2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
October 14-18, 2024. Abu Dhabi, UAE

Exploiting Hybrid Policy in Reinforcement Learning for Interpretable Temporal Logic Manipulation

Hao Zhang, Hao Wang, Xiucai Huang, Wenrui Chen, and Zhen Kan

2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) | 979-8-3503-7770-5/24/$31.00 ©2024 IEEE | DOI: 10.1109/IROS58592.2024.10802202

Abstract— **Reinforcement Learning (RL) based methods have been increasingly explored for robot learning. However, RL based methods often suffer from low sampling efficiency in the exploration phase, especially for long-horizon manipulation tasks, and generally neglect the semantic information from the task level, resulted in a delayed convergence or even tasks failure. To tackle these challenges, we propose a Temporal-Logic-guided Hybrid policy framework (HyTL) which leverages three-level decision layers to improve the agent's performance. Specifically, the task specifications are encoded via linear temporal logic (LTL) to improve performance and offer interpretability. And a waypoints planning module is designed with the feedback from the LTL-encoded task level as a high-level policy to improve the exploration efficiency. The middle-level policy selects which behavior primitives to execute, and the low-level policy specifies the corresponding parameters to interact with the environment. We evaluate HyTL on four challenging manipulation tasks, which demonstrate its effectiveness and interpretability. Our project is available at: https://sites.google.com/view/hytl-0257/.**

I. Introduction

A primary objective in robotic learning is to enable the robot to automatically plan key points to accomplish a challenging task like humans acting with instructions. To achieve such human-level intelligence, it is essential to understand the semantics of instructions and predict the important states from the task feedback is crucial. Among numerous learning algorithms, reinforcement learning (RL) [1] exhibited strong potential in various applications [2]–[4]. While RL-based methods have empowered the agent to complete tasks from simple to complex ones, a significant yet challenging topic is how effectively the robots can plan key states as sub-goals from the task level to ease the exploration burden and even improve the agent's performance for long-horizon tasks. In detail, there are three key challenges: 1) In contrast to traditional manipulation methods that learn from demonstration, how can the robot plan the key states on its own to reduce the burden on exploration? 2) When manipulating a long-horizon task, how to design an effective decision-making process to ease the exploration burden and incorporate the task semantics to facilitate the learning efficiency? 3) When considering long-horizon tasks, how to interpret the robot motion planning in task level?

Hierarchical reinforcement learning (HRL) that combines high-level policy facilitating the accomplishment of long-horizon tasks and low-level control policies has shown superior performance over conventional RL in a number of domains such as game scoring [5] and [6] as well as motion planning [7] and [8]. When performing HRL in the field of robot manipulation, a hierarchical policy was developed in [9] that chooses the action primitives and executes the corresponding parameters to accomplish challenging manipulation tasks. A trajectory-centric smoothness term is incorporated in [10] to enhance the generalization in manipulation by using dynamic time warping to align different trajectories to measure the distance. Empirical studies have found that the performance advantage of HRL is mainly attributed to the use of sub-goal for augmenting exploration. However, many existing hierarchical architectures are designed to be implemented directly at environment-level manipulation, where the feedback of interaction determines the quality of decision, lacking the connection with task semantics to guide the robotic manipulations.

Owing to the rich expressivity and capability, linear temporal logic (LTL) can describe a wide range of complex tasks combined by logically organized sub-tasks [11]–[13]. By transforming the LTL formula into an automaton, learning-based algorithms can be leveraged to solve manipulation tasks in robotic systems. For example, a non-deterministic Büchi automaton is exploited in a high resolution grasp network (HRG-Net) [14] to facilitate reactive human-robot collaboration in a locally observable transition system. Truncated LTL is transformed to a finite-state predicate automaton (FSPA) to facilitate the reward design and improve the performance of manipulation in [15]. Similar to the automaton, the representation module is also proposed by representing the LTL specification to guide the agent for complex tasks [16]. To augment the representation ability, the work of [17] uses Transformer [18] to express the semantics of LTL tasks for the robot's manipulation. However, it cannot provide the agent with specific guidelines about how to implement the LTL instructions from the task level to the concrete environment. The work of [19] develops a hierarchical setting to guide the robot to move in complex environments by waypoints. However, its architecture makes it difficult to address long-horizon manipulations and lacks the ability to provide interpretable guidance.

To bridge the gap, we consider planning the key points to ease the exploration burden of the policy from the task level, and incorporate the task semantics to provide meaningful

H. Zhang, Hao Wang, and Z. Kan (Corresponding Author) are with the Department of Automation at the University of Science and Technology of China, Hefei, Anhui, China, 230026. X. Huang is with the School of Automation, Chongqing University, Chongqing, China. W. Chen is with School of Robotics, Hunan University, Changsha, China.

This work was supported in part by National Key R&D Program of China under Grant 2022YFB4701400/4701403 and National Natural Science Foundation of China under Grant U201360.

interpretability for manipulations. The key contributions of this work are outlined below:

1) We develop a Temporal-Logic-guided Hybrid policy framework (HyTL), which not only leverages a hybrid decision-making process to facilitate the learning, but also incorporates task semantics to improve performance.

