Percieve With Confidence: Statistical Safety Assurances for Navigation with Learning-Based Perception

Zhiting Mei*, Anushri Dixit†, Meghan Booker‡§, Emily Zhou*,
Mariko Storey-Matsutani*, Allen Z. Ren*, Ola Shorinwa*, Anirudha Majumdar*
*Princeton University
†University of California, Los Angeles
‡Johns Hopkins University Applied Physics Laboratory
§Work conducted while at Princeton University
Contact: ani.majumdar@princeton.edu

***Abstract*—Rapid advances in perception have enabled large pre-trained models to be used out of the box for transforming high-dimensional, noisy, and partial observations of the world into rich occupancy representations. However, the reliability of these models and consequently their safe integration onto robots remains unknown, particularly when deployed in environments unseen during training. To provide safety guarantees, we rigorously quantify the uncertainty of pre-trained perception systems for object detection and scene completion via a novel calibration technique based on conformal prediction. Crucially, this procedure guarantees robustness to distribution shifts in states when perception outputs are used in conjunction with a planner. As a result, the calibrated perception system can be used in combination with *any* safe planner to provide an end-to-end statistical assurance on safety in unseen environments. We evaluate the resulting approach, *Perceive with Confidence* (PWC), in simulation and on hardware where a quadruped robot navigates through previously unseen static indoor environments. These experiments validate the safety assurances for obstacle avoidance provided by PWC. In simulation, our method reduces obstacle misdetection by 70% compared to uncalibrated perception models. While misdetections lead to collisions for baseline methods, our approach consistently achieves 100% safety. We further demonstrate reducing the conservatism of our method without sacrificing safety, achieving a 46% increase in success rates in challenging environments while maintaining 100% safety. In hardware experiments, our method improves empirical safety by 40% over baselines and reduces obstacle misdetection by 93.3%. The safety gap widens to 46.7% when navigation speed increases, highlighting our approach's robustness under more demanding conditions.**

I. Introduction

How can we decide if the outputs of a given perception system are sufficiently reliable for safety-critical robotic tasks such as autonomous navigation? Significant strides in perception over the past few years have enabled large pre-trained models to be used out of the box [1] for tasks such as *object detection* and *occupancy prediction*, which serve as a fundamental building block for navigation. However, current pre-trained models are still not reliable enough for safe integration into many real-world robotic systems. Despite being trained on vast amounts of data, these systems can often fail to generalize to novel environments [2–4]. In this paper, we ask: *how can we leverage the power of large pre-trained perception models while providing safety assurances for robot navigation?*

Consider a legged robot tasked with navigating a cluttered environment such as a home, office, or warehouse (Figure 1). A typical navigation pipeline for such a system consists of two modules: (i) a perception module that detects obstacles, and (ii) a planner that produces collision-free trajectories assuming accurate perception. However, there are two challenges associated with obtaining reliable outputs from the perception module. First, the environments in which we deploy our robots will be *unseen* during training, and thus require *generalization* to new obstacle geometries, appearances, and other environmental factors. Second, *closed-loop deployment* of the perception system in conjunction with a planner causes a shift in the distribution of *states* (e.g., relative locations to obstacles) that are visited by the robot. Since the robot's planner influences future states, the robot may view obstacles from unfamiliar relative poses (Figure 1), which can cause the perception system to fail.

In this paper, we address these challenges by performing rigorous *uncertainty quantification* for the outputs of a pre-trained perception system in order to achieve reliably safe (i.e., collision-free) navigation. We utilize techniques from *conformal prediction* [5] in order to lightly process the outputs of a pre-trained perception system in a way that provides a *formal assurance* on correctness, i.e., with a user-specified probability $1-\epsilon$, the processed perception outputs will correctly detect obstacles in a *new* environment. To enable this, we assume access to a relatively small-sized (e.g., $|\cdot| = 400$) dataset of environments that are representative of deployment environments with ground-truth obstacle annotations, and use these for *calibrating* the outputs of the perception system. Crucially, we propose a novel calibration technique that ensures robustness of the perception system to *any closed-loop distribution shift in states*. Hence, the calibrated outputs can be used in conjunction with *any* safe planner to provide an end-to-end statistical assurance on safety in new static

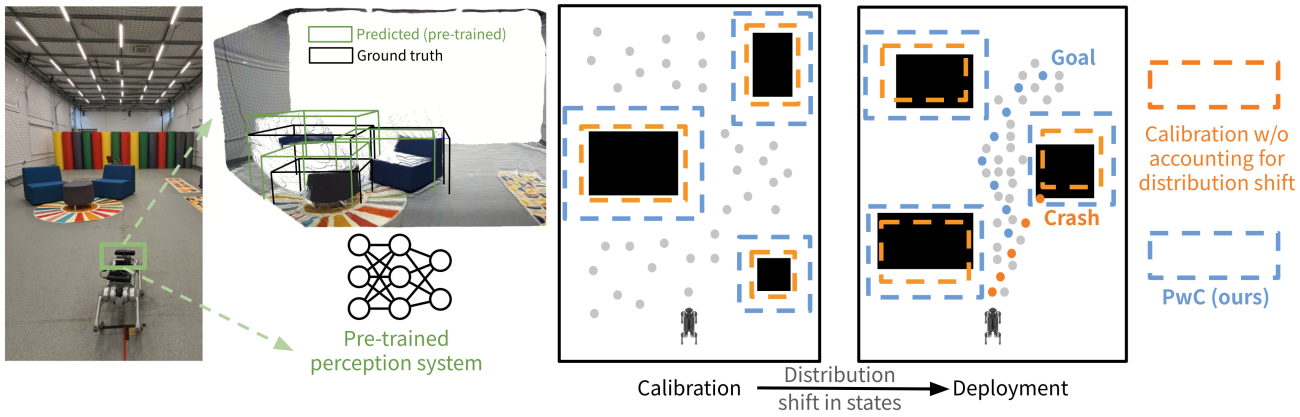


Fig. 1: PwC lightly processes the outputs of a pre-trained perception system (green bounding boxes) using conformal prediction in order to ensure a bounded misdetection rate despite *any* distribution shift in states (gray dots). The calibrated perception system (blue boxes) paired with a non-deterministic filter and a safe planner provide an end-to-end statistical assurance on safety in new test environments.

environments with a user-specified threshold $1-\epsilon$. To the best of our knowledge, this is the first work to calibrate a black-box perception system in a way that ensures robustness to closed-loop distribution shifts in order to provide end-to-end statistical assurances on safe navigation.

Our proposed framework, *Perceive with Confidence* (PwC), is amenable to different types of perception systems, e.g., bounding-box prediction perception systems and scene-completion perception systems. A preliminary version of this work, which was presented at the Conference on Robot Learning [6], focused on the calibration of bounding-box predictions from onboard robot perception systems. In the preliminary version, we demonstrate the end-to-end safety guarantees provided by PwC in simulated and hardware experiments on the Unitree Go1 quadruped navigating in indoor environments with objects that are unseen during calibration (Figure 1), where PwC achieves up to 40% increase in safety with only modest reductions in task completion rates compared to baselines that use the pre-trained perception model directly, fine-tune it on the calibration dataset, or utilize conformal prediction for uncertainty quantification but do not account for closed-loop distribution shift.

However, bounding boxes lack the high level of expressiveness required for high-accuracy occupancy predictions in environments with complex geometry. In this work, we extend PwC to perception systems that predict 3D occupancy maps, providing higher-fidelity scene representations for robot planning and navigation. Specifically, in this paper, we derive procedures for calibrating unsigned distance functions predicted by scene-completion models to safely decompose a robot's environment into free and occupied space. We call the resulting method PwC-NU-MCC, based on the scene-completion model used in this work, NU-MCC [7]. In addition, we present simulated experiments on a quadruped robot, demonstrating that PwC-NU-MCC outperforms the calibrated bounding-box predictor, improving the goal-reaching rate by 46%, particularly in scenes with intricate geometry (Section VII-B). Further, we present additional quadruped robot experiments, achieving faster navigation with the robot moving more than three times faster compared to our original experiments (Section VIII-B).

The rest of the paper is organized as follows: we first review relevant literature in Section II. In Section III, we formulate the problem of rigorous calibration of perception systems with safety guarantees. In Section IV, we provide a brief introduction to conformal prediction. Next, we discuss the offline calibration of perception systems in Section V followed by a discussion of the online perception and planning procedure Section VI. We present simulated and hardware experiments on a quadruped robot in Sections VII and VIII, respectively. We conclude in Section IX and provide additional discussion, including simulations and hardware results, in the Appendix.

II. Related Work

Safe planning. Collision avoidance is a crucial goal in autonomous navigation. Safe planning methods typically rely on the assumption that the robot has perfect knowledge of its state and environment [8]. Recent approaches have allowed for occlusion [9–12] or accounted for losing sight of a previously tracked object [13], but still require either perfect detection of seen objects or bounded sensor noise. Such assumptions are impractical for learning-based perception modules that can fail catastrophically in new environments.

Formal assurances for perception-based control. Proposed methods include control barrier functions (CBFs) [14, 15], verification methods on neural networks (NNs) [16, 17], and other learning-based methods [17–24]. However, these works either do not guarantee generalization to novel environments [16, 17], ignore closed-loop distribution shifts [20, 21], require end-to-end training and a good prior [22–24], or demand usage/design of specific controllers [14, 15, 18, 25]. Some make strong assumptions on the perception system [26, 27] that are unrealistic for deployment. In contrast, our method doesn't need any of the above, and is lightweight and modular, allowing for the use of any downstream safe planners to ensure end-to-end safety.

Conformal prediction. Conformal prediction (CP) [5, 28, 29]

is an uncertainty quantification framework particularly suitable for robotics applications [30–33] where learned modules are deployed in environments drawn from unknown distributions. In this work, we focus on providing uncertainty quantification for the perception system, which usually involves high-dimensional inputs and closed-loop distribution shifts. Prior works [20, 32, 34, 35] either provide guarantees for a single environment, assume known environments, or do not account for closed-loop distribution shifts. To the best of our knowledge, this is the first work to obtain end-to-end safety assurances for the perception and planning system in new environments while being robust to closed-loop distribution shifts and amenable to changes in the planner parameters.

III. Problem Formulation

Dynamics and environments. Suppose that the dynamics of the robot are described by $s_{t+1} = f_E(s_t, a_t)$, where $s_t \in \mathcal{S}$ is the robot's state at time-step t, $a_t \in \mathcal{A}$ is the action, and $E \in \mathcal{E}$ is the *environment* that the robot operates in during a given episode. We primarily focus on navigation with static obstacles; in this context, the environment E specifies the locations and geometries of objects. We assume that environments that the robot will be deployed in are drawn from an *unknown* distribution $\mathcal{D}_\mathcal{E}$, e.g., a distribution over possible rooms that the robot may be deployed in. We will make no assumptions on this distribution besides the ability to sample a finite dataset $D = \{E_1, \ldots, E_N\}$ of independent identically distributed (i.i.d.) environments from $\mathcal{D}_\mathcal{E}$.

Sensor and perception system. We consider a robot equipped with a sensor $\sigma : \mathcal{S} \times \mathcal{E} \to \mathcal{O}$ that provides observations $o_t = \sigma(s_t, E)$ (e.g., depth images) based on the robot's state and environment. We assume access to a pre-trained perception model $\phi : \mathcal{O} \to \mathcal{Z}$, which processes raw sensor observations o_t into an occupancy representation of the environment z_t. For example, models for 3D object detection can produce bounding boxes for obstacles [36], perform shape completion [37], or predict free space in the environment [38]. We demonstrate our framework with two types of perception models: (i) models for obstacle detection that output 3D bounding boxes, and (ii) models for scene reconstruction that output occupancy grid maps. The representations $(z_0, \ldots, z_t)$ up to the current time-step are aggregated into an overall representation $m_t \in \mathcal{M}$ (e.g., a map). We denote predicted occupied space in green and predicted free space in blue, except where noted otherwise.

Planner and Policy. The planner utilizes the environment representation m_t to compute actions for the given task. We denote the resulting end-to-end policy that utilizes a perception model ϕ by $\pi^\phi : \mathcal{O}^{t+1} \to \mathcal{Z}^{t+1} \to \mathcal{M} \to \mathcal{A}$, which maps histories of sensor observations to actions.

Safety and task performance. Let C_E^{safe} be a cost function that captures safety (e.g., obstacle avoidance). In addition, let $\mathcal{S}_{0,E}$ denote the allowable set of initial conditions in environment E. Then, $C_E^{\text{safe}}(\pi^\phi) \in \{0, 1\}$ assigns a cost of 0 if policy π^ϕ maintains safety from any initial state $s_0 \in \mathcal{S}_{0,E}$ when deployed over a given time horizon in environment E, and a cost of 1 otherwise. Although we only present safety-oriented cost functions here, additional cost functions can be used to capture task performance, e.g., C_E^{task} which minimizes the time to reach the goal.

Goal (statistical safety assurance). Our goal is to provide a statistical assurance on safety for the end-to-end policy π^ϕ. We propose a procedure that uses a finite dataset D of environments in order to produce a *calibrated* perception system $\tilde{\phi} : \mathcal{O} \xrightarrow{\phi} \mathcal{Z} \xrightarrow{\rho} \mathcal{Z}$. Our approach is modular: outputs of the calibrated perception system may be used with *any* safe planner (cf. Section VI) to provide probabilistic guarantees on safety, with:

$$C^{\text{safe}}_{\mathcal{D}_\mathcal{E}}(\pi^{\tilde{\phi}}) := \mathop{\mathbb{E}}_{E \sim \mathcal{D}_\mathcal{E}} \left[C_E^{\text{safe}}(\pi^{\tilde{\phi}}) \right] \leq \epsilon, \tag{1}$$

for a user-specified safety tolerance ϵ, while also post-processing outputs from ϕ as lightly (i.e., non-conservatively) as possible in order to allow the robot to optimize task performance.

IV. Background: Conformal Prediction

We leverage the theory of conformal prediction (CP) to perform rigorous uncertainty quantification for perception. Here, we provide a brief overview of conformal prediction and refer interested readers to [5, 39] for a more detailed discussion.

Given N i.i.d. (or *exchangeable*) samples $U_1, \ldots, U_N$ of a scalar random variable U, we compute the threshold, $\hat{q}_{1-\epsilon}$, such that the next sample, U_{test}, satisfies,

$$\mathbb{P}[U_{\text{test}} \leq \hat{q}_{1-\epsilon}] \geq 1 - \epsilon, \tag{2}$$
$$\hat{q}_{1-\epsilon} = \begin{cases} U_{(\lceil (N+1)(1-\epsilon) \rceil)} & \text{if } \lceil (N+1)(1-\epsilon) \rceil \leq N, \\ \infty & \text{otherwise,} \end{cases}$$

where $U_{(1)} \leq U_{(2)} \leq \ldots \leq U_{(N)}$ are the order statistics (sorted values) of the N samples $U_1, \ldots, U_N$. In the CP literature, the non-conformity score U represents a measure of the (in)correctness of a model.

The guarantee in (2) is *marginal*, i.e., (2) holds over the sampling of both the calibration dataset $U_1, \ldots, U_N$ and the test variable U_{test}. Hence, we will need to generate a fresh set of i.i.d. calibration data $\tilde{U}_1, \ldots, \tilde{U}_N$ for the guarantee to hold for a new sample $\tilde{U}_{\text{test}}$. However, in practice, one typically only has access to a single dataset of examples; inferences from this dataset must be used for all future predictions on test examples. In this work, we use the following dataset-conditional guarantee [28, 29] that doesn't require us to generate of N new samples for every test prediction and holds with probability $1 - \delta$ over the sampling of the calibration dataset:

$$\mathbb{P}[U_{\text{test}} \leq \hat{q}_{1-\epsilon} | U_1, \ldots, U_N] \geq \text{Beta}^{-1}_{N+1-v,v}(\delta), \tag{3}$$
$$v := \lfloor (N+1)\hat{\epsilon} \rfloor,$$

where, $\text{Beta}^{-1}_{N+1-v,v}(\delta)$ is the $\delta-$quantile of the Beta distribution with parameters $N + 1 - v$ and v, and we can choose $\hat{\epsilon}$ to achieve the desired $1 - \epsilon$ coverage.

