

Vision-Based Reactive Planning and Control of Quadruped Robots in Unstructured Dynamic Environments

Tangyu Qian, Zhangli Zhou, Shaocheng Wang, Zhijun Li, Chun-Yi Su, and Zhen Kan

Abstract—Quadruped robots have received increasing attention for the past few years. However, existing works primarily focus on static environments or assume the robot has full observations of the environment. This limits their practical applications since real-world environments are often dynamic and partially observable. To tackle these issues, vision-based reactive planning and control (V-RPC) is developed in this work. The V-RPC comprises two modules: offline pre-planning and online reactive planning. The pre-planning phase generates a reference trajectory over continuous workspace via sampling-based methods using prior environmental knowledge, given an LTL specification. The online reactive module dynamically adjusts the reference trajectory and control based on the robot's real-time visual perception to adapt to environmental changes.

I. INTRODUCTION

Recent years have witnessed great advances in the locomotion capability of quadruped robots like the MIT Cheetah [1] and ETH Anyman [2]. However, it's still rare to find one in daily scenarios, since the dynamic and unstructured environment poses significant challenges to existing methods [3]. Hence, this work is motivated to develop a reactive planning and control strategy for quadruped robots with temporal logic specifications and multi-modal perception to enable mission operation in unstructured dynamic environments.

Due to the capability of expressing complex robotic tasks beyond traditional point-to-point navigation [4], temporal logic-based motion planning has received significant attention [5]–[8]. However, most of previous studies either focused on static environments [9] or assumed full observations of the environment by the robot [10]–[12]. Recently, these methods were extended to dynamic environments [13]–[18]. Li et al. proposed a method that aims at finding the optimal solution within a limited time domain [19]. The proposed algorithm is a sampling-based approach that searches for local optimal paths in dynamic environments to accomplish timed logic tasks [20]. Despite recent progress, the aforementioned works are built upon a key assumption that the goal is implicitly coupled with a specified location. This may significantly limit its applicability, as the goal can be time-varying or even infeasible, especially in a dynamic environment. A more

This work was supported in part by the National Natural Science Foundation of China under Grant U2013601, 62173314 and CAAI-Huawei MindSpore Open Fund.

T. Qian, Z. Zhou, S. Wang, Z. Li, and Z. Kan (corresponding author) are with the Department of Automation, University of Science and Technology of China, Hefei, China.

C. Su is with the School of Automation and Guangdong Province Key Laboratory of Intelligent Decision and Cooperative Control, Guangdong University of Technology, Guangzhou, China

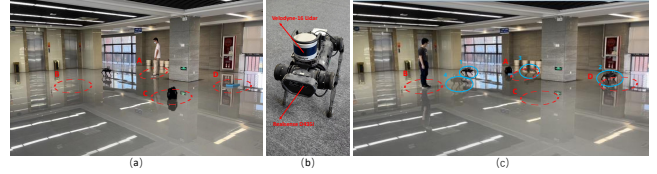


Fig. 1. (a) The experiment environment. (b) The Unitree A1 quadruped robot. (c) The visualized robot trajectory.

nature idea is to formulate the robot's motion planning task based on the target objects. When considering the motion control of quadruped robots, various algorithms have been developed. For instance, Cheetah can reach a top speed of 3.7m/s using model predictive control and whole body control [21]. A controlled backflip is achieved by Panther by representing the rotational dynamics using the rotation matrix [22]. Still, few prior works focus on solving tasks with complex temporal and logic constraints in unstructured and dynamic environments.

To this end, we consider a robot integrated with vision performing high-level linear temporal logic (LTL) tasks that encode position constraints to the target object, as shown in Fig. 1. To cope with unstructured dynamic environments, we developed a vision-based reactive planning and control (V-RPC) framework consisting of two layers: offline pre-planning and vision-based online reactive motion planning. The main contributions can be summarized as follows. We propose a novel task description that decouples the target object from the environment to enable efficient motion planning in unstructured dynamic environments. A real-time local scene perception algorithm is then developed, which can effectively detect and localize dynamic elements of environments and update robot's knowledge of the environment. Finally, targets of the motion control algorithm are determined by the scene perception module, which helps to adapt the robot to complex environments. Extensive numerical simulations and physical environments using the Unitree A1 quadruped robot are carried out to demonstrate the effectiveness of V-RPC. Fig. 1 shows the robot hardware and the experiment setup.

II. BACKGROUND

A. Preliminaries

Considering a robot operating in an unstructured dynamic environment Env , the interaction between the robot and Env can be captured by a weighted transition system (WTS).

