A Prototype of Probabilistic Inverse Graphics for Uncertainty-Aware Mobile Robot Manipulation, for Finding Bananas in Unknown Banana Lounges

Rachel Ma[§]
MIT

Xiaoyan Wang[§]
MIT

George Matheos[§]
MIT

***Abstract*—In this final project report for MIT's course 6.4212: Robotic Manipulation, we describe SPONANA. SPONANA is a prototype of an uncertainty-aware mobile manipulation system using probabilistic inverse graphics for belief-state generation. Uncertainty aware perception is a key component in mobile robots which must operate in open-world environments. Recent advances in inverse graphics, such as the development of NERFs [1], Gaussian splatting [2], and probabilistic-program based 3D scene interpretation [3], suggest that probabilistic inverse graphics may a be viable technology probabilistic robotic perception. Our SPONANA prototype verifies that modern probabilistic inverse graphics approches are robust and efficient enough to integrate into simple full-stack mobile manipulation systems comprising a number of modules, including probabilistic motion planners, antipodal grasp selectors, inverse-kinematics-based grasp executors, and belief-aware task-level controllers. We also extricate two ways SPONANA indicates the scope of probabilistic perception technology may need to be expanded to enable larger-scale robotic demonstrations. We ultimately apply SPONANA to a task of unparalleled importance: retrieving bananas from unknown banana lounges. We demonstrate that uncertainty awareness enables finding bananas as much as 1.5x faster than uncertainty-unaware approaches.**

I. Introduction

A seminal challenge for the field of robotics is developing mobile robots which can operate efficiently in open-universe environments. In such an environment, containing unknown objects, the robot will necessarily have uncertainty about what it will encounter, and to operate efficiently, it may need to take this uncertainty into account in its planning processes. In light of this grand challenge, a number of techniques have been introduced to enable robots to reason over belief states, probability distributions over world states, which quantify the robot's knowledge and uncertainty about the environment [4, 5, 6]. A prerequisite for the application of such techniques are perception algorithms which consume sensor data and output belief states. A long-standing proposal is to use probabilistic inverse graphics algorithms to consume raw data from robotic camera sensors and output posterior probability distributions over possible world states corresponding to these images, by performing inverse rendering [7, 8]. In recent years, computer vision approaches based on inverse rendering, such as NERFs [1, 9, 10] and 3D Gaussian splatting [2], have become state of the art in computer vision tasks such as scene reconstruction. Simultaneously, probabilistic inverse graphics systems have for the first time become mature enough to robustly interpret multi-object scenes in real-time, with uncertainty awareness [3, 11]. This confluence suggests that inverse graphics approaches may become important in uncertainty-aware robotic perception.

