2024 IEEE International Conference on Robotics and Automation (ICRA 2024)
May 13-17, 2024. Yokohama, Japan

Conformal Policy Learning for Sensorimotor Control under Distribution Shifts

Huang Huang[1]*, Satvik Sharma[1]*, Antonio Loquercio[1]*,
Anastasios Angelopoulos[1], Ken Goldberg[1], Jitendra Malik[1]

***Abstract*— This paper focuses on the problem of detecting and reacting to changes in the distribution of a sensorimotor controller's observables. The key idea is the design of policies that can take conformal quantiles as input, to detect distribution shifts with formal statistical guarantees, which we define as *conformal policy learning*. We show how to design such policies by using conformal quantiles to switch between base policies with different characteristics, e.g. safety or speed, or directly augmenting a policy observation with a quantile and training it with reinforcement learning. Theoretically, we show that such policies achieve the formal convergence guarantees in finite time. In addition, we thoroughly evaluate their advantages and limitations on two use cases: simulated autonomous driving and active perception with a physical quadruped. Empirical results demonstrate that our approach outperforms five baselines. It is also the simplest of the baseline strategies besides one ablation. Being easy to use, flexible, and with formal guarantees, our work demonstrates how conformal prediction can be an effective tool for sensorimotor learning under uncertainty.**

I. Introduction

As robots break out of lab-controlled conditions and start to operate for and around humans, it's critical to ensure their safety and reliability. When operating in the wild, such robots increasingly rely on machine learning systems: in modular and end-to-end systems, there is a neural network at the core of the decision-making process interpreting high-dimensional observations, making predictions, or directly predicting actions. However, when deployed in unstructured environments, such neural networks often encounter data distributions that change over time, negatively affecting their performance and, in turn, the safety of the overall system. We present an approach that allows robots to quantify such distribution shifts with formal statistical guarantees, and to use this information on downstream control tasks (Fig. 1).

Traditional approaches for estimating distribution shifts model the network activations and weights by parametric probability distributions [1], estimate uncertainties through sampling [2] or train an estimator of uncertainty by using a loss function [3], [4]. However, these methods tend to generate over-confident predictions [5] and provide neither rigorous statistical guarantees nor calibrated uncertainties. Conformal prediction [6] offers a principled approach to quantifying the uncertainty of black-box machine learning models, which has fueled a recent surge in its popularity [7], [8], [9], [10], [11], [12].

Conformal prediction works as follows. First, the user specifies a *score function*, $s(x, y)$, that measures the quality of a prediction $f(x)$ from a black-box model. For example, the score function can be the absolute distance between a prediction $f(x_i)$ and the ground truth y_i, i.e., $s(x_i, y_i) = |y_i - f(x_i)|$. Then, given a sample of exchangeable calibration data points $\{(X_1, Y_1)\}_{i=1}^{n}$, and a new data point (X, Y) with an unobserved label, a prediction set is formed for the label by inverting the score function as

$$\mathcal{C}(X) = \{y : s(X, y) \leq \hat{q}\}, \tag{1}$$

where $\hat{q}$ is chosen as the $(1 - \alpha)(1 + 1/n)$-quantile of the scores on the calibration data. This procedure results in prediction sets with valid coverage [6], [13],

$$\mathbb{P}(Y \in \mathcal{C}(X)) \geq 1 - \alpha, \tag{2}$$

where α is the error rate specified by the user. Despite the restrictive assumptions that the data points are exchangeable [6], [13], traditional conformal prediction has been applied in several robotics domains, e.g., to design sample-efficient alert systems for self-driving cars [14] and object-pose estimation [15], and language-based planning [16]. However, the data observed during the robot operation is more akin to a time series, whose distribution might change over time, due to factors such as environment changes (night-day), sensor degradation, or unforeseen operating conditions. This invalidates traditional conformal prediction since the data-generation process is not exchangeable.

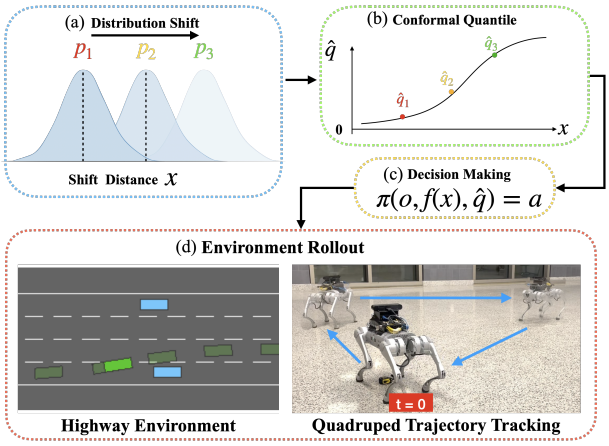

Fig. 1: Illustration of conformal policy learning. As the distribution shifts (a), the uncertainty of the prediction increases, reflected by an increasing quantile (b). (c) Our policy takes into the observation O, prediction $f(x)$ and the quantile $\hat{q}$ to decide the actions to take. (d) We evaluate this framework on two use cases: autonomous driving in simulation and vision-based trajectory tracking with a physical quadruped.