2) We design a novel waypoints planning module to ease the exploration burden, which exploits the task feedback to guide the agent interacting from the task level, empirically improving the agent's sampling efficiency.

3) By leveraging gradients and disentangling features of the task representation module, the interpretability of motion planning guided by LTL specifications is further improved.

4) We evaluate HyTL's performance on four challenging manipulation tasks with five baselines, which exhibit higher learning efficiency, better performance and interpretability compared to other methods especially to [17].

II. Preliminaries

A. Co-Safe LTL and LTL Progression

Co-safe LTL (sc-LTL) is a subclass of LTL satisfied by finite-horizon state trajectories [20]. An sc-LTL formula is built on a set of atomic propositions Π that can be true or false, standard Boolean operators such as $\wedge$ (conjunction), $\vee$ (disjunction), $\neg$ (negation), as well as temporal operators like $\Diamond$ (eventually). The semantics of an sc-LTL formula are interpreted over a word $\boldsymbol{\sigma} = \sigma_0\sigma_1...\sigma_n$, which is a finite sequence with $\sigma_i \in 2^{\Pi}$, where $i = 0, \ldots, n$.

LTL formulas can also be progressed along a sequence of truth assignments [21]–[23]. Specifically, give an LTL formula φ and a word $\boldsymbol{\sigma} = \sigma_0\sigma_1...$, the LTL progression $\mathrm{prog}\left(\sigma_i, \varphi\right)$ at step i, $\forall i = 0, 1, ...$, is defined as $\mathrm{prog}\left(\sigma_i, p\right) = \text{True}$ if $p \in \sigma_i$, where $p \in \Pi$ and $\mathrm{prog}\left(\sigma_i, p\right) = \text{False}$ otherwise.

B. Reinforcement Learning and Labeled PAMDP

A parameterized action Markov decision process (PAMDP) [5] provides a primitive-augmented RL framework to address long-horizon tasks. The whole dynamics between the agent and environment over the sc-LTL tasks φ can be modeled as a labeled PAMDP $\mathcal{M}_e = (S, T, \mathcal{H}, p_e, \Pi, L, R, \gamma, \mu)$, where S is the state space, T indicates the horizon, $\mathcal{H} = \{\mathfrak{h} : (k, x_k) \mid x_k \in \mathcal{X}_k \text{ for all } k \in \mathcal{K}\}$ is the discrete-continuous hybrid action space where $\mathcal{K} = \{1, ..., K\}$ is the set of discrete behavior primitives and $\mathcal{X}_k$ is the corresponding continuous parameter set for each $k \in \mathcal{K}$, $p_e(s'|s, \mathfrak{h})$ is the transition probability from $s \in S$ to $s' \in S$ under action $\mathfrak{h} = (k, x_k)$, Π is a set of atomic propositions, $L : S \to 2^{\Pi}$ is the labeling function, $R : S \to \mathbb{R}$ is the reward function, $\gamma \in (0, 1]$ is the discount factor, and μ is the initial state distribution. A hybrid policy π_e is exploited to interact with environment under the task φ, which outputs a hybrid action pair $\mathfrak{h}$ and receives the corresponding reward by $r_t = R(s_t)$.

III. Problem Formulation

In order to further elaborate the motivation of the proposed interpretable temporal-logic-guided hybrid decision-making framework, we will use the following example throughout the work to illustrate the main idea of our method.

Example 1. Consider a long-horizon manipulation skill of Peg Insertion from [9] as illustrated in Fig. 1, in which the robot needs to grasp the peg O_{peg} and inserts it into the hole G_{hole} guided by the planned waypoints. The set of propositions Π is $\{\mathsf{peg\_grasped}, \mathsf{hole\_reached}, \mathsf{peg\_inserted}\}$. Using above propositions, an sc-LTL task is $\varphi_{\mathsf{peg}} = \Diamond(\mathsf{peg\_grasped} \wedge \Diamond(\mathsf{hole\_reached} \wedge \Diamond \mathsf{peg\_inserted}))$, which requires the robot to sequentially grasp the peg, reach the hole and insert it into the hole.

In this work, we are interested in designing a temporal-logic guided hybrid policy architecture, which not only plans key waypoints based on the task semantics, but also guides the agent through the waypoints to choose appropriate behavioral primitives and the corresponding parameters to facilitate robot learning. By predicting the key states via the planning module, we hope to take advantage of its foresight to generate hypothetical goals to guide the agent, and ease the exploration burden of the robot's motion planning. Compared to hierarchical architectures [9] that are committed to manipulation, the waypoints generated by the planning module are gradually updated according to LTL instructions to guide the agent towards sub-goals as quickly as possible, resulting in a three-way improvement, in which the LTL instructions and environmental rewards boost the planning module for better waypoints, the waypoints guide the robot's manipulation, and the output actions improve Transformer for better tasks semantics.