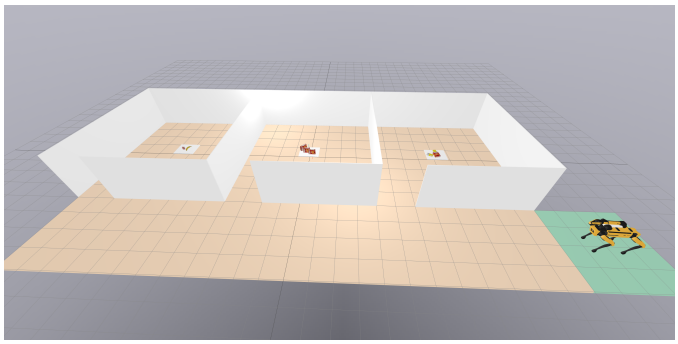


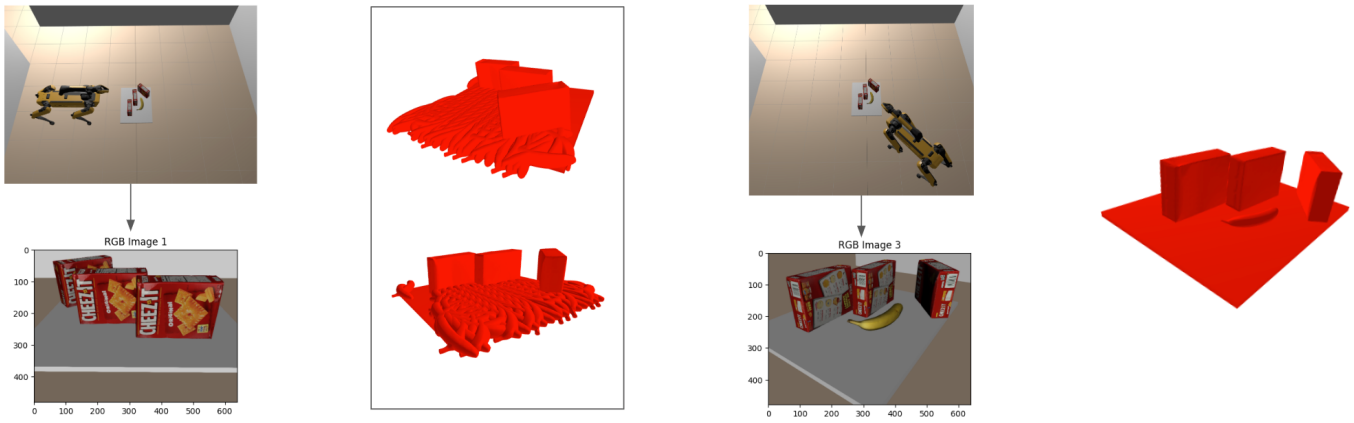
Fig. 1: The SPONANA task environment. A banana is hidden in known three-room environment with unknown objects cluttering the tables. During a simulation, Spot spawns in the starting area (green) and must explore the rooms, find and pick up the banana, and bring it to Spot's spawn location.

In our final project for MIT 6.4212: Robotic Manipulation, we prototyped, in simulation, a full-stack robotic controller for mobile manipulation in open-world environments, using probabilistic inverse graphics for belief-state generation. Figure 1 illustrates the task environment, and Figure 2 illustrates the system's use of probabilistic perception and uncertainty awareness.

A. Project objectives

1) Research objective: Our primary research objective in this work was to assess the viability at present of using probabilistic inverse graphics for perception in end-to-end robotic systems which must understand and interact with open-ended environments in an uncertainty-aware manner. We sought to elucidate two questions. First, are modern probabilistic inverse graphics systems such as [3] actually robust and efficient enough to integrate into end-to-end robotic systems that contain other complex modules, such as motion planners, grasp selectors, inverse-kinematics-based grasp executors, and

[§]Equal contribution



(a) Spot looks at the table and captures an image of it.

(b) Spot's belief state, rendered from two vantage points

(c) Spot views the table from a second angle.

(d) Spot's new belief state, with knowledge of the banana pose.

Fig. 2: Belief-aware banana-finding with Spot, with belief states generated via probabilistic inverse graphics. (a) Upon entering a room in the SPONANA environment, Spot looks at the table from a vantage point near the doorway, and captures an RGB-Depth image (RGB only shown). (b) Spot generates a belief state. The belief state contains knowledge of the poses of all the visible objects (here, the three cheez-it boxes and the table), as well as a collection of samples of possible locations for the target object (the banana), if it is on this table. This belief state reflects the knowledge that if the banana is on the table, it is hidden behind one of the cheez-it boxes. Spot considers what this belief state would look like, viewing the table from each corner opposite its current pose. (c) Spot determines that if it moves to the right hand side of the table, it has a higher probability of seeing the banana unoccluded. It moves to this corner of the table and captures a new image. (d) Spot updates its belief state, to reflect knowledge of the banana's pose.

task-level controllers? Second, what gaps remain in current approaches to probabilistic perception, which may need resolution before such approaches can be deployed in larger-scale open-ended robotics settings?

2) Technical objective: Our primary technical objective was for our mobile manipulator to achieve measurably greater task efficiency than baseline manipulators without uncertainty awareness. A key motivation for probabilistic inverse graphics approaches to perception is the ability to produce uncertainty-aware belief states, rather than point estimates of the world state. Therefore, in order for our prototype to be a satisfying test of the viability of such approaches for their intended use in robotics, the prototype must be able to attain improved task-level efficiency through uncertainty awareness.

B. Summary of results.

In Section IV, we describe SPONANA, a prototype controller for the Boston Dynamics Spot robot enabling it to search open-world environments to retrieve a pre-specified target object.