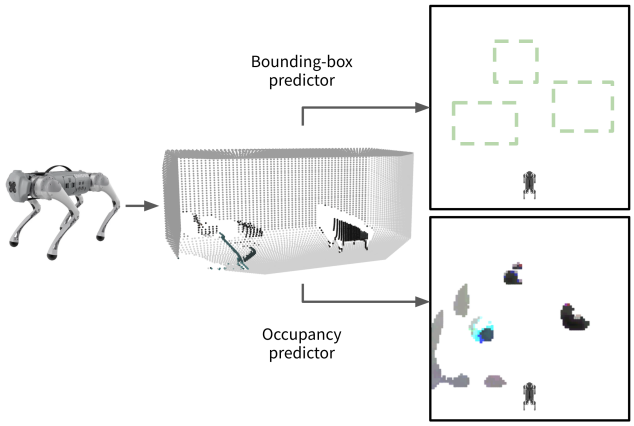


Fig. 2: Our proposed method is amenable to a range of perception systems, e.g., bounding-box predictors (top), which output a map with predicted bounding boxes (green dashed boxes), and occupancy predictors (bottom), which output a map with predicted occupied regions.

V. Calibrating the Perception System

In this section, we describe our approach to the uncertainty quantification of a pre-trained perception system. We focus on the challenges highlighted in Section I: providing statistical assurances on safe generalization to novel environments and ensuring that the offline calibration procedure is robust to shifts in the distribution of states induced by the online implementation of the planner.

We consider two types of perception systems: (i) perception systems that output bounding boxes predicting the locations of objects in the environment, and (ii) perception systems that perform scene reconstruction and output occupancy grid maps representing the occupancy of the environment. For example, Figure 2 illustrates the maps produced by (i) the bounding-box predictor and (ii) the occupancy prediction.

A. Misdetection Rate

Our key idea for ensuring *generalization* to new, unseen environments and tackling *the distribution shift arising from the closed-loop deployment* of the calibrated perception system with a plannner is to use a *policy-independent* misdetection cost, $\tilde{C}_E$, which considers worst-case errors across *all* states in an environment[1], $\tilde{C}_E(\phi) := \max_{s\in\mathcal{S}} \mathbb{1}_{\mathcal{X}^{\text{occ}} \not\subseteq \overline{\mathcal{X}}^{\text{occ}}_s}$. We present a calibration procedure that bounds this misdetection cost with high probability in a new environment, and thus guarantee the correctness of the calibrated perception system independent of the robot policy using CP. Moreover, we note that the perception system can be fine-tuned to the target deployment environments to reduce the nominal misdetection rate, which we discuss in Appendix A.

[1]It would be infeasible to consider *all* possible states in an environment. In practice, we use a sampling-based motion planner and consider a fixed set of samples for our calibration that could be used by any planner.

B. Calibration Procedure

Dataset. We assume access to a dataset of N i.i.d. environments $D = \{E_1, \dots E_N\} \sim \mathcal{D}_\mathcal{E}$ (cf. Section III). Let $\mathcal{X}_i$ denote the configuration space of environment E_i (e.g., x-y location). In each environment, E_i, we have access to the ground-truth occupied space $\mathcal{X}^{\text{occ}}_i$ and the predicted occupied space $\overline{\mathcal{X}}^{\text{occ}}_{s,i}$, generated by the pre-trained perception system ϕ for each state $s \in \mathcal{S}$. Care is required to ensure that the calibration environments are representative of deployment environments. As such, we construct the calibration dataset using real-world environments or create simulation environments using real-world data [40–42] to ensure sufficient variation in environmental factors (e.g., geometry and locations of obstacles, lighting, etc.).

Calibration. In each calibration environment E_i, we define a parameter q_i that monotonically scales the predicted occupied space to be $\overline{\mathcal{X}}^{\text{occ}}_{s,i}(q_i)$. In other words, as we increase q_i, $\overline{\mathcal{X}}^{\text{occ}}_{s,i}(q_i)$ expands monotonically. We find the q_i such that the ground truth occupied space is fully enclosed by the scaled prediction, i.e., $\mathcal{X}^{\text{occ}}_i \subseteq \overline{\mathcal{X}}^{\text{occ}}_{s,i}(q_i), \forall s \in \mathcal{S}$, where, $\mathcal{S}$ is assumed to be a finite, discrete set. In Section VII, we provide concrete examples on how to choose the parameter q for the two types of perception models considered. We define the *non-conformity score* for environment E_i to be the minimum required scaling parameter q_i in that environment:

$$U_i \;=\; \min_{q_i} \; q_i \quad \text{s.t} \quad \mathcal{X}^{\text{occ}}_i \subseteq \overline{\mathcal{X}}^{\text{occ}}_{s,i}(q_i), \forall s \in \mathcal{S}. \tag{4}$$

Observe that $U_i \leq 0 \implies \mathcal{X}^{\text{occ}}_i \subseteq \overline{\mathcal{X}}^{\text{occ}}_{s,i}$, $\forall s \in \mathcal{S}$ and a growing U_i signals a worse performance of the pre-trained perception system. We can compute the nonconformity scores for the i.i.d. sampled environments $\{E_1, \dots, E_N\}$ and the quantile $\hat{q}_{1-\epsilon} = \text{Quantile}\left(U_{(1)}, \dots, U_{(N)}; \frac{\lceil (N+1)(1-\hat{\epsilon}) \rceil}{N}\right)$. Here, $\hat{\epsilon}$ is the calibration threshold such that the dataset conditional guarantee (3) achieves the desired $(1-\epsilon)-$coverage with probability $1-\delta = 0.99$ over the sampling of the calibration dataset.

Proposition 1. *Consider the calibrated perception system $\tilde{\phi}$ that modifies every output of the perception system ϕ by scaling the predicted occupied space as $\overline{\mathcal{X}}^{\text{occ}}_{s,i}(q_i)$. With probability $1-\delta$ over the sampling of the dataset used for calibration, the calibrated perception system, $\tilde{\phi}$, is guaranteed to have an ϵ-bounded misdetection rate on* new *test environments:*

$$\mathop{\mathbb{E}}_{E_{test}\sim\mathcal{D}_\mathcal{E}}\left[\tilde{C}_{E_{test}}(\tilde{\phi}) | U_1, \dots, U_N\right] \leq \epsilon. \tag{5}$$

Proof: As seen in Section IV, conformal prediction gives us the following *dataset-conditional* guarantee on a new sample of the nonconformity score U_{test} corresponding to a test environment E_{test}. With probability $1-\delta$ over the sampling of $U_1, \dots, U_N$,

$$\mathbb{P}[U_{\text{test}} \leq \hat{q}_{1-\epsilon} | U_1, \dots, U_N] \geq \text{Beta}^{-1}_{N+1-v,v}(\delta).$$

We can rewrite the event $U_{\text{test}} \leq \hat{q}_{1-\epsilon}$ as:

$$\begin{aligned}
&\{U_{\text{test}} \leq \hat{q}_{1-\epsilon}\} \\
=&\left\{\hat{q}_{1-\epsilon} \geq \min_{q_{\text{test}}} q_{\text{test}} \mid \mathcal{X}^{\text{occ}}_{\text{test}} \subseteq \overline{\mathcal{X}}^{\text{occ}}_{s,\text{test}}(q_{\text{test}}), \forall s \in \mathcal{S}\right\} \\
=&\left\{\mathcal{X}^{\text{occ}}_{\text{test}} \subseteq \overline{\mathcal{X}}^{\text{occ}}_{s,\text{test}}(\hat{q}_{1-\epsilon}), \forall s \in \mathcal{S}\right\} \\
=&\left\{\tilde{C}_{E_{\text{test}}}(\tilde{\phi}) = 0\right\},
\end{aligned}$$

which gives us the desired result (5). ∎

Proposition 1 gives us a formal assurance on the correctness of the perception system *independent of the robot's policy*. As we describe below, the calibrated perception can thus be combined with *any* safe planner to bound the collision rate to ϵ. The calibrated perception outputs are guaranteed to be correct with probability $1-\epsilon$ over environments. Since we accounted for the perception error from every state in each environment, the resulting calibrated outputs are also guaranteed to be correct from every state in new test environments. Given that we have addressed the challenge of closed-loop distribution shift, we can now utilize this calibrated perception system with any safe planner to obtain a statistical assurance on robot safety.

C. Implementation with a limited field-of-view

A natural question that arises after following the calibration procedure described above is: what happens if the robot is not able to observe all objects in the environment from all states? This may happen due to a limited sensing capability or because some parts of the environment are occluded from view. We address this issue in our calibration procedure implementation by only taking into account perception errors for objects that are within the field-of-view of the robot in a given state, and masking any region of the ground-truth occupied space that is not visible to the robot, i.e., $\mathcal{X}^{\text{occ}}$ (which now depends on state s) is the ground-truth visible occupied region. Hence, the perception system correctness assurance stated above holds for all objects within the field-of-view of the robot at any given state. The presence of possibly occluded obstacles is dealt with by a safe planner, which we describe next.

VI. Perception and Planning

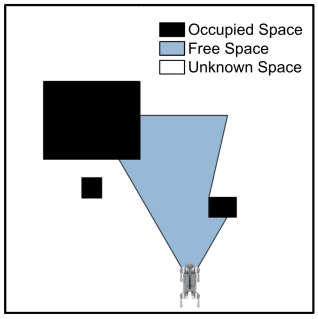


Fig. 3: The configuration space is partitioned into three.

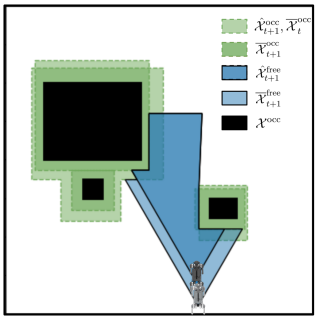


Fig. 4: The filter takes union over the free space.

We now focus on the online implementation of the method described in Section V to reduce conservatism when used in conjunction with a safe planner. In general, a safe planner takes into account the dynamics of the robot and produces plans in the state space $\mathcal{S}$. Let $\mathcal{X}$ be the configuration space of the robot (e.g., x-y location for a point). The configuration space of any given environment E can be partitioned into the ground-truth occupied space $\mathcal{X}^{\text{occ}}$, the known free space $\mathcal{X}^{\text{free}}$, and the unknown space $\mathcal{X}^{\text{unknown}}$ (Figure 3).

Non-deterministic filter. We utilize the assurance obtained from Section V to implement a *non-deterministic filter* [43, Ch. 11.2.2], which shrinks the occupied space and grows the known free space over time (Figure 4). Suppose the robot's perceived partition of the configuration space $\mathcal{X}$ at time t is denoted by the triplet $\{\overline{\mathcal{X}}^{\text{free}}_t, \overline{\mathcal{X}}^{\text{occ}}_t, \overline{\mathcal{X}}^{\text{unknown}}_t\}$, which represents the overall map m_t of the environment. At a new time step $t+1$, the robot's perception system returns a new estimation for the occupied space, $\hat{\mathcal{X}}^{\text{occ}}_{t+1}$. The filter intersects the occupied spaces: $\overline{\mathcal{X}}^{\text{occ}}_{t+1} = \overline{\mathcal{X}}^{\text{occ}}_t \cap \hat{\mathcal{X}}^{\text{occ}}_{t+1}$. We compute the new estimation of free space $\hat{\mathcal{X}}^{\text{free}}_{t+1}$ based on $\overline{\mathcal{X}}^{\text{occ}}_{t+1}$, considering occlusion and limited field of view. The new perceived free space is updated by taking the union: $\overline{\mathcal{X}}^{\text{free}}_{t+1} = \overline{\mathcal{X}}^{\text{free}}_t \cup \hat{\mathcal{X}}^{\text{free}}_{t+1}$.

The non-deterministic filter pairs effectively with our method in Section V for two key reasons: 1) it mitigates the conservatism of our expansion procedure for the predicted occupied space by intersecting $\overline{\mathcal{X}}^{\text{occ}}_t$, rapidly reducing its size even if the initial prediction with CP bounds appears generous; and 2) Proposition 1 ensures that with high probability in a new test environment, $\overline{\mathcal{X}}^{\text{free}}_t$ never intersects the true occupied space $\mathcal{X}^{\text{occ}}$. We demonstrate the rapid expansion of known free space in Figure 6 for our simulated setup (Section VII).

Safe planning. With our formal assurance on the estimated free space $\overline{\mathcal{X}}^{\text{free}}_t$, we can utilize *any* safe planner [44–46] to ensure end-to-end safety, as long as the planner includes a safety filter that takes into account the robot's dynamics in order to reject potentially unsafe actions with the assumption of known robot states and a static (but unknown) environment [8, Corollary 1.4].

For our simulation and hardware experiments, we use the safe planner proposed in [9] due to its approximate optimality. The safety filter in this case is an inevitable collision set (ICS) constraint [47], where the robot is forbidden to enter any state that will eventually result in collision no matter what control actions are taken. Within the known free space $\overline{\mathcal{X}}^{\text{free}}_t$, the robot plans using the fast marching tree algorithm (FMT*) [48] with dynamics [49]. If the goal is not visible within $\overline{\mathcal{X}}^{\text{free}}_t$, the robot plans to an intermediate goal on the boundary of its free space. The intermediate goals are chosen based on the cost-to-come from current robot state to the intermediate goal, and the distance-to-go from the intermediate goal to the actual goal. The robot replans whenever it receives a sensor update and an updated $\overline{\mathcal{X}}^{\text{free}}_{t+1}$ from its non-deterministic filter, and accounts for ICS constraints [50] in-between sensor updates.

Proposition 2. *For any user-specified safety tolerance ϵ, the* calibrated *perception system $\tilde{\phi}$ in Proposition 1 combined with any safe planner that chooses actions based on the outputs of*

the non-deterministic filter ensures the end-to-end safety for the overall policy $\pi^{\tilde{\phi}}$:

$$C^{safe}_{\mathcal{D}_\mathcal{E}}(\pi^{\tilde{\phi}}) := \mathop{\mathbb{E}}_{E\sim\mathcal{D}_\mathcal{E}}\left[C^{safe}_{E}(\pi^{\tilde{\phi}})\right] \leq \epsilon, \tag{6}$$

where $C^{safe}_{E}(\pi^{\tilde{\phi}})$ is the cost for safety from Section III.

Proof: As shown in Proposition 1, the misdetection rate of the calibrated perception system $\tilde{\phi}$ is ϵ-bounded on environments drawn from $\mathcal{D}$ at each time step t, where the robot is at state s_t. In other words, the predicted occupied space $\hat{\mathcal{X}}^{\text{occ}}_t$ at each time step contains the true occupied space $\mathcal{X}^{\text{occ}}$ with high probability across environments. Conversely, the predicted free space $\hat{\mathcal{X}}^{\text{free}}_t$ at each time step does not intersect with the true occupied space $\mathcal{X}^{\text{occ}}$ with high probability across environments. If we consider a safety-relevant misdetection cost at time step t:

$$\hat{C}^{\text{safe}}_{E}(\tilde{\phi}, s_t) = \begin{cases} 1 & \text{if } \mathcal{X}^{\text{occ}} \subseteq \hat{\mathcal{X}}^{\text{free}}_t \text{ (unsafe)}, \\ 0 & \text{otherwise}, \end{cases} \tag{7}$$

then the misdetection rate over the set of states should be ϵ-bounded across environments by Proposition 1:

$$\mathop{\mathbb{E}}_{E\sim\mathcal{D}_\mathcal{E}} \max_{t\in[0,T]} \hat{C}^{\text{safe}}_{E}(\tilde{\phi}, s_t) \leq \epsilon. \tag{8}$$

Because the expectation in Equation (8) is over the set of environments, the following statement holds in any new environment (with probability $1-\delta$ over the calibration dataset of environments),

$$\mathbb{P}\Big\{ \max_{t\in[0,T]} \hat{C}^{\text{safe}}_{E}(\tilde{\phi}, s_t) = 0 \Big\} \geq 1-\epsilon. \tag{9}$$

Given $m_t = \{\overline{\mathcal{X}}^{\text{free}}, \overline{\mathcal{X}}^{\text{occ}}, \overline{\mathcal{X}}^{\text{unknown}}\}$, a safe planner never drives the robot outside of the free space. Therefore, the safe planner guarantees $C^{\text{safe}}_{E}(\pi^{\tilde{\phi}}) \leq \tilde{C}^{\text{safe}}_{E}(\tilde{\phi})$.

$$\mathbb{P}\Big\{ C^{\text{safe}}_{E}(\pi^{\tilde{\phi}}) = 0 \Big\} \geq 1-\epsilon. \tag{10}$$

■

This result is a direct consequence of the formal assurance on the calibrated perception system that ensures correctness from *any* state in a new test environment (sampled i.i.d. from the same distribution as the calibration environments) with probability $1-\epsilon$ *over environments*.

