\mathcal{C}}$, by $\mathcal{O}, \mathcal{P}$ and $\mathcal{C}$ respectively. Then, similarly to [2], [11], [20], we describe each polygonal obstacle $P_i \in \mathcal{P} \subseteq \mathcal{O}$ by an *obstacle function*, $\beta_i(\mathbf{x})$, a real-valued map providing an implicit representation of the form $P_i = \{\mathbf{x} \in \mathbb{R}^2 \,|\, \beta_i(\mathbf{x}) \leq 0\}$ that the robot can construct online after it has localized P_i, following [24]. We also require the following assumptions.

Assumption 1 *1) Each obstacle $C_i \in \mathcal{C}$ has a positive clearance $d(C_i, C_j) > 0$ from any obstacle $C_j \in \mathcal{C}$, $j \neq i$. Also, $d(C_i, \partial\mathcal{F}) > 0$, $\forall C_i \in \mathcal{C}$.*

2) For each $P_i \in \mathcal{P}$, there exists $\varepsilon_i > 0$ such that the set $S_{\beta_i} := \{\mathbf{x} \,|\, \beta_i(\mathbf{x}) \leq \varepsilon_i\}$ has a positive clearance $d(S_{\beta_i}, C) > 0$ from any obstacle $C \in \mathcal{C}$.

Based on these assumptions and considering first-order dynamics $\dot{\mathbf{x}} = \mathbf{u}(\mathbf{x})$, the problem consists of finding a Lipschitz continuous controller $\mathbf{u} : \mathcal{F} \to \mathbb{R}^2$, that leaves the path-connected freespace $\mathcal{F}$ positively invariant and steers the robot to the (possibly moving) goal $\mathbf{x}_d \in \mathcal{F}$.

B. Overview of the Solution

We solve the aforementioned problem by interpolating a sequence of spaces between the physical and a topologically equivalent but geometrically simple model space, designing our control input in the model space, and transforming this input through the inverse of the diffeomorphism between the mapped and the model space (Section III) to find the commands in the physical space (Section IV).

To this end, we arrive at the central formal results (Proposition 1, Theorems 1-2) by employing the smooth "switch" and "deforming factor" construction given in the technical report [25, Eqns. (8) and (9) respectively], integrated into the prior hybrid systems navigational framework from [11], that had, in turn, relied on the method for generating differential drive inputs from [20]. Detailed descriptions of all these components are included, for readers' convenience, in the technical report [25, Appendices I, II].

C. Environment Representation and Technical Notation

Here, we establish notation for the four distinct representations of the environment that we will refer to as *planning spaces* [11], [20], as shown in Fig. 2. The robot navigates the physical space and discovers obstacles, that are dilated by the robot radius r and stored in the semantic space. Potentially overlapping obstacles in the semantic space are subsequently consolidated in real time to form the mapped space. A change of coordinates from this space is then employed to construct a geometrically simplified (but topologically equivalent) model space, by merging familiar obstacles overlapping with the boundary of the enclosing freespace $\partial\mathcal{F}_e$ to $\partial\mathcal{F}_e$, deforming other familiar obstacles to disks, and leaving unknown obstacles intact.

1) Physical Space: The *physical space* is a description of the actual workspace $\mathcal{W}_e$ punctured with the obstacles $\tilde{\mathcal{O}}$, and is inaccessible to the robot. The robot navigates this space toward $\mathbf{x}_d$, and discovers and localizes new obstacles along the way. We denote by $\tilde{\mathcal{P}}_{\mathcal{I}} := \{\tilde{P}_i\}_{i\in\mathcal{I}} \subseteq \tilde{\mathcal{P}}$ the set of physically "instantiated" familiar objects, i.e., all objects whose geometry and pose are known to the robot either beforehand (when considering, e.g., a known wall layout), or by performing online localization at execution time, using semantic mapping. This set is indexed by a set $\mathcal{I} \subseteq \mathcal{N}_{\mathcal{P}}$, also defining the modes of our hybrid controller (Section IV).

2) Semantic Space: The *semantic space* $\mathcal{F}^{\mathcal{I}}_{sem}$ describes the robot's constantly updated information about the environment, from the observable portions of a subset of unrecognized obstacles, together with the $|\mathcal{I}|$ instantiated familiar obstacles. We denote the *set of unrecognized obstacles in the semantic space* by $\mathcal{C}_{sem} := \{C_i\}_{i\in\mathcal{J}_{\mathcal{C}}}$, indexed by $\mathcal{J}_{\mathcal{C}} \subseteq \mathcal{N}_{\mathcal{C}}$, and the *set of familiar obstacles in the semantic space* by $\mathcal{P}^{\mathcal{I}}_{sem} := \bigsqcup_{i\in\mathcal{I}} P_i$. Here the robot is treated as a single point.

3) Mapped Space: The semantic space does not explicitly contain any topological information about the explored environment, since Assumption 1 does not exclude overlaps between obstacles in $\mathcal{P}$. Hence, we form the *mapped space*, $\mathcal{F}^{\mathcal{I}}_{map}$, by taking unions of elements of $\mathcal{P}^{\mathcal{I}}_{sem}$, making up a new set of *consolidated familiar obstacles* $\mathcal{P}^{\mathcal{I}}_{map} := \{P_i\}_{i\in\mathcal{J}^{\mathcal{I}}}$ indexed by $\mathcal{J}^{\mathcal{I}}$, with $|\mathcal{J}^{\mathcal{I}}| \leq |\mathcal{I}|$, along with copies of the unknown obstacles (i.e., $\mathcal{C}_{map} := \mathcal{C}_{sem}$).

Next, for any connected component P of $\mathcal{P}^{\mathcal{I}}_{map}$ that intersects the boundary of the enclosing freespace $\partial\mathcal{F}_e$, we take $B := P \cap \mathcal{F}_e$ and include B in a new set $\mathcal{B}^{\mathcal{I}}_{map}$, indexed by $\mathcal{J}^{\mathcal{I}}_{\mathcal{B}}$. The rest of the connected components in $\mathcal{P}^{\mathcal{I}}_{map}$, which do not intersect $\partial\mathcal{F}_e$, are included in a set $\mathcal{D}^{\mathcal{I}}_{map}$, indexed by $\mathcal{J}^{\mathcal{I}}_{\mathcal{D}}$. The idea here is that obstacles in $\mathcal{B}^{\mathcal{I}}_{map}$ should be merged to $\partial\mathcal{F}_e$, and obstacles in $\mathcal{D}^{\mathcal{I}}_{map}$ should be deformed to disks, since $\mathcal{F}^{\mathcal{I}}_{map}$ and $\mathcal{F}^{\mathcal{I}}_{model}$ need to be diffeomorphic.