1) Insights on the maturity of probabilistic inverse graphics in robotics: SPONANA successfully integrates probabilistic inverse graphics in a multi-module, end-to-end mobile manipulator. It represents an affirmative answer to our first question in Section I-A. In Section VII, we articulate several insights attained in the development of this prototype, about the potential for probabilistic inverse graphics to be scaled to larger-scale, real-world robots. We also highlight two desiderata for future probabilistic perception systems.

2) Efficiency of the belief-aware controller: We find that our uncertainty-aware Spot controller is 1.5x more efficient at the SPONANA task than a baseline controller without belief-state awareness. See Section VI for details.

II. SPONANA: SPOT FINDING BANANAS IN UNKNOWN BANANA LOUNGES

To pursue these objectives, we created a prototype of an end-to-end robotic controller for the Boston Dynamics Spot robot and tested it in simulation. The task is to can search an environment containing unknown objects until it successfully locates a pre-specified target object. Efficient task performance requires probabilitic reasoning about likely target object locations.

Choice of target object. A critical question in the development of this simulation was the choice of the initial target object with which to test the prototype. After careful deliberation, we selected the banana. The banana is a white fruit with a yellow peel enjoyed by humans and chimpanzees alike [12]. This choice was inspired by ongoing research at MIT aimed at expanding the capabilities of the Spot robot to include retrieving banana lounges from the MIT banana lounge. This task has been reported as among the most inspiring applications of robotics to date [13]. We hope our prototype can help inspire the next generation to pursue robotics, and bananas.

A. SPONANA task details

We consider a Spot robot operating in an environment with a known map, filled with unknown objects. Figure 1 illustrates the map with which we tested the system, comprising three rooms, each containing a table. Spot knows that a banana is on a table somewhere in the environment. Spot must search the rooms until it finds the banana, and then pick up the banana once it is found. The tables may contain non-banana objects that occlude the banana from the robot's field of view, requiring Spot to examine each table from multiple vantage points to find the banana.

III. Related Work

The SPONANA system is at the intersection of probabilistic perception and inverse graphics, belief-space planning and control, and mobile manipulation and open-universe environments.

1) Deep perception: Many popular approaches to 3D perception utilize deep learning systems, often trained using probabilistic objectives [14, 15, 16]. There is evidence that these approaches struggle to consistently return coherent representations of uncertainty over scene structure [11]. There is also evidence that probabilistic inverse graphics systems, both built on top of deep perception [11, 17], and built without deep learning [3], can achieve more robust 3D pose estimation more data-efficiently than deep learning approaches alone. Recent work has also indicated that non-neural inverse graphics approaches like Gaussian splatting [2] can achieve comparable or superior results in tasks like scene reconstruction to approaches based on neural networks. Such results indicate that it is worth understanding the viability of inverse graphics approaches for probabilistic perception in end-to-end robotic systems, motivating the SPONANA project.

2) Probabilistic inverse graphics: Inverse graphics frames the problem of 3D perception as the inverse problem of rendering [7, 8]. Recent inverse graphics approaches include NeRFs [1, 9, 10] and 3D Gaussian splatting [2]. Probabilistic inverse graphics frames the inverse graphics problem as Bayesian inference (Section V-A). A number of algorithms have been proposed for probabilistic inverse graphics in various domains [18, 19], many based on inference in probabilistic programs. SPONANA builds the most directly on 3DP3 [11] and Bayes3D [3], two recent systems for performing probabilistic inference over scenes of 3D objects from known and unknown categories, from RGB-Depth image data. Bayes3D in particular has demonstrated the viability of probabilistic inverse graphics for real-time perception, by utilizing GPU acceleration. SPONANA builds on the low-level Bayes3D inverse-graphics library, extending it to support inference over unknown arrangements of objects in a multi-room environment.

3) Open-world mobile robotics: Robotic controllers and planners for open-world environments are often built using belief-state planning or belief-aware controllers [4, 5, 6], as in SPONANA. For our initial prototype, we tested SPONANA in a manually developed Drake simulation environments; future work could extend SPONANA to operate in popular mobile robotic benchmark environments such as BEHAVIOR-1K [20], and Habitat [21], and HandMeThat [22]. As probabilistic inverse graphics systems mature, it will become possible to compare mobile robotic systems like SPONANA to those using other approaches to perception, such as [23, 24, 25, 26]. One interesting comparison dimension is data-efficiency; since inverse-graphics systems include a strong inductive bias describing how 2D images are projected 3D scenes, they can require far less training to achieve good performance [3].