We discuss extensions of our perception and planning approach to problems with sensor and dynamics uncertainty in Appendix A.

VII. Simulated Experiments

We evaluate our approach for vision-based navigation in the PyBullet simulator [51] using a diverse set of chairs from the 3D-Front dataset [42].

Simulation Environment. We specify the environment distribution by randomly placing $1-5$ chairs from the diverse 3D-Front dataset [42] in an 8 m $\times$8 m room (Figure 5). We construct the simulation environment using CAD models of *real* furniture pieces from the 3D-Front dataset [42], which contains a highly diverse array of industrial CAD models developed by professional designers.

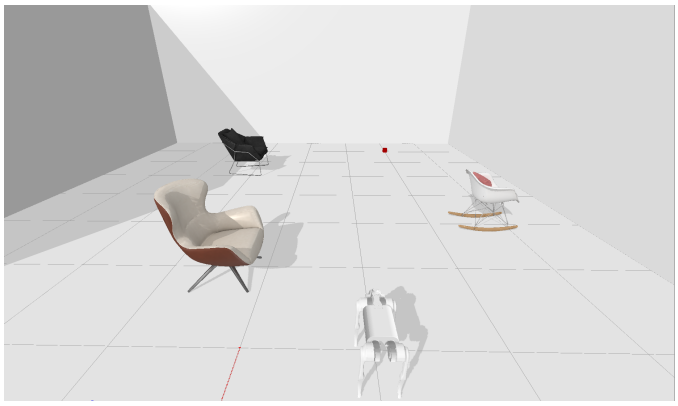


Fig. 5: Simulation environment in Pybullet.

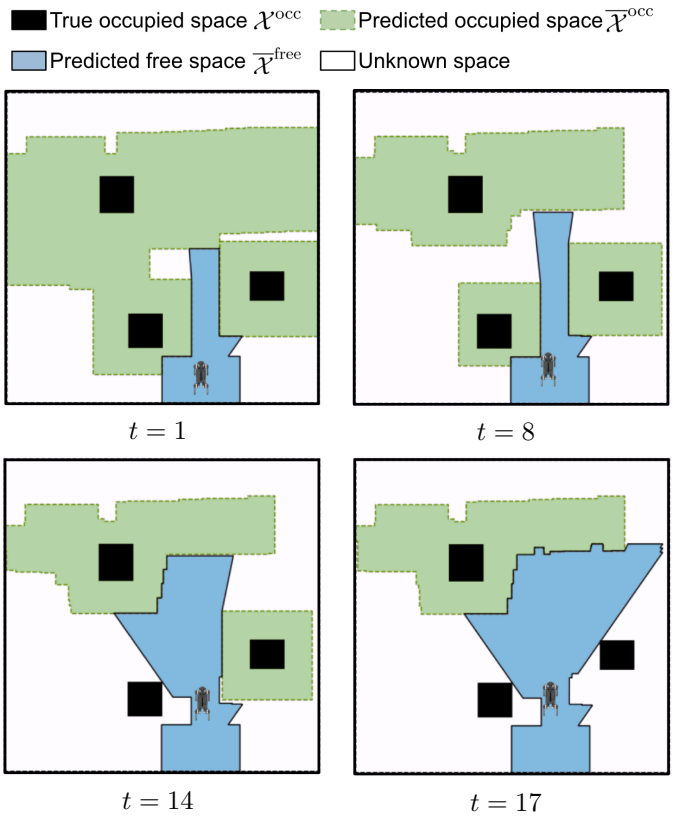


Fig. 6: Simulation and non-deterministic filter updates. The robot begins with large occupied space predictions due to the inflation obtained through offline calibration (Section V). After a few updates, the predicted occupied space $\overline{\mathcal{X}}^{\text{occ}}$ shrinks significantly.

Robot Platform. We evaluate each method on the Unitree Go1 quadruped robot, where we task the robot to navigate to a goal location that is about 7m away from the initial location of the robot. The robot camera has a field of view of 70° and a visibility range of $[1, 5]$ m.

Metrics for experiments. We utilize the following metrics for

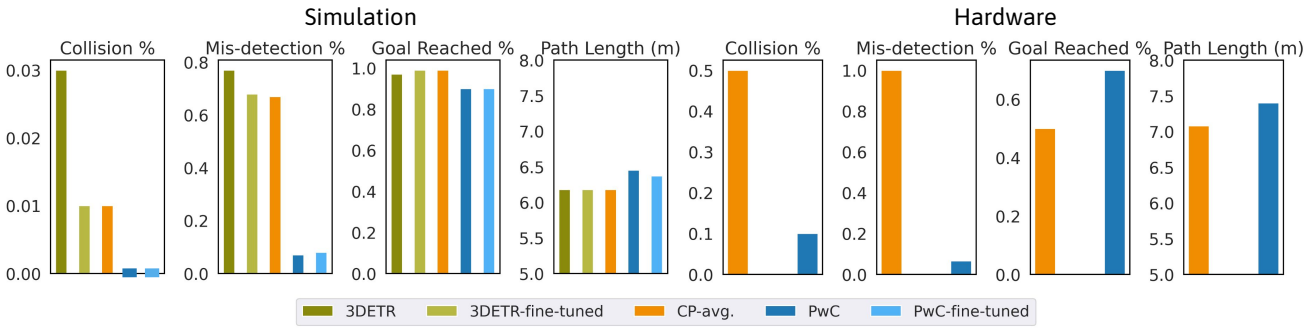


Fig. 7: [$\epsilon = 0.15$] **(Left)** Results for the simulated experiments across 100 new environments with 1 - 5 chairs (see Section VII). **(Right)** Results for the hardware trials across 30 different chair configurations with 4-8 chairs described in Section VIII. Here the path length is averaged only for successful trials for both PwC and CP-avg. due to the varying goal locations.

our simulation experiments: a trial is counted as a collision if the robot collides with an obstacle and we count a misdetection for a trial if the free space predicted by the planner has any intersection with the ground-truth occupancy of the obstacles. We say that the goal has been reached in a given trial if the robot is able to navigate to within 1 m around the goal in less than 140 s. We also record the average path length for trials in which the goal is reached.

A. *Bounding-Box Predictors*

We first consider perception systems that output bounding-boxes. For our implementation, we use the 3DETR end-to-end transformer model [52] as the pre-trained perception system. We compare our approach (*Perceive with Confidence* — PwC) to three baselines to illustrate its effectiveness in achieving a user-specified safety rate. First, we consider the most common approach of directly using the outputs of the perception system [52] in our planning pipeline. We call this baseline **3DETR**. Next, we consider the common practice of fine-tuning the outputs of the perception system using a small dataset of task-representative environments D_{tune} (cf. Section A-A). We call this perception system **3DETR-fine-tuned**. Lastly, we perform calibration using conformal prediction; however, instead of accounting for the closed-loop distribution shift, we bound the misdetection rate averaged across environments *and* states (similar to [20], which does not utilize conformal prediction, but quantifies expected errors in a perception system for a pre-defined distribution of states). We refer to this baseline as **CP- avg**. We consider two variations of our approach for comparison to the above baselines. First, we refine 3DETR outputs using our calibration procedure described in Section V. We call this approach **PwC**. Second, the 3DETR outputs are fine-tuned and calibrated using split conformal prediction as described in Appendix A-A; we call this approach **PwC-fine-tuned**.

Calibration and Planning. We implement the calibration procedure presented in Section V with the perception model ϕ instantiated as a bounding box predictor, mapping the observation o_t to a union of bounding boxes. Formally, we represent each bounding box j with the minimum and maximum coordinates in each dimension, $\left[d^{\min}, d^{\max}\right]_j$, where $d = (x, y)$ represents the spatial coordinates. The predicted occupied space is the union of 15 predicted most likely bounding boxes: $\overline{\mathcal{X}}^{\text{occ}} = \cup_{j=1}^{15}\left[d^{\min}, d^{\max}\right]_j$. The parameter q for this perception model is the inflation of the bounding boxes along each dimension. Therefore, in a given calibration environment E_i, from a given state s, and with a specific inflation parameter q_i, the predicted occupied region is defined as:

$$\overline{\mathcal{X}}^{\text{occ}}_{s,i}(q_i) := \cup_{j=1}^{15}\left[d^{\min}_{s,i} - q_i, d^{\max}_{s,i} + q_i\right]_j. \tag{11}$$

We collect a calibration dataset of 400 environments as specified. In the 8 m $\times$8 m room, we use a fixed set of 400 sampled configurations for the sampling-based motion planner and use the same set of samples for the calibration procedure. Similarly, we collect an additional fine-tuning dataset D_{tune} consisting of 100 environments. These environments include ones with occlusions of the goal and objects in the scene. With an allowable misdetection rate of $\epsilon = 0.15$, we obtain $\hat{q}_{0.85} = 0.75$ m for PwC, $\hat{q}_{0.85} = 0.65$ m for PwC-fine-tuned, and $\hat{q}_{0.85} = 0.05$ m for CP-avg. through calibration. The planner replans and obtains a new sensor observation to update the filter every 0.5 s or less (if the previous plan is already completed).

Misdetection Rate. We examine the misdetection rate, i.e., whether obstacles in the scene are classified as free space at any point during a trial, of our method, PwC, and the baseline CP-avg., which is also calibrated using conformal prediction but without accounting for the closed-loop distribution shift. We vary the allowable misdetection bound ϵ for each method and compute the rate of misdetections in 100 test environments. As seen in Figure 8, our method guarantees a misdetection rate lower than the threshold ϵ while CP-avg. violates this threshold for every ϵ considered.

Collision Rate. We compare PwC to the baselines in 100 new environments drawn from the same distribution as calibration environments. Figure 5 illustrates one such test environment. Figure 6 shows the evolution of the free space in this environment using PwC. Although the initial calibrated perception

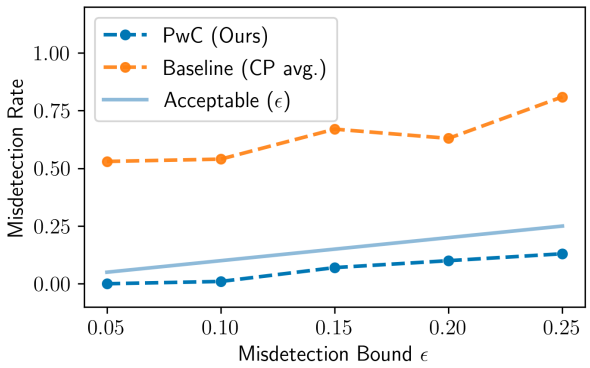


Fig. 8: As we relax the confidence threshold by increasing ϵ, the misdetection rate increases but remains bounded for PwC. The baseline method has a misdetection rate much higher than acceptable.

system outputs are inflated, the non-deterministic filter is able to expand the predicted free space in a few time steps and ensure that the robot can navigate without unnecessary conservatism, while guaranteeing safety. The results are summarized in Figure 7. We observe that our proposed approaches, PwC and PwC-fine-tuned, have no collisions in any environments. While the robot reaches the goal in a slightly lower percentage of environments compared to baselines, we emphasize that ours is the only approach that is able to ensure a low, statistically guaranteed misdetection rate across test environments.

Ablations. To further illustrate the effect of misdetections on safety, we consider a different distribution of environments wherein we randomly place a *single* chair in the straight line path between the initial position of the robot and the goal. For a safety threshold $1 - \epsilon = 0.85$, we compare PwC, CP-avg, and 3DETR. The results are provided in Figure 9 for 100 new test environments, wherein the goal is reached if the robot navigates to within 2 m of the goal. In these environments, the desired safety rate is not met by the baselines while our approach is still statistically guaranteed to be safe.

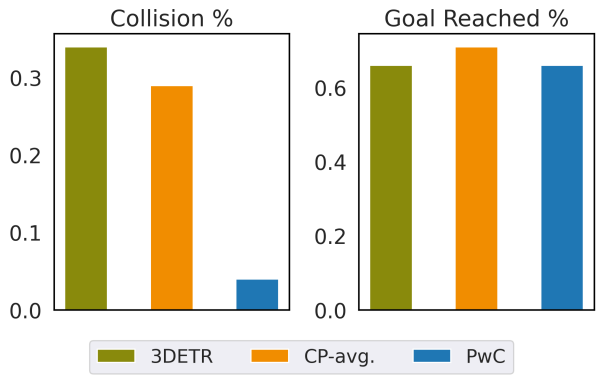


Fig. 9: A comparison between collision rates of different perception systems that use the same planner.

We provide additional simulation results in Appendix C that illustrate: 1) the effects of closed-loop distribution shifts on safety wherein PwC is robust to an increase in the level of closed-loop distribution shift, while the baseline CP-avg. is not, leading to higher collision rates; 2) the tradeoff in different partition sizes for fine-tuning using split-CP; 3) the effect of varying the allowable safety rate ϵ; 4) the effect of varying the number of sampled configurations; and 5) comparing PwC to a method of heuristically inflating bounding boxes.

B. Occupancy Predictors

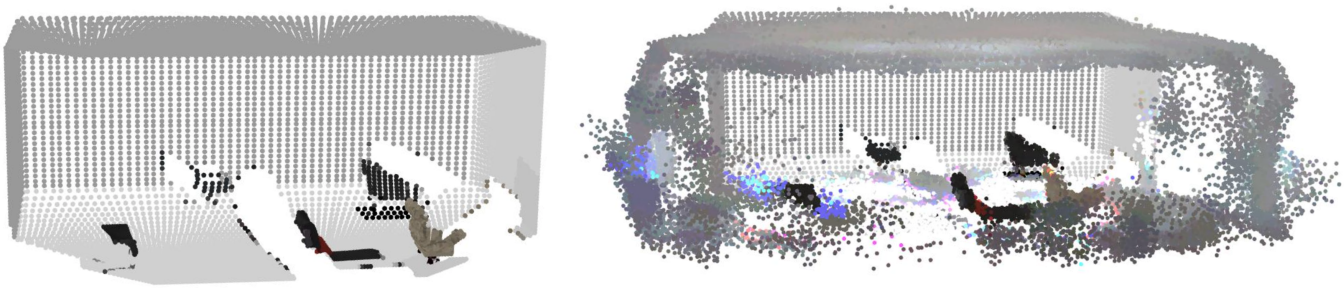
Now, we consider perception systems that predict occupancy maps. Specifically, we demonstrate the framework with the scene completion model NU-MCC [7]. NU-MCC takes an RGB-Depth image as input and predicts the value of the unsigned distance function (UDF) of each point in space. The original NU-MCC model includes a 3D reconstruction phase (Figure 10), where points with UDF less than a threshold, q, are kept and shifted to the surface of objects. Although this procedure results in better 3D visualizations, it breaks the correspondence between the threshold q and spatial coverage. In our method, we only use the predicted UDF from NU-MCC. In addition, we exponentially scale all the predicted UDFs to achieve a more uniformly covered range of UDF values.