By exploiting the LTL progression, an augmented PAMDP with an LTL instruction φ, namely the task-driven labeled PAMDP (TL–PAMDP), is developed as $\mathcal{M}_{\mathcal{H}}^{\varphi} \triangleq \left(\tilde{S}, \tilde{T}, \mathcal{H}, \tilde{p}, \Pi, L, \tilde{R}_{\Psi}, \gamma, \mu\right)$ like [17] (Detailed definitions can be found on our website), where the reward function is

$$\tilde{R}_{\varphi}(s, \varphi) = \begin{cases} r_{env} + r_{\varphi}, & \text{if } \mathrm{prog}(L(s), \varphi) = \text{True} \\ r_{env} - r_{\varphi}, & \text{if } \mathrm{prog}(L(s), \varphi) = \text{False}. \\ r_{env}, & otherwise \end{cases} \quad (1)$$

The problem of this work can be stated as follows.

Problem 1. Given a TL–PAMDP $\mathcal{M}_{\mathcal{H}}^{\varphi} \triangleq \left(\tilde{S}, \tilde{T}, \mathcal{H}, \tilde{p}, \Pi, L, \tilde{R}_{\Psi}, \gamma, \mu\right)$ corresponding to task φ, this work is aimed at finding an optimal policy π_{φ}^{*} over the LTL instruction φ, so that the return $\mathbb{E}\left[\sum_{k=0}^{\infty} \gamma^k r_{t+k} \mid S_t = s\right]$ under the policy π_{φ}^{*} can be maximized.

IV. Algorithm Design

To address Problem 1, we present a novel approach, namely **T**emporal-**L**ogic-guided **Hy**brid policy framework (HyTL), that offers a hybrid policy to plan waypoints, choose primitives, and determine parameters based on

LTL representation encoded by Transformer. Section IV-A presents how the plan module in HyTL generates hypothetical waypoints to improve sampling efficiency. Section IV-B explains in detail how hybrid decision-making architecture facilitates the agent's training in long-horizon manipulations. Section IV-C shows how AttCAT interprets the LTL instruction for the robot's manipulation. The overall method of HyTL is illustrated in Fig. 2, which first extracts the LTL representation φ_θ by the task representation module, then samples a series of waypoints $\boldsymbol{w}$ based on the initial state s_0 by the waypoints planning module, and outputs the action pair $\mathfrak{h} = (k, x_k)$ by selecting the appropriate action primitive k following the primitives choosing module and determines the corresponding parameters x_k following the parameterization determining module.

A. Plan Waypoints from Task Feedback

One of the main challenges in solving Problem 1 is sampling efficiency in the exploration phase when utilizing RL for long-horizon manipulations. To address this problem, [17] considers augmenting the hierarchical policy with task semantics to enable agent making decisions corresponding to the sub-goal and accelerate learning. However, it is difficult for this approach to map abstract task semantics into a concrete manipulation. Therefore, we design a planning module that incorporates task feedback to generate waypoints to guide the agent by combining task rewards.

Waypoints Planning Module. As shown in Fig. 1, we exploit the Gated Recurrent Unit (GRU) as a predictor in the form of a residual connection to incrementally construct waypoints to guide agents. Specifically, given an initial state s_0 as the initial waypoints w_0 and hidden state h_0, the sequential waypoints $\boldsymbol{w} = w_0 w_1 \ldots$ can be generated by the deviation between subgoals as $w_{i+1} = w_i + \frac{\partial w_i}{\partial t}$ where $\frac{\partial \mathrm{w_i}}{\partial \mathrm{t}} = \mathrm{GRU}\left(w_i, h_i\right)$ and $w_i \in \mathbb{R}^3$. Since it's hard for GRU to update directly through rewards, we model the plan module as a stochastic policy with Gaussian distribution inspired by [19], i.e., $\pi_{w_\zeta} = \mathcal{N}(\overline{w}, \sigma)$ with weights ζ where $\overline{w}$ is the mean of waypoint w_i and σ is predicted by a linear transform from the hidden state h_i. Thus a sequential waypoints $\boldsymbol{w} = w_0 w_1 \ldots$ can be sampled from $\pi_{w_\zeta}(s_0)$ and if the agent reaches w_i, w_{i+1} will be exploited to guide. To bridge the environment and the feedback of the task (i.e., satisfy the LTL task φ), we design the following loss function

$$J_{\pi_w}(\zeta) = \underset{\boldsymbol{w} \sim \pi_{w_\zeta}}{E} \left[-\log \pi_{w_\zeta} \cdot \tilde{R}_\varphi \left(\boldsymbol{s}, \varphi \right) \right]$$

to update the planning module, where $\tilde{R}_\varphi \left(\boldsymbol{s}, \varphi \right)$ is the cumulative reward from the state sequence $\boldsymbol{s} = s_0 s_1 \ldots$ corresponding to $\boldsymbol{w}$. Thus when the planning layer is updated, it not only uses the environment-level reward r_{env}, but also exploits the task-level reward r_φ to continuously optimize the waypoints.

As shown in Fig. 2(c), by using the waypoint planning module as a high-level policy layer, not only can the task-level semantics be taken over to guide the agent at the environment, but also the residual structure can be exploited to provide the agent with more flexible guidance by setting up more waypoints in hard-to-explore areas.