IV. The SPONANA System

A. SPONANA System Design

The SPONANA controller is built from four modules: the task-level controller `TaskController`, the `Navigator` module responsible for motion planning and controlling Spot's position in the world, the `Grasper` module responsible for grasp planning and execution, and the `Perception` module which performs probabilistic inverse graphics to generate belief states over the world state from RGB-Depth images. Figure 3 schematizes the input/output interfaces of each module. The following subsections will describe the interface and implementation of each module in turn.

1) Notation: We adopt the following notation for the values communicated between modules:

1) 1_{Name} denotes a boolean value with a given name.
2) $q_{\text{JointCollection}}$ denotes the configuration of a set of joints.
3) $\text{Image}_{\text{RGB}}$ and $\text{Image}_{\text{Depth}}$ denote Red/Green/Blue and Depth images, represented as values in $\mathbb{R}^{3\times\text{width}\times\text{height}}$ and $\mathbb{R}^{\text{width}\times\text{height}}$.
4) X_{WB} denotes the pose of body B in the world frame W.

2) Low-level interface to Spot: We model Spot's base as a mobile prismatic base, of which we can read and command x and y positions and the orientation rz. We assume ground truth access to the camera pose (see Sec. VII-B) The low-level robot interface receives commanded base positions, $q_{\text{BaseCommanded}}$, and a 7DOF desired arm configuration, $q_{\text{ArmCommanded}}$, from the `Navigator` and `Grasper` systems respectively.

3) Simulation environment: We used Drake [27]. Figure 1 illustrates the simulation environment. In each simulation, each table in the environment is populated with a manually-specified or randomly-generated configuration of YCB objects, and one table is populated with a banana.

B. Belief-aware task-level control

Algorithm 1 describes the logic implemented by the SPONANA task-level controller to find and pick up the target object. (For brevity, we elide the straightforward logic used to return the target object to Spot's starting region in the environment.) The controller receives boolean signal $1_{\text{NavigationComplete}}$ from the `Navigator` system, boolean signals $1_{\text{PerceptionComplete}}$ and $1_{\text{TargetFound}}$ from the `Perception` system, a vector $\vec{p}_{\text{visibility}}$ from the `Perception` subsystem, and boolean signal $1_{\text{GraspingComplete}}$ from the `Grasper` system.

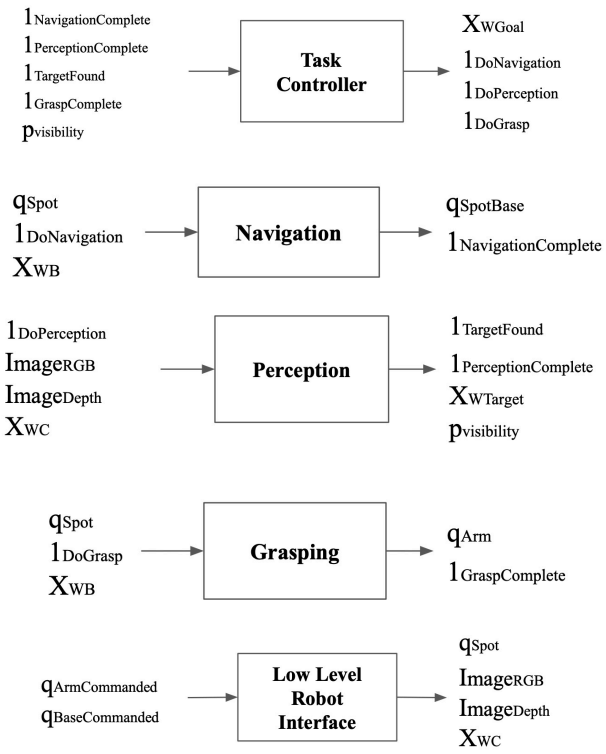


Fig. 3: The input/output interface for the 4 subsystems of the SPONANA controller, and the SPONANA controller's interface to the robot.