4) Model Space: The *model space* $\mathcal{F}^{\mathcal{I}}_{model}$ is a topologically equivalent but geometrically simplified version of the mapped space $\mathcal{F}^{\mathcal{I}}_{map}$. It has the same boundary as $\mathcal{F}_e$ and consists of copies of the sensed fragments of the $|\mathcal{J}_{\mathcal{C}}|$ unrecognized visible obstacles in $\mathcal{C}_{map}$, and a collection of $|\mathcal{J}^{\mathcal{I}}_{\mathcal{D}}|$ Euclidean disks corresponding to the $|\mathcal{J}^{\mathcal{I}}_{\mathcal{D}}|$ consolidated obstacles in $\mathcal{D}^{\mathcal{I}}_{map}$ that are deformed to disks. Obstacles in $\mathcal{B}^{\mathcal{I}}_{map}$ are merged into $\partial\mathcal{F}_e$, to make $\mathcal{F}^{\mathcal{I}}_{map}$ and $\mathcal{F}^{\mathcal{I}}_{model}$ topologically equivalent, through a map $\mathbf{h}^{\mathcal{I}}$, described next.

III. DIFFEOMORPHISM CONSTRUCTION

Here, we describe our method of constructing the diffeomorphism, $\mathbf{h}^{\mathcal{I}}$, between $\mathcal{F}^{\mathcal{I}}_{map}$ and $\mathcal{F}^{\mathcal{I}}_{model}$. We assume that the robot has recognized and localized the $|\mathcal{J}^{\mathcal{I}}|$ obstacles in $\mathcal{P}^{\mathcal{I}}_{map}$, and has, therefore, identified obstacles to be merged to the boundary of the enclosing freespace $\partial\mathcal{F}_e$, stored in $\mathcal{B}^{\mathcal{I}}_{map}$, and obstacles to be deformed to disks, stored in $\mathcal{D}^{\mathcal{I}}_{map}$.

A. Obstacle Representation and Convex Decomposition

As a natural extension to doubly reactive algorithms for environments cluttered with convex obstacles [3], [5], we assume that the robot has access to the convex decomposition

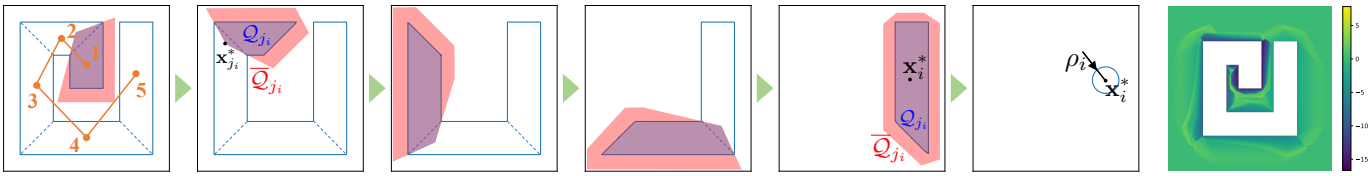

Fig. 3: Diffeomorphism construction via direct convex decomposition: Any arbitrary convex decomposition (e.g., [22]) defines a tree $\mathcal{T}_{P_i} := (\mathcal{V}_{P_i}, \mathcal{E}_{P_i})$ (left), which induces the sequence of purging transformations that map the polygon's boundary and exterior to the boundary and exterior of an equivalent disk. The purging transformation for each convex piece $j_i \in \mathcal{V}_{P_i}$ is defined by a pair of convex polygons $\mathcal{Q}_{j_i}, \overline{\mathcal{Q}}_{j_i}$ that limit the effect of the diffeomorphism to a neighborhood of j_i. The final map is guaranteed to be smooth, as shown by a visualization of its determinant in logarithmic scale (right).

of each obstacle $P \in \mathcal{P}^{\mathcal{I}}_{map}$. For polygons without holes, we are interested in decompositions that do not introduce Steiner points (i.e., additional points except for the polygon vertices), as this guarantees the dual graph of the convex partition to be a tree. Here, we acquire this convex decomposition using Greene's method [22] and its C++ implementation in CGAL [26], operating in $\mathcal{O}(r^2 n^2)$ time, with n the number of polygon vertices r the number of reflex vertices.

As shown in Fig. 3, convex partioning results in a *tree of convex polygons* $\mathcal{T}_{P_i} := (\mathcal{V}_{P_i}, \mathcal{E}_{P_i})$ corresponding to P_i, with $\mathcal{V}_{P_i}$ a set of vertices identified with convex polygons (i.e., vertices of the dual of the formal partition) and $\mathcal{E}_{P_i}$ a set of edges encoding polygon adjacency. Therefore, we can pick any polygon as root and construct $\mathcal{T}_{P_i}$ based on the adjacency properties induced by the dual graph of the decomposition, as shown in Fig. 3. If $P_i \in \mathcal{D}^{\mathcal{I}}_{map}$, we pick as root the polygon with the largest surface area, whereas if $P_i \in \mathcal{B}^{\mathcal{I}}_{map}$, we pick as root any polygon adjacent to $\partial\mathcal{F}_e$.

B. The Map Between the Mapped and the Model Space

As shown in Fig. 3, the map $\mathbf{h}^{\mathcal{I}}$ between the mapped and the model space is constructed in several steps, involving the successive application of purging transformations by composition, during execution time, for all leaf polygons of all obstacles P in $\mathcal{B}^{\mathcal{I}}_{map}$ and $\mathcal{D}^{\mathcal{I}}_{map}$, in any order, until their root polygons are reached. We denote by $\hat{\mathcal{F}}^{\mathcal{I}}_{map}$ this final intermediate space, where all obstacles in $\mathcal{F}^{\mathcal{I}}_{map}$ have been deformed to their root polygons. We denote by $\mathcal{F}^{\mathcal{I}}_{map,j_i}$ and $\mathcal{F}^{\mathcal{I}}_{map,p(j_i)}$ the intermediate spaces before and after the purging transformation of leaf polygon $j_i \in \mathcal{V}_{P_i}$ respectively.

We begin our exposition with a description of the purging transformation $\mathbf{h}^{\mathcal{I}}_{j_i} : \mathcal{F}^{\mathcal{I}}_{map,j_i} \to \mathcal{F}^{\mathcal{I}}_{map,p(j_i)}$ that maps the boundary of a leaf polygon $j_i \in \mathcal{V}_{P_i}$ onto the boundary of its parent, $p(j_i)$, and continue with a description of the map $\hat{\mathbf{h}}^{\mathcal{I}} : \hat{\mathcal{F}}^{\mathcal{I}}_{map} \to \mathcal{F}^{\mathcal{I}}_{model}$ that maps the boundaries of root polygons of obstacles in $\mathcal{B}^{\mathcal{I}}_{map}$ and $\mathcal{D}^{\mathcal{I}}_{map}$ to $\mathcal{F}_e$ and the corresponding disks in $\mathcal{F}^{\mathcal{I}}_{model}$ respectively.

