2022 IEEE International Conference on Robotics and Automation (ICRA)
May 23-27, 2022. Philadelphia, PA, USA

Reactive Informative Planning for Mobile Manipulation Tasks under Sensing and Environmental Uncertainty

Mariliza Tzes[1], Vasileios Vasilopoulos[2], Yiannis Kantaros[3] and George J. Pappas[1]

***Abstract*— In this paper we address mobile manipulation planning problems in the presence of sensing and environmental uncertainty. In particular, we consider mobile sensing manipulators operating in environments with unknown geometry and uncertain movable objects, while being responsible for accomplishing tasks requiring grasping and releasing objects in a logical fashion. Existing algorithms either do not scale well or neglect sensing and/or environmental uncertainty. To face these challenges, we propose a hybrid control architecture, where a symbolic controller generates high-level manipulation commands (e.g., grasp an object) based on environmental feedback, an informative planner designs paths to actively decrease the uncertainty of objects of interest, and a continuous reactive controller tracks the sparse waypoints comprising the informative paths while avoiding a priori unknown obstacles. The overall architecture can handle environmental and sensing uncertainty online, as the robot explores its workspace. Using numerical simulations, we show that the proposed architecture can handle tasks of increased complexity while responding to unanticipated adverse configurations.**

I. INTRODUCTION

Task and motion planning (TAMP) has emerged as the 'backbone' of robotic manipulation, widely seen in industrial and service applications. Specifically, rearrangement planning has recently received increasing attention [1]–[6], but limitations still exist due to the provable NP-hardness of the overall problem [7]. One of the key challenges in the rearrangement planning scenarios is the handling of uncertainties over the task domain. Consider the scenario where the *mobile* robotic manipulator is assigned to grasp an object of unknown location while avoiding unexpected conditions of the environment, e.g. unfamiliar obstacles.

In this work, we propose an architecture for addressing *mobile* manipulation task planning problems in the presence of environmental and sensing uncertainty. In particular, we consider mobile manipulators equipped with noisy sensors being responsible for accomplishing high level manipulation tasks, captured by Linear Temporal Logic (LTL) formulas, in environments with unknown geometry and movable objects that are located at uncertain positions. In Fig. 1 we illustrate an example of the task domain. To address such problems, we propose a novel hybrid architecture, seen in Fig. 2, which can handle unanticipated conditions in the robot's workspace. Particularly, the proposed method consists of a symbolic controller generating high-level manipulation commands (e.g., grasp an object), an informative planner generating sequences of (sparse) waypoints to actively decrease the uncertainty of the objects associated with the symbolic commands, and a reactive controller to follow the informative waypoints while avoiding a priori unknown obstacles. The transition from the symbolic to the continuous reactive controller is online and can respond to unanticipated conditions such as movable objects prohibiting task accomplishment by pushing them out of the way (see [8]). Despite the problem complexity, our architecture is provable complete under specific conditions, and scales well with the task complexity.

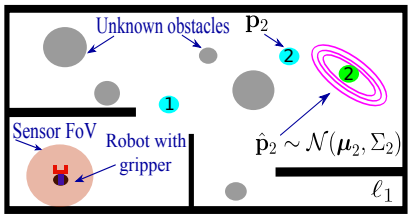


Fig. 1: An illustration of the uncertainty over the task domain. The robot is equipped with a gripper and a limited range onboard sensor (orange) for localizing the uncertain movable objects (cyan) and avoid unknown, unanticipated obstacles (grey) while navigating in a partially known environment (black).

A. Mobile Manipulation under Uncertainty

Most of the existing works focus on *known* environments [9], [10], making them applicable only to very specific problem instances. Recent works on uncertain environments [11]–[13] propose deep learning approaches that design visually guided rearrangement planning algorithms. In [12], the authors introduce a visual state prediction engine, that predicts a workspace state offline. Common in these works is that the robot's camera has access to the entire workspace, decodes environmental uncertainties offline, and executes the manipulation task based on that decoded environment.

Another suggested way to approach rearrangement planning under uncertainty is to model the problem as a *partially observable Markov decision process* (POMDP), where the robot holds a belief about the state of the workspace and intentionally selects actions that reduce its uncertainty about the world [14], [15]. Garrett et. al. [14] utilize a Particle Filter to update the belief distribution over the object's position and online replanning is required after each action of the robotic manipulator, resulting in limited applicability when increasing the number of movable objects. An interesting approach is proposed in [16], where a motion planner is defined in object-centric coordinates, enabling the derivation of a controller that can react to (small) perturbations.

[1]GRASP Lab, University of Pennsylvania, Philadelphia, PA 19104, USA, {mtzes, pappasg}@seas.upenn.edu.
[2]Computer Science and Artificial Intelligence Laboratory (CSAIL), MIT, Cambridge, MA 02139, vvasilo@mit.edu.
[3]Department of Electrical and Systems Engineering, Washington University in St. Louis, St. Louis, MO, 63112, ioannisk@wustl.edu.
This work was supported by AFOSR grant FA9550-19-1-0265.

2022 IEEE International Conference on Robotics and Automation (ICRA) | 978-1-7281-9681-7/22/$31.00 ©2022 IEEE | DOI: 10.1109/ICRA46639.2022.9811642

Authorized licensed use limited to: University of Pennsylvania. Downloaded on October 29,2026 at 14:55:04 UTC from IEEE Xplore. Restrictions apply.

Our scheme differs from the existing literature in that it formally defines the uncertainty over the task domain and incorporates the *active sensing* part, where the algorithm provides directions on to where the *mobile* robotic manipulator should take measurements *on-the-fly* to reduce its uncertainty, while navigating through unknown obstacles to satisfy a complex manipulation task encoded in an LTL formula.

B. Reactive Temporal Logic Planning

Examples of reactive temporal logic planning algorithms in partially unknown environments have been developed in [17]–[26]. Reactive to LTL specifications planning algorithms are proposed in [23], [24], as well. Specifically, in [23], [24] the robot reacts to the environment while the task specification captures this reactivity. Correctness of these algorithms is guaranteed if the robot operates in an environment that satisfies the assumptions that were explicitly modeled in the task specification. Unlike our approach, common in all these works is that they assume perfect sensors while often relying on discrete abstractions of the robot dynamics [27], [28]. Additionally, the above works neglect active interaction with the environment to satisfy the logic specification.