*Equal contribution, [1] University of California, Berkeley

Recently, new forms of conformal prediction have emerged to handle entirely non-exchangeable settings such as time-series prediction [17], [18], [19], [20], [21], [22]. The standard setting, in this case, is the *adversarial sequence model*, in which the data points $\{(x_t, y_t)\}_{t\in[T]}$ are arbitrary deterministic objects (e.g., real numbers) devoid of any probabilistic meaning. Therefore, methods that can provide robustness in that setting are insulated against *all possible realizations of future data* — including those drawn by an omniscient adversary strategizing against the agent with full knowledge of its current and future plans. Such adversarial settings have been popular since the early literature on calibration [23] and were introduced into the realm of conformal prediction by [17]. Note that these approaches lose the guarantee at every time-step and instead provide *long-term coverage*. That is, for some sequence of sets $\mathcal{C}_t$, letting $\text{err}_t = 1\{y_t \notin \mathcal{C}_t(x_t)\}$, they achieve

$$\frac{1}{T}\sum_{i=1}^{T} \text{err}_t = \alpha + o(1).$$

.

Such a setting, more akin to real-world conditions, has quickly captured the attention of roboticists for applications in multi-agent motion-planning problems [24], [25], [26]. In these examples, prediction sets for the other agents' positions, are incorporated as safety constraints into the planner. However, simpler conformal methods with tighter prediction sets and greater stability have been developed—namely, the conformal PID control method [22]. This approach for generating prediction sets has only been applied in traditional time-series prediction settings, where the data distribution at time t is unaffected by the predictions and decisions made at times $1, \ldots, t-1$. Furthermore, the utility of these prediction sets are somewhat limited, as they require a robotic system in which constraints are easy to incorporate. Ad exemplum, policies trained with deep learning *cannot* accept the sort of mathematical constraints that are common in optimization.

The main methodological innovation of this paper is *conformal policy learning*: the design of policies that can take conformal quantiles as input. Our results indicate we can train reinforcement learning policies that, in practice, satisfy the conditions needed [27] to achieve formal coverage guarantees in finite time; we call such policies *conformal policies* and show the first practical examples of their existence.

We explore conformal policies of two forms. The first is a naive switching mechanism, falling back to a safe policy whenever the quantile exceeds a user-defined threshold [28]. The switching policy has safety guarantees under the relatively weak assumption that the safe policy is *eventually safe*, i.e., that it will have low risk when deployed for long enough [27]. However, its reliance on a hand-designed threshold for switching between policies can make tuning challenging. In the second, we aim to remove such user-defined threshold by directly augmenting the observation of a policy with conformal quantiles, and training such policy with reinforcement learning. To our knowledge, this strategy is novel and demonstrates an interesting new form of learning under uncertainty, although its practical performance is not yet as good as that of the switching policy.

We demonstrate these ideas in two compelling use cases: a simulated autonomous driving scenario, where an autonomous car must drive as fast as possible while avoiding other vehicles; and a real-world active perception problem with a quadruped, with the goal of actively controlling the robot's speed over the trajectory to maximize the quality of vision-based state estimation. Comparison against many strong baselines in simulation shows that our approach empirically achieves the desired guarantees while significantly outperforming these baselines. In addition, we report the interesting finding that learning a policy with conformal quantiles as input systematically underperforms the simple switching strategy in out-of-distribution settings, indicating that the learning procedure fails to capture the relation between the quantile and the overall policy uncertainty. We provide an in-depth study of this phenomenon and a set of hypotheses underpinning this behavior.

II. Related Work

Robotics research has allocated a lot of attention to decision-making under uncertainty in the context of machine learning approaches [29]. Given the recent surge in popularity of conformal prediction, several works showed how to use it to add statistical guarantees in perception [15] and planning [24], [25], [26], [14]. Recent work shows that conformal quantiles can control the hallucination of large language models, making the latter better in learning-based planning [16]. However, these approaches do not provide exhaustive studies on how conformal quantiles could be used directly for control in real-world robotics scenarios.

Conversely, PAC-Bayes generalization theory has been used for deriving conditional bounds to influence decision-making on vision-based aerial navigation and manipulation in Farid et al [30]. However, this approach focused on studying generalization *in the training distribution*, while our work focuses on studying the problem of generalization under distribution shifts. In addition, they lack a study on how such bounds could be directly used by a policy trained with reinforcement learning.

In robotics, uncertainty has not only been used in the context of safety but also to speed up the reinforcement learning on a real robot [31], combine optimal control with neural networks [32], [28], or to increase the efficiency of learning in manipulation [33]. However, these works neither provide formal guarantees nor an easy and interpretable way to change the behavior of the control policy as a function of its safety. Conversely, our work can use conformal quantiles to directly affect the controller's performance with a single and interpretable user-specified parameter, α, i.e., the desired error rate. This gives the user the possibility to have more greedy behavior, hence favoring exploration, or a more conservative attitude, promoting safety. In addition, our work is empirically shown to achieve the desired error rate, as predicted by the underlying theory.

III. METHOD

We describe our methodology, first focusing on its theoretical foundation in conformal PID control and then develop the idea of conformal policy learning — i.e., the design of policies that take conformal quantiles as input.

A. Conformal control methodology

Producing the sets. Similarly to classical conformal prediction, we will proceed at time t by forming prediction sets $\mathcal{C}_t$ in the following way:

$$\mathcal{C}_t = \{y : s(x_t, y) \leq q_t\}, \tag{3}$$

where the quantile q_t is up to us to choose. The set construction in Eq. (3) is almost the same as that of standard conformal prediction except that q_t is *not* an empirical quantile of the previous data points. Instead, it will be dynamically modulated using conformal P control, i.e., quantile tracking [22]. *Why is the parameter q_t so important?* Because the miscoverage event err_t abides by the equivalence $\text{err}_t = 1 \iff s(x_t, y_t) > q_t$. Therefore, picking the right sequence of quantile updates is critical to achieving coverage.