Algorithm 1 Belief-aware task-level controller

Require: Fixed parameters: $\vec{X}_{WT}, \vec{X}_{TC}, p_{\text{min}}$
Require: Inputs from other modules: $1_{\text{NavigationComplete}}$, $1_{\text{TargetObjectFound}}$, $1_{\text{PerceptionDone}}$, $1_{\text{GraspingComplete}}$, $\vec{p}_{\text{visibility}}$
Require: Controller State: Action, idx_{Room}, $\text{idx}_{\text{LocInRoom}}$
if Action = Navigate **then**
  if $1_{\text{NavigationComplete}} = 1$ **then**
    Action $\leftarrow$ RunPerception
  end if
else if Action = RunPerception **then**
  if $1_{\text{PerceptionDone}} = 1$ **then**
    if $1_{\text{TargetObjectFound}} = 1$ **then**
      Action $\leftarrow$ RunGrasping
    else
      ▷ Decide next position to explore.
      if $\exists i : \vec{p}_{\text{visibility}}[i] > p_{\text{min}}$ **then**
        $\text{idx}_{\text{LocInRoom}} \leftarrow \text{argmax}_i \vec{p}_{\text{visibility}}[i]$
      else
        $\text{idx}_{\text{Room}} \leftarrow \text{idx}_{\text{Room}} + 1$
        $\text{idx}_{\text{LocInRoom}} \leftarrow 0$
      end if
    end if
  end if
else if Action = RunGrasping **then**
  if $1_{\text{GraspingComplete}} = 1$ **then**
    Action $\leftarrow$ Done
  end if
else
end if
▷ Set the boolean output ports based on the current action.
Output $1_{\text{RunPerception}} = 1$ **if** Action = RunPerception **else** 0
Output $1_{\text{DoNavigation}} = 1$ **if** Action = Navgiate **else** 0
Output $1_{\text{DoGrasp}} = 1$ **if** Action = Grasping **else** 0
▷ Set the goal-to-navigate-to output to the desired camera pose in the world frame.
Output $X_{\text{WGoal}} = \vec{X}_{WT}[\text{idx}_{\text{Room}}] \times \vec{X}_{TC}[\text{idx}_{\text{LocInRoom}}]$
Update controller state to $(\text{Action}, \text{idx}_{\text{Room}}, \text{idx}_{\text{LocInRoom}})$

It outputs boolean signal $1_{\text{DoNavigation}}$ which it can use to activate the `Navigator`, signal $1_{\text{DoPerception}}$ to activate the `Perception` system, and 1_{DoGrasp} to activate the `Grasper`. It also outputs the next pose X_{WGoal} to navigate to to the `Navigator`.

The controller is instantiated with knowledge of the table poses in the environment, $\vec{X}_{WT}$. In our simulations, this is a length-3 vector where $\vec{X}_{WT}[i]$ is the pose of the ith table (the table in the ith room). The controller is also instantiated with a vector $\vec{X}_{TC}$ of camera poses in the table frame; these are poses at which the controller will have Spot inspect each table to determine what objects are on it. The vector $\vec{p}_{\text{visibility}}$ relates to $\vec{X}_{TC}$ as follows. For each index i, $\vec{p}_{\text{visibility}}[i]$ is an estimate of the posterior probability that (1) the target object is on the current table under inspection, and (2) if Spot goes to pose $\vec{X}_{TC}[i]$ relative to the current table being inspected, it will see the target object.

The belief-aware controller explores the the rooms in the environment one by one. In each room, it first navigates to pose $\vec{X}_{TC}[0]$ relative to the current table it is inspecting (Figure 2a). It then calls the `Perception` system, which updates its belief about the table contents, and thereby outputs an updated value of $1_{\text{TargetFound}}$ and an updated posterior probability vector $\vec{p}_{\text{visibility}}$. If the target was found, the controller activates the grasping module. If not, the controller checks whether the new visibility probability vector indicates that there is sufficiently high probability of seeing the target object from some other location. (This is done by thresholding w.r.t. a fixed value p_{min}.) This will occur when Spot cannot see the whole table from its initial viewpoint (for instance, due to objects on the table occluding part of the table, as in Figure 2). If so, Spot navigates to the pose at the table where it believes it has the highest probability of finding the target. The controller implementation utilizes 3 state variables: the current Action it is executing (either RunPerception, Navigate, RunGrasping, or Done), the index of the current room it is inspecting (idx_{Room}, which is an index for array $\vec{X}_{WT}$), and the index of the current pose it is inspecting the table from in that room ($\text{idx}_{\text{LocInRoom}}$, which is an index for array $\vec{X}_{TC}$).