C. Contribution

This paper proposes the first planning and control architecture for mobile manipulation tasks in the presence of environmental and sensing uncertainty. To the best of our knowledge, the most relevant work to the one presented here is the recent work by the authors [8], which, unlike this work, considers perfect sensors and known movable objects. The symbolic and reactive controllers maintain respective correctness and collision avoidance guarantees as in [8], and in this work we present the conditions for the informative planner's correctness. Additionally, we provide a variety of simulation examples that illustrate the efficacy of the proposed algorithm for accomplishing complex manipulation tasks in unknown environments.

II. PROBLEM DESCRIPTION

We consider a first-order, nonholonomically-constrained, disk-shaped robot of radius $r \in \mathbb{R}_{>0}$ that resides in a closed, compact, polygonal and typically non-convex workspace $\Omega \subset \mathbb{R}^2$. The robot's rigid placement is denoted by $\overline{\mathbf{x}}(t) := (\mathbf{x}(t), \psi(t)) \in \mathbb{R}^2 \times S^1$ where $\mathbf{x}(t) \in \mathbb{R}^2$ and $\psi(t) \in S^1$ are the robot's position and orientation respectively at time t and its input vector $\overline{\mathbf{u}}(t) := (v(t), \omega(t))$ consists of a fore-aft and an angular velocity command. The robot's rigid placement $\bar{\mathbf{x}}(t)$ is assumed to be perfectly known at each time t.

The workspace Ω contains a finite collection of (i) disk-shaped *movable objects* denoted by $\tilde{\mathcal{M}} := \{\tilde{\mathcal{M}}_i\}_{i\in\{1,\dots,N_M\}}$ with a vector of radii $(\rho_1, \dots, \rho_{N_M}) \in \mathbb{R}^{N_M}$ and (ii) *disjoint obstacles* of unknown number and placement, denoted by $\tilde{\mathcal{O}}$. A subset $\tilde{\mathcal{P}} \subseteq \tilde{\mathcal{O}}$ of these obstacles are assumed to have a recognizable polygonal geometry, that the robot can instantly identify and localize as in [29], but a completely random shape. The remaining obstacles $\tilde{\mathcal{C}} := \tilde{\mathcal{O}} \backslash \tilde{\mathcal{P}}$ are assumed to be strongly convex, following [8], but completely

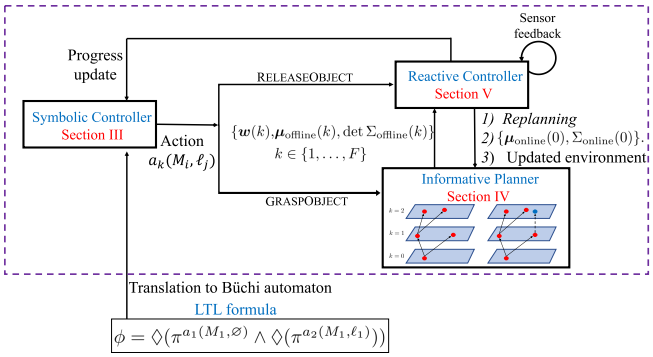


Fig. 2: **System architecture**: The task is encoded in an LTL formula and translated offline to a Büchi Automaton (Symbolic Controller - Section III). During execution time, the Symbolic Controller generates the symbolic actions and activates either the Informative Planner (Section IV) or the Reactive Controller for action implementation (Section V-A). If the action is GRASPOBJECT, the Informative Planner gets activated to generate a sequence of waypoints $\boldsymbol{w}(k)$ that actively decrease the uncertainty of the object associated with the command. The waypoints are coupled with the statistics $(\boldsymbol{\mu}_{\text{offline}}(k), \Sigma_{\text{offline}}(k))$ of the à-posteriori Gaussian distributions, which are transmitted to the Reactive Controller for execution and get updated (using a Kalman Filter) once the robot reaches each particular waypoint. If the action is RELEASEOBJECT, the Reactive Controller gets immediately activated and translates the symbolic action into a navigation command towards a target $\mathbf{x}^*$. For each action, the Reactive Controller either allows the robot to probably converge to designated targets while avoiding obstacles in the environment, or switches to *Replanning* mode and triggers the Informative Planner to evaluate new waypoints when specific conditions are met (Section V-B).

unknown. As in [8], [29]–[31], 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. Unlike our previous work [8], the positions of objects in $\tilde{\mathcal{M}}$ are also assumed to be uncertain. Instead, the robot holds a Gaussian distribution over the objects' locations, as described in Section II-A. An illustrative example of the robot's workspace can be seen in Fig. 1. The robot is equipped with a gripper to move the objects $\tilde{\mathcal{M}}$ that can either be engaged ($g = 1$) or disengaged ($g = 0$) and a sensor (e.g., camera) which allows it to take measurements associated with the unknown object $\tilde{\mathcal{M}}_i$. Hereafter, we assume that the robot can generate measurements as per the following *observation model*:

$$\boldsymbol{y}(t) = M(\mathbf{x}(t))\boldsymbol{p} + \boldsymbol{v}(t) \tag{1}$$

where $\boldsymbol{y}(t) = [\boldsymbol{y}_1(t)^T, \dots, \boldsymbol{y}_{N_M}^T]^T$, $\boldsymbol{y}_i(t)$ is the measurement signal at time t received from position $\mathbf{x}(t)$, associated with object $\tilde{\mathcal{M}}_i$, $\boldsymbol{p} = [\boldsymbol{p}_1^T, \dots, \boldsymbol{p}_{N_M}^T]^T, \boldsymbol{p}_i \in \mathbb{R}^2$ are the objects' true positions and $\boldsymbol{v}(t) \sim \mathcal{N}(\mathbf{0}, \boldsymbol{R}(t))$ is a sensor-state dependent measurement noise, whose covariance matrix is $\mathbf{R}(t)$. Notice that the observation model in (1) is linear with respect to the objects' locations $\mathbf{p}$ but not necessarily to the robot's position $\mathbf{x}(t)$. The latter is a reasonable model for some sensors (e.g. cameras).