1) The map between $\mathcal{F}^{\mathcal{I}}_{map,j_i}$ and $\mathcal{F}^{\mathcal{I}}_{map,p(j_i)}$: We first find admissible centers $\mathbf{x}^*_{j_i}$, and polygonal collars $\overline{\mathcal{Q}}_{j_i}$, that encompass the actual polygon $\mathcal{Q}_{j_i}$, and limit the effect of the purging transformation in their interior, while keeping its value equal to the identity everywhere else (see Fig. 3).

Definition 1 An admissible center for the purging transformation of the leaf polygon $j_i \in \mathcal{V}_{P_i}$, denoted by $\mathbf{x}^*_{j_i}$, is a point in $p(j_i)$ such that the polygon $\mathcal{Q}_{j_i}$ with vertices the original vertices of j_i and $\mathbf{x}^*_{j_i}$ is convex.

Definition 2 An admissible polygonal collar for the purging transformation of the leaf polygon j_i is a convex polygon $\overline{\mathcal{Q}}_{j_i}$ such that:

1) $\overline{\mathcal{Q}}_{j_i}$ does not intersect the interior of any polygon $k \in \mathcal{V}_P$ with $k \neq j_i, p(j_i), \forall P \in \mathcal{F}^{\mathcal{I}}_{map,j_i}$, or any $C \in \mathcal{C}_{map}$.
2) $\mathcal{Q}_{j_i} \subset \overline{\mathcal{Q}}_{j_i}$, and $\overline{\mathcal{Q}}_{j_i} \backslash \mathcal{Q}_{j_i} \subset \mathcal{F}^{\mathcal{I}}_{map,j_i}$.

Examples are shown in Fig. 3. As in [11], we also construct implicit functions $\gamma_{j_i}(\mathbf{x}), \delta_{j_i}(\mathbf{x})$ corresponding to the leaf polygon $j_i \in \mathcal{V}_{P_i}$ such that $\mathcal{Q}_{j_i} = \{\mathbf{x} \in \mathbb{R}^2 \,|\, \gamma_{j_i}(\mathbf{x}) \leq 0\}$ and $\overline{\mathcal{Q}}_{j_i} = \{\mathbf{x} \in \mathbb{R}^2 \,|\, \delta_{j_i}(\mathbf{x}) \geq 0\}$, using tools from [24].

Based on these definitions, we construct the C^∞ *switch of the purging transformation for the leaf polygon* $j_i \in \mathcal{V}_{P_i}$ as a function $\sigma_{j_i} : \mathcal{F}^{\mathcal{I}}_{map,j_i} \to \mathbb{R}$, equal to 1 on the boundary of $\mathcal{Q}_{j_i}$, equal to 0 outside $\overline{\mathcal{Q}}_{j_i}$ and smoothly varying (except the polygon vertices) between 0 and 1 everywhere else (see [25, Eqn. (8)] in [25, Appendix II]). Finally, we define the *deforming factors* as the functions $\nu_{j_i} : \mathcal{F}^{\mathcal{I}}_{map,j_i} \to \mathbb{R}$, responsible for mapping the boundary of the leaf polygon j_i onto the boundary of its parent $p(j_i)$ (see [25, Eqn. (9)] in [25, Appendix II]). We can now construct the map between $\mathcal{F}^{\mathcal{I}}_{map,j_i}$ and $\mathcal{F}^{\mathcal{I}}_{map,p(j_i)}$ as in [11], [20]

$$\mathbf{h}^{\mathcal{I}}_{j_i}(\mathbf{x}) := \sigma_{j_i}(\mathbf{x}) \left(\mathbf{x}^*_{j_i} + \nu_{j_i}(\mathbf{x})(\mathbf{x} - \mathbf{x}^*_{j_i})\right) + (1 - \sigma_{j_i}(\mathbf{x}))\, \mathbf{x}$$

Proposition 1 *The map $\mathbf{h}^{\mathcal{I}}_{j_i}$ is a C^∞ diffeomorphism between $\mathcal{F}^{\mathcal{I}}_{map,j_i}$ and $\mathcal{F}^{\mathcal{I}}_{map,p(j_i)}$ away from the polygon vertices of j_i, none of which lies in the interior of $\mathcal{F}^{\mathcal{I}}_{map,j_i}$.*

Proof. Included in [25, Appendix II]. ■

We denote by $\mathbf{g}^{\mathcal{I}} : \mathcal{F}^{\mathcal{I}}_{map} \to \hat{\mathcal{F}}^{\mathcal{I}}_{map}$ the map between $\mathcal{F}^{\mathcal{I}}_{map}$ and $\hat{\mathcal{F}}^{\mathcal{I}}_{map}$, arising from the composition of purging transformations $\mathbf{h}^{\mathcal{I}}_{j_i} : \mathcal{F}^{\mathcal{I}}_{map,j_i} \to \mathcal{F}^{\mathcal{I}}_{map,p(j_i)}$.

2) The Map Between $\hat{\mathcal{F}}^{\mathcal{I}}_{map}$ and $\mathcal{F}^{\mathcal{I}}_{model}$: Here, for each root polygon r_i, we define the polygonal collar and the C^∞ *switch of the transformation* $\sigma_{r_i} : \hat{\mathcal{F}}^{\mathcal{I}}_{map} \to \mathcal{F}^{\mathcal{I}}_{map}$ as in Definition 2 and [25, Eqn. (8)] (see [25, Appendix II]) respectively, and we distinguish between obstacles in $\mathcal{B}^{\mathcal{I}}_{map}$ and in $\mathcal{D}^{\mathcal{I}}_{map}$ for the definition of the centers (see Fig. 3).

Definition 3 An admissible center for the transformation of:

1) the root polygon r_i, corresponding to $P_i \in \mathcal{D}^{\mathcal{I}}_{map}$, is a point $\mathbf{x}^*_i$ in the interior of r_i (here identified with $\mathcal{Q}_{r_i}$).
2) the root polygon r_i, corresponding to $P_i \in \mathcal{B}^{\mathcal{I}}_{map}$, is a point $\mathbf{x}^*_i \in \mathbb{R}^2 \backslash \mathcal{F}_e$, such that the polygon $\mathcal{Q}_{r_i}$ with vertices the original vertices of r_i and $\mathbf{x}^*_i$ is convex.