Tracking the quantile. q_t updates as following:

$$q_{t+1} = q_t - \eta \nabla \rho_{1-\alpha}(\text{err}_t - \alpha), \tag{4}$$

where ρ is the quantile loss (sometimes referred to as the "pinball loss") at level $1-\alpha$. This update can be seen as a simple form of P-control where the feedback signal is err_t and we would like it close to $1-\alpha$.

Proof of validity. A simplified and slightly improved proof of the result in [22] is given (this analysis strategy was first developed in [17]).

Theorem 1: Let the scores $s(x_1, y_1), \ldots, s(x_T, y_T)$ be bounded between $[0, B]$. Then we have, uniformly for all positive integers W and T_0 satisfying $T_0 + W \leq T$ and all realizations $(x_1, y_1), \ldots, (x_T, y_T)$, that

$$\frac{1}{W} \sum_{t=T_0}^{T_0+W} \text{err}_t \leq \alpha + \frac{B+\eta}{\eta W}. \tag{5}$$

Proof: Akin to Proposition 5 of [17], we telescope the sum in (4) to get

$$\begin{aligned} \frac{q_{T_0+W} - q_{T_0}}{\eta W} &= \frac{1}{W} \sum_{t=T_0}^{T_0+W} \nabla \rho_{1-\alpha}(\text{err}_t - \alpha) \\ &= \frac{1}{W} \sum_{t=T_0}^{T_0+W} \text{err}_t - \alpha, \end{aligned} \tag{6}$$

where the last step used the definition of the gradient of the quantile loss. Rearranging terms, the implication is that

$$\frac{1}{W} \sum_{t=T_0}^{T_0+W} \text{err}_t \leq \alpha + \frac{q_{T_0+W} - q_{T_0}}{\eta W} \leq \alpha + \frac{B+\eta}{\eta W}, \tag{7}$$

where the last inequality holds because $q_{T_0+W} \leq B + \alpha\eta$ and $q_{T_0} \geq -\eta(1-\alpha)$. ∎

The nature of the result is *multi-resolution* in the sense that for all possible windows in time, the coverage will be met, and has a finite sample correction at level $(B+\eta)/\eta W$. In practice, we use a (slightly) adaptive version of the quantile tracker with $\eta_t = 0.1\hat{B}_t$. It is trivial to get a guarantee in this setting as well. Furthermore, a two-sided guarantee is available by a similar proof strategy.

B. Conformal Policies for Sensorimotor Control

We introduce the formal framework of a conformal policy. The policy is a sensorimotor control algorithm $\pi(x_s, f, q)$ that observes a state x_s, a conformal quantile q, and a predictor f. The latter predicts future values that could influence decision-making, e.g. the future geometry of the terrain for locomotion [34], the future position of other agents [35], or the low-level dynamics of the robot [36]. We do not make any specific assumption on the nature of the policy or the predictor. For a conformal policy to have formal safety guarantees, it must be *eventually safe*:

Assumption A1 (Eventually safe.): A policy $\pi(x_s, f, q)$ is eventually safe if there exist $\alpha^{\text{safe}} < \alpha$, q^{safe}, and $K \in \mathbb{N}$ satisfying

$$\begin{aligned} &\left\{\forall k \in [K], q_k \geq q^{\text{safe}}\right\} \\ &\qquad \implies \frac{1}{K} \sum_{k=1}^{K} \mathbf{1}\{s(x_k, y_k) > q_k\} \leq \alpha^{\text{safe}}. \end{aligned} \tag{8}$$

Here, the policy is run with quantile q_k to produce the values (x_{k+1}, y_{k+1}).

The assumption may at first look strange because the inequality does not involve π. However, one must remember that the actions of π produce the future values of x_k and y_k. The assumption therefore encodes the idea that if the policy makes enough errors, it can revert to a situation where its prediction error is lower. For example, the robot can slow down, making the task of perception and scene understanding easier. Under this assumption, the sets will cover at the correct frequency:

Corollary 1: Under Assumption A1, we have that

$$\frac{1}{T} \sum_{t=1}^{T} \text{err}_t \leq \alpha + o(1). \tag{9}$$

Proof: Assumption A1 implies the decision policy is safe over time [27] with respect to the miscoverage loss. By Theorem 1 of the same, the miscoverage loss is controlled. ∎

The validity of switching policies is also addressed as the special case of this proposition.

Of course, it is worth noting that a tacit assumption in our problem setup is the existence of online feedback, i.e., that the predictor can verify its forecasts against the ground-truth after a limited amount of time. Though not entirely benign, this assumption is valid in several real-world scenarios. For example, a legged robot will observe whether its predictions about the terrain geometry are valid after walking over it [34], or an agent could verify the accuracy of trajectory predictions for other agents several time steps after the prediction intervals are formed [35].