Assumption 1 *The covariance matrix $\mathbf{R}(t)$ is assumed to be known for all time instants t, a common assumption for the application of Kalman Filters for state estimation.*

A. Kalman Filter for Position Estimation

As mentioned above, the robot does not know ahead of time the exact locations of the movable obstacles $\mathbf{p}$. Thus,

Authorized licensed use limited to: University of Pennsylvania. Downloaded on October 29,2026 at 14:55:04 UTC from IEEE Xplore. Restrictions apply.

the robot must maintain a belief over their locations and intentionally select actions that reduce its uncertainty about the world. Given a Gaussian prior distribution for the objects' positions $\boldsymbol{p}$, i.e., $\hat{\boldsymbol{p}}(0) \sim \mathcal{N}(\boldsymbol{\mu}(0), \Sigma(0))$ and measurements denoted by $\boldsymbol{y}_{0:t}$ that the robot has collected until a time instant t, the robot computes the à-posteriori Gaussian distribution denoted by $\hat{\boldsymbol{p}}(t) \sim \mathcal{N}(\boldsymbol{\mu}(t|\boldsymbol{y}_{0:t}), \Sigma(t|\boldsymbol{y}_{0:t}))$, where $\boldsymbol{\mu}(t|\boldsymbol{y}_{0:t})$ and $\Sigma(t|\boldsymbol{y}_{0:t}))$ denote the à-posteriori mean and covariance matrix respectively. To compute the local Gaussian distribution, a Kalman Filter can be used.

B. Specifying Complex Manipulation Tasks

The robot needs to accomplish a mobile manipulation task, by visiting regions of interest $\ell_j \subseteq \Omega$, where $j \in \{1, \ldots, L\}$, for some $L > 0$, and applying one of the following three manipulation actions $a_k(\tilde{\mathcal{M}}_i, \ell_j) \in \mathcal{A}$, with $\tilde{\mathcal{M}}_i \in \mathcal{M}$ referring to a movable object, defined as follows:

•GRASPOBJECT($\tilde{\mathcal{M}}_i$) instructing the robot to grasp the movable object $\tilde{\mathcal{M}}_i$, labeled as $a_1(\tilde{\mathcal{M}}_i, \varnothing)$, with $\varnothing$ denoting that no region is associated with this action. For the safety of the grasping operation, the robot must first localize the object. Given that the robot holds and updates a Gaussian distribution over the object's position, we can define its uncertainty over this position at time t as the determinant of the à-posteriori covariance matrix, $\det \Sigma_i(t|\mathbf{y}_{0:t})$, where $\Sigma_i(t|\mathbf{y}_{0:t})$ is the covariance matrix of the marginal distribution of the joint à-posteriori Gaussian distribution, corresponding to object $\tilde{\mathcal{M}}_i$, as defined in Section II-A. Alternative uncertainty measures could be used, such as the trace or maximum eigenvalue of $\Sigma_i(t|\mathbf{y}_{0:t})$. Once the robot takes the appropriate measurements at designated waypoints derived from the informative planner (Section IV) and manages to reduce its uncertainty below a user-specified threshold ϵ, i.e. $\det \Sigma_i(t|\mathbf{y}_{0:t}) \leq \epsilon$, we assume that the robot can safely grasp the object.

•RELEASEOBJECT($\tilde{\mathcal{M}}_i, \ell_j$) instructing the robot to push the (assumed already grasped) object $\tilde{\mathcal{M}}_i$ toward its designated goal position, ℓ_j, labeled as $a_2(\tilde{\mathcal{M}}_i, \ell_j)$.

•DISSASSEMBLEOBJECT($\tilde{\mathcal{M}}_j, \mathbf{x}^{**}$) gets activated by the *Fix* mode, introduced in [8, Section IV], which is triggered when the completion of the RELEASEOBJECT($\tilde{\mathcal{M}}_i, \ell_j$) is blocked by object $\tilde{\mathcal{M}}_j$. The action is responsible for moving the blocking object $\tilde{\mathcal{M}}_j$ to a different location $\mathbf{x}^{**}$.

An example of a manipulation task can be seen in Fig. 1 where the robot is asked to grab obstacle $\tilde{\mathcal{M}}_1$ and release it in region ℓ_1. We will capture such manipulation tasks via Linear Temporal Logic (LTL) specifications. Specifically, we use atomic predicates of the form $\pi^{a_k(\tilde{\mathcal{M}}_i, \ell_j)}$, which are true when the robot applies the action $a_k(\tilde{\mathcal{M}}_i, \ell_j)$ and false until the robot achieves that action. Note that these atomic predicates allow us to specify temporal logic specifications defined over manipulation primitives and, unlike related works [10], [32], are entirely agnostic to the geometry of the environment. We define LTL formulas by collecting such predicates in a set $\mathcal{AP}$ of atomic propositions. For example, the scenario of Fig. 1 can be described as a sequencing task [33] with the following LTL formula: $\phi = \Diamond(\pi^{a_1(\tilde{M}_1, \varnothing)} \wedge \Diamond(\pi^{a_2(\tilde{M}_1, \ell_1)}))$ where $\Diamond$ and $\wedge$ refer to the 'eventually' and 'AND' operator. LTL formulas are satisfied over an infinite sequence of states [34]. Unlike related works where a state is defined to be the robot position, e.g., [23], here we follow the approach of [8] and define a state by the manipulation action $a_k(\tilde{\mathcal{M}}_i, \ell_j)$ that the robot applies. Also, similarly to our prior work [8], we exclude the 'next' operator from the syntax, since it is not meaningful for practical robotics applications [35], as well as the negation operator[1].

Remark 1 The robot does not know where the objects are exactly, before grasping them. However, if it manages to grasp an object and releases it, it will remember the last location of the released obstacle.

C. Problem Statement

Given a robot of initial rigid placement $\overline{\mathbf{x}}(0)$, equipped with a sensor (e.g., camera) capable of receiving noisy measurements as per the observation model (1) and a prior Gaussian distribution $\hat{\mathbf{p}}(0)$ over the objects' positions $\mathbf{p}$, design a hybrid control architecture that (i) computes in real time a set of informative waypoints $\mathcal{W}$, that will allow the robot to actively reduce its uncertainty over the positions $\mathbf{p}$, and (ii) designs an infinite sequence of symbolic/discrete manipulation actions that satisfy ϕ and a continuous-time controller to execute it while avoiding all obstacles.

D. Hybrid Controller Architecture

In this Section, we briefly provide an overview of our proposed hybrid controller architecture, seen also in Fig. 2. The manipulation task, expressed in ϕ, is fed to the symbolic controller (Section III) which translates it online into a sequence of symbolic actions. The robot nominally executes each of these actions sequentially. The informative controller (Section IV) is called to compute a sequence of waypoints that if followed, active reduction of the uncertainty over the objects' positions is guaranteed. The reactive controller (Section V) receives the waypoints, follows them by avoiding obstacles in the unknown environment and activates, under special occasions, the *Replanning* mode.

III. SYMBOLIC CONTROLLER

In this Section we present the discrete controller that generates online a sequence of manipulation commands in the form of actions as described in Section II. A detailed construction of the controller can be found in [8, Sec. III].