Definition 1. A WTS $\mathcal{T} = (X, x_0, \rightarrow_{\mathcal{T}}, AP, L_X, C_{\mathcal{T}})$ in Env is a tuple, where X is the geometric space of Env ; x_0

is the initial state of robot; $\rightarrow_{\mathcal{T}} \subseteq X \times X$ is the geometric transition relation s.t. $(x, x') \in \rightarrow_{\mathcal{T}}$ if $\text{dist}(x, x') \leq \eta$, where η is a predefined maximum step size, and the transition from x to x' is collision-free; AP is the set of atomic propositions indicating the labels of regions; $L_X : X \rightarrow AP$ is the time-varying labeling function that returns an atomic proposition satisfied at the current location x ; and $C_{\mathcal{T}} : (\rightarrow_{\mathcal{T}}) \rightarrow \mathbb{R}^+$ is the geometric Euclidean distance, i.e., $C_{\mathcal{T}}(x, x') = \text{dist}(x, x')$, $\forall (x, x') \in \rightarrow_{\mathcal{T}}$.

Due to the rich expressivity, LTL is used throughout this work to describe the high-level robot missions.

Definition 2. [23] (Linear temporal logic) LTL is a formal language over a set of atomic propositions AP and combinations of Boolean and temporal operators. The syntax of LTL is defined as:

$$\phi := \text{true} \mid ap \mid \phi_1 \wedge \phi_2 \mid \neg \phi_1 \mid \phi_1 \bigcirc \phi_2 \mid \phi_1 \mathcal{U} \phi_2,$$

where $ap \in AP$ is an atomic proposition, *true*, *negation* \neg , and *conjunction* \wedge are propositional logic operators, and *next* \bigcirc and *until* \mathcal{U} are temporal operators. Based on that, other propositional logic operators such as *false*, *disjunction* \vee , *implication* \rightarrow , and temporal operators such as *always* \square and *eventually* \diamond , can be defined.

Any LTL formula can be converted to a Non-deterministic Büchi Automata (NBA).

Definition 3. An NBA over 2^{AP} is a tuple $\mathcal{B} = (Q, Q_0, \Sigma, \rightarrow_{\mathcal{B}}, Q_F)$, where Q is the set of states, $Q_0 \subseteq Q$ is the set of initial states, $\Sigma = 2^{AP}$ is the finite alphabet, $\rightarrow_{\mathcal{B}} \subseteq Q \times \Sigma \times Q$ is the transition relation, and $Q_F \subseteq Q$ is the set of accepting states.

B. Gait Representation

The gait of a quadruped robot refers to the relative temporal relationship between stance and swing phase of 4 dependent legs. The gait can be divided into different categories based on the duty factor and relative phase. The duty factor ρ is the quotient of stance duration T_{stance} by the full gait cycle duration T_{gait} , where $T_{\text{gait}} = T_{\text{stance}} + T_{\text{swing}}$ with T_{swing} representing the swing duration. The relative phase φ describes the ratio of the time difference between legs in the whole gait cycle. If a leg is chosen as reference, its relative phase is set to 0. To define the gaits, we use the array $[\varphi_{LF}, \varphi_{LH}, \varphi_{RH}, \varphi_{RF}]$ to represent the relative phases. Further details about the gait control are referred to Sec IV-B.3.b.

III. PROBLEM FORMULATION

Consider a quadruped robot operating in an unstructured dynamic environment Env and equipped with an RGB-D camera. Initially, the robot only has a preliminary map \mathcal{M} of the environment. The problem to be solved is formally stated as follows.

Problem 1. Given a constrained task ϕ_c and a quadruped robot with the initial position x_0 , the goal is to design a vision-based reactive motion planning strategy such that ϕ_c is ensured to be completed in the dynamic environment Env .

Due to the consideration of unstructured dynamic environments, there are two main challenges in solving Problem 1: the *inefficiency challenges* and the *incompleteness challenges*. The inefficiency challenges represent the emergence of a new planning more efficient in accomplishing the task. And the incompleteness challenges indicate infeasible subtasks and thus a modified planning is needed.

Assumption 1. The constrained task ϕ_c can be accomplished regardless how the environment Env changes.

Assumption 1 is mild and reasonable. Otherwise, the task ϕ_c cannot be completed no matter how the motion planning is designed and there is no need for further discussion.

IV. METHOD

The developed V-RPC consists of a two-layer planning, as shown in Fig. 2. Sec. IV-A introduces a pre-planning method as a top layer, to generate global trajectories for LTL satisfaction. By taking offline planning as reference trajectories, as the bottom layer, reactive local planning and control based on sensor fusions is developed in Sec. IV-B.

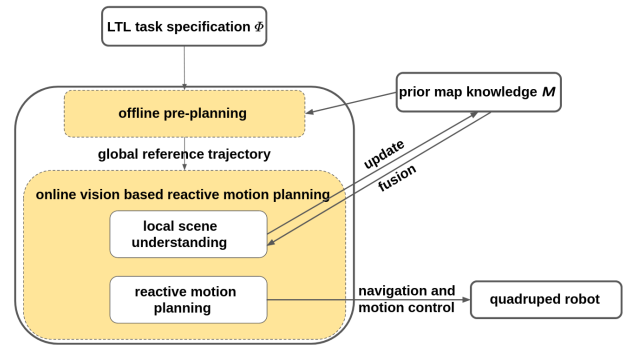


Fig. 2. Framework of V-RPC: vision-based reactive planning and control.