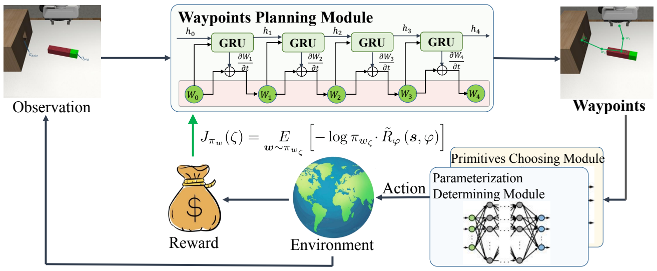
Waypoints Planning Module
h_0
h_1
h_2
h_3
h_4
GRU
GRU
GRU
GRU
$\frac{\partial W_1}{\partial t}$
$\frac{\partial W_2}{\partial t}$
$\frac{\partial W_3}{\partial t}$
$\frac{\partial W_4}{\partial t}$
W_0
W_1
W_2
W_3
W_4
G_{hole}
O_{peg}
Observation
Waypoints
$J_{\pi_w}(\zeta) = \underset{\boldsymbol{w} \sim \pi_{w_\zeta}}{E} \left[-\log \pi_{w_\zeta} \cdot \tilde{R}_\varphi \left(\boldsymbol{s}, \varphi \right) \right]$
Primitives Choosing Module
Parameterization Determining Module
Action
Reward
Environment

Fig. 1: The waypoints planning module updated via interactions.

B. Hybrid Policy based on LTL Instructions

Another challenge in solving Problem 1 is to design an effective decision-making architecture to improve the agent's performance in long-horizon manipulations. To address this challenge, the work of [9] pre-built five behavioral primitives (**reach**, **grasp**, **push**, **release**, and **atomic**), and designed a hierarchical architecture to reduce the agent's exploration burden by first selecting appropriate primitives through a high-level policy and then determining the parameters of primitives via low-level policies. However, [9] focuses on the design of the decision structure, neglecting how task-level information can help the agent accomplish the task. To address this problem, based on LTL representation, this work proposes a hybrid decision-making framework that not only incorporates task semantics to improve the learning efficiency, but also plans waypoints from the task level to guide the agent to accomplish long-horizon tasks. Except for Waypoints Planning Module mentioned in Section IV-A, the HyTL framework shown in Fig .2 comprises following key modules.

Task Representation Module. To incorporate the task semantics, we use Transfomer [18] to encode LTL representation. Given an input $X_\varphi = (x_0, x_1, \ldots)$ generated by the LTL task φ where $x_{t, t=0,1,\ldots}$, represents the operator or proposition, X_φ is preprocessed using the word embedding E as $X_{\mathrm{E}} = [x_0\mathrm{E}; x_1\mathrm{E}; \ldots; x_N\mathrm{E}] \in \mathbb{R}^{B \times M \times D}$ where B is the batch size, $M = N + 1$ is the length of input X_φ and D is the model dimension of Transformer. X_{E} is then combined with the frequency-based positional embedding $E_{pos} \in \mathbb{R}^{B \times M \times D}$ to utilize the sequence order. Then the LTL representations encoded by Transformer can be computed by following steps:

$$\begin{aligned} X_0 &= X_{\mathrm{E}} + E_{pos}, \\ X_l^{'} &= \mathrm{MSA}(\mathrm{LN}(X_{l-1})) + X_{l-1}, \quad && l = 1, \ldots, L \\ X_l &= \mathrm{MLP}(\mathrm{LN}(X_l^{'})) + X_l^{'}, && l = 1, \ldots, L \\ Y &= \mathrm{LN}(X_l) \end{aligned} \tag{2}$$

where MSA denotes the multi-head self-attention, LN means the layer norm, MLP represents the position-wise fully connected feed-forward sub-layers, and Y is the output of the final layer from the Transformer encoder. The outline of Transformer Encoder for HyTL is shown in Fig. 2(a). Thus incorporating the LTL representation into HyTL, which not only helps to improve the agent's performance, but also lays the foundation for the interpretability of motion planning shown in Section IV-C.

Primitives Choosing Module. Based on the predicted waypoint w_i as guidance, the primitive policy outputs a behavioral primitive appropriate for the current state and the progressed LTL instruction. Specifically, the behavior primitive k can be chosen by primitive policy $\pi_{k_\psi}(k|s,\varphi_\theta,w)$ with weights ψ conditioning on the waypoint w. As a middle-level policy layer, the primitives choosing module not only considers the information from the task level and environment to offer the suitable primitive, but also guides the low-level parameter module on what to do for long-horizon manipulations.

Parameterization Determining Module. Conditioning on the current state, task representations, predicted waypoints, and selected primitives, the parameterized policy outputs an appropriate set of parameters to ensure effective interactions between the behavioral primitives and the environment. Specifically, the primitive parameter x can be determined by parameter policy $\pi_{p_\xi}(x|s,\varphi_\theta,w,k)$ with weights ξ conditioning on the waypoint w and primitive k. As the last layer interacts with the environment, the parameterization determining module, based on the above conditions, instructs the agent on how to interact with the environment to accomplish the tasks.