A. Construction of the Symbolic Controller

Initially, the LTL specification ϕ, constructed using a set of atomic predicates $\mathcal{AP}$, is translated into a Non-deterministic Büchi Automaton (NBA) using the tool in [36]. The NBA contains state-space and transitions among states. The LTL formula is satisfied if starting from the initial state, the robot generates an infinite sequence of observations (i.e., atomic predicates that become true) that results in an infinite sequence of transitions so that the final state is visited infinitely

[1]Since the negation operator is excluded, safety requirements, such as obstacle avoidance, cannot be captured by the LTL formula; nevertheless, the proposed method can still handle safety constraints by construction of the (continuous-time) reactive, vector field controller in Section V.

Authorized licensed use limited to: University of Pennsylvania. Downloaded on October 29,2026 at 14:55:04 UTC from IEEE Xplore. Restrictions apply.

often. The NBA states can be used to measure the progress the robot has made in terms of accomplishing the assigned mission, using a distance metric described in [8, App. I]. Specifically given that the robot lies at a specific NBA state $\mathbf{s}_B(t)$, the metric yields the next state $\mathbf{s}_B^{\text{next}}$ that decreases the distance to a state that accomplishes the assigned task. Once the target NBA state is selected, a symbolic action in the form of a manipulation task that achieves it is generated and navigation commands are passed either to the informative planner or to the reactive controller.

When the symbolic action is satisfied, a new target NBA state is selected and a corresponding manipulation command is generated. In case of incompleteness, (e.g. the reactive controller can not reach an object) the controller picks another symbolic action that could drive the robot from the state $\mathbf{s}_B(t)$ to the selected state $\mathbf{s}_B^{\text{next}}$. If no commands exist, the symbolic controller picks a different state $\mathbf{s}_B^{\text{next}}$ that could drive the robot towards the final NBA state. If there are no such automaton states, the symbolic controller returns a message stating that the robot cannot accomplish the task.

IV. INFORMATIVE PLANNER

In this Section, we present a sampling-based path planner [37] that generates informative waypoints so that the uncertainty of the object of interest drops below a user-specified threshold; see Section II-B. The informative planner gets activated each time a new grasping command is received from the symbolic controller or the reactive controller turns on the *Replanning* mode; see Section V-B. The robot temporarily pauses and waits for the transmission of new waypoints from the informative planner, before resuming the execution of the action using the reactive controller.

A. Construction of the Informative Path Planner

The path planner receives from the symbolic controller the action GRASPOBJECT($\tilde{\mathcal{M}}_i$) and translates it into a navigation command towards a collision-free location $\mathbf{x}^*$ on the boundary of the closed ball $\overline{\mathsf{B}(\boldsymbol{\mu}_{i,\text{offline}}(0), r+\rho_i)}$, where $\boldsymbol{\mu}_{i,\text{offline}}(0)$ is the mean of the marginal distribution of the joint prior Gaussian distribution $\hat{\mathbf{p}}(0) \sim \mathcal{N}(\boldsymbol{\mu}_{\text{offline}}(0), \Sigma_{\text{offline}}(0))$ (see Section II-A) when the informative planner is called. The task of the planner is to design an informative path that would drive the robot towards the target $\mathbf{x}^*$ and allow it to take measurements to reduce the uncertainty it has over the object's position. Given the initial robot's position $\mathbf{x}(0)$ and a prior distribution of the objects positions $\mathbf{p}(0)$, our goal is to compute a planning horizon F and a sequence of waypoints $\boldsymbol{w}(k) \in \mathbb{R}^2$, for all time instants $k = \{0, \dots, F\}$, which solves the deterministic optimal control problem in (2), where $\boldsymbol{\delta}(k) \in \mathcal{U}$ is an actuation input selected from a finite set $\mathcal{U}$ of admissible actuation inputs. In (2a), $\boldsymbol{w}_{0:F}$ stands for the sequence of waypoints from $k = 0$ until $k = F$. The objective (2a) captures the cumulative uncertainty in the estimation of $\mathbf{p}_i(k)$ after fusing information collected by the robot from $k = 0$ up to time F. The constraint (2b) requires the final uncertainty of $\mathbf{p}_i(F)$ to be below a user-specified threshold ϵ and the second constraint (2c) requires that the returned waypoints should lie in obstacle-free areas. The constraint (2d) computes the following waypoint after applying the actuation input $\boldsymbol{\delta}(k)$ and constraint (2e) sets the first waypoint equal to the initial position of the robot. In (2f), $\xi(\cdot)$ stands for the Kalman Filter Ricatti map used to compute the covariance matrices given the robot state[2].

$$\min_{F, \boldsymbol{w}_{0:F}} \left[J(F, \boldsymbol{w}_{0:F}) = \sum_{k=0}^{F} \det \Sigma_{i,\text{offline}}(k+1) \right] \tag{2a}$$

$$\det \Sigma_{i,\text{offline}}(F+1) \leq \epsilon \tag{2b}$$

$$\boldsymbol{w}(k+1) \in \mathcal{F}, \tag{2c}$$

$$\boldsymbol{w}(k+1) = \boldsymbol{w}(k) + \boldsymbol{\delta}(k), \tag{2d}$$

$$\boldsymbol{w}(0) = \mathbf{x}(0), \tag{2e}$$

$$\Sigma_{\text{offline}}(k+1) = \xi(\boldsymbol{w}(k), \Sigma_{\text{offline}}(k)) \tag{2f}$$

To solve (2) we employ a sampling-based algorithm that incrementally constructs a directed tree that explores both the information and the physical space. To what follows, we denote the tree as $\mathcal{G} = \{\mathcal{V}, \mathcal{E}, J_{\mathcal{G}}\}$, where $\mathcal{V}$ is the set of nodes and $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ denotes the set of edges. Each node of the tree contains states of the form $\mathbf{q}(k) = [\mathbf{x}(k), \Sigma_{\text{offline}}(k)]$ and the function $J_{\mathcal{G}} \colon \mathcal{V} \to \mathbb{R}_+$ assigns the cost of reaching node $\mathbf{q}(k) \in \mathcal{V}$ from the root of the tree. The root of the tree is defined as $\mathbf{q}(0) = [\mathbf{x}(0), \Sigma_{\text{offline}}(0)]$, where $\mathbf{x}(0), \Sigma_{\text{offline}}(0)$ are the robot's initial position and prior covariance respectively. The cost of the root is initialized as $J_{\mathcal{G}}(\mathbf{q}(0)) = \det \Sigma_{\text{offline}}(0)$, while the cost of node $\mathbf{q}(k+1)$ is equal to $J_{\mathcal{G}}(\mathbf{q}(k+1)) = J_{\mathcal{G}}(\mathbf{q}(k)) + \det \Sigma_{\text{offline}}(k+1)$, where the node $\mathbf{q}(k)$ is the parent node of $\mathbf{q}(k+1)$. Applying the cost function recursively results in the objective function (2a). A detailed description for the construction of the tree $\mathcal{G}$ is provided in [37, Section III]. Except for the waypoints $\boldsymbol{w} \in \mathcal{W}$, the informative planner sends their corresponding mean $\boldsymbol{\mu}_{\text{offline}}$ and covariance matrix Σ_{offline} from the à-posteriori gaussian distribution, computed during the design of the informative path, i.e. the set $\{\boldsymbol{w}(k), \boldsymbol{\mu}_{\text{offline}}(k), \det \Sigma_{\text{offline}}(k)\},\ k \in \{1, \dots, F\}$. Once a sequence of informative waypoints has been generated, we append to the end of this sequence the target $\mathbf{x}^*$ for grasping.