A. Offline Motion Planning

In this section, the robot's a prior knowledge of the environment \mathcal{M} and LTL task specification ϕ are used to generate a pre-planning. Due to the time-varying labeling functions in the dynamic environment, the target objects are specified using atomic propositions with constraints, such as ϕ_c . Note that ϕ_c can be converted to ϕ .

Definition 4. Given the WTS \mathcal{T} and the NBA \mathcal{B} , the product Büchi automaton (PBA) is a tuple $P = \mathcal{T} \times \mathcal{B} = (Q_P, Q_P^0, \rightarrow_P, Q_P^F, C_P, L_P)$, where $Q_P = X \times Q$ is the set of infinite product states, $Q_P^0 = x_0 \times Q_0$ is the set of initial states; $\rightarrow_P \subseteq Q_P \times 2^{AP} \times Q_P$ is the transition relation defined by the rule: $\frac{x \rightarrow_{\mathcal{T}} x' \wedge q \xrightarrow{L_X(x)} q'}{q_P = (x, q) \rightarrow_P q'_P = (x', q')}$, where $q_P \rightarrow_P q'_P$ denotes the transition $(q_P, q'_P) \in \rightarrow_P$, $Q_P^F = X \times Q_F$ is the set of accepting states, $C_P : (\rightarrow_P) \rightarrow \mathbb{R}^+$ is the cost function defined as the cost in the geometric space, e.g., $C_P(q_P = (x, q), q'_P = (x', q')) = C_{\mathcal{T}}(x, x')$, $\forall (q_P, q'_P) \in \rightarrow_P$, and $L_P : Q_P \rightarrow AP$ is the labelling function s.t. $L_P(q_P) = L_X(x)$, $\forall q_P = (x, q)$.

A valid trace $\tau_P = q_P^0 q_P^1 q_P^2 \dots$ of a PBA is called accepting, if it visits Q_P^F infinitely often. The corresponding accepting word $\tau_o = o_0 o_1 o_2 \dots, \forall o_i = L_P(q_P^i)$ satisfies the corresponding LTL formula ϕ . Let τ_F denote an accepting trace and $proj|_X : Q_P \rightarrow X$ a function that projects the product state space into the workspace, i.e., $proj|_X(q_P) = x, \forall q_P = (x, q)$. Using the projection, we can extract a trajectory $\tau_T = proj|_X(\tau_F)$ that satisfies the LTL formula. More details are presented in [23]. Therefore, the goal of pre-planning is to find an accepting path τ_P of PBA, with minimum accumulative geometric cost C_P .

If the state space of WTS is continuous, it's impossible to explicitly construct a PBA. Hence, the sampling-based method TL-RRT* from [24] is employed to track PBA on-the-fly, which can generate the feasible optimal path for LTL satisfaction. The resulting trajectory is a lasso-type sequence in the form of prefix-suffix structure, i.e. $\tau_F = \tau_P^{pre} [\tau_P^{suf}]^\omega$, where the prefix part $\tau_P^{pre} = q_P^0 q_P^1 \dots q_P^K$ is only executed once, and the suffix part $\tau_P^{suf} = q_P^K q_P^{K+1} \dots q_P^{K+M}$ with $q_P^K = q_P^{K+M}$ is executed infinitely often. Following prior work [6], we can decompose the optimal path $\tau_F = \tau_P^{pre} [\tau_P^{suf}]^\omega$ based on automaton components into a sequence of goal-reaching trajectories i.e., $\tau_F = \tau_0 \tau_1 \dots \tau_K [\tau_{K+1} \dots \tau_{K+l}]^\omega$. Each τ_i can be presented as an optimal solution of reachability navigation expressed as a simple LTL formula $\phi_{i,F} = \Box \neg \mathcal{O} \wedge \phi_{g_i}$, \mathcal{O} represent obstacles. We refer readers for more details about the decomposition procedure in [6].

In the following section, we take each τ_i as a reference path and synthesize a vision-based reactive motion planning algorithm through RGB-camera to satisfy each $\phi_{i,F}$ in dynamic environments.

B. Online Reactive Motion Planning

1) *Multimodal Sensing*: We implement vision sensors to enable the robot to continuously update its prior knowledge. To extract the high-level representation of the scene from the color image r of the RGB-D camera, the darknet from [25] is used to detect the classes of task-relevant objects in the field of view of camera. All 2D bounding boxes B related to the task are then obtained. Point cloud is segmented and clustered using the density-based spatial clustering of applications with noise [26] so that the centroids and the respective convex hull dimensions can be obtained.