Calibration and Planning. To define the non-conformity score, we find the smallest threshold q_i in each environment E_i such that the predicted occupancy covers the ground truth occupancy. Formally, the observation acquired at time step t is denoted o_t, represented by an RGB-depth image. The perception model, denoted ϕ, is the combination of (i) the adapted NU-MCC model which predicts UDF for each point in space, (ii) filtering out points in space with UDF $> q$, and (iii) projecting the resulting pointcloud onto a 2D occupancy grid as a bird's eye-view. This perception pipeline is summarized in Figure 11. Thus, ϕ maps o_t to the occupancy representation of the environment, $\mathcal{Z}$. In our implementation, $\mathcal{Z}$ is an $n \times n$ grid with boolean entries, with 1 representing occupied and 0 otherwise. We use $P \in \mathcal{Z}$ to denote one point on the grid in $\mathcal{Z}$. In a given calibration environment E_i, from a given state s, and a specific threshold q_i, the predicted occupancy is defined as:

$$\overline{\mathcal{X}}^{\text{occ}}_{s,i}(q_i) \coloneqq \{P \in \mathcal{Z} \mid \text{UDF}(P) \leq q_i\}. \tag{12}$$

As q_i increases, $\overline{\mathcal{X}}^{\text{occ}}_{s,i}(q_i)$ expands monotonically, satisfying the property stated in Section V. Therefore, proposition 1 follows, guaranteeing that the calibrated system $\hat{\phi}$ predicts occupancies that cover the ground truth with high probability.

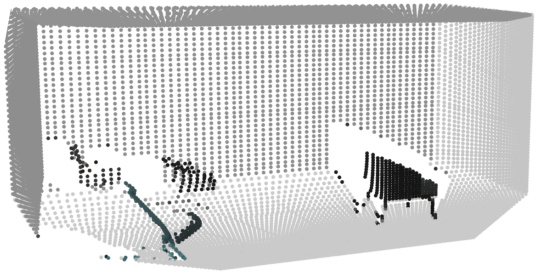
We generate a calibration dataset consisting of 300 environments from the distribution described in Section VII. In addition, the chairs are randomly rotated about the z-axis (Figure 13, middle and right). We use a fixed set of 812 sampled configurations, same as the set used by the sampling-based planner described in Section VI, modified for the occupancy map setting. With an allowable misdetection rate of $\epsilon = 0.15$, we obtain $\hat{q}_{0.85}$ for PwC-NU-MCC and NU-MCC-CP-avg through calibration, and use the (exponentiated) default for NU-MCC. For PwC calibrated on task distribution with rotated



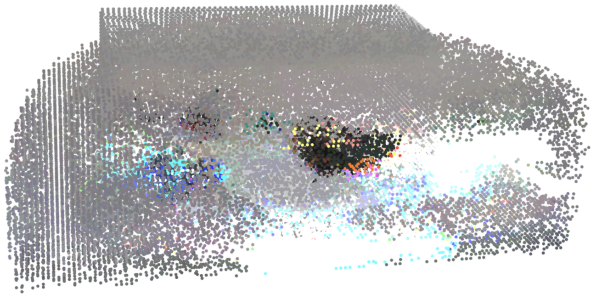
(a) Original RGB-Depth observation.

(b) Reconstructed Scene.

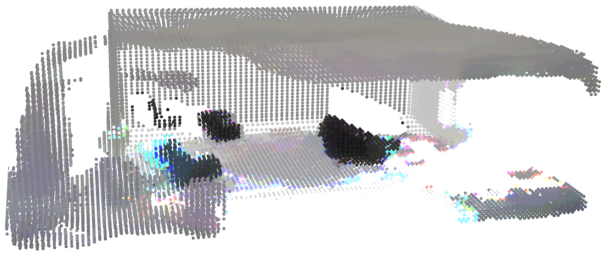
Fig. 10: NU-MCC scene completion with 3D reconstruction.



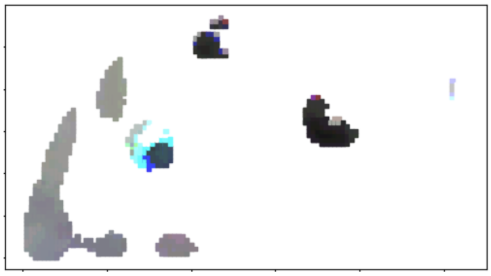
Step 1: Perceived RGB-Depth image.



Step 2: compute UDF for sampled points.



Step 3: filter out points with UDF > q.



Step 4: project the filtered pointcloud to 2D.

Fig. 11: Perception model ϕ based on NU-MCC, showing the construction of calibrated occupancy maps with provable guarantees on the correctness of the resulting map.

chairs and more states, we obtain $\hat{q}_{0.85} = 1.10$ m. Note that the $\hat{q}_{1-\epsilon}$ for PwC stands for the bounding box inflation rather than the UDF threshold.

Results. Figure 12 summarizes the simulation results. We compare our method based on occupancy prediction (PwC-NU-MCC) against the non-calibrated version (NU-MCC), as well as the method applying conformal prediction without accounting for closed-loop distribution shift (NU-MCC-CP-avg). We also compare against our method based on bounding box predictors, as described in Section V (PwC). We use the same metrics as described in Section VII.

For the results shown in Figure 12, we use a test dataset of 100 environments from the same distribution as the calibration dataset. The rotated chairs are no longer axis-aligned on the xy-plane, causing unnecessary conservatism when using the bounding box representation. Indeed, Figure 12 shows that PwC-NU-MCC has a much higher success rate (40% improvement) and shorter path length compared to PwC, while the safety rate is maintained. The four methods in the figure are arranged in the order of least to most conservative from left to right, showing a significant drop in collision rate and mis-detection rate with our methods, which also fall within the guarantee of less than 15%.

Figure 13 shows the trajectory of the robot in the same simulation environment, using three different perception modules. The left plot shows PwC as described in Section VII-A, while the middle and right plots show PwC-NU-MCC and NU-MCC respectively. For PwC, the bounding boxes are unnecessarily

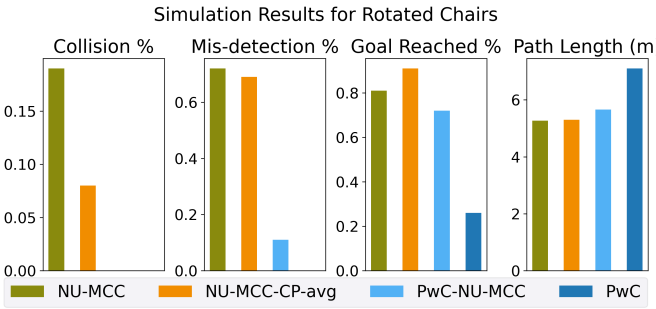


Fig. 12: Results for the simulated experiments with occupancy predictors, across 100 environments with rotated chairs.

inflated, causing the robot to get stuck in the overly conservative estimate of free space. PwC-NU-MCC preserves the safety guarantee while characterizing the true free space much more accurately, reaching the goal safely. NU-MCC overestimates the free space and collides with the obstacle.

VIII. Hardware Experiments

Now, we validate the end-to-end statistical safety assurance of our approach on a quadruped robot in the task of vision-based navigation with two sets of experiments. As in our simulation setup in Section VII, the robot is tasked with navigating to a goal location while avoiding different chairs placed in varying configurations across an 8 m $\times 8$ m room. We conduct two sets of experiments, which we term "nominal" and "fast". In the nominal experiments, the robot navigates with an average forward speed of 0.4 m/s, whereas in the fast experiments, we speed up the robot to 1.5 m/s. In both sets of experiments, we utilize the perception system calibrated in simulation with a guaranteed safety rate of $1 - \epsilon = 0.85$, as described in Section VII-A. Our calibration in simulation environments with realistic and diverse environments ensures that the performance of the perception system remains similar in its simulation and hardware implementations. We compare our PwC method against CP-avg. (defined in Section VII). We run the nominal experiments across 30 different physical environments (60 trials total) and run the fast experiments across 15 environments (30 trials total). One challenge is to ensure a minimal sim-to-real gap for perception. In order to address this, we utilize depth measurements as the robot's sensory input. This choice facilitates a small sim-to-real gap, as observed in prior work [53, 54].

A. Experiment Setup

We represent the robot's state as $s_t = [x, y, v_x, v_y]^T$ where x and y are its position in the environment and v_x and v_y are the respective velocities (See Figure 14 for the coordinate system). For each trial, the robot is initialized around position $[4, 0]$ m (with the origin set to the bottom left corner of the room) and has 60 seconds to reach the goal. For the nominal experiments, the robot replans every second in a receding horizon manner using the safe planner described in Section VI. The goals are varied every 10 environments and include positions $[2, 7]$ m, $[4, 7]$ m, and $[6, 7]$ m, with a radius of 1 m. For the fast experiments, the robot replans every 0.8 s. The goal is set at $[6, 7]$ m, and the radius is increased to 1.5 m.

Hardware. We use the Unitree Go1 quadruped robot with fully onboard sensing and computation. The robot is equipped with a ZED 2i RGB-D camera and a ZED Box computer attached to the base of the robot as shown in the top row of Figure 14. The Zed 2i provides the Go1 with point cloud observations with a $70°$ field of view and a visibility range of [1, 5]m. The Zed 2i also uses vision-inertial odometry to provide accurate positional state estimates in the environment. The Zed Box includes an 8-core ARM processor and a 16GB Orin NX GPU. This allows us to process the point cloud observations in order to produce bounding boxes using the pre-trained 3DETR model [52]. The bounding boxes are aggregated over time to update the estimated free, occupied, and unknown spaces as described in Section VI. The safe planner described in Section VI is used to output Cartesian velocity commands bounded at a speed of 0.8m/s; these commands are sent from the Zed Box over UDP to the Go1's processor. Our method is implemented in real-time on the Zed Box hardware with replanning every 0.5 seconds of which the non-deterministic filter takes 0.00025 seconds to run. The dynamics of the Go1 are estimated using MATLAB's System Identification Toolbox [55] and are provided in Appendix D-A.

Environments. For both sets of experiments, we test the robot in different environments, consisting of various chair configurations and geometries in an 8 m $\times 8$ m room. Configurations range from random, occluded goal, occluded chairs, clustered chairs, and narrow paths (approximately 1.8 m in width leaving 0.4 m of available free space for PwC to find). For the nominal experiments, each environment has between 4 and 8 chairs present. See Appendix D-B and D-C for the unseen chairs used in testing and the environment configurations respectively. We use a Vicon motion capture system to log the ground-truth placement and bounding boxes of the chairs for each environment. For the fast experiments, each environment has 6 chairs present. Since recording ground-truth data introduces latency that prevents the robot from reaching its target velocity of 1.5 m/s, we report only collision and goal-reach rates for this set of experiments.

B. Results

For PwC, we used the $\hat{q}_{0.85} = 0.73$ m threshold found in simulation to inflate the predicted bounding boxes returned from 3DETR in order to achieve 85% confidence that our robot will remain safe in new environments. We summarize key statistics of PwC compared to CP-avg. ($\hat{q}_{0.85} = 0.02$ m) across 30 different environments in Figure 7 (right). Importantly, our trials demonstrate that our confidence bound holds on hardware in real environments and without being too conservative. PwC was safe through 90% of the trials and also had comparable path length to the baseline. Meanwhile, the baseline struggled in the real environments by having misdetections in each trial and colliding with a chair in half of the trials. See

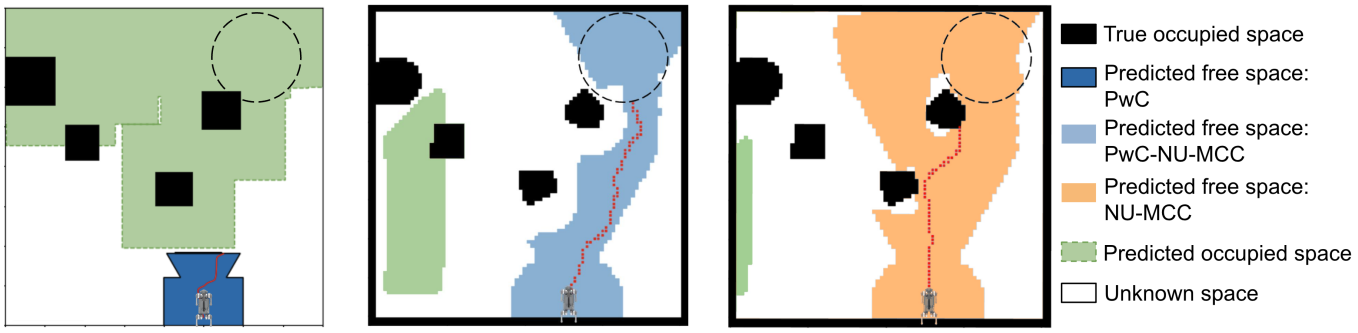


Fig. 13: Comparison of trajectories in the same environment using three different perception systems: **(Left)** PwC with 3DETR and bounding box representations, **(Middle)** PwC-NU-MCC, **(Right)** NU-MCC. The robot marks the start position, and the dashed circle represents the goal area. The blue region shows predicted free space, and the black regions represent the ground truth occupied space, either as bounding boxes or as occupancy grids. The robot's trajectory is marked in red. Among all methods, only PwC-NU-MCC enables the robot to safely reach the goal.

Figure 14 for trajectories and free space estimations through several environments with narrow spaces, occluded chairs, and occluded goals. The supplementary video contains full example trials.

PwC's low misdetection rate and higher success rate in these trials emphasize the efficacy of the bounding box inflation provided by CP paired with the non-deterministic filter. This principled pairing inflates the (potentially poor) bounding box detections to properly capture obstacles but quickly shrinks the occupied space with the filter such that the robot can still navigate effectively.

For faster navigation, we employ two complementary strategies: minimizing idle planning time via concurrent planning and execution, and increasing the robot's velocity in the pre-sampled configuration space. First, we implement a concurrent planning and execution framework using threading. The planning process is divided into two stages: an initial policy computed from the starting state and a continually updated future policy computed from the robot's predicted future state. At the start of each trial, the robot calculates an initial policy to reach the goal. During execution, a separate thread uses the robot's predicted future state, the state at the end of the current policy, to concurrently generate the next policy phase. This synchronization of execution and planning minimizes idle planning time. To support the increased speed, we re-sample the configuration space by keeping the positions unchanged but scaling up the speed, so that the calibration results would still hold. The re-sampled states, along with pre-computed reachability sets using the robot dynamics, are used to generate planned trajectories at high speed.

As a result, we increase the robot's average forward speed from about 0.4 m/s to 1.5 m/s and reduce the average task completion time in similar environments from about 28 seconds to 8 seconds. These performance gains are achieved without significant compromise in the safety rate of hardware validations of our previous method. We present the accelerated hardware results across 15 new environments (different from those shown in Appendix D-C) and compare the collision rate and success rate against the CP-avg. baseline, as shown in Table I.

TABLE I: Results for accelerated hardware experiments with PwC and CP-avg.

Method	Collision	Goal Reached
Accelerated PwC	20%	53.3%
CP-avg.	66.7%	33.3%

IX. Discussion and Conclusions

We present a modular framework, PwC, for rigorously quantifying the uncertainty of a pre-trained perception model in order to provide an end-to-end statistical safety assurance for perception-based navigation tasks. Notably, our statistical assurance holds for generalization to new environmental factors (e.g., new obstacle geometries and configurations) and allows for the distribution shift of states that may occur during closed-loop deployment of the perception system with the planner. Additionally, we address the conservatism introduced by the inflation of bounding boxes, by applying PwC to occupancy predictors and achieving much better performance without sacrificing the safety assurance. We validate the theoretical safety assurances provided by PwC with our simulation and hardware experiments, demonstrating significant empirical improvements in safety compared to baseline approaches that do not consider closed-loop distribution shift.