IV. Simulation Experiments

A. Experimental Setup

We evaluate our approach in a simulation highway environment, where an autonomous driving agent must drive as fast as possible while avoiding other vehicles [37] (Fig. 1). We instantiate the environment with 4 lanes, 50 other cars, and the ego vehicle. The action space of the ego-vehicle consists of 5 discrete actions: switching to the left lane, switching to the right lane, speeding up, slowing down, and maintaining the current speed. At each time step, the observation consists of the relative x and y positions and relative x and y velocities of the closest five vehicles to the ego-vehicle, normalized between -1 and 1. The ego-vehicle can only observe other agents in front of it. The agent can move in a specified speed range. A rollout stops when the ego-vehicle collides with other vehicles or after 40 seconds.

B. Base Policies Training

We consider two base policies: π_{safe} trained only with a non-collision reward; and π_{speed} trained with both the non-collision reward and the speed reward. The minimum speed of the ego-vehicle is $20m/s$, so no policy can immediately stop. We train both π_{safe} and π_{speed} with Deep Q-Network (DQN) [38] for 200K steps. π_{safe} has a collision reward of -1 and π_{speed} has an additional high-speed reward of 0.5. Both policies are 2-layer MLPs with neurons of [256,256].

C. Predictor Training

We train a predictor to forecast the nearest distance to the surrounding vehicles l three steps into the future, with the input of the history of the past five observations. We normalize l into the $[0, 1]$ interval by $\tilde{l} = 2/(1 + e^{(0.1l)})$. We collect a dataset of 30k rollouts with π_{speed} and π_{safe}. During the data collection, the ego-vehicle speed range is between $20m/s$ and $30m/s$. We use a 3-layer MLP followed by a Sigmoid layer as the predictor with 64, 128 and 64 neurons for each layer. The predictor is trained until convergence with an MSE loss and a learning rate of 0.001.

D. Conformal Policies

We use two policies for evaluation. The first, π_{switch}, switches between π_{safe} and π_{speed} as a function of a threshold q_{safe} of the predicted future distance $\tilde{l}_t$ and its estimated quantile $\tilde{q}_t$ at $1 - \alpha = 90\%$ coverage computed with [22]. Specifically,

$$\pi_{switch}(x, \tilde{l}_t, \tilde{q}_t) = \begin{cases} \pi_{safe}(x), & \text{if } (\tilde{l}_t + \tilde{q}_t) \geq q_{\text{safe}} \\ \pi_{speed}(x), & \text{otherwise.} \end{cases} \quad (10)$$

π_{switch} satisfies Assumption A1 by design. The second, $\pi_{RL}(x, \tilde{l}_t, \tilde{q}_t)$, is directly trained with DQN with the same reward of π_{speed}. Ideally, π_{RL} will learn to modulate its speed as a function of its uncertainty. Note, however, that this design is not strictly guaranteed to satisfy Assumption A1.

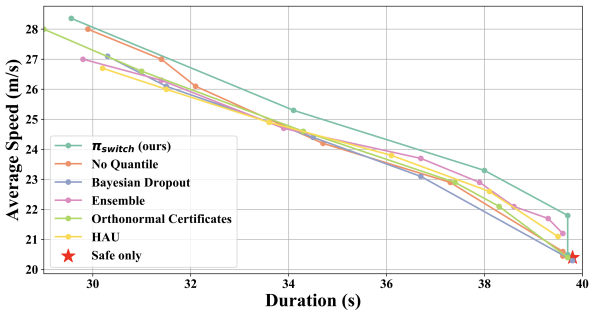

Fig. 2: Pareto plot for the simulation environments comparing the performance of π_{switch} to other baselines. Each method has its own Pareto curve, constructed by varying the switching threshold q_{safe}. Each point in the curve is the average speed and the rollout duration at a particular q_{safe} over 500 realizations. The red star is the performance of π_{safe}. Our approach always achieves a higher speed than the baselines at the same collision frequency.

E. Baselines

We compare against the following baselines for uncertainty estimation of $\tilde{l}$. **Bayesian Dropout [2]:** We add a 0.1 dropout rate at each linear layer of the predictor and compute uncertainties from five runs. **Ensemble [39]:** We train five networks with five disjoints subsets of the data. **HAU [4]:** Heteroscedastic Aleatoric Uncertainty (HAU). **No Quantile**: This is equivalent to π_{switch} but without quantiles in (10). Essentialy, the policy switches to π_{safe} if $\tilde{l}_t \geq q_{safe}$. **Orthonormal Certificates [40]:** We learn a set of linear functions that map the training data to zero. We use the predictor's second-to-last hidden layer as a feature representation for the input and project these features to a 100-dimensional space representing the linear certificates. We train this additional linear layer with the target label being 0 for 100 epochs.

F. Evaluation Results

We consider two metrics: the average rollout duration and the ego-vehicle's average rollout speed. We compute the metrics by averaging the results over 500 different environment realizations. We train the predictor and the base policies within a speed range of $[20, 30]m/s$. We then evaluate all policies by changing the speed range to $[20, 40]m/s$. This will result in the policy observing both in and out of distribution states during evaluation. At evaluation time, we vary the switching threshold q_{safe} in Eq. (10) to study the trade-off between safety and speed that each method provides. Results are reported in Fig. 2. Our approach consistently outperforms the baselines and enables faster driving for each safety constraint. The improvement provided by our approach is larger for more stringent values of q_{safe}, indicating that it better detects and reacts to dangerous conditions. Finally, our approach adds relatively no extra computation (only the update of Eq. (4)) and requires no changes to the model, and yet outperforms ensemble methods, which increase the runtime computation budget fivefold. Beyond better performance, our approach also provides a formal guarantee of $1 - \alpha = 90\%$