2) *Scene Understanding*: Human motion plays an essential role in the environment. Therefore, we came up with a way that utilized a cylinder with a radius of 0.2m and 2m high to determine the best position based on the occupancy of the point cloud. We defined the occupancy value in terms of the fraction $frac$, where $frac = \frac{n}{100 \times N} \%$, where N refers to the number of point clouds belonging to humans, and n represents the number of point clouds belonging to both humans and cylinders. Generally, the more the point cloud occupies the cylinder's volume after clustering, the more accurate the prediction is.

Furthermore, we devise a method to determine whether the drifting of the target object's predicted position results

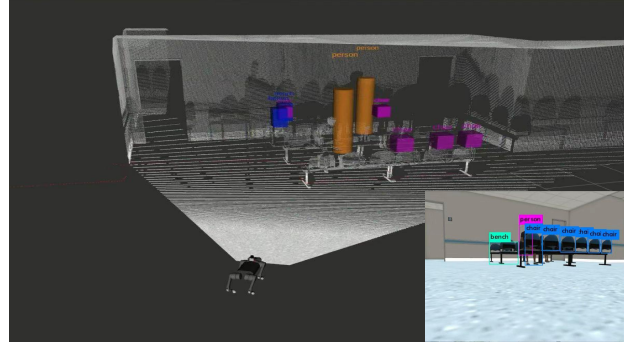


Fig. 3. Rviz visualization: The robot uses local visual information to update its own knowledge of the environment

from the target object's movement or the jitter induced by the robot movement. To address this issue, we set a threshold value to the movement speed of dynamic objects in the environment. If the drift exceeds this threshold, the object is considered dynamic and, conversely, a drift caused by robot bumps and sensor errors. The aforementioned solutions can significantly enhance the accuracy of predicting target objects and differentiate drifting caused by measurement errors or dynamic objects.

3) Reactive Motion Planning and Control :

a) *Planning*: Given an LTL task specification ϕ corresponding to the constrained task ϕ_c , the labeling function L_X^B is used to record the position constraint of the target objects. We can generate offline trajectories π that satisfies ϕ according to Section IV-A. Then, the trajectory π can be projected onto \mathcal{T} to obtain a list of target object's location $locate$ that need to be accessed, along with the corresponding list of position constraints $bind$. In the following, we show how to adjust offline reference trajectories for dynamic environments using online vision-based reactive motion planning module.

Two mechanisms i.e., the greedy mechanism and the gate mechanism are introduced to deal with the *inefficiency challenges* and *incompleteness challenges* in Section III.

The greedy mechanism deals with two main types of situations. In the first situation, when the robot intended to go to $locate[i]$ to execute the subtask x_i , it may find a new location x^* closer to the current state $x(t)$ while satisfying position constraints $bind$. Mathematically, $dist(locate[i], x(t)) > dist(x^*, x(t)) \wedge x^* \in bind[i]$. At this point, rather than continue with the previous goal $locate[i]$, x^* will be selected instead as the new goal. Another situation is that the current subtask cannot be completed and the robot's prior knowledge does not provide valid information. In this case, the robot goes to the vicinity of the next subtask to find the current target object, and if it finds a solution to complete the current subtask on the way, it can execute it.

The gate mechanism: When the intended location of the subtask $locate[i]$ is reached, the subtask may not be completed as expected due to the change of environment, and it is necessary to find a new target point $N(x_i)$ based on \mathcal{M} . This point is the closest point to the robot that can satisfy

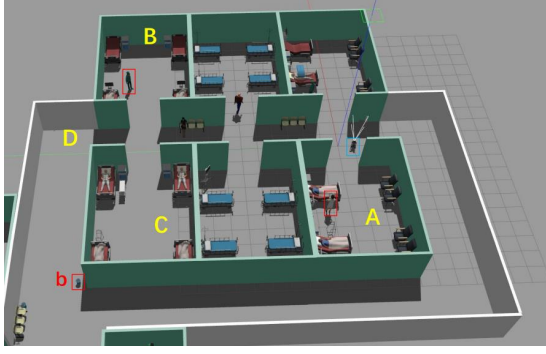


Fig. 4. A quadruped robot is asked to perform the task ϕ_c in a hospital environment. The robot needs go to room A to get the prescription and delivers it to the patient in room B. After taking the medicine, the medicine box should be dropped in the trash bin b in the platform D and the medicine should be returned to the nurse in room A.

the requirement $bind[i]$. Combining the above steps, we can obtain the optimal trajectory τ^* that satisfies the task ϕ_c on \mathcal{T} . It is easy to see that this reaction process is not on the automaton, so there is no need to reconstruct the automaton or re-search such a time-consuming and labor-intensive operation when a dynamic response occurs in the environment. Thanks mainly to the target object rather than the position in the environment used for the LTL task description, the robot is able to choose how to carry out reactive planning according to the task autonomously.

b) Control: In Sec. IV-B, we introduced how to detect and classify different objects. Afterwards, the resulting information is utilized for reactive navigation planning. In this paragraph, prior map knowledge and object detection information are fused to implement scene understanding-guided reactive motion control.