Limitations and Future Work. One limitation of our work is the assumption of static obstacles. As a future direction, we are interested in quantifying uncertainty in both the state of agents moving in the environment and predictions of their *semantic labels* (e.g., "pedestrian" vs. "bicyclist"), and utilizing game-theoretic planning techniques that account for the uncertainty in the agents' current state and future motion. Additionally, while our definition of safety is limited to collision avoidance, we are interested in extending it to richer settings such as navigation on limited surface area. Lastly, we are interested in uncertainty quantification for perception models that support tasks beyond

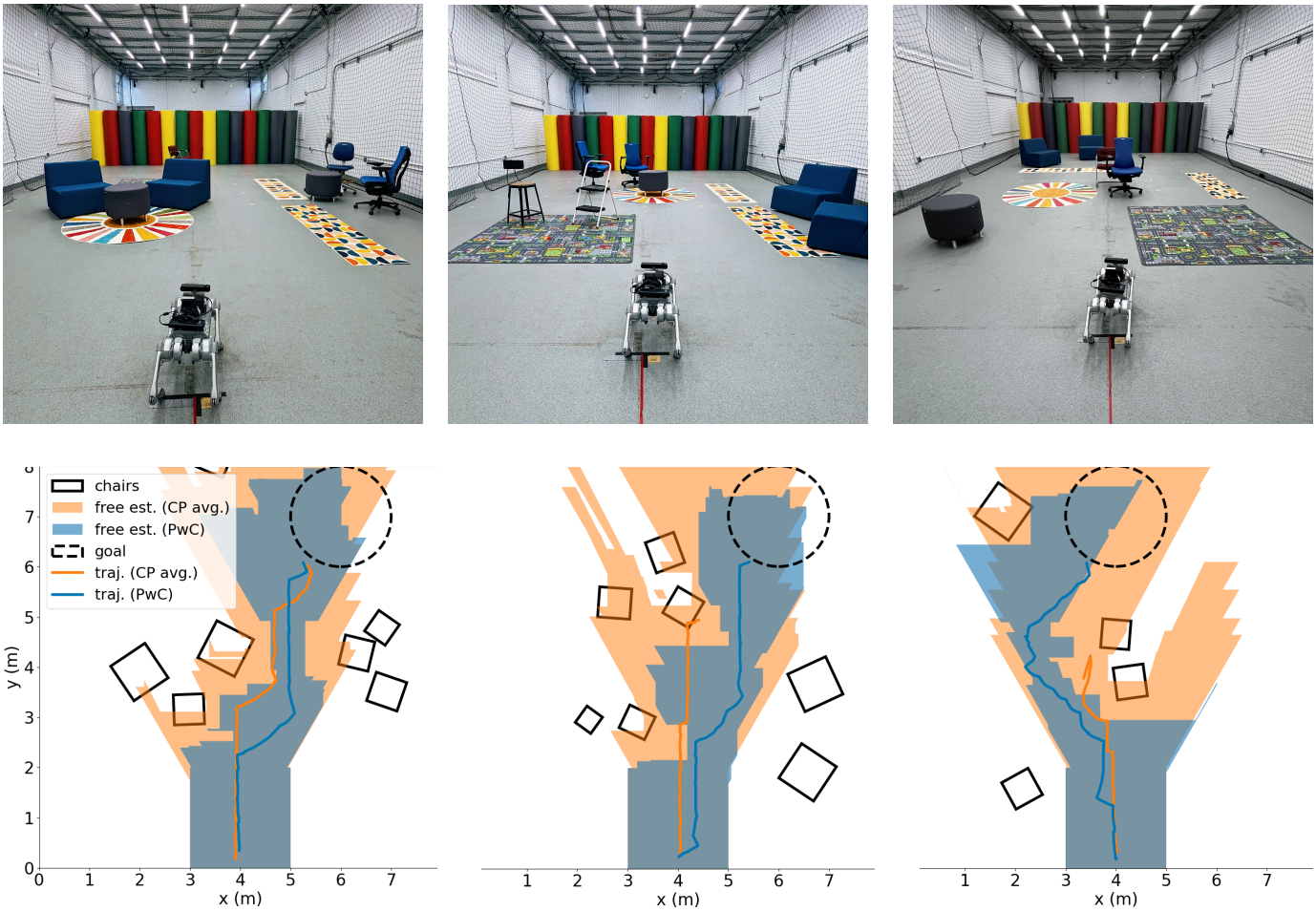


Fig. 14: Hardware trial results. **(Top)** The physical layouts of three example environments. **(Bottom)** The robot trajectories performed in these environments. Estimated free space is shaded, and robot trajectories are represented by solid lines: our method in blue and the baseline in orange. Our method (PwC) successfully navigates to the goal through challenging areas, whereas the baseline misdetects free spaces, leading to collisions in some cases.

point-to-point navigation, e.g., calibrating the outputs of multi-modal foundation models for language-instructed navigation where we ensure accurate detection of target objects as well as semantically unsafe regions [56]. We expect that rigorous uncertainty quantification is a necessary step towards fully leveraging the power of large foundation models [1] while safely integrating them into future robotic systems.

Acknowledgments

The authors were partially supported by the NSF CAREER Award [#2044149] and the Office of Naval Research [N00014-21-1-2803]. The authors would like to thank Alec Farid and David Snyder for helpful discussions on this work.

References

[1] Roya Firoozi, Johnathan Tucker, Stephen Tian, Anirudha Majumdar, Jiankai Sun, Weiyu Liu, Yuke Zhu, Shuran Song, Ashish Kapoor, Karol Hausman, Brian Ichter, Danny Driess, Jiajun Wu, Cewu Lu, and Mac Schwager. Foundation models in robotics: Applications, challenges, and the future. *arXiv preprint arXiv:2312.07843*, 2023.

[2] Eduardo Arnold, Omar Y Al-Jarrah, Mehrdad Dianati, Saber Fallah, David Oxtoby, and Alex Mouzakitis. A survey on 3d object detection methods for autonomous driving applications. *IEEE Transactions on Intelligent Transportation Systems*, 20(10):3782–3795, 2019.

[3] An Wang, Mobarakol Islam, Mengya Xu, Yang Zhang, and Hongliang Ren. SAM meets robotic surgery: An empirical study in robustness perspective. *arXiv preprint arXiv:2304.14674*, 2023.

[4] Niko Sünderhauf, Oliver Brock, Walter Scheirer, Raia Hadsell, Dieter Fox, Jürgen Leitner, Ben Upcroft, Pieter Abbeel, Wolfram Burgard, Michael Milford, et al. The limits and potentials of deep learning for robotics. *The International journal of robotics research*, 37(4-5):405–420, 2018.

[5] Vladimir Vovk, Alexander Gammerman, and Glenn Shafer. *Algorithmic learning in a random world*, volume 29. Springer, 2005.

[6] Anushri Dixit, Zhiting Mei, Meghan Booker, Mariko Storey-Matsutani, Allen Z Ren, and Anirudha Majumdar. Perceive with confidence: Statistical safety assurances for navigation with learning-based perception. In *8th Annual Conference on Robot Learning*, 2024.

[7] Stefan Lionar, Xiangyu Xu, Min Lin, and Gim Hee Lee. Nu-mcc: Multiview compressive coding with neighborhood decoder and repulsive udf. In *Advances in neural information processing systems*, 2023.

[8] Kai-Chieh Hsu, Haimin Hu, and Jaime Fernández Fisac. The safety filter: A unified view of safety-critical control in autonomous systems. *Annual Review of Control, Robotics, and Autonomous Systems (ARCRAS)*, 2023.

[9] Lucas Janson, Tommy Hu, and Marco Pavone. Safe motion planning in unknown environments: Optimality benchmarks and tractable policies. *arXiv preprint arXiv:1804.05804*, 2018.

[10] Zixu Zhang and Jaime Fernández Fisac. Safe occlusion-aware autonomous driving via game-theoretic active perception. *arXiv preprint arXiv:2105.08169*, 2021.

[11] Charles Packer, Nicholas Rhinehart, Rowan Thomas McAllister, Matthew A Wright, Xin Wang, Jeff He, Sergey Levine, and Joseph E Gonzalez. Is anyone there? Learning a planner contingent on perceptual uncertainty. In *Proceedings of the Conference on Robot Learning*, pages 1607–1617. PMLR, 2023.

[12] Markus Koschi and Matthias Althoff. Set-based prediction of traffic participants considering occlusions and traffic rules. *IEEE Transactions on Intelligent Vehicles*, 6(2): 249–265, 2020.

[13] Forrest Laine, Chiu-Yuan Chiu, and Claire Tomlin. Eyes-closed safety kernels: Safety for autonomous systems under loss of observability. *arXiv preprint arXiv:2005.07144*, 2020.

[14] Sarah Dean, Andrew Taylor, Ryan Cosner, Benjamin Recht, and Aaron Ames. Guaranteeing safety of learned perception modules via measurement-robust control barrier functions. In *Proceedings of the Learning for Dynamics and Control (L4DC) Conference*, pages 654–670. PMLR, 2021.

[15] Charles Dawson, Bethany Lowenkamp, Dylan Goff, and Chuchu Fan. Learning safe, generalizable perception-based hybrid control with certificates. *arXiv preprint arXiv:2201.00932*, 2022.

[16] Chiao Hsieh, Yangge Li, Dawei Sun, Keyur Joshi, Sasa Misailovic, and Sayan Mitra. Verifying controllers with vision-based perception using safe approximate abstractions. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 41(11):4205–4216, 2022.

[17] Sydney M Katz, Anthony L Corso, Christopher A Strong, and Mykel J Kochenderfer. Verification of image-based neural network controllers using generative models. *Journal of Aerospace Information Systems*, 19(9):574–584, 2022.

[18] Shromona Ghosh, Yash Vardhan Pant, Hadi Ravanbakhsh, and Sanjit A Seshia. Counterexample-guided synthesis of perception models and control. In *Proceedings of the IEEE American Control Conference*, pages 3447–3454. IEEE, 2021.

[19] Sarah Dean and Benjamin Recht. Certainty equivalent perception-based control. In *Proceedings of the Learning for Dynamics and Control (L4DC) Conference*, pages 399–411. PMLR, 2021.

[20] Dawei Sun, Benjamin C Yang, and Sayan Mitra. Learning-based perception contracts and applications. *arXiv preprint arXiv:2309.13515*, 2023.

[21] YuXuan Liu, Nikhil Mishra, Maximilian Sieb, Yide Shentu, Pieter Abbeel, and Xi Chen. Autoregressive uncertainty modeling for 3d bounding box prediction. In *European Conference on Computer Vision*, pages 673–694. Springer, 2022.

[22] Anirudha Majumdar, Alec Farid, and Anoopkumar Sonar. PAC-Bayes control: Learning policies that provably generalize to novel environments. *The International Journal of Robotics Research*, 40(2-3):574–593, 2021.

[23] Alec Farid, Sushant Veer, and Anirudha Majumdar. Task-driven out-of-distribution detection with statistical guarantees for robot learning. In *Conference on Robot Learning*, pages 970–980. PMLR, 2022.

[24] Alec Farid, David Snyder, Allen Ren, and Anirudha Majumdar. Failure prediction with statistical guarantees for vision-based robot control. In *Proceedings of Robotics: Science and Systems*, 2022.

[25] Rohan Sinha, Edward Schmerling, and Marco Pavone. Closing the loop on runtime monitors with fallback-safe mpc. In *2023 62nd IEEE Conference on Decision and Control (CDC)*, pages 6533–6540. IEEE, 2023.

[26] Sarah Dean, Nikolai Matni, Benjamin Recht, and Vickie Ye. Robust guarantees for perception-based control. In *Proceedings of the Learning for Dynamics and Control (L4DC) Conference*, pages 350–360. PMLR, 2020.

[27] Glen Chou, Necmiye Ozay, and Dmitry Berenson. Safe output feedback motion planning from images via learned perception modules and contraction theory. In *Algorithmic Foundations of Robotics XV*, pages 349–367. Springer International Publishing, 2023.

[28] Vladimir Vovk. Conditional validity of inductive conformal predictors. In *Asian Conference on Machine Learning*, pages 475–490. PMLR, 2012.

[29] Anastasios N. Angelopoulos and Stephen Bates. A gentle introduction to conformal prediction and distribution-free uncertainty quantification. *arXiv preprint arXiv:2107.07511*, 2022.

[30] Allen Z Ren, Anushri Dixit, Alexandra Bodrova, Sumeet Singh, Stephen Tu, Noah Brown, Peng Xu, Leila Takayama, Fei Xia, Jake Varley, Zhenjia Xu, Dorsa Sadigh, Andy Zeng, and Anirudha Majumdar. Robots that ask for help: Uncertainty alignment for large language model

planners. *arXiv preprint arXiv:2307.01928*, 2023.

[31] Lars Lindemann, Matthew Cleaveland, Gihyun Shim, and George J Pappas. Safe planning in dynamic environments using conformal prediction. *IEEE Robotics and Automation Letters*, 2023.

[32] Anushri Dixit, Lars Lindemann, Skylar X Wei, Matthew Cleaveland, George J Pappas, and Joel W Burdick. Adaptive conformal prediction for motion planning among dynamic agents. In *Proceedings of the Learning for Dynamics and Control (L4DC) Conference*, pages 300–314. PMLR, 2023.

[33] Rachel Luo, Shengjia Zhao, Jonathan Kuck, Boris Ivanovic, Silvio Savarese, Edward Schmerling, and Marco Pavone. Sample-efficient safety assurances using conformal prediction. In *International Workshop on the Algorithmic Foundations of Robotics*, pages 149–169. Springer, 2022.

[34] Shuo Yang, George J Pappas, Rahul Mangharam, and Lars Lindemann. Safe perception-based control under stochastic sensor uncertainty using conformal prediction. *arXiv preprint arXiv:2304.00194*, 2023.

[35] Sangdon Park, Osbert Bastani, Nikolai Matni, and Insup Lee. PAC confidence sets for deep neural networks via calibrated prediction. *arXiv preprint arXiv:2001.00106*, 2019.

[36] Shay Aharon, Louis-Dupont, Ofri Masad, Kate Yurkova, Lotem Fridman, Lkdci, Eugene Khvedchenya, Ran Rubin, Natan Bagrov, Borys Tymchenko, Tomer Keren, Alexander Zhilko, and Eran-Deci. Super-gradients, 2021. URL https://zenodo.org/record/7789328.

[37] Chao-Yuan Wu, Justin Johnson, Jitendra Malik, Christoph Feichtenhofer, and Georgia Gkioxari. Multiview compressive coding for 3D reconstruction. *arXiv preprint arXiv:2301.08247*, 2023.

[38] Evangelos Chatzipantazis, Stefanos Pertigkiozoglou, Edgar Dobriban, and Kostas Daniilidis. SE(3)-equivariant attention networks for shape reconstruction in function space. *arXiv preprint arXiv:2204.02394*, 2022.

[39] Anastasios N. Angelopoulos, Stephen Bates, Adam Fisch, Lihua Lei, and Tal Schuster. Conformal Risk Control, April 2023. URL http://arxiv.org/abs/2208.02814. arXiv:2208.02814 [cs, math, stat].

[40] Mikaela Angelina Uy, Quang-Hieu Pham, Binh-Son Hua, Duc Thanh Nguyen, and Sai-Kit Yeung. Revisiting point cloud classification: A new benchmark dataset and classification model on real-world data. In *International Conference on Computer Vision (ICCV)*, 2019.

[41] Berk Calli, James Bruce, Aaron Walsman, Kurt Konolige, Siddhartha Srinivasa, Pieter Abbeel, and Aaron Dollar. Yale-cmu-berkeley dataset for robotic manipulation research. *The International Journal of Robotics Research*, 36:027836491770071, 04 2017. doi: 10.1177/0278364917700714.

[42] Huan Fu, Bowen Cai, Lin Gao, Ling-Xiao Zhang, Jiaming Wang, Cao Li, Qixun Zeng, Chengyue Sun, Rongfei Jia, Binqiang Zhao, et al. 3D-Front: 3D furnished rooms with layouts and semantics. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 10933–10942, 2021.