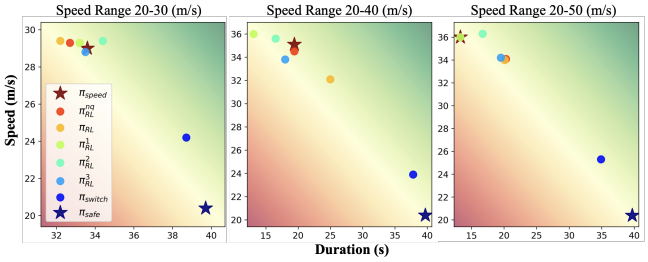

Fig. 3: Three subplots showing the policy evaluated on 20-30 m/s (**Left**), 20-40 m/s (**Middle**), and 20-50 m/s (**Right**). The top right green color indicates ideal behavior with high speed and duration. Red shading indicates worse behaviour. The stars represent policies that do not use conformal quantiles, like π_{speed} and π_{safe}.

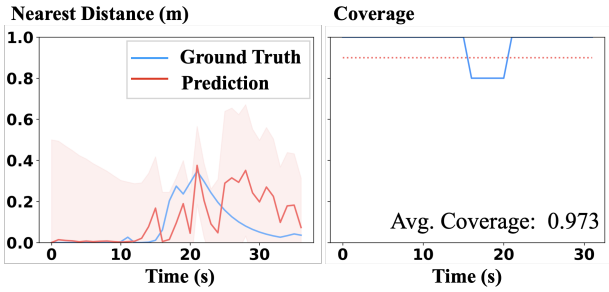

Fig. 4: **Left**: Prediction and conformal quantile estimation in one simulation rollout for speed range of [20,30]m/s. The ground truth value of $\tilde{l}$ is shown in blue and the prediction and quantile are shown in red. **Right**: Coverage with a sliding window size of 5 time steps. We show the coverage for $\tilde{l}$ for speed range of [20,30]m/s, which is higher than the specified value 0.9 (red dotted line) most of the time.

coverage, which we verify empirically in Fig. 4.

In Figure 3, we compare different variants of conformal policies trained end-to-end with RL to π_{speed} and π_{switch}. π_{RL} represents an agent that observes, in addition to the state x_t, the future prediction $\tilde{l}_t$ and its conformal quantile $\tilde{q}_t$. The most important baseline for π_{RL} is the agent π^{nq}_{RL}, which observes only $\tilde{l}_t$ and has no access to quantile uncertainties. We additionally ablate a set of reward designs for π_{RL} (all the following have access to both $\tilde{l}_t$ and $\tilde{q}_t$). π^1_{RL} is trained with an additional penalty on the norm of $\tilde{q}_t$. This should push the agent to stay in known states, actively avoiding distribution shifts. π^2_{RL} is trained with a penalty of $\tilde{q}_t \cdot \tilde{l}_t$. This should push the agent only to become conservative when the probability of collision rises (due to an increase in $\tilde{l}_t$), and quantile regression detects the state as out of distribution. The rationale is that, in distribution, the agent does not need to minimize the distance to other cars. π^3_{RL} is trained with both a penalty on the norm $\tilde{q}_t$ and a penalty on $\tilde{q}_t \cdot \tilde{l}_t$. For the latter three agents, we keep the basic reward constant but select, for each agent separately, the weighting coefficient of the penalty terms to maximize their performance.

All policies are trained in the speed range $[20, 30]m/s$ with DQN for 200K iterations and evaluated in three different speed ranges, increasingly out of the training distribution. Figure 3 indicates that π_{switch} is the safest agent by a significant margin as the performance is closest to the blue star for all three subplots. π_{switch} also achieves the best trade-off between speed and safety, indicated by its position in the green area. No agents trained with RL manage to use the predictions and quantiles as effectively as a switching policy. This indicates that traditional RL algorithms can fail to account for the *known unknowns*. Our hypothesis is that at training time (by definition), the agent only sees data in distribution, thereby with a small $\tilde{q}_t$. Therefore, it does not learn how to act when $\tilde{q}_t$ is large. Understanding how to make the policy take further advantage of conformal quantiles would be interesting for future work.

V. Physical Experiments

A. Experimental Setup

We evaluate our approach on an active perception problem on a physical quadruped. We only consider conformal policies with switching behaviors, i.e., π_{switch}, given their better performance with respect to end-to-end conformal policies like π_{RL}, as shown in the previous simulation experiments.

We attach a RealSense T265 tracking camera on the front of a Unitree Go1 quadruped with an onboard Nvidia Jetson Xavier NX. We use the built-in SLAM module of the camera to estimate the quadruped state. The objective is to have the quadruped follow a given trajectory as accurately and fast as possible. The built-in SLAM module may generate SLAM errors correlated to the quadruped planned path and speed, meaning the algorithm cannot estimate the current state. We design π_{switch} to speed up or slow down the quadruped to trade off the trajectory tracking quality with speed.