The gaits can be classified as dynamic ones or static gaits. Depending on the scene understanding, the quadruped robot switches between this two kinds of gaits. The static gait is defined as having at least three legs in contact with the ground at the same moment. Therefore the center of pressure (CoP) of torso always projects within the support polygon generated by the stance legs, allowing for better stability and adaptability when encountering unstructured terrains. Having at most two legs in contact with the ground at the same time, the dynamic gait gives rise to faster velocity and improved energy efficiency in despite of the underactuation.

In this work, we consider dynamic trot gait and static walk gait. In the trot gait, the two diagonal legs move at the same time. Due to the diagonal symmetry, relative phase φ requires to be $[0, 0.5, 0, 0.5]$. When duty factor exceeds 0.5, two full stance phases appear, leading to better stability and slower speed, and vice versa. That is, the smaller the duty cycle, the faster the motion speed. Since a full flight phase might induce an impact to the torso, we choose $\rho = 0.5$ with $T_{gait} = 0.6s$. for the trade-off between motion speed and stability.

Four legs swing in turn in a certain order when executing the walk gait. By default static gait, the duty factor should be no less than 0.75. Hence, We set duty factor as $\rho = 0.75$ for higher motion frequency. Meanwhile, the relative phase

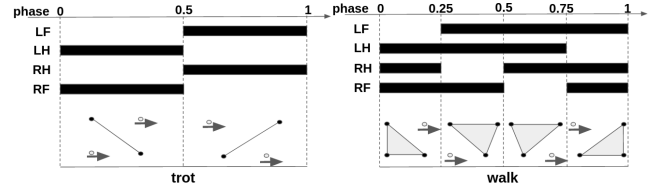


Fig. 5. The grey triangle represents the support polygon. The black blocks represent the stance action, while the white represents the swing action. The solid dot represents a stance foot. The hollow dot represents a swing foot and the arrows indicate the swing direction.

is set as $[0, 0.75, 0.25, 0.5]$ for the sake of gait alignment. Additionally, T_{gait} is set to 1.2s.

Since different stance poses in one gait can be treated as finite predefined modes of the switched system, following [27] and [28], the gait control of a quadruped robot is formulated as a nonlinear model predictive control (NMPC) problem

$$\min_{u_k} \sum_{k=0}^{N-1} l_k(x_k, u_k) \quad (1a)$$

$$s.t. \quad x_0 = x(0) \quad (1b)$$

$$\dot{x} = f(x(t), u(t), t) \quad (1c)$$

$$g_k(x_k, u_k) = 0, \quad k = 0, \dots, N \quad (1d)$$

$$h_k(x_k, u_k) \geq 0, \quad k = 0, \dots, N \quad (1e)$$

where $x = [h_{com}^T, q_b^T, q_j^T]^T \in \mathbb{R}^{24}$ denotes the system states and $u = [f_c^T, v_j^T]^T \in \mathbb{R}^{24}$ denotes the inputs, where h_{com} is the collection of normalized centrodial momentum, q_b is the generalized coordinate of base, q_j and v_j are the joint positions and velocities, and f_c consists of contact forces at four contact points. In the NMPC, (1a) is the tracking cost to follow a given trajectory in N horizons, (1b) is the initial state given by the state estimator, (1c) represents the centrodial dynamics, (1d) and (1e) are the equality and inequality constraints for specific robot tasks. The NMPC solver [?] computes the optimal trajectory of the switched system. The optimal trajectory x^* and u^* are then tracked by a whole body controller (WBC) in a hierarchical optimization fashion [29] to get the feed-forward torque τ_{ff} . Finally, the motor torque is computed by a PID controller.

V. SIMULATION AND EXPERIMENT

In this section, we experimentally validate the effectiveness of vision-based reactive planning and control (V-RPC) framework in simulation and real-world environments.

A. Simulation

1) Reactive Navigation: The constrained task is specified as $\phi_c = \square \diamond nurse^{Room_A} \wedge \square \diamond doctor^{Room_B \vee Room_C} \wedge \square \diamond can \wedge \square \diamond nurse^{Room_A}$, where *nurse* represents nurses, *doctor* represents doctors, and *can* represents trash cans. The superscript represents position constraints and empty means no constraints. The task ϕ_c requires the quadruped robot to go to room A to find the nurse to consult the patient's condition, and then go to room B or room C to find a doctor and get medicine. Before bringing the medicine to the nurse

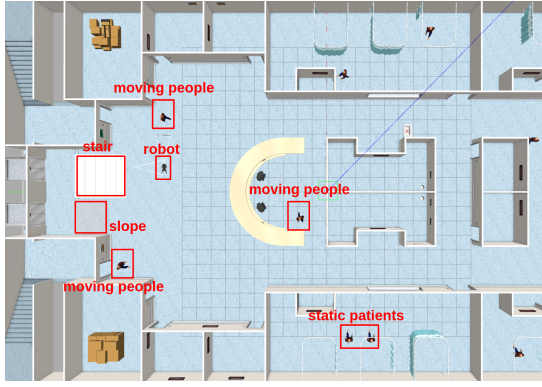


Fig. 6. The bird view of the second hospital simulation environment.