In this work, we opt for SAC [24] as the RL backbone to update in the framework. Let $Q_{\varphi_\theta}(s,\mathfrak{h},w)$ and $Q_{\varphi_\theta^{'}}(s,\mathfrak{h},w)$ be the Q-value function of task φ and $\varphi^{'}$, and let $\pi_{w_\zeta}(s)$, $\pi_{k_\psi}(k|s,\varphi_\theta,w)$ and $\pi_{p_\xi}(x|s,\varphi_\theta,w,k)$ be the hybrid policy networks. The loss for critic, plan policy, primitive policy, and parameter policy in adapted SAC are then designed respectively as

$$\begin{aligned}
J_Q &= (Q_{\varphi_\theta} - (\tilde{R}_\varphi + \gamma(Q_{\varphi_\theta^{'}} - \alpha_k \log \pi_{k_\psi} - \alpha_p \log \pi_{p_\xi})))^2, \\
J_{\pi_w}(\zeta) &= \underset{\boldsymbol{w}\sim\pi_{w_\zeta}}{E}\left[-\log \pi_{w_\zeta}\cdot \tilde{R}_\varphi(\boldsymbol{s},\varphi)\right], \\
J_{\pi_k}(\psi) &= \underset{\boldsymbol{w}\sim\pi_{w_\zeta}}{E}\ \underset{k\sim\pi_{k_\psi}}{E}\left[\alpha_k \log \pi_{k_\psi} - \underset{x\sim\pi_{p_\xi}}{E}\left[Q_{\varphi_\theta}\right]\right], \\
J_{\pi_p}(\xi) &= \underset{\boldsymbol{w}\sim\pi_{w_\zeta}}{E}\ \underset{k\sim\pi_{k_\psi}}{E}\ \underset{x\sim\pi_{p_\xi}}{E}\left[\alpha_p \log \pi_{p_\xi} - Q_{\varphi_\theta}\right].
\end{aligned} \tag{3}$$

Note that φ_θ and $\varphi_\theta^{'}$ encoded by Transformer are indirectly updated by back-propagation of the above equations.

The pseudo-code is outlined in Alg. 1. In the exploration phase, Transformer first extracts the LTL representation φ_θ and concatenates the valve with the state s_0 (line 4). Then before the agent interacts with the environment, the waypoint planning module samples a series of waypoints $\boldsymbol{w}$ based on the initial state s_0 and incorporates w_i as part of the observation to guide the agent's exploration (line 6). Based on the above observation, the agent selects the appropriate action primitive following π_{k_ψ} and determines the corresponding parameters by π_{p_ξ} (line 11). During the interaction, the operator prog tracks the original instructions φ and checks whether the LTL task φ is complete (lines 7-10). In the training phase, HyTL updates all neural network weights following (3) (lines 14-16).

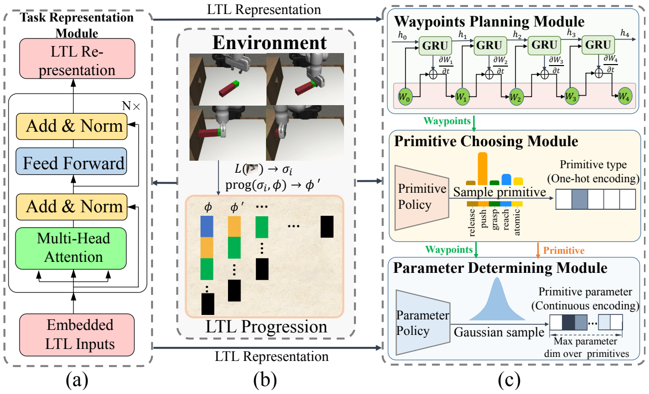

Fig. 2: The framework of HyTL. (a) The outline of Task Representation Module. (b) The LTL progression for progressing LTL formulas. (c) The hybrid decision-making process.

Algorithm 1 Temporal-Logic-guided Hybrid policy (HyTL)

```
 1: procedure INPUT:(The PAMDP $\mathcal{M}_e$ with the LTL specification $\varphi$)
        Output:  An approximately optimal policy $\pi_\varphi^*(\mathfrak{h} \mid s,\varphi)$ for the TL-PAMDP $\mathcal{M}_\mathcal{H}^\varphi$
        Initialization:  All neural network weights
 2:     for iteration 1,2,...,N do {Exploration Phase}
 3:         for episode 1,2,...,M do
 4:             Initialize timer $t \leftarrow 0$ and episode $s_0$, and augment the state $s_0$ with $\varphi_\theta$ encoded by Transformer
 5:             while episode not terminated do
 6:                 Sample waypoints $\boldsymbol{w} = w_0 w_1 \ldots$ from $\pi_{w_\zeta}(s_0)$ and guide the state $s_t$ by the waypoint $w_i$
 7:                 $\varphi^{'} \leftarrow \text{prog}(L(s),\varphi)$
 8:                 if $\varphi^{'} \in \{\mathsf{True},\mathsf{False}\}$ or $s \in T$ then
 9:                     Break
10:                 end if
11:                 Gather data from $\varphi$ following $\pi_{k_\psi}$ and $\pi_{p_\xi}$, and guide the state $s_t$ by the waypoint $w_{i+1}$ if the agent reached $w_i$
12:             end while
13:         end for
14:         for training step 1,2,...,K do {Training Phase}
15:             Update all neural network weights by (3)
16:         end for
17:     end for
18: end procedure
```

C. Interpret Manipulation via AttCAT

Another advantage of representing LTL instructions with Transformer is that it provides interpretability for robot motion planning. When LTL specifications are encoded via Transformer [25], their interpretability can be represented by the value of heads' weights on different propositions. However, [25] only sums the weights of all heads, ignoring the effect of the gradients flowing in the Transformer architecture. Inspired by [26], this work further explores the impact of LTL representation on robot motion planning by interpreting Transformer via Attentive Class Activation Tokens (AttCAT), which not only utilizes the inputs' gradients combined with attention weights to generate impact scores, but also disentangles features flowing between intermediate layers of Transformer.