We use the built-in MPC to control the quadruped by providing the robot linear velocity commands in the x and y direction and the yaw angular velocity command. Given a tracking trajectory consisting of waypoints $(x_i, y_i, \theta_i), i \in [0, ..., N]$, where x_i, y_i, θ_i is the robot coordinates and orientation in the world frame, the robot commands are computed by a P controller such that $v_x(t) = kp_x \cdot (\hat{x_t} - x_i), v_y(t) = kp_y \cdot (\hat{y_t} - y_i), v_\theta(t) = kp_\theta \cdot (\hat{\theta_t} - \theta_i)$. $(\hat{x_t}, \hat{y_t}, \hat{\theta_t})$ are the estimated robot coordinates and orientations given by SLAM, and (x_i, y_i, θ_i) is the nearest waypoint from the given trajectory to the current robot state. By adaptively changing P gains kp_x, kp_y, kp_θ, our policy can speed up or slow down the quadruped to track fast or avoid SLAM errors, respectively. The policy runs at 10 Hz.

B. Predictor Training

We train a predictor to forecast whether the SLAM system will issue a tracking error. Specifically, the predictor takes in the observation of robot states and velocity commands of the past 5 steps, the current nearest waypoint, the next 3 waypoints to track, and SLAM errors received in the past 5 steps. Then, it predicts whether a SLAM error will occur at time step $t+2$, which we define as $\tilde{l}_t$. The predictor is trained on 43 randomly sampled trajectories. Each trajectory is generated by linearly interpolating two keypoints uniformly sampled from $[-2, 2]m$ to create a zig-zag trajectory, which the robot follows with kp of 1.5. The predictor is a one-layer

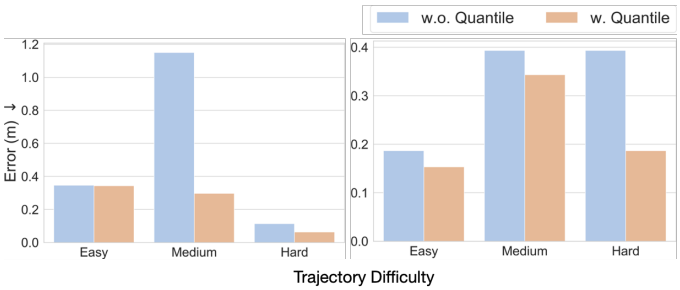

Fig. 5: Tracking error for in (left) and out (right) distribution.

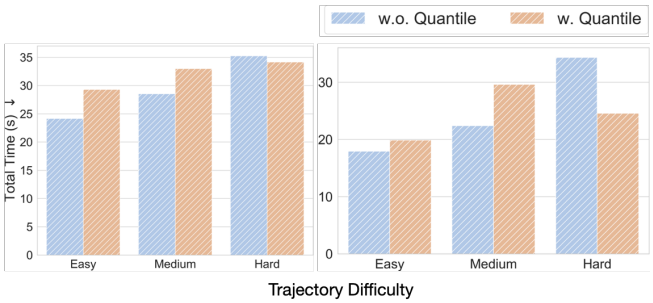

Fig. 6: Tracking time for in (left) and out (right) distribution.

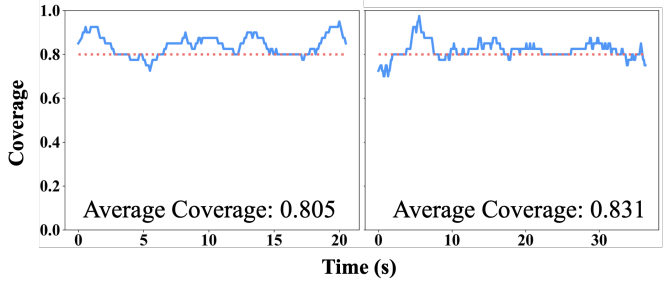

Fig. 7: Coverage plots with a sliding window size of 4 seconds generated with our method. We show the coverage for SLAM error prediction for a hard trajectory with within distribution P gains (**left**) and with out of the distribution P gains (**right**). In both cases, the average coverage over the entire rollout (labeled at the bottom right) is higher than the specified value 0.8, marked by the red dotted line.

MLP with 128 neurons and a single output. We also train the predictor with weighted binary cross entropy, upweighting the positive samples by factor of 5.

C. Policy Design

Our policy π_{switch} takes the prediction $\tilde{l}_t$ and the quantile $\hat{q}_t$ at $1-\alpha = 80\%$ coverage given by the conformal PID controller. π_{switch} uses these two to modulate the robot's speed. When $\tilde{l}_t + \tilde{q}_t \geq q_{safe}$, π_{switch} slows down the quadruped by updating $kp(t) = clip(0.8 * kp(t-1), kp_{min}, kp_{max})$. Conversely, when $\tilde{l}_t + \tilde{q}_t \leq q_{safe}$, i.e. the probability of failure is low, the policy speeds up the robot by increasing the P gain to $kp(t) = clip(1.1 * kp(t-1), kp_{min}, kp_{max})$. $q_{safe} = 0.8$ is a specified threshold and kp_{min}, kp_{max} are the lower and upper bound of kp. In practice, we notice the predictor can give many false positives, impacting the overall performance. π_{switch} detects the false positive by checking if $\tilde{l}_t - \tilde{q}_t < (1 - q_{safe})$ and $\tilde{l} + \tilde{q}_t > 1.1$. If this condition is satisfied, π_{switch} detects a false positive and speeds up the quadruped instead of slowing down. We use a baseline policy **No Quantile**, which modulates the robot speed only according to $\tilde{l}_t$ and q_{safe}, without any prediction uncertainty.