in room *A*, it needs to throw the box of medicine into a random trash can. In Fig. 4, the target objects where the robot performs its task are marked with red boxes. Given the current environment, a feasible solution for ϕ_c is to go from room *A* to room *B*, then goes to the trash can *b* in the corridor *D* to throw the trash, and finally returns to *A*. If *Env* is dynamic and unstructured, as shown in Fig. 4, the doctor may walk around between room *B* and room *C* and there might exist other trash cans *a* in the corridor *D*. Hence, the robot may not find the doctor when arriving at room *B*. This will trigger the gate mechanism. Specifically, the current subtask is considered as not completed yet, even if the robot arrives at room *B* as pre-planned. The gate mechanism declines the state transition in NBA \mathcal{B} , as shown in Fig. 8(b). Since the subtask of finding a doctor to take the medicine could not be completed as desired, the robot needs to re-plan its motion by going to room *C* to look for the doctor. When reaching room *C* and finding the doctor, the gate mechanism confirms the completion of the current subtask and the task then ϕ_c proceeds, as shown in the state transition in Fig. 8(c)). In Fig. 8(d), the robot finds another trash can *a* on its way to the trash can *b*. Since *a* is closer than *b* to its current location, the greedy mechanism kicks in allowing the robot to choose the trash can *a* instead to throw the medicine package. After the medicine package has been trashed at *a*, the gate mechanism confirms the completion of task *can* and proceeds to the last subtask. More details are referred to the experiment video¹.

2) *Reactive Control*: In this section, we introduce a scene-understanding guided reactive motion control policy. To demonstrate the reactive control module, as shown in Fig. 6, a bigger and more complicated hospital environment is created. The quadruped robot needs to switch gaits to accommodate different terrains while performing high-level navigation tasks. We consider two terrains, the flat ground and stairs, in the hospital environment, which can be detected by YOLO-V3 implemented with MindSpore. On the flat ground, the quadruped robot uses the trot gait for motion efficiency. If stairs are encountered, the gait is switched to walk to ensure torso stability when climbing up the stairs or going down the stairs. The patient status is also considered when determining

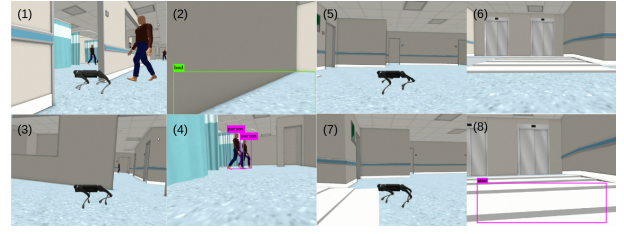


Fig. 7. Gazebo visualization of the reactive control when encountering environment changes.

the gait of the quadruped robot since in the hospital scene the robot needs to carry out tasks simultaneously without disturbing patients at rest. As shown in Fig. 7, if a static (e.g., sleeping) patient is detected in the ward, the quadruped robot will move quietly by switching to the walk gait to avoid the noise induced by the switching between the flight phase and the stance phase when trotting. If multiple moving people are detected, the quadruped robot will switch back to the trot gait for better motion.

B. Experiment

In the experiment, the constrained task is $\phi_c = \diamond(water \wedge \circ(\diamond(bag \wedge \circ(\diamond(human^{Room_B \vee Room_A})))) \vee \diamond(bag \wedge \circ(\diamond(water \wedge \circ(\diamond(human^{Room_B \vee Room_A}))))$, where *water* represents the water bottle, *bag* represents the school bag and *human* represents the person. The task ϕ_c requires the quadruped robot to first find the water bottle and school bag and then hand them to the human, who might be in either area *A* or area *B*. Similarly, ϕ_c is first translated to an LTL task specification $\phi = \diamond(water \wedge \circ(\diamond(bag \wedge \circ(\diamond(human)))) \vee \diamond(bag \wedge \circ(\diamond(water \wedge \circ(\diamond(human))))$ for off-line pre-planning. As shown in Fig. 1(c), since the target objects are out of the robot's field of view, neither the school bag nor the water bottle can be observed by the robot at the initial position 1. The robot goes to area *C* and area *D* sequentially according to the pre-planned trajectory. Unfortunately, none of the target objects can be observed and thus the gate mechanism determines that the subtask cannot be completed. Hence, the robot needs to explore the environment to find a way to complete the subtask. Since no position constraints are associated with the water bottle and the school bag and no relevant information available in its prior knowledge, the greedy mechanism is activated to guide the robot to complete the next subtask. Later, at the corner the robot finds the school bag and water bottle in area *A* and the human in area *B*. The robot then goes to area *A* to complete the task of packing the school bag and water bottle. After the gate mechanism confirms that two subtasks have been finished, the robot goes to area *B* to hand the water and the school bag to the human. More details are referred to the experiment video².