Based on (2), we can write columns of X_l separately as $\mathfrak{x}_i^l$, $i = 0,1,...,M$. To analyze the effect of i-th token on the output y^c across different proposition class c in the embedded

LTL input X_0, the relationship between y^c and $\mathfrak{x}_i^L$ can be modeled by a linear relationship $y^c = \sum_i^{N+1} w_i^c \cdot \mathfrak{x}_i^L$, where $w_i^c = \frac{\partial y^c}{\partial \mathfrak{x}_i^L}$ is the linear coefficient vector of $\mathfrak{x}_i^L$ capturing the impact of i-th token on the target class c.

By considering the interaction among tokens, the impact score of the i-th token towards class c, denoted by AttCAT, can be formulated through the weighted combination:

$$\mathrm{AttCAT_i} = \sum_{l=1}^{L} E_H(\alpha_\mathrm{i}^l \cdot (w_i^c \odot \mathfrak{x}_i^l)),$$

where $\odot$ denotes the Hadamard product, α_i^l represents the attention weights of the i-th token at l-th layer, and $E_H(\cdot)$ stands for the mean over multiple heads.

When provided with a pre-trained Transformer $TF(\cdot)$, an input LTL instruction φ, and the interpretability method $\mathrm{AttCAT}(\cdot)$, the magnitude of impact can be calculated by $|\mathrm{AttCAT}(TF(\varphi))|$, reflecting each token's contribution. By the visualization input tokens scores, the most impactful token on the output can be identified.

V. CASE STUDIES

In this section, the performance of the HyTL framework is evaluated in comparison to state-of-the-art algorithms through scenarios. We specifically represent the following aspects. **1) Performance:** how effectively does HyTL surpass the state-of-the-art algorithm in long-horizon manipulations? **2) Architecture:** How good is the waypoints planning module for algorithmic enhancement? **3) Interpretable:** How well can the agent comprehend the LTL instruction?

A. Baselines and Tasks Setting

Baselines. To demonstrate the efficacy of the HyTL framework, we empirically compare its performance against five baselines. 1) Our RL backbone **SAC** from [24] is the first baseline, which only executes atomic primitive. 2) The second baseline is **Maple** from [9] which augments the traditional RL methods with action primitives and corresponding parameters to improve the exploration efficiency. 3) The third baseline is $\mathbf{Maple_{way}}$, which is based on Maple and exploits the planning module to ease the exploration burden. 4) The fourth baseline is $\mathbf{Maple_{LTL2Action}}$,which further exploits the encoder from [27] to represent semantics of LTL for improving sampling efficiency. 5) The fifth baseline is $\mathbf{TARPs_{TF-LTL}}$, which is a manipulation skill learning algorithm from [17] augmenting RL with temporal logic and hybrid action space.

Tasks Setting. To evaluate the algorithm performances, four challenging manipulations in [9] are employed, including **Stack**, **Nut Assembly**, **Cleanup** and **Peg Insertion**. The corresponding task descriptions and LTL instructions are stated in Table. II of [17] and Example. 1.

B. Experimental Results and Analysis

(1) Main Results. Fig. 3 illustrates the results performed by different approaches over 6 seeds. As shown in Fig. 3, it is observed that 1) algorithms guided by waypoints ($\mathrm{Maple_{way}}$ and HyTL) are more efficiently sampled and converge faster than those without waypoints (Maple, $\mathrm{Maple_{LTL2Action}}$, and $\mathrm{TARPs_{TF-LTL}}$); 2) HyTL exhibits better performance relative to $\mathrm{TARPs_{TF-LTL}}$ and $\mathrm{Maple_{LTL2Action}}$ which only incorporate the task semantics; 3) On the most challenging task Peg Insertion, HyTL demonstrates the shortest converge episode, with over 30% reduction compared to the best previous work $\mathrm{TARPs_{TF-LTL}}$ [17].

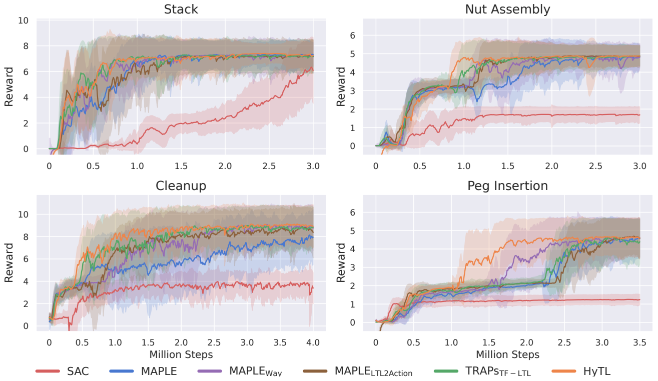

Fig. 3: Plots of normalized reward curves for four manipulations.