Theorem 1 (**Probabilistic Completeness** [37]) *If there exists a solution to Problem 2, then the Informative Planner is probabilistically complete, i.e., feasible waypoints* $\boldsymbol{w}_{0:F} = \boldsymbol{w}(0), \dots, \boldsymbol{w}(F)$ *will be found with probability* 1.

Remark 2 The optimal control problem (2) has resulted from a stochastic optimal control problem discussed in [37, Section II] that depends on sensor measurements. Then, due to the linearity and Gaussian assumptions made about the sensor model (1), a separation principle presented in [38] is employed, which allows the conversion of the stochastic optimal control problem into the deterministic one shown in (2) that does not depend on the measurements $\mathbf{y}_{0:k}$. Therefore, each time informative paths are required to decrease the

[2]The initial robot position and prior distribution refer to the time instant where the informative planner was called. We use k to denote the offline time instances and distinguish them from the online time instances t.

Authorized licensed use limited to: University of Pennsylvania. Downloaded on October 29,2026 at 14:55:04 UTC from IEEE Xplore. Restrictions apply.

uncertainty of an object, the robot can pause and compute the informative waypoints as per (2) without the need of measurements. To emphasize this, in (2), we use the subscript 'offline' in the computed covariance matrices.

Remark 3 The formulation of (2) requires the covariance matrix Σ_{offline} to be updated using the Kalman Filter update rule. Since the update rule and in general the entire problem does not depend on the measurements $\mathbf{y}_{0:k}$, the covariance matrix Σ_{offline} gets updated offline. However, the mean $\boldsymbol{\mu}_{\text{offline}}(k)$ is not updated during the computation of the informative waypoints and remains equal to the prior mean $\boldsymbol{\mu}_{\text{offline}}(0)$ for all time instances k. On the contrary, during online execution, each time the robot reaches a waypoint $\boldsymbol{w}(k)$, the reactive controller will update both the mean $\boldsymbol{\mu}_{\text{online}}(k+1|\mathbf{y}_{0:k})$ and the covariance matrix $\Sigma_{\text{online}}(k+1|\mathbf{y}_{0:k})$ based on the received measurements; see Section V.

V. REACTIVE CONTROLLER

In this Section, we briefly describe the continuous, reactive controller [29], responsible for driving the robot to track either a single target $\mathbf{x}^*$ or a set of waypoints $\mathcal{W}$. The reactive controller receives from the symbolic controller the action RELEASEOBJECT$(\tilde{\mathcal{M}}_i, \ell_j)$ and translates the action into a navigation command towards target $\mathbf{x}^*$, selected as the centroid of region ℓ_j. Upon receiving a symbolic action GRASPOBJECT$(\tilde{\mathcal{M}}_i)$, the reactive controller first awaits from the informative planner a set of waypoints $\mathcal{W}$, and then sequentially drives the robot to each target-waypoint $\boldsymbol{w} \in \mathcal{W}$. When the robot reaches the final waypoint, the reactive controller creates the gripping command $g = 1$, and requests a new action from the symbolic controller. Next, we briefly describe the online implementation of the controller, and then analyze the conditions under which the reactive controller triggers *replanning* from the informative planner.

A. Action Implementation

The reactive vector field controller from our prior work [29] allows either a fully-actuated or a differential-drive robot to probably converge to a designated fixed target while avoiding obstacles in the environment. Specifically, the known components of the environment (e.g., walls) and any sensed fragments of the unknown obstacles are stored in the mapped space from which a change of coordinates $\mathbf{h}$ deforms the mapped space to yield a geometrically simple but topologically equivalent model space. A constructed vector field in this model space is then transformed in realtime through the diffeomorphism $\mathbf{h}$ to generate the input in the physical space. When the robot grips an object, we use the method from [39], where the vector-field controller designs control policies $\mathbf{u}_{i,c}$ for the center $\mathbf{x}_{i,c}$ of the circumscribed disk of radius $(\rho_i + r)$, enclosing the robot and the object. The inputs are then transformed into the differential drive commands $\bar{\mathbf{u}} = (v, \omega)$ using the Jacobian $\mathbf{T}_{i,c}(\psi)$ of the gripping contact, i.e., $\bar{\mathbf{u}} \coloneqq \mathbf{T}_{i,c}(\psi)^{-1}\mathbf{u}_{i,c}$, $\dot{\mathbf{x}}_{i,c} = \mathbf{T}_{i,c}(\psi)\bar{\mathbf{u}}$.