[43] Steven M LaValle. Planning algorithms. *Cambridge University Press*, 2:3671–3678, 2006.

[44] Tom Schouwenaars, Éric Féron, and Jonathan How. Safe receding horizon path planning for autonomous vehicles. In *Proceedings of the Annual Allerton Conference on Communication Control and Computing*, volume 40, pages 295–304. The University; 1998, 2002.

[45] Sara Bouraine, Thierry Fraichard, and Ouahiba Azouaoui. Real-time safe path planning for robot navigation in unknown dynamic environments. In *Conference on Computing Systems and Applications*, 2016.

[46] Èric Pairet, Juan David Hernández, Marc Carreras, Yvan Petillot, and Morteza Lahijanian. Online mapping and motion planning under uncertainty for safe navigation in unknown environments. *IEEE Transactions on Automation Science and Engineering*, 19(4):3356–3378, 2021.

[47] Thierry Fraichard and Hajime Asama. Inevitable collision states. A step towards safer robots? In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 388–393, 2003.

[48] Lucas Janson, Edward Schmerling, Ashley Clark, and Marco Pavone. Fast marching tree: A fast marching sampling-based method for optimal motion planning in many dimensions. *The International Journal of Robotics Research*, 34(7):883–921, 2015. ISSN 0278-3649.

[49] Edward Schmerling, Lucas Janson, and Marco Pavone. Optimal sampling-based motion planning under differential constraints: The driftless case. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 2368–2375. IEEE, 2015.

[50] Edward Schmerling, Lucas Janson, and Marco Pavone. Optimal sampling-based motion planning under differential constraints: The drift case with linear affine dynamics. In *Proceedings of the IEEE Conference on Decision and Control*, pages 2574–2581. IEEE, 2015.

[51] Erwin Coumans and Yunfei Bai. Pybullet, a Python module for physics simulation for games, robotics and machine learning. http://pybullet.org, 2016–2022.

[52] Ishan Misra, Rohit Girdhar, and Armand Joulin. An end-to-end transformer model for 3D object detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2906–2917, 2021.

[53] Antonio Loquercio, Elia Kaufmann, René Ranftl, Matthias Müller, Vladlen Koltun, and Davide Scaramuzza. Learning high-speed flight in the wild. *Science Robotics*, 6(59): eabg5810, 2021.

[54] Theophile Gervet, Soumith Chintala, Dhruv Batra, Jitendra Malik, and Devendra Singh Chaplot. Navigating to objects in the real world. *Science Robotics*, 8(79): eadf6991, 2023.

[55] The MathWorks Inc. MATLAB version: 9.13.0 (r2022b), 2022. URL https://www.mathworks.com.

[56] Leonardo Santos, Zirui Li, Lasse Peters, Somil Bansal,

and Andrea Bajcsy. Updating robot safety representations online from natural language feedback. *arXiv preprint arXiv:2409.14580*, 2024.

[57] Hamid Rezatofighi, Nathan Tsoi, JunYoung Gwak, Amir Sadeghian, Ian Reid, and Silvio Savarese. Generalized intersection over union: A metric and a loss for bounding box regression. In *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 658–666, 2019.

[58] Kirill Neklyudov, Dmitry Molchanov, Arsenii Ashukha, and Dmitry Vetrov. Variance networks: When expectation does not meet your expectations. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=B1GAUs0cKQ.

[59] Sergey Levine, Aviral Kumar, George Tucker, and Justin Fu. Offline reinforcement learning: Tutorial, review, and perspectives on open problems. *arXiv preprint arXiv:2005.01643*, 2020.

[60] Stéphane Ross, Geoffrey Gordon, and Drew Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning. In *Proceedings of the International Conference on Artificial Intelligence and Statistics*, volume 15, pages 627–635. PMLR, 2011.

APPENDIX A
PERCEPTION AND PLANNING EXTENSIONS

In this section, we outline a few extensions to the basic technical approach described in Sections V and VI: (i) fine-tuning a pre-trained perception model and (ii) incorporating sensor and dynamics uncertainty.

A. Fine-Tuning a Pre-Trained Perception Model

In Section V, we assumed access to a pre-trained perception model ϕ that outputs occupancy predictions of the environment. The conformal prediction-based uncertainty quantification procedure then uses the calibration dataset $D = \{E_1, \ldots, E_N\}$ of environments to produce a calibrated perception system $\tilde{\phi}$, which lightly processes the outputs of ϕ scaling with a parameter q. In practice, it may also be useful to *fine-tune* ϕ for our target deployment environments before performing uncertainty quantification.

This can be achieved using *split conformal prediction* [29], where one splits the overall dataset D into $D = D_{\text{tune}} \cup D_{\text{cal}}$. If the perception model takes the form of a neural network ϕ_w parameterized by weights w, we can use D_{tune} to fine-tune w (or the weights of a residual network). We can then utilize D_{cal} in order to perform the CP-based calibration as described in Section V. We demonstrate the fine-tuning process for the case of bounding box predictions in Section VII, and show that this additional fine-tuning step before calibration can reduce the conservatism of outputs and improve end-to-end success rates.

The typical choice of loss function for training a bounding box predictor is the *generalized intersection-over-union (gIoU) loss* [57], a differentiable version of the IoU loss: given a ground-truth bounding box A and a predicted box B, one computes $L(A, B) := |A \cap B|/|A \cup B|$. However, while this loss is popular in computer vision, it is not suitable for robot navigation. In particular, the IoU loss is *symmetric*: it does not distinguish between the ground-truth and predicted bounding box and thus does not encourage the predicted box to *contain* the ground-truth box. We propose a modification to the gIoU loss in Appendix B, which encourages that the predicted bounding box encloses the ground-truth box while also ensuring that the predicted box is not too large. Similar to the gIoU loss, this loss is (almost-everywhere) differentiable and scale invariant. We utilize this loss for fine-tuning in our experiments (Section VII). However, one could use any other method for finetuning not limited to training a simple neural network with the gIoU loss [58].

B. Sensor Errors and Dynamics Uncertainty

In Section III, we modeled the robot's sensor as a deterministic mapping $\sigma : \mathcal{S} \times \mathcal{E} \to \mathcal{O}$, which provides observations from a particular state in a given environment. This formulation allows us to also incorporate sensor errors. Specifically, any errors or randomness in the sensor can be formally included as part of the environment $E \in \mathcal{E}$. Thus, in addition to sampling environmental variables such as obstacle locations, geometries, etc., each environment E also samples random variables that prescribe sensor errors from each state $s \in \mathcal{S}$ in the environment. This way of modeling sensor errors allows: (i) σ to be deterministic (since all sources of randomness are included in E), (ii) the sensor errors to be dependent on the relative pose of the robot relative to obstacles (e.g., modeling the fact that depth estimates are often further from ground-truth depth values as distance increases), and (iii) the modeling of correlations in sensor errors from different locations (e.g., capturing the fact that sensor errors from nearby robot locations can be highly correlated). Modeling time-varying sensor errors (i.e., different sensor errors from the robot state at different times) is not as immediate, but could potentially be incorporated by augmenting the state space $\mathcal{S}$ to include the time-step.

In addition to errors in sensing, one can also account for uncertainty in the dynamics of the robot by using a robust planner (see [8] for an overview). In the experiments described in Section VIII, we incorporate uncertainty by generating plans that prevent the robot from entering the inevitable collision set (cf. Section VI) even with bounded uncertainty in the dynamics.

APPENDIX B
LOSS FUNCTION FOR FINE-TUNING

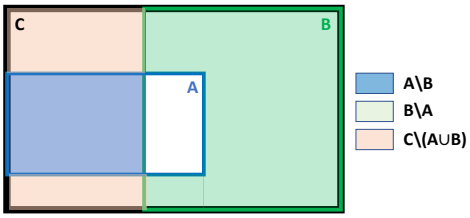


Fig. 15: Visualization of different terms in the loss function for a single object setting.

We use an almost-everywhere differentiable loss function for training. The loss function seeks to ensure that the predicted shape (e.g., bounding box) encloses the ground truth shape while also ensuring that the predicted shape is not too large.

Let's consider the simplest setting wherein we have one object in the scene and we are making a single prediction. In this case, A denotes the (convex) ground-truth shape and B denotes the (convex) predicted shape. Let C denote the convex hull of A and B. Our loss function is a weighted combination of three terms,

$$\begin{aligned} L \coloneqq\ & w_1 l_1 + w_2 l_2 + w_3 l_3 \\ =\ & w_1 \frac{|A \backslash B|}{|A|} + w_2 \frac{|B \backslash A|}{|B|} + w_3 \frac{|C \backslash (A \cup B)|}{|C|}. \end{aligned}$$

The first term is the most important; it tries to ensure that B encloses A. The second term tries to make sure that B is not much larger than it needs to be, see Figure 15. The first and second terms are sufficient if A and B are overlapping. However, if they do not overlap, there is no gradient information provided by the first two terms. Following [57], we introduce a third loss term in order to provide gradient information when the shapes do not intersect. The loss terms l_1, l_2, l_3 are each bounded within $[0, 1]$. Hence, if we choose w_1, w_2, w_3 such that $\sum_i w_i = 1$, then the overall loss is also bounded within $[0, 1]$.

Now let's consider the setting wherein, A denotes the union of multiple ground-truth bounding boxes (say we have m objects in the scene) and B is the union of all the predicted bounding boxes (we predict n boxes). We consider all the individual bounding box predictions $B_i, \forall i \in \{1, \dots n\}$ and associate the closest *visible* ground-truth bounding box A_i to each prediction. Now we can define C_i as the convex hull of A_i and B_i and the resulting loss function, L_i,

$$L_i := w_1 \frac{|A_i \backslash B_i|}{|A_i|} + w_2 \frac{|B_i \backslash A_i|}{|B_i|} + w_3 \frac{|C_i \backslash (A_i \cup B_i)|}{|C_i|}.$$

Hence, the overall loss is,

$$L = \frac{1}{n} \sum_{i=1}^{n} L_i.$$

Please refer to [57, Appendix 4.3] for instructions on how to compute the loss analytically for axis-aligned bounding boxes.

Appendix C
Ablations

We provide additional simulation results to illustrate the effects of: (1) closed-loop distribution shifts on safety wherein PwC is robust to an increase in the level of closed-loop distribution shift while the baseline, CP-avg., is not which leads to higher collision rates for CP-avg., (2) the tradeoff in different partition sizes for fine-tuning using split-CP, (3) the effect of varying ϵ on the safety rate, (4) impact of using different number of sampled configurations for calibration and online planning, and (5) comparison against additional uncertainty-aware perception systems that use a heuristic notion of uncertainty

Effects of closed-loop distribution shift on misdetections. In addition to the challenge of generalization, we highlight another challenge that any uncertainty quantification method for perception must tackle. Suppose we fix a policy π^{ϕ} (that uses perception system ϕ) and collect a dataset of observations in different calibration environments from the states that result from applying π^{ϕ}. We can use ground-truth bounding boxes in these environments to produce a calibrated perception system $\tilde{\phi}$ with a statistical assurance on correctness for the distribution of observations induced by π^{ϕ}. However, if we now apply the policy $\pi^{\tilde{\phi}}$ using the *calibrated* perception system $\tilde{\phi}$, the resulting distribution of states will be *different* from the distribution that forms the calibration dataset, thus invalidating the statistical assurance. We refer to this challenge as *closed-loop distribution shift*, which is similar to challenges that arise in offline reinforcement learning [59] and imitation learning [60].

To illustrate the effect of closed-loop distribution shifts on misdetections, we used exactly the same setup described above to obtain the simulation results in Figure 7. We changed the planner cost to have a different weighting on the cost-to-go. For one setting, we chose a weight $w = 1$ on the cost-to-go, which is the same as the weighting on the cost-to-come. In another setting, we chose a weight $w = 10$ on the cost-to-go, and hence a $10\times$ more emphasis on the cost-to-go compared to the cost-to-come. Table II shows the KL-divergence between the states visited by the planner and the sampling distribution of states for calibration as a measure of the closed-loop distribution shift. Increasing closed-loop shifts lead to higher misdetections. One can see that a simple change in the planner parameters can lead to potentially large changes in the safety rates for CP-avg. The closed-loop shift we may see in practice is unknown apriori. Hence, it is difficult to make any statements on the planner safety in closed-loop despite using CP for calibration of the perception system. PwC, on the other hand, is robust to the closed-loop shifts and can still satisfy the misdetection and safety assurance regardless of the planner parameters used.

TABLE II: A comparison of the effect of changing the planner parameters on CP-avg. and PwC.

Method	Collision	Mis-detection	KL-divergence
CP-avg. ($w = 1$)	14%	54%	2.09
CP-avg. ($w = 10$)	2%	64%	2.72
PwC ($w = 1$)	0%	0%	1.48
PwC ($w = 10$)	0%	2%	2.04

Effect of finetuning dataset size. Upon collecting a calibration dataset of about 400 environments, as described in the experiment setup in Section VII, we may choose to use a smaller subset of the calibration dataset to further finetune the pre-trained perception model to perform better in the types of environments we are interested in deploying the robot in. We consider the effect of different dataset split sizes for finetuning and then calibration. Using a larger set of environments for finetuning $|D_{\text{tune}}|$ may result in a better tuned model, but will leave fewer environments for calibration, $|D_{\text{cal}}|$, resulting in a more conservative $\hat{\epsilon}$ and $\hat{q}_{1-\epsilon}$ that satisfies the dataset-conditional guarantee (3), and vice versa. This trade-off is seen in Table III, where we observe the best performance when we have an equal split between finetuning and calibration.

Effects of varying ϵ on safety rate. We compare our method, PwC, to the baseline CP-avg. We vary the allowable safety rate ϵ for each method, and compute the rate of safety in 100 test environments. As seen in Table IV and Figure 8, our method not only guarantees that the rate of misdetections to be bounded, but also the safety rate. The safety rate of PwC is also consistently better than that of CP-avg.

Effect of varying the number of sampled configurations. For our experiments, we used a fixed set of 2000 sampled configurations. However, depending on the planner configuration requirements and desired speed of computation, the user may decide to have a different number of configuration samples for calibration and planning. We study the change in the CP inflation, $\hat{q}_{0.85}$, the resulting collision, misdetection, and task completion (reaching goal) rates. As we can see in Table V, in our case, we have far fewer misdetections with fewer samples, but we also observe a decrease in number of times the robot reaches the goal. We suspect that with fewer samples of configurations (consisting of x, y, v_x, v_y), it is harder for the sampling-based motion planner to find feasible

TABLE III: A comparison of the effect of various partition sizes for fine-tuning and calibration for PwC.

Split size ($\lvert D_{\textbf{tune}}\rvert + \lvert D_{\textbf{cal}}\rvert$)	$\hat{q}_{0.85}$ **(in m)**	**Collision**	**Misdetection**	**Goal Reached**
$100 + 300$	0.68	0%	1%	89%
$200 + 200$	**0.64**	0%	**1%**	**94%**
$300 + 100$	0.93	0%	2%	76%

TABLE IV: A comparison of the safety rates of CP-avg. and PwC when we vary the confidence threshold ϵ.

ϵ	**CP-avg.**	**PwC**
0.20	95%	**100%**
0.10	98%	**99%**
0.15	99%	**100%**
0.10	98%	**100%**
0.05	98%	**100%**

TABLE V: A comparison of the CP inflation $\hat{q}_{0.85}$ when we vary the number of sampled configurations.

# samples	$\hat{\mathbf{q}}_{\mathbf{0.85}}$	**Collision**	**Mis-detection**	**Goal Reached**
1050	0.7086	0%	1%	47%
1500	**0.6910**	0%	**0%**	57%
2000	0.75	0%	7%	**90%**

paths. On the other hand, we also observe a less conservative $\hat{q}_{0.85}$ when we use fewer samples; this is presumably also a result of using fewer samples to compute the non-conformity score that comprises of the worst-case perception error across all configurations.