Finally, we define the *deforming factors* $\nu_{r_i} : \hat{\mathcal{F}}^{\mathcal{I}}_{map} \to \mathbb{R}$ as in Section III-B1 for obstacles in $\mathcal{B}^{\mathcal{I}}_{map}$, and as the function

$\nu_{r_i}(\mathbf{x}) := \frac{\rho_i}{||\mathbf{x}-\mathbf{x}_i^*||}$ for obstacles in $\mathcal{D}_{map}^{\mathcal{I}}$ (see Fig. 3). We construct the map between $\hat{\mathcal{F}}_{map}^{\mathcal{I}}$ and $\mathcal{F}_{model}^{\mathcal{I}}$ as

$$\hat{\mathbf{h}}^{\mathcal{I}}(\mathbf{x}) := \sum_{i \in \mathcal{J}_{\mathcal{B}}^{\mathcal{I}} \cup \mathcal{J}_{\mathcal{D}}^{\mathcal{I}}} \sigma_{r_i}(\mathbf{x}) \left[\mathbf{x}_i^* + \nu_{r_i}(\mathbf{x})(\mathbf{x} - \mathbf{x}_i^*)\right] + \sigma_d(\mathbf{x})\mathbf{x}$$

with $\sigma_d(\mathbf{x}) := 1 - \sum_{i \in \mathcal{J}_{\mathcal{B}}^{\mathcal{I}} \cup \mathcal{J}_{\mathcal{D}}^{\mathcal{I}}} \sigma_{r_i}(\mathbf{x})$. It should be noted that Definitions 2 and 3 guarantee that, at any point in the workspace, at most one switch σ_{r_i} will be greater than zero which, in turn, guarantees that the diffeomorphism computation is essentially "local", and allows scaling to multiple obstacles in the mapped space $\mathcal{F}_{map}^{\mathcal{I}}$.

We can similarly arrive at the following result.

Proposition 2 *The map $\hat{\mathbf{h}}^{\mathcal{I}}$ is a C^∞ diffeomorphism between $\hat{\mathcal{F}}_{map}^{\mathcal{I}}$ and $\mathcal{F}_{model}^{\mathcal{I}}$ away from any sharp corners, none of which lie in the interior of $\hat{\mathcal{F}}_{map}^{\mathcal{I}}$.*

3) The Map Between $\mathcal{F}_{map}^{\mathcal{I}}$ and $\mathcal{F}_{model}^{\mathcal{I}}$: Based on the construction of $\mathbf{g}^{\mathcal{I}} : \mathcal{F}_{map}^{\mathcal{I}} \to \hat{\mathcal{F}}_{map}^{\mathcal{I}}$ and $\hat{\mathbf{h}}^{\mathcal{I}} : \hat{\mathcal{F}}_{map}^{\mathcal{I}} \to \mathcal{F}_{model}^{\mathcal{I}}$, we can finally write the map between the mapped space and the model space as the function $\mathbf{h}^{\mathcal{I}} : \mathcal{F}_{map}^{\mathcal{I}} \to \mathcal{F}_{model}^{\mathcal{I}}$ given by $\mathbf{h}^{\mathcal{I}}(\mathbf{x}) = \hat{\mathbf{h}}^{\mathcal{I}} \circ \mathbf{g}^{\mathcal{I}}(\mathbf{x})$. Since both $\mathbf{g}^{\mathcal{I}}$ and $\hat{\mathbf{h}}^{\mathcal{I}}$ are C^∞ diffeomorphisms away from sharp corners, it is straightforward to show that the map $\mathbf{h}^{\mathcal{I}}$ is a C^∞ diffeomorphism between $\mathcal{F}_{map}^{\mathcal{I}}$ and $\mathcal{F}_{model}^{\mathcal{I}}$ away from any sharp corners, none of which lie in the interior of $\mathcal{F}_{map}^{\mathcal{I}}$.

IV. REACTIVE PLANNING ALGORITHM

The analysis in Section III describes the diffeomorphism construction between $\mathcal{F}_{map}^{\mathcal{I}}$ and $\mathcal{F}_{model}^{\mathcal{I}}$ for a given index set $\mathcal{I}$ of instantiated familiar obstacles. However, the onboard sensor might incorporate new obstacles in the semantic map, updating $\mathcal{I}$. Therefore, as in [11], we give a hybrid systems description of our reactive controller, where each mode is defined by an index set $\mathcal{I} \in 2^{\mathcal{N}_{\mathcal{P}}}$ of familiar obstacles stored in the semantic map, the guards describe the sensor trigger events where a previously "unexplored" obstacle is discovered and incorporated in the semantic map (thereby changing $\mathcal{P}_{map}^{\mathcal{I}}$, and $\mathcal{D}_{map}^{\mathcal{I}}$, $\mathcal{B}_{map}^{\mathcal{I}}$) [11, Eqns. (31),(35)], and the resets describe transitions to new modes that are equal to the identity in the physical space, but might result in discrete "jumps" of the robot position in the model space [11, Eqns. (32), (36)]. In this work, this hybrid systems structure is not modified, and we just focus on each mode $\mathcal{I}$ separately.

For a fully actuated particle with dynamics $\dot{\mathbf{x}} = \mathbf{u}(\mathbf{x}), \mathbf{u} \in \mathbb{R}^2$, the control law in each mode $\mathcal{I}$ is given as

$$\mathbf{u}^{\mathcal{I}}(\mathbf{x}) = k \left[D_{\mathbf{x}}\mathbf{h}^{\mathcal{I}}\right]^{-1} \cdot \left(\mathbf{v}^{\mathcal{I}} \circ \mathbf{h}^{\mathcal{I}}(\mathbf{x})\right) \tag{1}$$

with $D_{\mathbf{x}}$ denoting the derivative operator with respect to $\mathbf{x}$, and the control input in the model space given as [5]

$$\mathbf{v}^{\mathcal{I}}(\mathbf{y}) = -\left(\mathbf{y} - \Pi_{\mathcal{LF}(\mathbf{y})}(\mathbf{y}_d)\right) \tag{2}$$

Here, $\mathbf{y} = \mathbf{h}^{\mathcal{I}}(\mathbf{x}) \in \mathcal{F}_{model}^{\mathcal{I}}$ and $\mathbf{y}_d = \mathbf{h}^{\mathcal{I}}(\mathbf{x}_d)$ denote the robot and goal position in the model space respectively, and $\Pi_{\mathcal{LF}(\mathbf{y})}(\mathbf{y}_d)$ denotes the projection onto the convex *local freespace* for $\mathbf{y}$, $\mathcal{LF}(\mathbf{y})$, defined as the Voronoi cell in [5, Eqn. (25)], separating $\mathbf{y}$ from all the model space obstacles (see Fig. 2). We use the following definition to define a slowly moving, non-adversarial moving target.