B. Replanning conditions

The reactive controller is further responsible for detecting whether or not a replanning request should be forwarded to the informative planner. As mentioned in Section IV, the reactive controller receives the set $\{\boldsymbol{w}(k), \boldsymbol{\mu}(k), \det\Sigma(k)\}$, $k \in \{1, \ldots, F\}$. Before the online execution of the path, the robot sets its prior gaussian distribution equal to the prior distribution of the informative planner, i.e., $\boldsymbol{\mu}_{\text{online}}(0) = \boldsymbol{\mu}_{\text{offline}}(0), \Sigma_{\text{online}}(0) = \Sigma_{\text{offline}}(0)$. When the robot reaches waypoint $\boldsymbol{w}(k)$, driven by the reactive controller, it updates the gaussian distribution using Kalman Filter and measurements received at each one of the previous $k-1$ waypoints, and computes the à-posteriori mean $\boldsymbol{\mu}_{\text{online}}(k|\mathbf{y}_{0:k})$ and covariance matrix $\Sigma_{\text{online}}(k|\mathbf{y}_{0:k})$. Using this information, the *Replanning* mode gets activated if at waypoint $\boldsymbol{w}(k)$: (i) the robot expected smaller uncertainty for object $\tilde{\mathcal{M}}_i$, i.e., $\det\Sigma_{i,\text{online}}(k|\mathbf{y}_{0:k}) > \det\Sigma_{i,\text{offline}}(k) + \epsilon_\Sigma$, for some small $\epsilon_\Sigma > 0$, (ii) the estimated position of the object $\tilde{\mathcal{M}}_i$ deviates from the offline estimation, i.e., $\| \boldsymbol{\mu}_{i,\text{online}}(k|\mathbf{y}_{0:k}) - \boldsymbol{\mu}_{i,\text{offline}}(k) \|_2 > \epsilon_\mu$, for some small $\epsilon_\mu > 0$, and (iii) the upcoming waypoint, $\boldsymbol{w}(k+1)$, lies inside an initially unknown but now sensed obstacle. Illustrative examples of the replanning conditions can be seen in Fig. 3a-3c. If the *Replanning* mode is activated, the reactive controller transfers its latest estimation about the location of the objects to the informative planner, which uses it as its prior distribution and the updated environment.

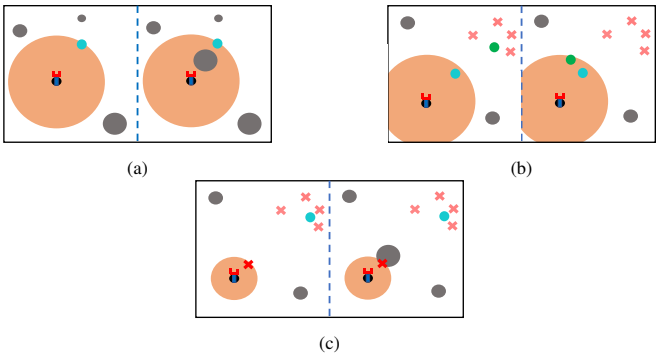


Fig. 3: Figs. 3a-3c represent examples of the three replanning conditions described in Section V-B. On the left and right subparts of the figures we include examples of the estimated and actual configutations of the world, respectively. In Fig. 3a the informative planner was not aware of the existence of the gray obstacle and thus expected to sense the object and returned a smaller uncertainty than the one computed from the reactive controller. In Fig. 3b, the estimated position of the object is depicted in green and there is a large deviation between the informative planner's and the reactive controller's expected locations, $\boldsymbol{\mu}_{i,\text{offline}}, \boldsymbol{\mu}_{i,\text{online}}$ respectively. Finally, Fig. 3c visualizes the case of an invalid waypoint.

VI. ILLUSTRATIVE SIMULATIONS

In this Section, we present numerical experiments that illustrate the performance of our proposed architecture for rearrangement planning (see Fig. 2). All case studies have been implemented using Python 3.7 (Informative Planner and Reactive Controller) and MATLAB 2020a (Symbolic Controller), on a Macbook Pro, 2.7 GHz Quad-Core Intel Core i7, 16GB RAM. In each case, we assume that the robot can take noisy measurements with a camera that provides the xy-coordinates of any *visible* movable object[3] within its sensing range, expressed in the robot's frame.

[3]For simulation purposes, we consider an object to be visible if the straight-line path from the robot to the true object location is collision-free.

Authorized licensed use limited to: University of Pennsylvania. Downloaded on October 29,2026 at 14:55:04 UTC from IEEE Xplore. Restrictions apply.

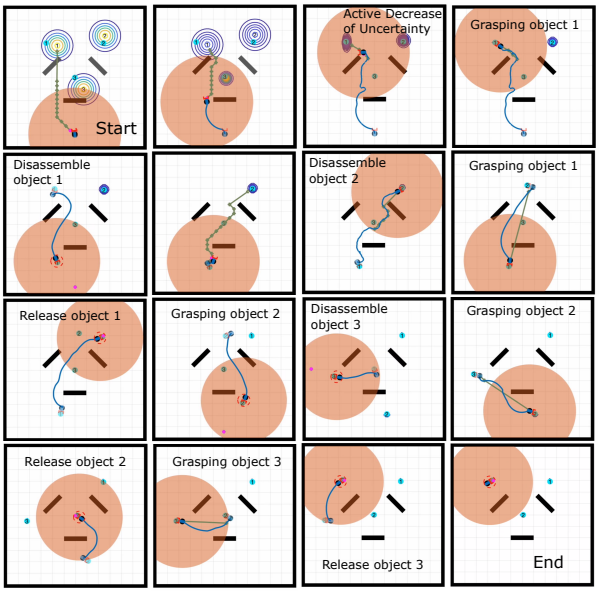


Fig. 4: An illustrative execution of the clockwise rearrangement of three movable objects (cyan), in an environment cluttered with some unanticipated obstacles (initially dark grey and then black upon sensed and localized).

The additive, measurement noise is Gaussian as defined in the observation model (1), where $\mathbf{R} = (0.05 \odot \mathbf{p})^2\mathbb{I}$. The robot perfectly localizes object $\tilde{\mathcal{M}}_i$ when the uncertainty drops below threshold ϵ, set to $1e^{-3}$. Finally, the *Replanning* thresholds $\epsilon_\Sigma, \epsilon_{\boldsymbol{\mu}}$ are set as $1e^{-3}$ and $0.3m$ respectively.

A. Rearrangement Task

We applied our proposed architecture in a complex rearrangement task, where the robot is required to rotate clockwise three movable objects. In this case, the reactive controller activates the *Fix* mode and executes the DISSASSEMBLEOBJECT$(\tilde{\mathcal{M}}_j, \mathbf{x}^{**})$ action, invisible to the symbolic controller, before resuming the execution of the initially received action. Fig. 4 illustrates successive snapshots of the rearrangement planning scenario. At each figure, we show the marginal à-posteriori gaussian distributions for the objects' positions, as computed by the reactive controller, using the online Kalman Filter. Initially, the robot follows the informative waypoints (green) to execute GRASPOBJECT(1). During its movement, it actively reduces the uncertainty it has over the objects' locations $\mathbf{p}$, while the mean $\boldsymbol{\mu}_{1,\text{online}}$ converges to the actual position $\mathbf{p}_1$. Next, the *Fix* mode gets activated, where the robot executes sequentially the actions DISSASSEMBLEOBJECT$(1, \mathbf{x}^{**})$, and DISSASSEMBLEOBJECT$(2, \mathbf{x}^{***})$. Finally, the robot resumes the action RELEASEOBJECT$(1, 2)$ and follows a similar procedure for the placement of objects 2 and 3.