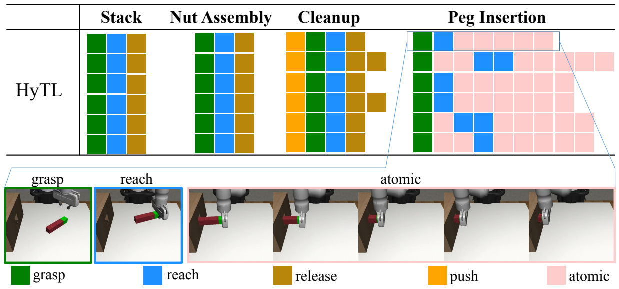

Fig. 4: The visualization of action sketches utilizing HyTL.

From the above observation, we conjecture that the unstable representation of the task module at the beginning of training affects the performance of the agents, which rely heavily on the task representation to improve the sampling efficiency. In contrast, the performance of HyTL, which has a hybrid decision architecture design, is not only affected by the task module's representation, but also relies on the waypoints generated by the planning module for guidance. Once either the planning module or the task module has been effectively updated, the gradient decreases rapidly in the direction that contributes to task completion. It is due to this complementary design in the hybrid decision architecture that HyTL can demonstrate higher learning efficiency.

2) Primitive Compositionality Quantification. Since primitives are utilized in HyTL like Maple, we use the compositionality score from [9] to evaluate the degree of action primitives over different methods. The action sketches of HyTL with 6 seeds are visualized in Fig. 4, which shows the different action primitives that HyTL selects and combines in accomplishing above four manipulation tasks. The compositionality scores are shown in Table I, in which higher scores reflect better compositionality and more stable performance. As shown in Table I, $\mathrm{Maple_{way}}$ and HyTL can select and combine more appropriate behavior

TABLE I: We present compositionality scores to reflect the compositionality and stability of the algorithms in four scenarios.

Compositionality Score	Maple	$\text{Maple}_{\text{way}}$	$\text{Maple}_{\text{LTL2Action}}$	$\text{TARPs}_{\text{TF-LTL}}$	HyTL
Stack	0.71	1.00	0.79	0.92	**1.00**
Nut Assembly	0.75	1.00	0.85	1.00	**1.00**
Cleanup	0.73	0.84	0.75	0.83	**0.89**
Peg Insertion	0.24	**0.91**	0.83	0.81	0.86

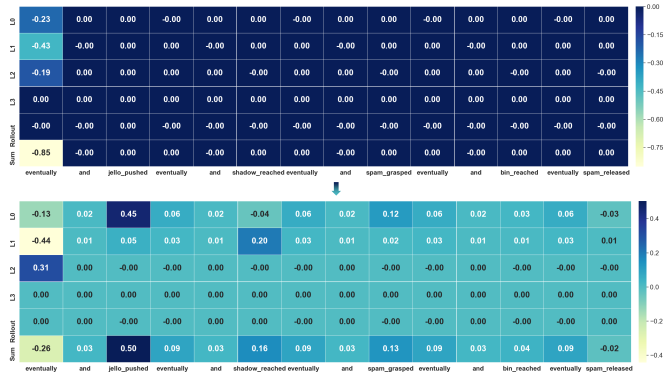

Fig. 5: We illustrate the heatmap of the task φ_{cleanup} by normalizing impact scores from different Transformer layers.

primitives by guiding within waypoints, resulting in higher compositionality scores than other methods.

(3) Interpretability via AttCAT. To further illustrate the agent's understanding of the LTL task, Fig. 5 illustrates the heatmap of the task φ_{cleanup}. As shown in the top table of Fig. 5, the cumulative scores for all tokens except for the eventually(-0.85) are almost zero, indicating that the agent lacks a clear understanding of the LTL instruction at the beginning of the training. Upon the convergence of Transforme, higher impact scores from all layers focus on the token jello_pushed(+0.50) as shown in the bottom row of Fig. 5, which suggests that the agent is more likely to move directly to the position corresponding to the jello_pushed proposition.

VI. CONCLUSIONS

In this work, we present an interpretable temporal-logic-guided hybrid decision-making framework to improve the agent's performance on four challenging manipulation tasks. In particular, a novel waypoints planning module is designed to ease the exploration burden, which exploits the task feedback to guide the agent interacting from the task level. And the hybrid decision-making process with three-level decision layers is proposed to facilitate learning of agent's manipulations. In addition, the interpretability of motion planning guided by LTL specifications is further improved by leveraging gradients and disentangling features of the task representation module. Experimental results and analysis demonstrate that HyTL improves the agent's sampling efficiency and offers reasonable interpretability. Future work will consider extending the method of HyTL to more challenging tasks, such as dexterous manipulations.

REFERENCES

[1] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. MIT press, 2018.

[2] F. Jenelten, J. He, F. Farshidian, and M. Hutter, "Dtc: Deep tracking control," *Sci. Robot.*, vol. 9, no. 86, p. eadh5401, 2024.

[3] S. Cheng and D. Xu, "League: Guided skill learning and abstraction for long-horizon manipulation," *IEEE Robot. Autom. Lett.*, 2023.