Definition 4 The smooth function $\mathbf{x}_d : \mathbb{R} \to \mathcal{F}_{map}^{\mathcal{I}}$ is a *non-adversarial target* if its model space velocity, given as $\dot{\mathbf{y}}_d := D_{\mathbf{x}}\mathbf{h}^{\mathcal{I}}(\mathbf{x}_d) \cdot \dot{\mathbf{x}}_d$, always satisfies either $(\mathbf{h}^{\mathcal{I}}(\mathbf{x}) - \mathbf{h}^{\mathcal{I}}(\mathbf{x}_d))^\top \dot{\mathbf{y}}_d \geq 0$, or $||\dot{\mathbf{y}}_d|| \leq k \frac{||\mathbf{h}^{\mathcal{I}}(\mathbf{x}) - \Pi_{\mathsf{B}(\mathbf{h}^{\mathcal{I}}(\mathbf{x}), 0.5d(\mathbf{h}^{\mathcal{I}}(\mathbf{x}), \partial\mathcal{F}_{model}^{\mathcal{I}}))}(\mathbf{h}^{\mathcal{I}}(\mathbf{x}_d))||^2}{||\mathbf{h}^{\mathcal{I}}(\mathbf{x}) - \mathbf{h}^{\mathcal{I}}(\mathbf{x}_d)||}$.

Intuitively, this Definition requires the moving target to slow down when the robot gets too close to obstacles (i.e., when $d(\mathbf{h}^{\mathcal{I}}(\mathbf{x}), \partial\mathcal{F}_{model}^{\mathcal{I}})$ becomes small) or the target itself (i.e., when $\Pi_{\mathsf{B}(\mathbf{h}^{\mathcal{I}}(\mathbf{x}), 0.5d(\mathbf{h}^{\mathcal{I}}(\mathbf{x}), \partial\mathcal{F}_{model}^{\mathcal{I}}))}(\mathbf{h}^{\mathcal{I}}(\mathbf{x}_d)) = \mathbf{h}^{\mathcal{I}}(\mathbf{x}_d)$), proportionally to the control gain k, unless the target approaches the robot (i.e., $(\mathbf{h}^{\mathcal{I}}(\mathbf{x}) - \mathbf{h}^{\mathcal{I}}(\mathbf{x}_d))^\top \dot{\mathbf{y}}_d \geq 0$). We use Definition 4 to arrive at the following central result.

Theorem 1 *With $\mathcal{I}$ the terminal mode of the hybrid controller[4], the reactive controller in* (1) *leaves the freespace $\mathcal{F}_{map}^{\mathcal{I}}$ positively invariant, and:*

1) *tracks $\mathbf{x}_d$ by not increasing $||\mathbf{h}^{\mathcal{I}}(\mathbf{x}) - \mathbf{h}^{\mathcal{I}}(\mathbf{x}_d)||$, if $\mathbf{x}_d$ is a non-adversarial target (see Definition 4).*
2) *asymptotically reaches a constant $\mathbf{x}_d$ with its unique continuously differentiable flow, from almost any placement $\mathbf{x} \in \mathcal{F}_{map}^{\mathcal{I}}$, while strictly decreasing $||\mathbf{h}^{\mathcal{I}}(\mathbf{x}) - \mathbf{h}^{\mathcal{I}}(\mathbf{x}_d)||$ along the way.*

Proof. Included in [25, Appendix II]. ■

In [20], we extended our algorithm to differential drive robots, by constructing a smooth diffeomorphism $\overline{\mathbf{h}}^{\mathcal{I}} : \mathcal{F}_{map}^{\mathcal{I}} \times S^1 \to \mathcal{F}_{model}^{\mathcal{I}} \times S^1$ away from sharp corners. We include the details of the construction in [25, Appendix I], and present our main result below, whose proof is similar to that of Theorem 1 and is omitted for brevity.

Theorem 2 *With $\mathcal{I}$ the terminal mode of the hybrid controller[4], the reactive controller for differential drive robots (see [25, Eqn. (3)] in [25, Appendix I]) leaves the freespace $\mathcal{F}_{map}^{\mathcal{I}} \times S^1$ positively invariant, and:*

1) *tracks $\mathbf{x}_d$ by not increasing $||\mathbf{h}^{\mathcal{I}}(\mathbf{x}) - \mathbf{h}^{\mathcal{I}}(\mathbf{x}_d)||$, if $\mathbf{x}_d$ is a non-adversarial target (see Definition 4).*
2) *asymptotically reaches a constant $\mathbf{x}_d$ with its unique continuously differentiable flow, from almost any robot configuration in $\mathcal{F}_{map}^{\mathcal{I}} \times S^1$, without increasing $||\mathbf{h}^{\mathcal{I}}(\mathbf{x}) - \mathbf{h}^{\mathcal{I}}(\mathbf{x}_d)||$ along the way.*

V. NUMERICAL STUDIES

In this Section, we present numerical studies run in MATLAB using `ode45`, that illustrate our formal results. Our reactive controller is implemented in Python and communicates with MATLAB using the standard MATLAB-Python interface. For our numerical results, we assume perfect robot state estimation and localization of obstacles, using a fixed range sensor that can instantly identify and localize either the entirety of familiar obstacles that intersect its footprint, or the fragments of unknown obstacles within its range.

[4]The *terminal mode* of the hybrid system is indexed by the improper subset, $\mathcal{I} = \mathcal{N}_{\mathcal{P}}$, where all familiar obstacles in the workspace have been instantiated in the set $\mathcal{P}_{sem}^{\mathcal{I}}$.

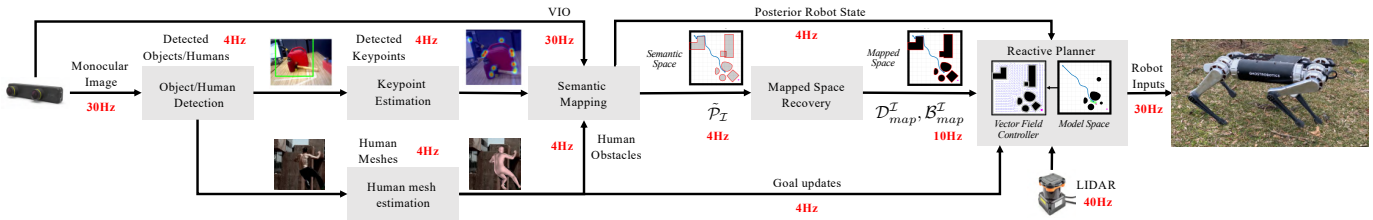

Fig. 4: The online reactive planning architecture: Advancing beyond [11], camera output is run through a perceptual pipeline incorporating three separate neural networks (run onboard at 4Hz) whose function is to: (a) detect familiar obstacles and humans [27]; (b) localize corresponding semantic keypoints [9]; and (c) perform a 3D human mesh estimation [21]. Keypoint locations on the image, other detected geometric features, and an egomotion estimate provided by visual inertial odometry are used by the semantic mapping module [10] to give updated robot ($\mathbf{x}$) and obstacle poses ($\tilde{\mathcal{P}}_{\mathcal{I}}$). The reactive planner, now streamlined to run onboard at 3x the rate of the corresponding module in [11], merges consolidated obstacles in $\mathcal{D}^{\mathcal{I}}_{map}, \mathcal{B}^{\mathcal{I}}_{map}$ (recovered from $\tilde{\mathcal{P}}_{\mathcal{I}}$), along with LIDAR data for unknown obstacles, to provide the robot inputs and close the control loop. In this new architecture, the estimated human meshes are used to update the target's position in the reported human tracking experiments, detect a specific human gesture or pose related to the experiment's semantics, or (optionally) introduce additional obstacles in the semantic mapping module for some out-of-scope experiments.