VI. CONCLUSIONS

This work presents a vision-based reactive motion planning and control framework in unstructured dynamic environments.

¹<https://youtu.be/CbEbulAc004>

²<https://youtu.be/EE5MKiAuezc>

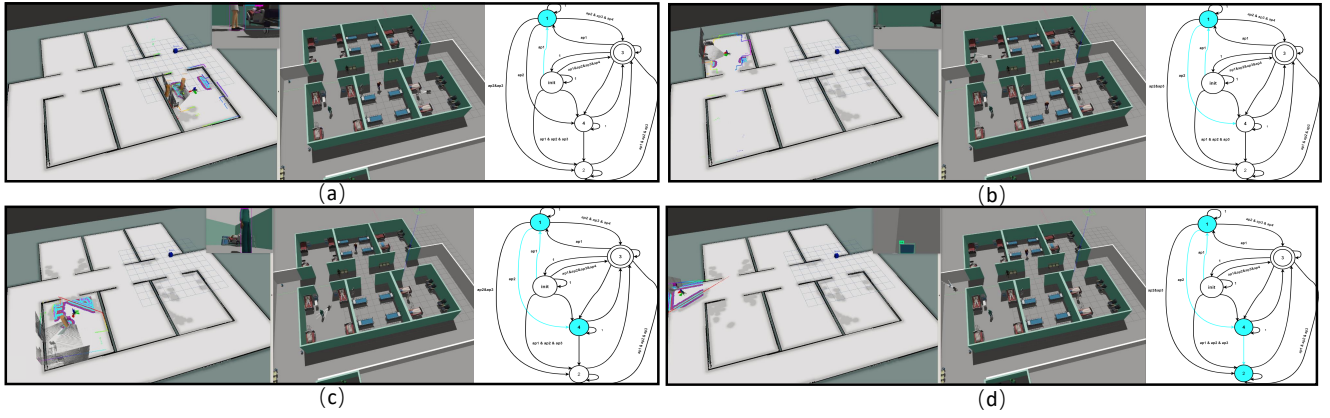


Fig. 8. Visualization of the task ϕ_c . (a) The successful completion of $nurse^{Room A}$, (b) The gate mechanism declines the state transition in NBA B . (c) The successful completion of $doctor^{Room B \vee Room C}$. (d) The successful completion of can .

The effectiveness of our approach has been demonstrated via numerical simulation and physical experiments. Future work will be extended to heterogeneous multi-robot systems for more challenging collaborative tasks.