[4] W. Qiu, W. Mao, and H. Zhu, "Instructing goal-conditioned reinforcement learning agents with temporal logic objectives," *Adv. neural inf. process. syst.*, vol. 36, 2023.

[5] W. Masson, P. Ranchod, and G. Konidaris, "Reinforcement learning with parameterized actions," in *Proc. AAAI Conf. Artif. Intell.*, vol. 30, no. 1, 2016.

[6] Z. Fan, R. Su, W. Zhang, and Y. Yu, "Hybrid actor-critic reinforcement learning in parameterized action space," *arXiv preprint arXiv:1903.01344*, 2019.

[7] B. Wang, Z. Liu, Q. Li, and A. Prorok, "Mobile robot path planning in dynamic environments through globally guided reinforcement learning," *IEEE Robot. Autom. Lett.*, vol. 5, no. 4, pp. 6932–6939, 2020.

[8] D. Shah, P. Xu, Y. Lu, T. Xiao, A. Toshev, S. Levine, and B. Ichter, "Value function spaces: Skill-centric state abstractions for long-horizon reasoning," *arXiv preprint arXiv:2111.03189*, 2021.

[9] S. Nasiriany, H. Liu, and Y. Zhu, "Augmenting reinforcement learning with behavior primitives for diverse manipulation tasks," in *Proc. IEEE Int. Conf. Rob. Autom.* IEEE, 2022, pp. 7477–7484.

[10] H. Fu, S. Yu, S. Tiwari, M. Littman, and G. Konidaris, "Meta-learning parameterized skills," 2023.

[11] C. Belta, A. Bicchi, M. Egerstedt, E. Frazzoli, E. Klavins, and G. J. Pappas, "Symbolic planning and control of robot motion," *IEEE Robot. Autom. Mag.*, vol. 14, no. 1, pp. 61–70, 2007.

[12] C. Baier and J.-P. Katoen, *Principles of model checking*. MIT press, 2008.

[13] M. Cai, S. Xiao, Z. Li, and Z. Kan, "Optimal probabilistic motion planning with potential infeasible ltl constraints," *IEEE Trans. Autom. Control*, 2022, to appear.

[14] Z. Zhou, S. Wang, Z. Chen, M. Cai, H. Wang, Z. Li, and Z. Kan, "Local observation based reactive temporal logic planning of human-robot systems," *IEEE Trans. Autom. Sci. Eng.*, pp. 1–13, 2023.

[15] X. Li, Z. Serlin, G. Yang, and C. Belta, "A formal methods approach to interpretable reinforcement learning for robotic planning," *Sci. Robot.*, vol. 4, no. 37, 2019.

[16] Y.-L. Kuo, B. Katz, and A. Barbu, "Encoding formulas as deep networks: Reinforcement learning for zero-shot execution of ltl formulas," in *IEEE/RSJ Int. Conf. Intell. Robot. Syst.*, 2020, pp. 5604–5610.

[17] H. Wang, H. Zhang, L. Li, Z. Kan, and Y. Song, "Task-driven reinforcement learning with action primitives for long-horizon manipulation skills," *IEEE Trans. Cybern.*, 2023.

[18] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," *Adv. neural inf. process. syst*, vol. 30, 2017.

[19] Z. Xiong, J. Eappen, D. Lawson, A. H. Qureshi, and S. Jagannathan, "Co-learning planning and control policies using differentiable formal task constraints," *arXiv preprint arXiv:2303.01346*, 2023.

[20] O. Kupferman and M. Y. Vardi, "Model checking of safety properties," *Form. Methods Syst. Des.*, vol. 19, no. 3, pp. 291–314, 2001.

[21] H. Zhang and Z. Kan, "Temporal logic guided meta q-learning of multiple tasks," *IEEE Robot. Autom. Lett.*, vol. 7, no. 3, pp. 8194–8201, 2022.

[22] F. Bacchus and F. Kabanza, "Using temporal logics to express search control knowledge for planning," *Artif Intell*, vol. 116, no. 1-2, pp. 123–191, 2000.

[23] M. Tuli, A. Li, P. Vaezipoor, T. Klassen, S. Sanner, and S. McIlraith, "Learning to follow instructions in text-based games," *Proc. Adv. Neural Inf. Process. Syst.*, vol. 35, pp. 19 441–19 455, 2022.

[24] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine, "Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor," in *Int. Conf. Mach. Learn.* PMLR, 2018, pp. 1861–1870.

[25] H. Zhang, H. Wang, and Z. Kan, "Exploiting transformer in sparse reward reinforcement learning for interpretable temporal logic motion planning," *IEEE Robot. Autom. Lett.*, 2023.

[26] Y. Qiang, D. Pan, C. Li, X. Li, R. Jang, and D. Zhu, "Attcat: Explaining transformers via attentive class activation tokens," *Adv. neural inf. process. syst*, vol. 35, pp. 5052–5064, 2022.

[27] P. Vaezipoor, A. C. Li, R. A. T. Icarte, and S. A. Mcilraith, "Ltl2action: Generalizing ltl instructions for multi-task rl," in *Proc. Int. Conf. Machin. Learn.* PMLR, 2021, pp. 10 497–10 508.