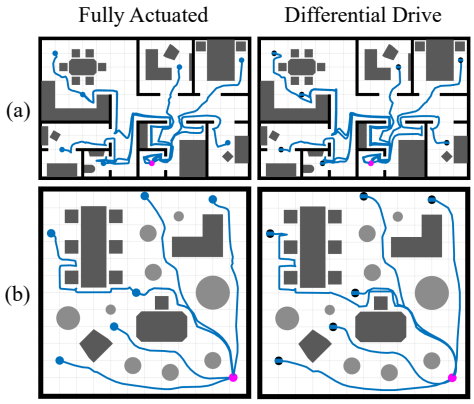

Fig. 5: Top: Navigation in an indoor layout cluttered with multiple familiar obstacles and previously unknown pose. - Bottom: Navigation in a room cluttered with known non-convex (dark grey) and unknown convex (light grey) obstacles. Simulations are run from different initial conditions.

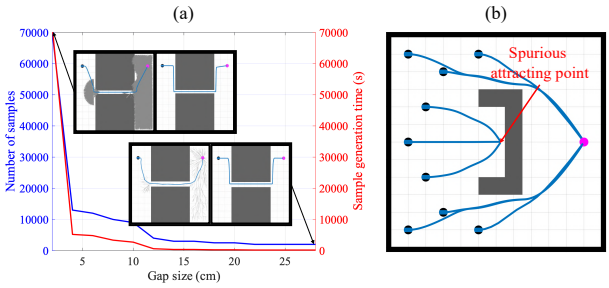

Fig. 6: (a) Minimum number of (offline computed) samples needed for successful online implementation of RRT^X [23] in an unexplored environment with two familiar obstacles forming a narrow passage. The number becomes increasingly large as the gap becomes smaller. The robot diameter is 50cm. (b) Illustration of a graceful failure of our proposed algorithm. The sole non-convex but unknown encountered obstacle creates a spurious attracting equilibrium state that traps a subset of initial conditions. However, collision avoidance is always guaranteed by the onboard sensor.

A. Illustrations of the Navigation Framework

We begin by illustrating the performance of our reactive planning framework in two different settings (Fig. 5), for both a fully actuated and a differential drive robot, also included in the accompanying video submission. In the first case (Fig. 5-a), the robot is tasked with moving to a predefined location in an environment resembling an apartment layout with known walls, cluttered with several familiar obstacles of unknown location and pose, from different initial conditions. In the second case (Fig. 5-b), the robot navigates a room cluttered with both familiar and unknown obstacles from several initial conditions. In both cases, the robot avoids all the obstacles and safely converges to the target. The robot radius used in our simulation studies is 0.2m.

B. Comparison with RRT^X [23]

In the second set of numerical results, we compare our reactive controller with a state-of-the-art path replanning algorithm, RRT^X [23]. We choose to compare against this specific algorithm instead of another sampling-based method for static environments (e.g., RRT* [6]), since both our reactive controller and RRT^X are dynamic in nature; they are capable of incorporating new information about the environment and modifying the robot's behavior appropriately. For our simulations, we assume that RRT^X possesses the same sensory apparatus with our algorithm; an "oracle" that can instantly identify and localize nearby obstacles. The computed paths are then reactively tracked using [28].

Fig. 6-a exemplifies the (well-known [29]) performance degradation of RRT^X in the presence of narrow passages: as the corridor narrows (while always larger than the robot's diameter), the minimum number of (offline-computed) samples needed for successful replanning and safe navigation increases in a nonlinear manner. In consequence of this dramatically growing time-to-completeness, our video demonstrates a potentially catastrophic failure of the associated replanner: in the presence of multiple narrow passages, it cycles repeatedly as it searches for possible alternative openings, before eventually (and only after increasingly protracted cycling) reporting failure (incorrectly) and halting. On the contrary, our algorithm always guarantees safe passage to the target through compliant environments – and Fig. 6-b illustrates its graceful failure for settings that violate Assumption 1. The non-compliant (novel but not convex) obstacle creates an attracting equilibrium state that traps a set of initial conditions whose area becomes arbitrarily large as its "shadow" (the associated basin of attraction) grows. However, the presence of a Lyapunov function precludes the possibility of any cycling behavior: failure to achieve the goal (and the diagnosis of a non-compliant environment) is readily identified.

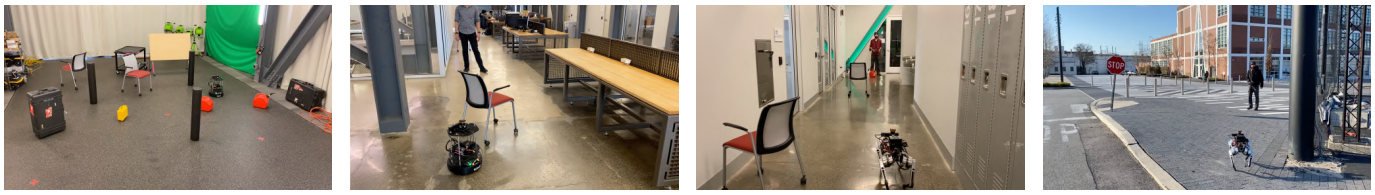

Fig. 7: Types of environments used in our experiments. Visual context is included in the supplementary video submission.

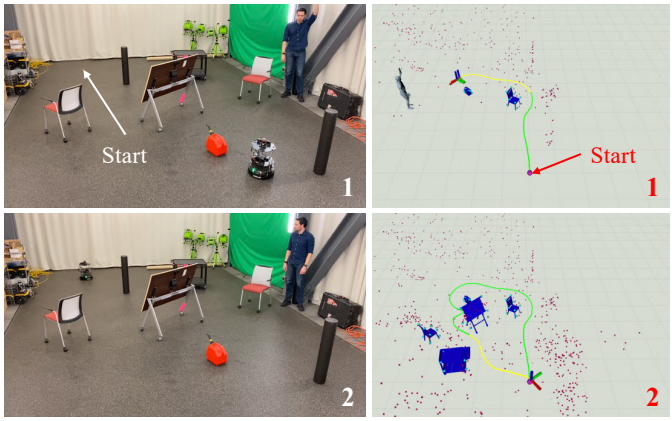

Fig. 8: Top: Turtlebot reactively follows a human until a stop gesture is given and detected – Bottom: Turtlebot safely returns to its starting position.