B. Demonstration of Replanning Scenarios

In this Section, we present simulation examples that trigger the *Replanning* mode, in the simple scenario where the robot is tasked with repositioning a movable object to a predefined location (labeled R1 in Fig. 5). On the left subparts of Fig. 5, we present the state of the world just before the *Replanning* gets triggered and on the right the updated informative waypoints. In Fig. 5a, an initially unknown obstacle lays between the robot and the expected location of the object. Initially, the informative planner wrongly assumes that the object can be sensed, and thus the uncertainty $\det \Sigma_{\text{offline}}$ is expected to decrease. However, during online execution, the object's visibility is blocked by the obstacle, and thus its uncertainty $\det \Sigma_{\text{online}}$ is larger than expected, i.e., Assumption 1 is violated due to unknown obstacles. In Fig. 5b, a replan is triggered because the expected location of the object $\boldsymbol{\mu}_{\text{online}}$ is much different than the one computed offline, $\boldsymbol{\mu}_{\text{offline}}$, making grasping of the object unsafe. Finally, in Fig. 5c the waypoint to be tracked (magenta) is infeasible. The reactive controller senses the unanticipated obstacle, the informative planner's world gets updated and valid waypoints are recomputed.

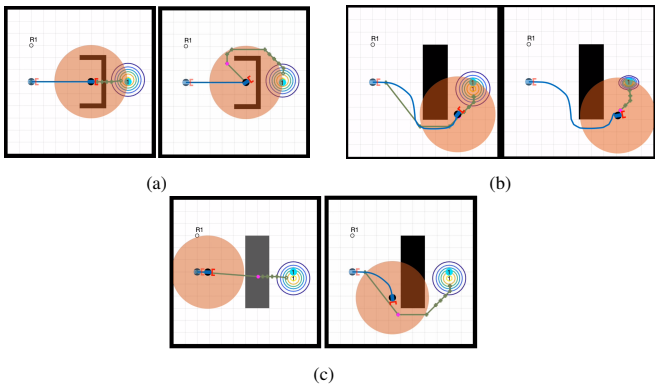


Fig. 5: Illustrative examples that lead to triggering the *Replanning* mode, described in Section V-B.

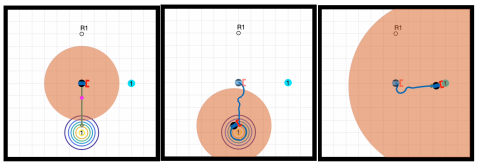


Fig. 6: Illustration of an extreme case where the prior estimated location of the object is far away from the true object location (Left). The architecture reports failure for a small sensing range (Middle) but succeeds with a larger range (Right).

C. Demonstration of Failure Mode

Finally, in Fig 6, we report an extreme case where our architecture fails. The prior expected location, $\boldsymbol{\mu}_{\text{offline}}$ (center of the Gaussian) is far away from the actual object position (cyan), as can be seen in the left subfigure. In the middle subfigure, the robot has a small sensing range and cannot take a measurement from the object in order to call for *Replanning*, thus ending up detouring from the target. The larger sensing range in the right subfigure allows the robot to replan and complete its action.

VII. CONCLUSION

In this paper, we introduce a novel hybrid architecture for rearrangement planning under sensing and environmental uncertainty, that can design informative paths for *actively* reducing the uncertainty a *mobile* robotic manipulator maintains over the task domain, while executing complex manipulation tasks. Future work will focus on extending the presented architecture to multi-robot systems performing collaborative manipulation tasks as well as on experimental validation.

Authorized licensed use limited to: University of Pennsylvania. Downloaded on October 29,2026 at 14:55:04 UTC from IEEE Xplore. Restrictions apply.

REFERENCES

[1] C. R. Garrett, T. Lozano-Perez, and L. P. Kaelbling, "Sampling-based methods for factored task and motion planning," *The International Journal of Robotics Research*, vol. 37, no. 13-14, pp. 1796–1825, 2018.

[2] J. Lee, Y. Cho, C. Nam, J. Park, and C. Kim, "Efficient obstacle rearrangement for object manipulation tasks in cluttered environments," in *2019 International Conference on Robotics and Automation (ICRA)*. IEEE, 2019, pp. 183–189.

[3] M. Toussaint, "Logic-geometric programming: An optimization-based approach to combined task and motion planning," in *Twenty-Fourth International Joint Conference on Artificial Intelligence*, 2015.

[4] S. Srivastava, E. Fang, L. Riano, R. Chitnis, S. Russell, and P. Abbeel, "Combined task and motion planning through an extensible planner-independent interface layer," in *2014 IEEE International Conference on Robotics and Automation (ICRA)*, 2014, pp. 639–646.

[5] S. Li, D. Park, Y. Sung, J. A. Shah, and N. Roy, "Reactive task and motion planning under temporal logic specifications," *arXiv preprint arXiv:2103.14464*, 2021.

[6] A. Adu-Bredu, N. Devraj, P.-H. Lin, Z. Zeng, and O. C. Jenkins, "Probabilistic inference in planning for partially observable long horizon problems," in *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2021, pp. 3154–3161.

[7] G. Wilfong, "Motion planning in the presence of movable obstacles," *Annals of Mathematics and Artificial Intelligence*, vol. 3, no. 1, pp. 131–150, 1991.

[8] V. Vasilopoulos, Y. Kantaros, G. J. Pappas, and D. E. Koditschek, "Reactive planning for mobile manipulation tasks in unexplored semantic environments," in *IEEE International Conference on Robotics and Automation*, 2021.

[9] S. Srivastava, E. Fang, L. Riano, R. Chitnis, S. Russell, and P. Abbeel, "Combined task and motion planning through an extensible planner-independent interface layer," in *IEEE International Conference on Robotics and Automation*, 2014, pp. 639–646.

[10] K. He, M. Lahijanian, L. E. Kavraki, and M. Y. Vardi, "Towards manipulation planning with temporal logic specifications," in *IEEE International Conference on Robotics and Automation*, 2015, pp. 346–352.

[11] D. Driess, J.-S. Ha, R. Tedrake, and M. Toussaint, "Learning geometric reasoning and control for long-horizon tasks from visual input," in *Proc. of the IEEE International Conference on Robotics and Automation (ICRA)*, 2021.