D. Evaluation Results

We evaluate the above policy on three closed trajectories with different difficulties. We use a triangle as the easy trajectory, a rectangle as the medium hard trajectory, and a hexagon as the hard trajectory. When the distance between the robot's estimated position and the last waypoint is lower than 0.1m and at least 75% of waypoints are achieved, the robot stops tracking. As it's hard to obtain the ground-truth quadruped positions, we denote the tracking error as the distance between the start and end points of the quadruped and record the time to finish tracking. The predictor is trained with data collected with a fixed P gain of 1.5. Experiments *in distribution* have $kp_{min} = 1.2$, $kp_{max} = 1.8$, therefore quite close to the predictor training data. Experiments *out of distribution* have $kp_{min} = 0.8$, $kp_{max} = 4$, much beyond what the predictor was trained for. For both in-distribution and out-of-distribution experiments, we repeat each trajectory three times and compare the tracking error and time for our policy and the baseline policy.

The results are summarized in Figure 5, 6 and 7. As shown in Figure 5, in both in-distribution and out-of-distribution scenarios, the policy using both the prediction and the quantile achieves a lower tracking error with a slightly longer tracking time. For the easy trajectory, π_{switch} achieves a similar tracking error to the baseline policy for in distribution experiment. The benefit of quantile shows up for more difficult trajectories and for the out-of-distribution experiments, where tracking error difference is more significant. π_{switch} is much better at tracking error for the in distribution medium trajectory where the predictor has a high uncertainty. As shown in Figure 6, for easy and medium trajectory, π_{switch} takes longer to track as it slows down more to avoid SLAM errors. π_{switch} is faster than the No Quantile policy for the hard trajectory with out-of-distribution states. This is because the predictor is overly conservative on this out-of-distribution trajectory and generates many false positives. Figure 7 shows the average coverage over a window of 4s for a hard trajectory for both in-distribution and out-of-distribution states using π_{switch}. In both cases, the coverage is higher than the desired threshold of $1 - \alpha = 0.8$, empirically validating the formal guarantee from the conformal quantiles.

VI. Conclusions

We studied conformal policy learning, a method for safe robotic control under distribution shift. The theoretical results prove that the conformal policy is sure to quickly exit regimes where its internal notion of uncertainty is flawed, causing its error rate to be too high. Physical and simulation experiments validate these claims in robotic control setups and show practical benefits compared to strong baselines under distribution shift. Designing algorithms to improve the performance of RL controllers when provided with explicit uncertainty measures would be an interesting future work.

REFERENCES

[1] J. M. Hernández-Lobato and R. Adams, "Probabilistic backpropagation for scalable learning of bayesian neural networks," in *International Conference on Machine Learning*, 2015, pp. 1861–1869.

[2] Y. Gal and Z. Ghahramani, "Dropout as a bayesian approximation: Representing model uncertainty in deep learning," in *international conference on machine learning*. PMLR, 2016, pp. 1050–1059.

[3] R. Koenker and G. Bassett Jr, "Regression quantiles," *Econometrica: journal of the Econometric Society*, pp. 33–50, 1978.

[4] A. Kendall and Y. Gal, "What uncertainties do we need in bayesian deep learning for computer vision?" *Advances in neural information processing systems*, vol. 30, 2017.

[5] O. Ian, "Risk versus uncertainty in deep learning: Bayes, bootstrap and the dangers of dropout," in *Advances in Neural Information Processing Systems Workshops*, 2016.

[6] V. Vovk, A. Gammerman, and G. Shafer, *Algorithmic Learning in a Random World*. Springer, 2005.

[7] J. Lei, M. G'Sell, A. Rinaldo, R. J. Tibshirani, and L. Wasserman, "Distribution-free predictive inference for regression," *Journal of the American Statistical Association*, vol. 113, no. 523, pp. 1094–1111, 2018.

[8] A. Fisch, T. Schuster, T. S. Jaakkola, and R. Barzilay, "Efficient conformal prediction via cascaded inference with expanded admission," in *International Conference on Learning Representations*, 2021. [Online]. Available: https://openreview.net/forum?id=tnSo6VRLmT

[9] Y. Romano, E. Patterson, and E. J. Candès, "Conformalized quantile regression," in *Advances in Neural Information Processing Systems*, 2019.

[10] A. N. Angelopoulos, S. Bates, J. Malik, and M. I. Jordan, "Uncertainty sets for image classifiers using conformal prediction," in *International Conference on Learning Representations (ICLR)*, 2021.

[11] Y. Romano, M. Sesia, and E. Candès, "Classification with valid and adaptive coverage," in *Advances in Neural Information Processing Systems*, H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, Eds., vol. 33. Curran Associates, Inc., 2020, pp. 3581–3591.

[12] A. N. Angelopoulos and S. Bates, "A gentle introduction to conformal prediction and distribution-free uncertainty quantification," 2021. [Online]. Available: https://arxiv.org/abs/2107.07511

[13] H. Papadopoulos, K. Proedrou, V. Vovk, and A. Gammerman, "Inductive confidence machines for regression," in *Machine Learning: European Conference on Machine Learning*, 2002, pp. 345–356.

[14] R. Luo, S. Zhao, J. Kuck, B. Ivanovic, S. Savarese, E. Schmerling, and M. Pavone, "Sample-efficient safety assurances using conformal prediction," in *International Workshop on the Algorithmic Foundations of Robotics*. Springer, 2022, pp. 149–169.