VI. EXPERIMENTS

A. Experimental Setup

Our experimental layout is summarized in Fig. 4. Since the algorithms introduced in this paper take the form of first-order vector fields, we mainly use a quasi-static platform, the Turtlebot robot [30] for our physical experiments. We suggest the robustness of these feedback controllers by performing several experiments on the more dynamic Ghost Spirit legged robot [8], using a rough approximation to the quasi-static differential drive motion model. In both cases, the main computer is an Nvidia Jetson AGX Xavier GPU unit, responsible for running our perception and navigation algorithms, during execution time. This GPU unit communicates with a Hokuyo LIDAR, used to detect unknown obstacles, and a ZED Mini stereo camera, used for visual-inertial state estimation and for detecting humans and familiar obstacles.

Our perception pipeline, run onboard the Nvidia Jetson AGX Xavier at 4Hz, supports the detection and 3D pose estimation of objects and humans, who, for the purposes of this paper, are used as moving targets. We use the YOLOv3 detector [27] to detect 2D bounding boxes on the image which are then processed based on the class of the detected object. If one of the specified object classes is detected, then we follow the semantic keypoints approach of [9] to estimate keypoints of the object on the image plane[5]. The familiar object classes (as defined in Section II) used in our experiments are chair, table, ladder, cart, gascan and pelican case, although this dictionary can increase depending on the user's needs. The training data for the particular instances of interest are collected with a semi-automatic procedure, similarly to [9]. Given the bounding box and keypoint annotations for each image, the two networks are trained with their default configurations until convergence. On the other hand, if the bounding box corresponds to a person detection, then we use the approach of [21], that provides us with the 3D mesh of the detected person.

Our semantic mapping infrastructure relies on the algorithm presented in [10], and is implemented in C++. This algorithm fuses inertial information (here provided by the position tracking implementation from StereoLabs on the ZED Mini stereo camera), geometric (i.e., geometric features on the 2D image), and semantic information (i.e., the detected keypoints and the associated object labels as described above) to give a posterior estimate for both the robot state and the associated poses of all tracked objects, by simultaneously solving the data association problem arising when several objects of the same class exist in the map.

Finally, our reactive controller, running online and onboard the Nvidia Jetson AGX Xavier GPU unit at 30Hz, is also implemented in C++ using Boost Geometry [31] for the underlying polygon operations, and communicates with our perception pipelines using ROS, as shown in Fig. 4.

B. Empirical Results

As also reported in the supplementary video, we distinguish between two classes of physical experiments in several different environments shown in Fig. 7; tracking either a predefined static target or a moving human, and tracking a given semantic target (e.g., approach a desired object).

1) Geometric tracking of a (moving) target amidst obstacles: Fig. 1 shows Spirit tracking a human in a previously unexplored environment, cluttered with both catalogued obstacles (whose number and placement is unknown in advance) as well as completely unknown obstacles. The robot uses familiar obstacles to both localize itself against them [10] and reactively navigate around them. Fig. 7 summarizes the wide diversity of à-priori unexplored environments, with different lighting conditions, successfully navigated indoors (by Turtlebot and Spirit) and outdoors (by Spirit), while tracking humans[6]. along thousands of body lengths.

Note that the formal results of Section IV require that unknown obstacles be convex. However, here we clutter the environment with a mix of unknown obstacles – some convex, but others of more complicated non-convex shapes (e.g., unknown walls) – to establish empirical robustness in urban environments that are out of scope of the underlying theory.

[5]Note that while both the YOLOv3 detector [27] and the keypoint estimation algorithm [9] are empirically very robust (e.g., particularly against partial occlusions), they could be easily replaced with other state-of-the-art algorithms that provide reasonable robustness against partial occlusions.

[6]Collision avoidance when the robot gets close to the tracked human is guaranteed with the use of the onboard LIDAR; the human is treated as an unknown obstacle and the robot tries to keep separation and avoid collision (with formal guarantees assuming the conditions of Definition 4)

In such settings, that move beyond the formal assumptions outlined in Section II, the robot might converge to undesired local minima behind non-convex obstacles from a subset of (unfavorable) initial conditions (see Fig. 6-b); however, collision avoidance is still guaranteed by the onboard LIDAR.

As anticipated, the few failures we recorded were associated with the inability of the SLAM algorithm to localize the robot in long, featureless environments. However, it should be noted that even when the robot or object localization process fails, collision avoidance is still guaranteed with the use of the onboard LIDAR. Nevertheless, collisions could result with obstacles that cannot be detected by the 2D horizontal LIDAR (e.g., the red gascan shown in Fig. 8). One could still think of extensions to the presented sensory infrastructure (e.g., the use of a 3D LIDAR) that could at least still guarantee safety under such circumstances.

2) Logical reaction using predefined semantics: In the second set of experimental runs, we exploit the new online semantic capabilities to introduce logic in our reactive tracking process. For example, Fig. 8 depicts a tracking task requiring the robot to respond to the human's stop signal (raised left or right hand) by returning to its starting position. The supplementary video presents several other semantically specified tasks requiring autonomous reactions of both a logical as well as geometric nature (all involving negotiation of novel environments from the arbitrary geometric circumstances associated with different contexts of logical triggers).

VII. CONCLUSION AND FUTURE WORK

This paper presents a reactive planner that can provably safely semantically engage non-adversarial moving targets in planar workspaces, cluttered with an arbitrary mix of catalogued obstacles, using both a wheeled robot and a dynamic legged platform. Future work seeks to extend past hierarchical mobile manipulation schemes using early versions of this architecture [32] to incorporate both more dexterous manipulation [33] as well as logically complex abstract specification (e.g., using temporal logic [34]). In the longer term, we believe that concepts from the literature on convex decomposition of polyhedra [35] may afford a generalization beyond our present restriction to 2D workspaces toward the challenge of navigating partially known environments in higher dimension.

REFERENCES

[1] M. Farber, "Topology of Robot Motion Planning," *Morse Theor. Methods in Nonl. Anal. and in Sympl. Topology*, pp. 185–230, 2006.

[2] E. Rimon and D. E. Koditschek, "Exact Robot Navigation Using Artificial Potential Functions," *IEEE Trans. Robotics and Automation*, vol. 8, no. 5, pp. 501–518, 1992.