[12] Y. Labbé, S. Zagoruyko, I. Kalevatykh, I. Laptev, J. Carpentier, M. Aubry, and J. Sivic, "Monte-carlo tree search for efficient visually guided rearrangement planning," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 3715–3722, 2020.

[13] W. Yuan, J. A. Stork, D. Kragic, M. Y. Wang, and K. Hang, "Rearrangement with nonprehensile manipulation using deep reinforcement learning," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, 2018, pp. 270–277.

[14] C. R. Garrett, C. Paxton, T. Lozano-Pérez, L. P. Kaelbling, and D. Fox, "Online replanning in belief space for partially observable task and motion problems," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2020, pp. 5678–5684.

[15] L. P. Kaelbling and T. Lozano-Pérez, "Integrated task and motion planning in belief space," *The International Journal of Robotics Research*, vol. 32, no. 9-10, pp. 1194–1227, 2013.

[16] T. Migimatsu and J. Bohg, "Object-centric task and motion planning in dynamic environments," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 844–851, 2020.

[17] M. Guo, K. H. Johansson, and D. V. Dimarogonas, "Revising motion planning under linear temporal logic specifications in partially known workspaces," in *IEEE International Conference on Robotics and Automation*, 2013, pp. 5025–5032.

[18] M. Guo and D. V. Dimarogonas, "Multi-agent plan reconfiguration under local ltl specifications," *The International Journal of Robotics Research*, vol. 34, no. 2, pp. 218–235, 2015.

[19] M. R. Maly, M. Lahijanian, L. E. Kavraki, H. Kress-Gazit, and M. Y. Vardi, "Iterative temporal motion planning for hybrid systems in partially unknown environments," in *The 16th International Conference on Hybrid Systems: Computation and Control*, 2013, pp. 353–362.

[20] M. Lahijanian, M. R. Maly, D. Fried, L. E. Kavraki, H. Kress-Gazit, and M. Y. Vardi, "Iterative temporal planning in uncertain environments with partial satisfaction guarantees," *IEEE Transactions on Robotics*, vol. 32, no. 3, pp. 583–599, 2016.

[21] S. C. Livingston, R. M. Murray, and J. W. Burdick, "Backtracking temporal logic synthesis for uncertain environments," in *IEEE International Conference on Robotics and Automation*, 2012, pp. 5163–5170.

[22] S. C. Livingston, P. Prabhakar, A. B. Jose, and R. M. Murray, "Patching task-level robot controllers based on a local μ-calculus formula," in *IEEE International Conference on Robotics and Automation*, 2013, pp. 4588–4595.

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

[24] J. Alonso-Mora, J. A. DeCastro, V. Raman, D. Rus, and H. Kress-Gazit, "Reactive mission and motion planning with deadlock resolution avoiding dynamic obstacles," *Autonomous Robots*, vol. 42, no. 4, pp. 801–824, 2018.

[25] Y. Kantaros and G. J. Pappas, "Optimal temporal logic planning for multi-robot systems in uncertain semantic maps," in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Macau, China, November 2019, pp. 4127–4132.

[26] Y. Kantaros, M. Malencia, V. Kumar, and G. J. Pappas, "Reactive temporal logic planning for multiple robots in unknown environments," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2020, pp. 11 479–11 485.

[27] C. Belta, V. Isler, and G. J. Pappas, "Discrete abstractions for robot motion planning and control in polygonal environments," *IEEE Transactions on Robotics*, vol. 21, no. 5, pp. 864–874, 2005.

[28] G. Pola, A. Girard, and P. Tabuada, "Approximately bisimilar symbolic models for nonlinear control systems," *Automatica*, vol. 44, no. 10, pp. 2508–2516, 2008.

[29] V. Vasilopoulos, G. Pavlakos, S. L. Bowman, J. D. Caporale, K. Daniilidis, G. J. Pappas, and D. E. Koditschek, "Reactive Semantic Planning in Unexplored Semantic Environments Using Deep Perceptual Feedback," *IEEE Robotics and Automation Letters*, vol. 5, no. 3, pp. 4455–4462, 2020.

[30] O. Arslan and D. E. Koditschek, "Sensor-Based Reactive Navigation in Unknown Convex Sphere Worlds," *The International Journal of Robotics Research*, vol. 38, no. 1-2, pp. 196–223, July 2018.

[31] V. Vasilopoulos, G. Pavlakos, K. Schmeckpeper, K. Daniilidis, and D. E. Koditschek, "Reactive Navigation in Partially Familiar Planar Environments Using Semantic Perceptual Feedback," *Under review, arXiv: 2002.08946*, 2020.

[32] Y. Shoukry, P. Nuzzo, A. L. Sangiovanni-Vincentelli, S. A. Seshia, G. J. Pappas, and P. Tabuada, "SMC: Satisfiability Modulo Convex Programming," *Proceedings of the IEEE*, vol. 106, no. 9, pp. 1655–1679, 2018.

[33] G. E. Fainekos, H. Kress-Gazit, and G. J. Pappas, "Hybrid controllers for path planning: A temporal logic approach," in *44th IEEE Conference on Decision and Control, European Control Conference, (CDC-ECC)*, Seville, Spain, 2005, pp. 4885–4890.

[34] C. Baier and J.-P. Katoen, *Principles of Model Checking*. MIT Press, Cambridge, MA, 2008.

[35] M. Kloetzer and C. Belta, "A Fully Automated Framework for Control of Linear Systems from Temporal Logic Specifications," *IEEE Transactions on Automatic Control*, vol. 53, no. 1, pp. 287–297, 2008.

[36] P. Gastin and D. Oddoux, "Fast ltl to büchi automata translation," in *International Conference on Computer Aided Verification*. Springer, 2001, pp. 53–65.

[37] M. Tzes, Y. Kantaros, and G. J. Pappas, "Distributed sampling-based planning for non-myopic active information gathering," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Prague, Czech Republic, October 2021, pp. 5872–5877.

[38] N. Atanasov, J. Le Ny, K. Daniilidis, and G. J. Pappas, "Information acquisition with sensing robots: Algorithms and error bounds," in *2014 IEEE International Conference on Robotics and Automation (ICRA)*, 2014, pp. 6447–6454.

[39] V. Vasilopoulos, W. Vega-Brown, O. Arslan, N. Roy, and D. E. Koditschek, "Sensor-Based Reactive Symbolic Planning in Partially Known Environments," in *IEEE International Conference on Robotics and Automation*, 2018, pp. 5683–5690.

Authorized licensed use limited to: University of Pennsylvania. Downloaded on October 29,2026 at 14:55:04 UTC from IEEE Xplore. Restrictions apply.