REFERENCES

- [1] G. Bledt, M. J. Powell, B. Katz, J. Di Carlo, P. M. Wensing, and S. Kim, "Mit cheetah 3: Design and control of a robust, dynamic quadruped robot," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2018, pp. 2245–2252.
- [2] P. Fankhauser and M. Hutter, "Anymal: a unique quadruped robot conquering harsh environments," *Research Features*, no. 126, pp. 54–57, 2018.
- [3] M. Leslie, "Robots tackle darpa underground challenge," pp. 2–4, 2022.
- [4] Y. Li, R. Cui, Z. Li, and D. Xu, "Neural network approximation based near-optimal motion planning with kinodynamic constraints using rrt," *IEEE Transactions on Industrial Electronics*, vol. 65, no. 11, pp. 8718–8729, 2018.
- [5] M. Cai, M. Hasanbeig, S. Xiao, A. Abate, and Z. Kan, "Modular deep reinforcement learning for continuous motion planning with temporal logic," *IEEE Robot. Autom. Lett.*, vol. 6, no. 4, pp. 7973–7980, 2021.
- [6] M. Cai, E. Aasi, C. Belta, and C.-I. Vasile, "Overcoming exploration: Deep reinforcement learning for continuous control in cluttered environments from temporal logic specifications," *IEEE Robotics and Automation Letters*, vol. 8, no. 4, pp. 2158–2165, 2023.
- [7] G. Chen, M. Liu, and Z. Kong, "Temporal-logic-based semantic fault diagnosis with time-series data from industrial internet of things," *IEEE Trans. Ind. Electron.*, vol. 68, no. 5, pp. 4393–4403, 2020.
- [8] X. Zhou, T. Yang, Y. Zou, S. Li, and H. Fang, "Multiple subformulae cooperative control for multiagent systems under conflicting signal temporal logic tasks," *IEEE Transactions on Industrial Electronics*, vol. 70, no. 9, pp. 9357–9367, 2022.
- [9] Y. Chen, X. C. Ding, A. Stefanescu, and C. Belta, "Formal approach to the deployment of distributed robotic teams," *IEEE Transactions on Robotics*, vol. 28, no. 1, pp. 158–171, 2011.
- [10] M. Cai, S. Xiao, Z. Li, and Z. Kan, "Optimal probabilistic motion planning with potential infeasible ltl constraints," *IEEE Trans. Autom. Control*, vol. 68, no. 1, pp. 301–316, 2023.
- [11] Z. Li, G. Li, X. Wu, Z. Kan, H. Su, and Y. Liu, "Asymmetric cooperation control of dual-arm exoskeletons using human collaborative manipulation models," *IEEE Transactions on Cybernetics*, vol. 52, no. 11, pp. 12 126–12 139, 2022.
- [12] Z. Li, C. Deng, and K. Zhao, "Human-cooperative control of a wearable walking exoskeleton for enhancing climbing stair activities," *IEEE Transactions on Industrial Electronics*, vol. 67, no. 4, pp. 3086–3095, 2019.
- [13] C. I. Vasile, X. Li, and C. Belta, "Reactive sampling-based path planning with temporal logic specifications," *Int. J. Robot. Res.*, p. 0278364920918919, 2020.
- [14] Y. Li, E. M. Shahrivar, and J. Liu, "Safe linear temporal logic motion planning in dynamic environments," in *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2021, pp. 9818–9825.
- [15] M. Cai, H. Peng, Z. Li, and Z. Kan, "Learning-based probabilistic ltl motion planning with environment and motion uncertainties," *IEEE Transactions on Automatic Control*, vol. 66, no. 5, pp. 2386–2392, 2020.
- [16] Z. Li, X. Li, Q. Li, H. Su, Z. Kan, and W. He, "Human-in-the-loop control of soft exosuits using impedance learning on different terrains," *IEEE Transactions on Robotics*, vol. 38, no. 5, pp. 2979–2993, 2022.
- [17] Z. Li, K. Zhao, L. Zhang, X. Wu, T. Zhang, Q. Li, X. Li, and C.-Y. Su, "Human-in-the-loop control of a wearable lower limb exoskeleton for stable dynamic walking," *IEEE/ASME transactions on mechatronics*, vol. 26, no. 5, pp. 2700–2711, 2020.
- [18] Z. Li, Q. Li, P. Huang, H. Xia, and G. Li, "Human-in-the-loop adaptive control of a soft exo-suit with actuator dynamics and ankle impedance adaptation," *IEEE Transactions on Cybernetics*, 2023.
- [19] Z. Li, M. Cai, S. Xiao, and Z. Kan, "Online motion planning with soft metric interval temporal logic in unknown dynamic environment," *IEEE Control Syst. Lett.*, vol. 6, pp. 2293–2298, 2022.
- [20] M. Otte and E. Frazzoli, "Rrtx: Asymptotically optimal single-query sampling-based motion planning with quick replanning," *Int. J. Robot. Res.*, vol. 35, no. 7, pp. 797–822, 2016.
- [21] D. Kim, J. Di Carlo, B. Katz, G. Bledt, and S. Kim, "Highly dynamic quadruped locomotion via whole-body impulse control and model predictive control," *arXiv preprint arXiv:1909.06586*, 2019.
- [22] Y. Ding, A. Pandala, C. Li, Y.-H. Shin, and H.-W. Park, "Representation-free model predictive control for dynamic motions in quadrupeds," *IEEE Transactions on Robotics*, vol. 37, no. 4, pp. 1154–1171, 2021.
- [23] C. Baier and J.-P. Katoen, *Principles of model checking*. MIT press, 2008.
- [24] X. Luo, Y. Kantaros, and M. M. Zavlanos, "An abstraction-free method for multirobot temporal logic optimal control synthesis," *IEEE Trans. Rob.*, vol. 37, no. 5, pp. 1487–1507, 2021.
- [25] A. Bochkovskiy, C.-Y. Wang, and H.-Y. M. Liao, "Yolov4: Optimal speed and accuracy of object detection," 2020.
- [26] M. Ester, H.-P. Kriegel, J. Sander, X. Xu *et al.*, "A density-based algorithm for discovering clusters in large spatial databases with noise," in *kdd*, vol. 96, no. 34, 1996, pp. 226–231.
- [27] Q. Liao, Z. Li, A. Thirugnanam, J. Zeng, and K. Sreenath, "Walking in narrow spaces: Safety-critical locomotion control for quadrupedal robots with duality-based optimization," *arXiv preprint arXiv:2212.14199*, 2022.
- [28] J.-P. Sleiman, F. Farshidian, M. V. Minniti, and M. Hutter, "A unified mpc framework for whole-body dynamic locomotion and manipulation," *IEEE Robotics and Automation Letters*, vol. 6, no. 3, pp. 4688–4695, 2021.
- [29] C. D. Bellicoso, C. Gehring, J. Hwangbo, P. Fankhauser, and M. Hutter, "Perception-less terrain adaptation through whole body control and hierarchical optimization," in *2016 IEEE-RAS 16th International Conference on Humanoid Robots (Humanoids)*. IEEE, 2016, pp. 558–564.