Comparison to heuristic inflation. We compare PwC to the baseline method of inflating the bounding box predictions based on some heuristic confidence level, i.e., we scale the bounding box with 1 - confidence (so we scale the boxes where we are less confident by a larger amount). As shown in Table VI, While this baseline demonstrates a higher completion rate, both the collision rate and the misdetection rate increase significantly, leading to unsafe situations. Further, while our method provides a statistical safety guarantee, the baseline method does not admit any formal assurance.

TABLE VI: A comparison of the effects of using heuristic inflation versus PwC.

Method	**Collision**	**Misdetection**	**Goal Reached**
PwC	**0%**	**7%**	90%
Heuristic	3%	67%	**97%**

APPENDIX D
EXPERIMENT SETUP

A. System Identification

To perform system identification of the Unitree Go1 quadruped robot, we collected trajectories using a Vicon motion capture system. We then used MATLAB's system identification toolbox [55]. Specifically, we provided an initial linear ODE grey box model guess and then used prediction error minimization (PEM) for refinement. The resulting system is shown in (13) where x and y describe the positional state of the robot in the environment, v_x and v_y describe the respective velocities, and u_x and u_y describe the respective commanded velocities.

B. Chair Test Dataset

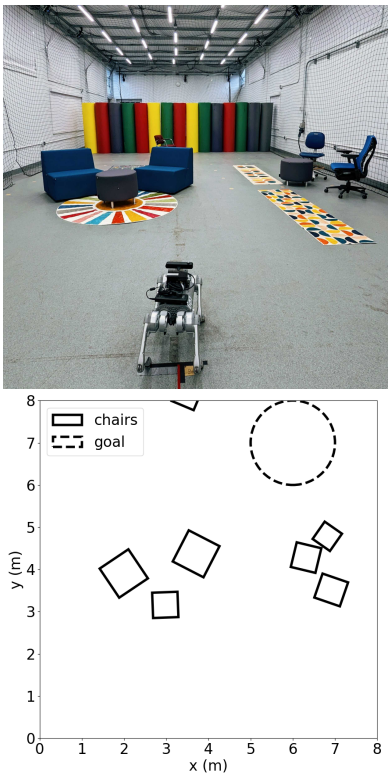
Our test dataset of chairs for the first set of experiments conducted in Section VIII included 8 chairs with diverse sizes and geometries unseen in training and calibration for the perception system. Test chairs are shown below in Figure 16.

C. Environments

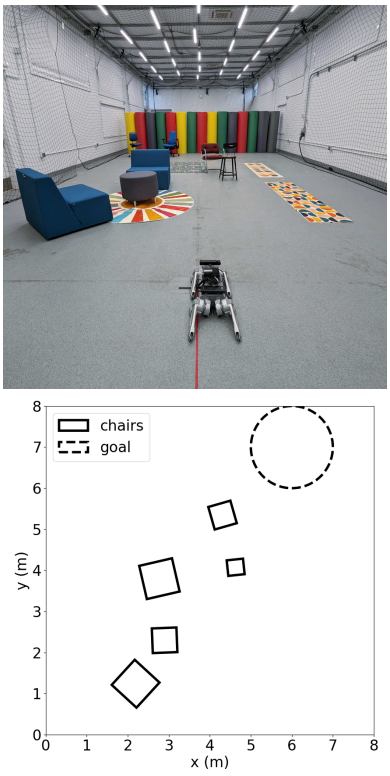
As described in Section VIII, in the first set of experiments, the robot was tested in 30 unique environments with varying furniture configurations and goals. The following 30 figures show an image of each configuration, accompanied by a bird's-eye map of the obstacle and goal locations.

Fig. 16: New, unseen test chairs used in original hardware experiments. In the fast experiments, only chairs 1, 2, and 5 (left to right) were used.

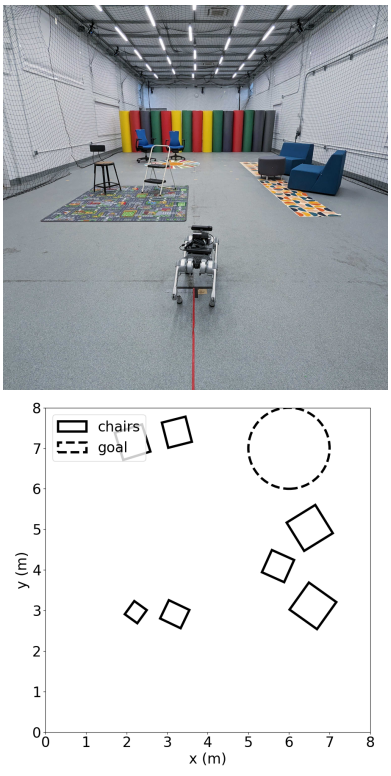
$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{v}_x \\ \dot{v}_y \end{bmatrix} = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & -2.5170 & 0.1353 \\ 0 & 0 & -0.5197 & -3.9680 \end{bmatrix} \begin{bmatrix} x \\ y \\ v_x \\ v_y \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 2.3350 & 0 \\ 0 & 4.6510 \end{bmatrix} \begin{bmatrix} u_x \\ u_y \end{bmatrix} \tag{13}$$