[3] S. Paternain, D. E. Koditschek, and A. Ribeiro, "Navigation Functions for Convex Potentials in a Space with Convex Obstacles," *IEEE Trans. Automatic Control*, 2017.

[4] B. D. Ilhan, A. M. Johnson, and D. E. Koditschek, "Autonomous legged hill ascent," *J. Field Robotics*, vol. 35 (5), pp. 802–832, 2018.

[5] O. Arslan and D. E. Koditschek, "Sensor-Based Reactive Navigation in Unknown Convex Sphere Worlds," *Int. J. Robotics Research*, vol. 38, no. 1-2, pp. 196–223, Jul 2018.

[6] S. Karaman and E. Frazzoli, "High-speed flight in an ergodic forest," in *IEEE Int. Conf. Robotics and Automation*, 2012, pp. 2899–2906.

[7] I. Noreen, A. Khan, and Z. Habib, "Optimal path planning using RRT* based approaches: a survey and future directions," *Int. J. Advanced Computer Science and Applications*, vol. 7, no. 11, pp. 97–107, 2016.

[8] Ghost Robotics, "Spirit 40." [Online]. Available: http://ghostrobotics.io

[9] G. Pavlakos, X. Zhou, A. Chan, K. G. Derpanis, and K. Daniilidis, "6-DoF object pose from semantic keypoints," in *IEEE Int. Conf. Robotics and Automation*, May 2017, pp. 2011–2018.

[10] S. L. Bowman, N. Atanasov, K. Daniilidis, and G. J. Pappas, "Probabilistic data association for semantic SLAM," in *IEEE Int. Conf. Robotics and Automation*, May 2017, pp. 1722–1729.

[11] V. Vasilopoulos *et al.*, "Reactive Navigation in Partially Familiar Planar Environments Using Semantic Perceptual Feedback," *Under review, arXiv: 2002.08946*, 2020.

[12] S. Gupta, J. Davidson, S. Levine, R. Sukthankar, and J. Malik, "Cognitive Mapping and Planning for Visual Navigation," in *IEEE CVPR*, 2017, pp. 7272–7281.

[13] N. Savinov, A. Dosovitskiy, and V. Koltun, "Semi-parametric topological memory for navigation," *arXiv: 1803.00653*, 2018.

[14] W. B. Shen, D. Xu, Y. Zhu, L. J. Guibas, L. Fei-Fei, and S. Savarese, "Situational Fusion of Visual Representation for Visual Navigation," in *IEEE Int. Conf. Computer Vision*, 2019, pp. 2881–2890.

[15] X. Meng, N. Ratliff, Y. Xiang, and D. Fox, "Neural Autonomous Navigation with Riemannian Motion Policy," in *IEEE Int. Conf. Robotics and Automation*, 2019, pp. 8860–8866.

[16] L. Janson, T. Hu, and M. Pavone, "Safe Motion Planning in Unknown Environments: Optimality Benchmarks and Tractable Policies," in *Robotics: Science and Systems*, 2018.

[17] M. Lahijanian *et al.*, "Iterative Temporal Planning in Uncertain Environments With Partial Satisfaction Guarantees," *IEEE Trans. Robotics*, vol. 32, no. 3, pp. 583–599, 2016.

[18] A. Bajcsy, S. Bansal, E. Bronstein, V. Tolani, and C. J. Tomlin, "An Efficient Reachability-Based Framework for Provably Safe Autonomous Navigation in Unknown Environments," *arXiv: 1905.00532*, 2019.

[19] S. Kousik *et al.*, "Bridging the Gap Between Safety and Real-Time Performance in Receding-Horizon Trajectory Design for Mobile Robots," *arXiv: 1809.06746*, 2018.

[20] V. Vasilopoulos and D. E. Koditschek, "Reactive Navigation in Partially Known Non-Convex Environments," in *13th Int. Workshop on the Algorithmic Foundations of Robotics (WAFR)*, 2018.

[21] N. Kolotouros, G. Pavlakos, M. J. Black, and K. Daniilidis, "Learning to reconstruct 3D human pose and shape via model-fitting in the loop," in *IEEE Int. Conf. Computer Vision*, 2019, pp. 2252–2261.

[22] D. H. Greene, "The decomposition of polygons into convex parts," *Computational Geometry*, vol. 1, pp. 235–259, 1983.

[23] M. Otte and E. Frazzoli, "RRT^X: Asymptotically optimal single-query sampling-based motion planning with quick replanning," *Int. J. of Robotics Research*, vol. 35, no. 7, pp. 797–822, 2015.

[24] V. Shapiro, "Semi-analytic geometry with R-functions," *Acta Numerica*, vol. 16, pp. 239–303, 2007.

[25] V. Vasilopoulos *et al.*, "Technical Report: Reactive Semantic Planning in Unexplored Semantic Environments Using Deep Perceptual Feedback," *Technical Report, arXiv: 2002.12349*, 2020.

[26] The CGAL Project, *CGAL User and Reference Manual*, 4.14 ed. CGAL Editorial Board, 2019. [Online]. Available: https://doc.cgal.org

[27] J. Redmon and A. Farhadi, "Yolov3: An incremental improvement," *arXiv: 1804.02767*, 2018.

[28] O. Arslan and D. E. Koditschek, "Smooth Extensions of Feedback Motion Planners via Reference Governors," in *IEEE Int. Conf. Robotics and Automation*, 2017, pp. 4414–4421.

[29] J. C. Latombe and D. Hsu, "Randomized single-query motion planning in expansive spaces," Ph.D. dissertation, Stanford University, 2000.

[30] TurtleBot2, "Open-source robot development kit for apps on wheels," 2019. [Online]. Available: https://www.turtlebot.com/turtlebot2/

[31] B. Schling, *The Boost C++ Libraries*. XML Press, 2011.

[32] V. Vasilopoulos *et al.*, "Sensor-Based Reactive Execution of Symbolic Rearrangement Plans by a Legged Mobile Manipulator," in *IEEE/RSJ Int. Conf. Intelligent Robots and Systems*, 2018, pp. 3298–3305.

[33] T. T. Topping, V. Vasilopoulos, A. De, and D. E. Koditschek, "Composition of templates for transitional pedipulation behaviors," in *Int. Symposium on Robotics Research*, 2019.

[34] H. Kress-Gazit, G. E. Fainekos, and G. J. Pappas, "Temporal-logic-based reactive mission and motion planning," *IEEE Trans. Robotics*, vol. 25, no. 6, pp. 1370–1381, 2009.

[35] J.-M. Lien and N. M. Amato, "Approximate convex decomposition of polyhedra," in *ACM Symp. Solid and Phys. Model.*, 2007, pp. 121–131.