[15] H. Yang and M. Pavone, "Object pose estimation with statistical guarantees: Conformal keypoint detection and geometric uncertainty propagation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 8947–8958.

[16] A. Z. Ren, A. Dixit, A. Bodrova, S. Singh, S. Tu, N. Brown, P. Xu, L. Takayama, F. Xia, J. Varley *et al.*, "Robots that ask for help: Uncertainty alignment for large language model planners," *arXiv preprint arXiv:2307.01928*, 2023.

[17] I. Gibbs and E. J. Candès, "Adaptive conformal inference under distribution shift," in *Advances in Neural Information Processing Systems*, 2021.

[18] ——, "Conformal inference for online prediction with arbitrary distribution shifts," *arXiv preprint arXiv:2208.08401*, 2022.

[19] M. Zaffran, O. Féron, Y. Goude, J. Josse, and A. Dieuleveut, "Adaptive conformal predictions for time series," in *International Conference on Machine Learning*, 2022.

[20] O. Bastani, V. Gupta, C. Jung, G. Noarov, R. Ramalingam, and A. Roth, "Practical adversarial multivalid conformal prediction," *Advances in Neural Information Processing Systems*, vol. 35, pp. 29 362–29 373, 2022.

[21] A. Bhatnagar, H. Wang, C. Xiong, and Y. Bai, "Improved online conformal prediction via strongly adaptive online learning," *arXiv preprint arXiv:2302.07869*, 2023.

[22] A. N. Angelopoulos, E. J. Candès, and R. J. Tibshirani, "Conformal pid control for time series prediction," *arXiv preprint arXiv:2307.16895*, 2023.

[23] A. P. Dawid, "The well-calibrated bayesian," *Journal of the American Statistical Association*, vol. 77, no. 379, pp. 605–610, 1982.

[24] A. Dixit, L. Lindemann, S. X. Wei, M. Cleaveland, G. J. Pappas, and J. W. Burdick, "Adaptive conformal prediction for motion planning among dynamic agents," in *Learning for Dynamics and Control Conference*. PMLR, 2023, pp. 300–314.

[25] L. Lindemann, M. Cleaveland, G. Shim, and G. J. Pappas, "Safe planning in dynamic environments using conformal prediction," *IEEE Robotics and Automation Letters*, 2023.

[26] A. Muthali, H. Shen, S. Deglurkar, M. H. Lim, R. Roelofs, A. Faust, and C. Tomlin, "Multi-agent reachability calibration with conformal prediction," *arXiv preprint arXiv:2304.00432*, 2023.

[27] J. Lekeufack, A. N. Angelopoulos, A. Bajcsy, M. I. Jordan, and J. Malik, "Conformal decision theory: Safe autonomous decisions from imperfect predictions," *Published online at https://conformal-decision.github.io/static/pdf/submission.pdf*, 2023.

[28] A. Loquercio, M. Segu, and D. Scaramuzza, "A general framework for uncertainty estimation in deep learning," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 3153–3160, 2020.

[29] N. Sünderhauf, O. Brock, W. Scheirer, R. Hadsell, D. Fox, J. Leitner, B. Upcroft, P. Abbeel, W. Burgard, M. Milford, and P. Corke, "The limits and potentials of deep learning for robotics," *Int. Journ. on Robotics Research*, vol. 37, no. 4-5, pp. 405–420, 2018.

[30] A. Farid, D. Snyder, A. Z. Ren, and A. Majumdar, "Failure prediction with statistical guarantees for vision-based robot control," *arXiv preprint arXiv:2202.05894*, 2022.

[31] G. Kahn, A. Villaflor, B. Ding, P. Abbeel, and S. Levine, "Self-supervised deep reinforcement learning with generalized computation graphs for robot navigation," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, 2018.

[32] E. Kaufmann, M. Gehrig, P. Foehn, R. Ranftl, A. Dosovitskiy, V. Koltun, and D. Scaramuzza, "Beauty and the beast: Optimal methods meet learning for drone racing," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2019.

[33] K. Chua, R. Calandra, R. McAllister, and S. Levine, "Deep reinforcement learning in a handful of trials using probabilistic dynamics models," in *Advances in Neural Information Processing Systems*, 2018, pp. 4754–4765.

[34] A. Loquercio, A. Kumar, and J. Malik, "Learning visual locomotion with cross-modal supervision," in *IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2023, pp. 7295–7302.

[35] A. Bajcsy, A. Loquercio, A. Kumar, and J. Malik, "Learning vision-based pursuit-evasion robot policies," *arXiv preprint arXiv:2308.16185*, 2023.

[36] L. Smith, Y. Cao, and S. Levine, "Grow your limits: Continuous improvement with real-world rl for robotic locomotion," *arXiv preprint arXiv:2310.17634*, 2023.

[37] E. Leurent, "An environment for autonomous driving decision-making," https://github.com/eleurent/highway-env, 2018.

[38] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski *et al.*, "Human-level control through deep reinforcement learning," *nature*, vol. 518, no. 7540, pp. 529–533, 2015.

[39] B. Lakshminarayanan, A. Pritzel, and C. Blundell, "Simple and scalable predictive uncertainty estimation using deep ensembles," *Advances in neural information processing systems*, vol. 30, 2017.

[40] N. Tagasovska and D. Lopez-Paz, "Single-model uncertainties for deep learning," in *Advances in Neural Information Processing Systems*, vol. 32, 2019.