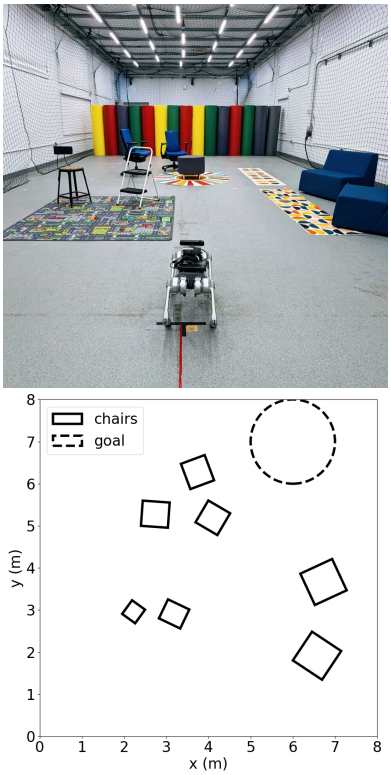
(1) Environment 1



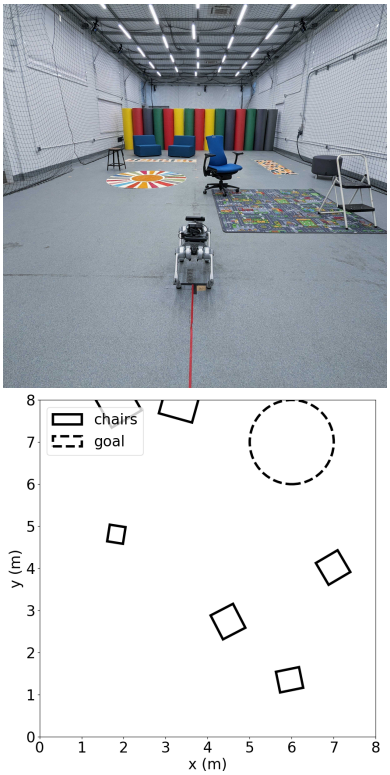
(2) Environment 2



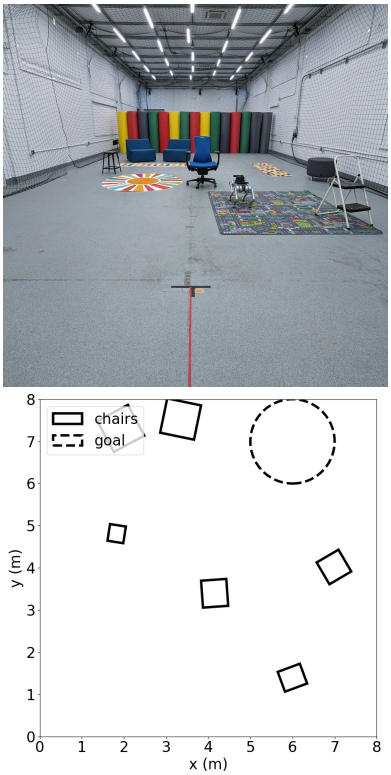
(3) Environment 3



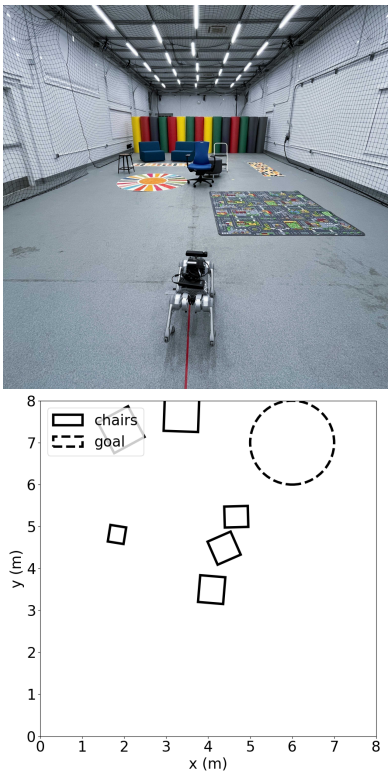
(4) Environment 4



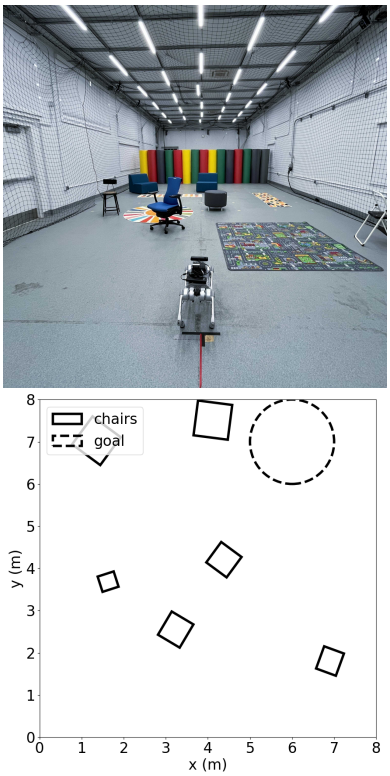
(5) Environment 5



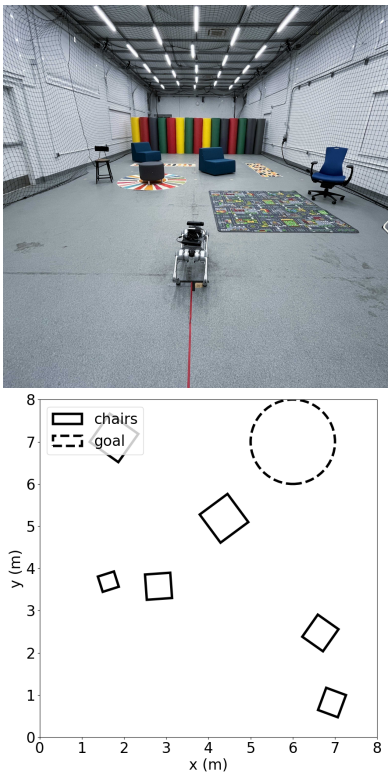
(6) Environment 6



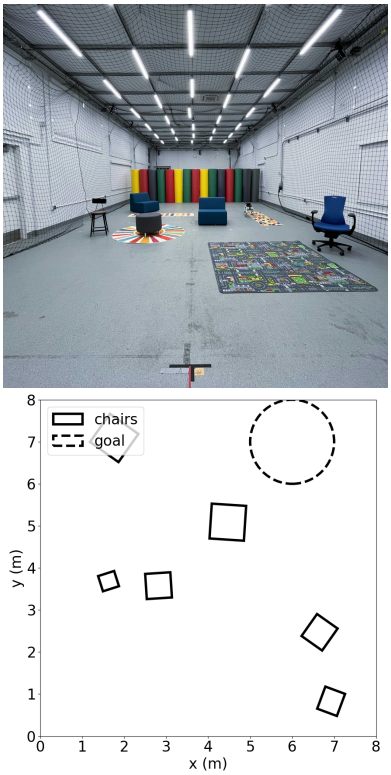
(7) Environment 7



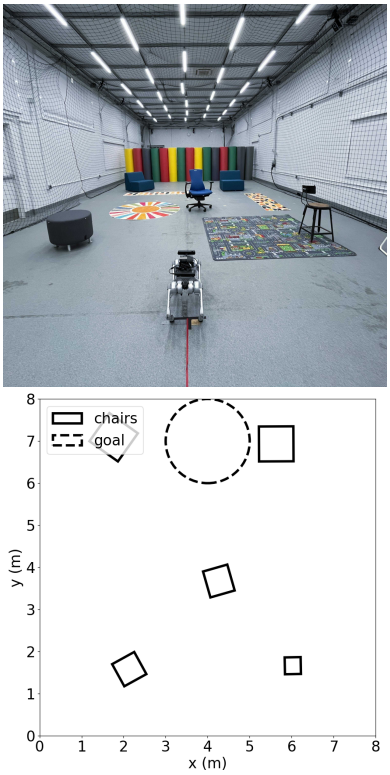
(8) Environment 8



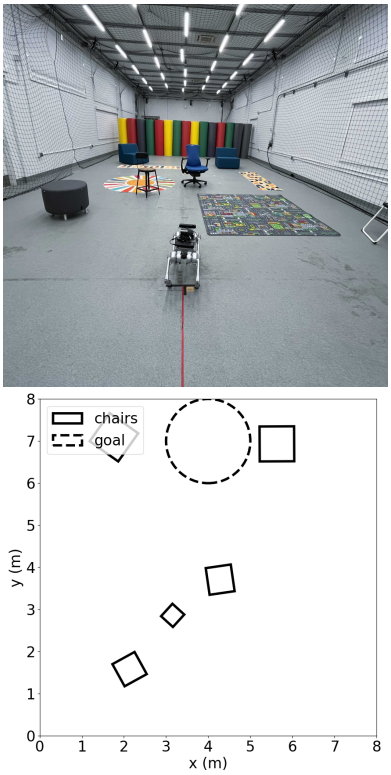
(9) Environment 9



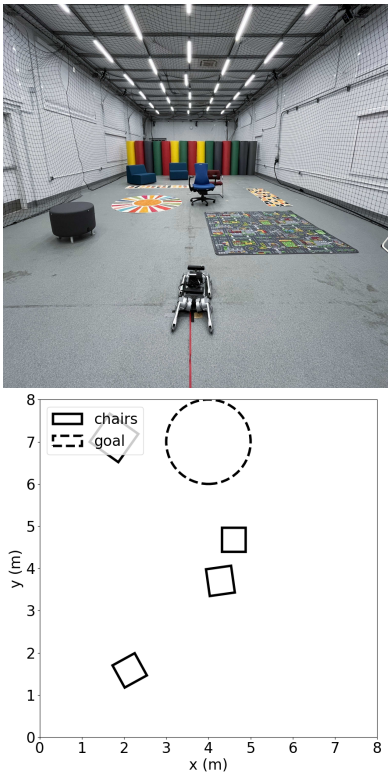
(10) Environment 10



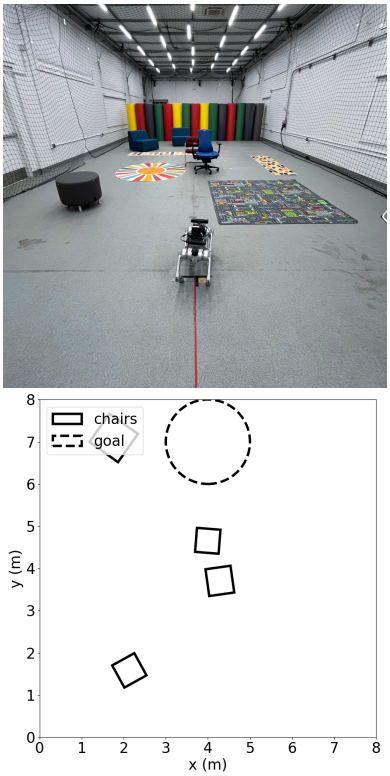
(11) Environment 11



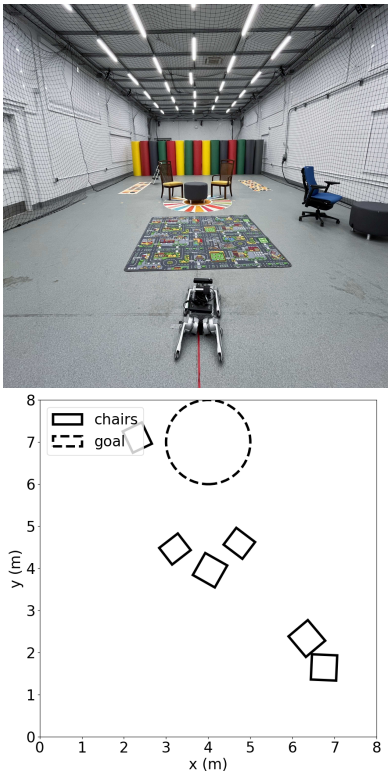
(12) Environment 12



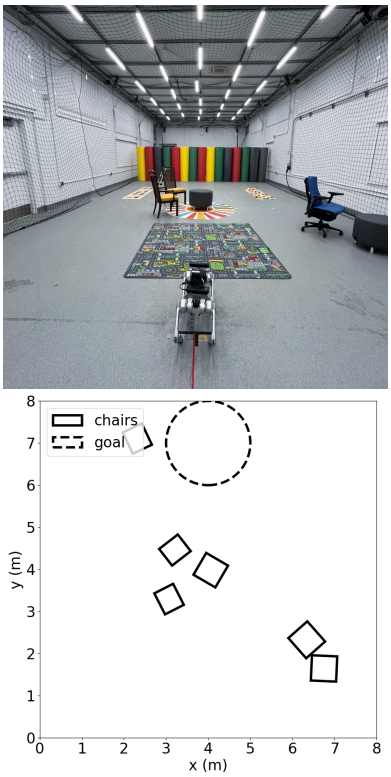
(13) Environment 13



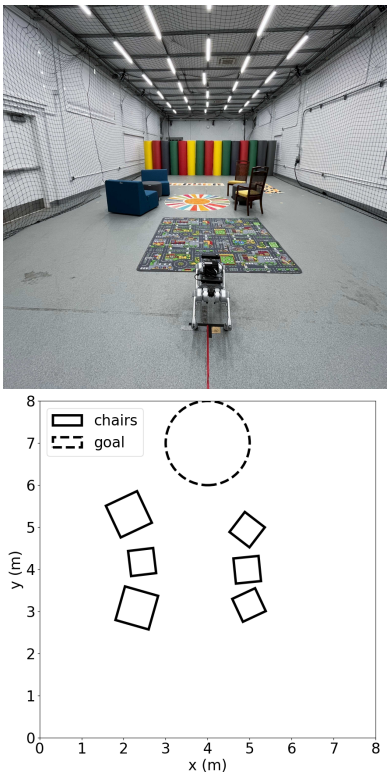
(14) Environment 14



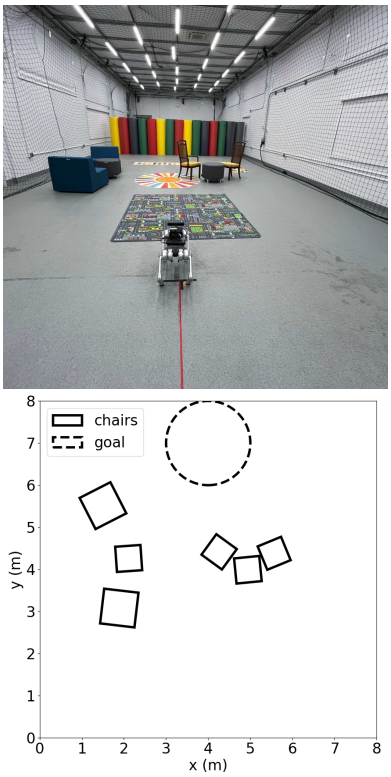
(15) Environment 15



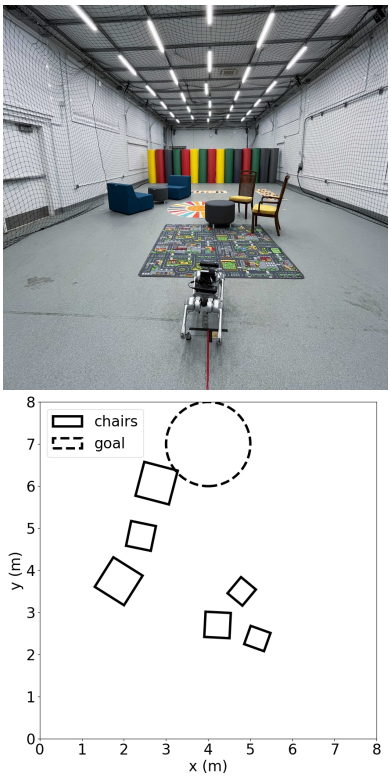
(16) Environment 16



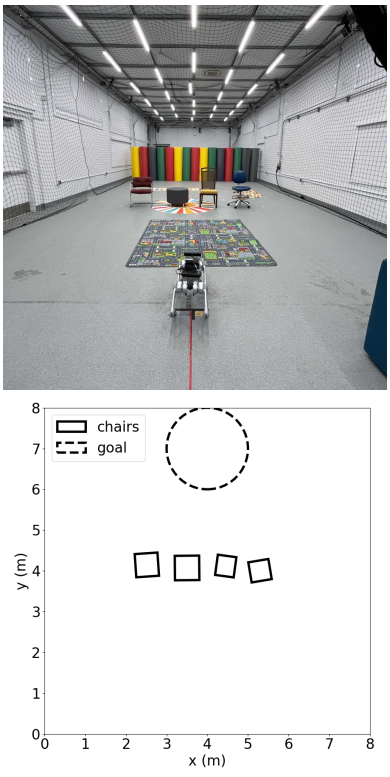
(17) Environment 17



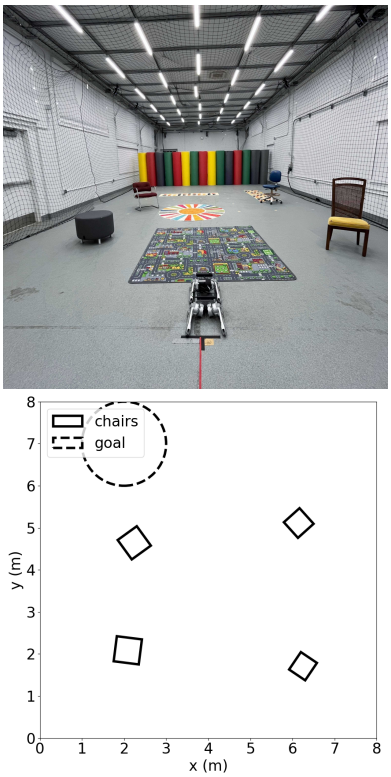
(18) Environment 18



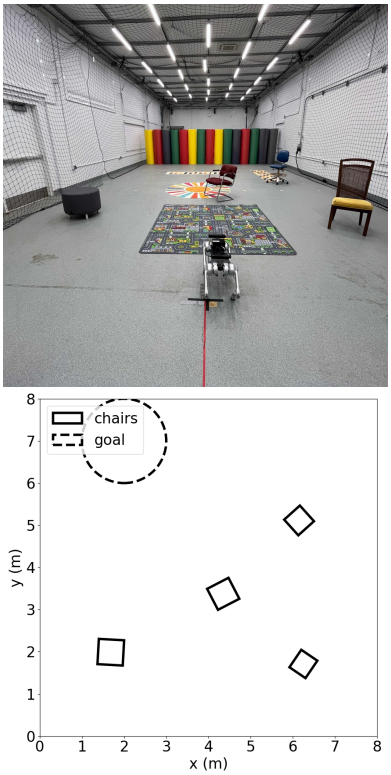
(19) Environment 19



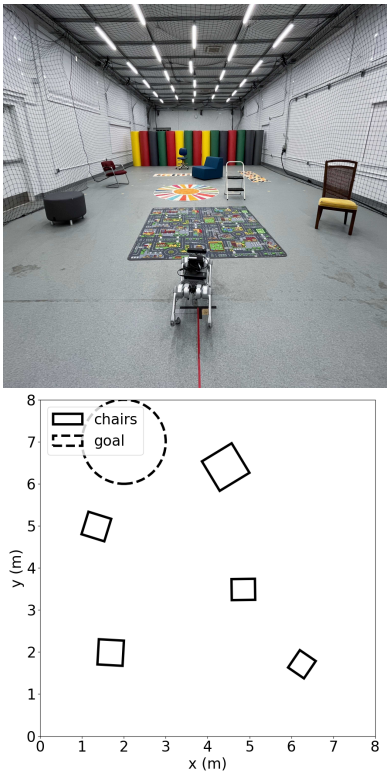
(20) Environment 20



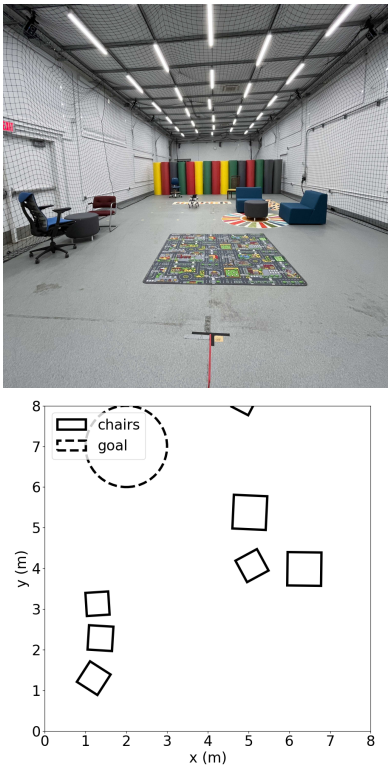
(21) Environment 21



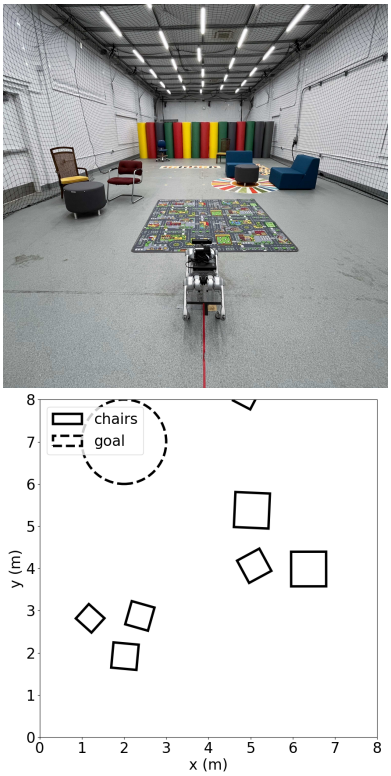
(22) Environment 22



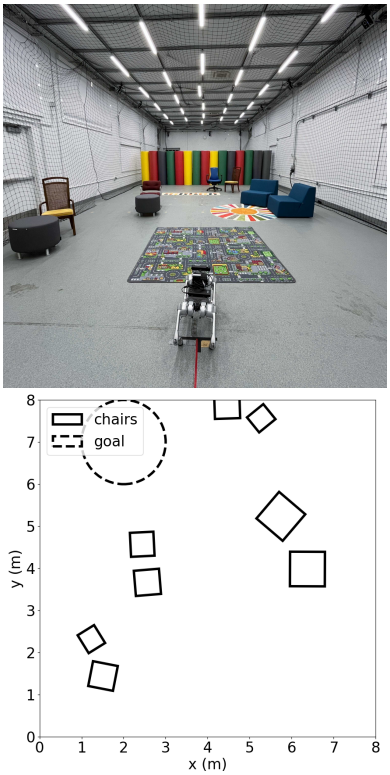
(23) Environment 23



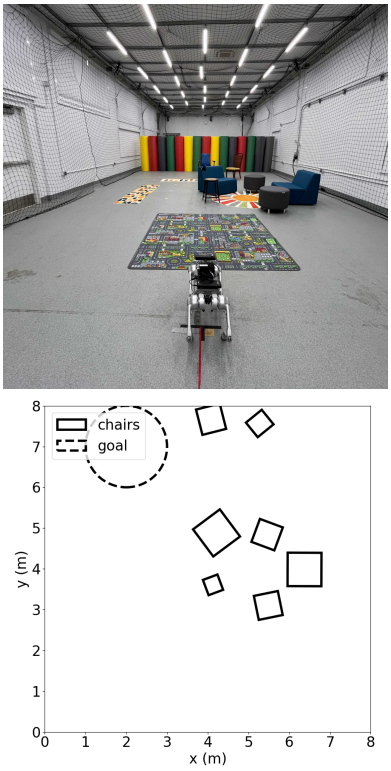
(24) Environment 24



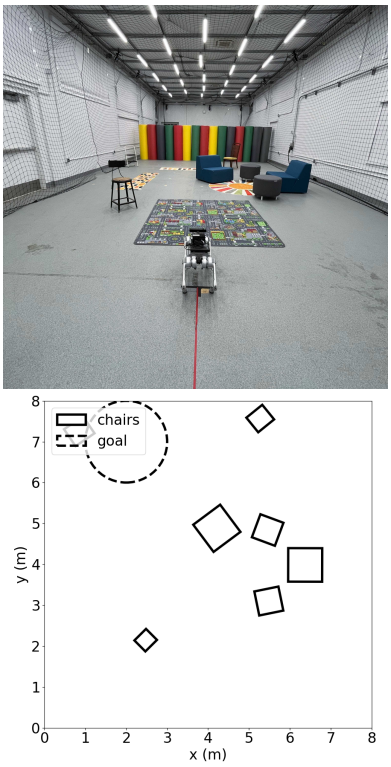
(25) Environment 25



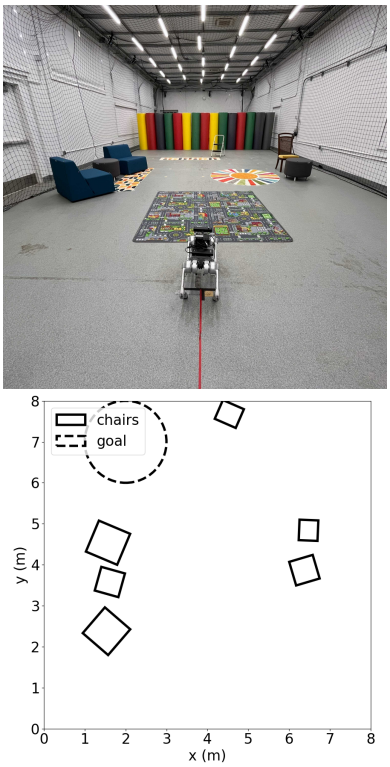
(26) Environment 26



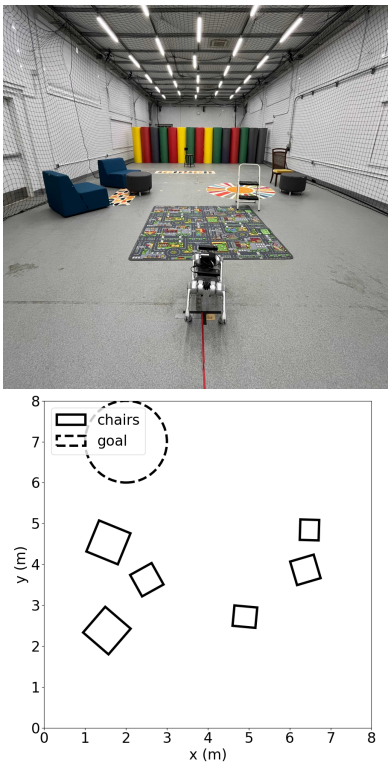
(27) Environment 27



(28) Environment 28



(29) Environment 29



(30) Environment 30