C. Perception

The `Perception` module is responsible for maintaining a belief state. This belief state is a probability distribution over world states. We describe our implementation of probabilistic perception in Section V. The module recieves RGB and depth images $\text{Image}_{\text{RGB}}$ and $\text{Image}_{\text{Depth}}$ from the robot, an indicator $1_{\text{DoPerception}}$ from the `TaskController`, and the current Spot camera pose X_{WC}. (We give Spot ground-truth access to the camera pose from the simulator; see Sec VII-B for commentary.) The module outputs $1_{\text{PerceptionComplete}}$, $1_{\text{TargetFound}}$, and $\vec{p}_{\text{visibilty}}$, as described in Sec. IV-B.

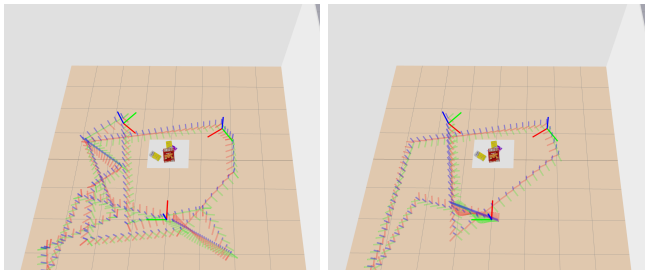


Fig. 4: Comparison of basic RRT trajectory (left) and the improved trajectory generated by RRT with shortcutting (right).

D. Navigation

The `Navigation` module is responsible for Spot's navigation in the environment. It performs motion planning using a rapidly-exploring random tree [28] with shortcutting (Figure 4), and executes the motion plan by sending controls to the robot base. The module accepts the boolean flag $1_{\text{DoNavigation}}$ as input, as well as the pose $X_{W\text{Goal}}$ to navigate to. Specifically, $X_{W\text{Goal}}$ is the pose Spot's camera should be at after navigation is complete. The module outputs a Boolean $1_{\text{NavigationComplete}}$ to interface with the task controller.

E. Grasping

The grasping system operates in 3 steps: (1) antipodal grasp selection, (2) gripper-frame grasp trajectory generation, and (3) iterative inverse kinematics for grasp execution. Antipodal grasp selection is performed per Ch. 5 of [29], on a point-cloud generated of the known target object mesh at the inferred target object location. A gripper-frame grasp trajectory template is filled in based on Spot's current gripper position and the target gripper pose. Figure 5 illustrates an example gripper-frame trajectory used from an end-to-end SPONANA simulation. Finally, the grasp is executed by iteratively running inverse kinematics via mathematical optimization, to move the gripper to the next frame in the gripper trajectory with minimal deviation from the current joint configuration q_{Arm}. The grasping module is currently the least robust SPONANA subsystem; see Section VII-B for details. The inputs of the module include the current 10DOF Spot configuration q_{Spot} (describing both the base and arm positions) from the robot, an activation signal 1_{DoGrasp} from the `Task Controller`, and the pose X_{WB} of the target object to grasp. The primary output is the new commanded 7DOF arm configuration q_{Arm}. The module also outputs a $1_{\text{GraspComplete}}$ indicator.

V. Perception via Probabilistic Inverse Graphics

A. Background

1) Belief states: A belief state is defined relative to a probability distribution P on $\mathcal{S} \times \mathcal{O}$, where $\mathcal{S}$ is the set of possible world states, and $\mathcal{O}$ is the set of possible observations of the environment. In SPONANA, $\mathcal{S}$ is the space of ways objects can be arranged in the world, and $\mathcal{O}$ is the set of sequences of images the Spot robot may have received at the current moment in the simulation. After observing value $o \in \mathcal{O}$, a perfect robot's belief state would be $P(s = \cdot | O)$, the posterior on the world state $s \in \mathcal{S}$ given observation o.

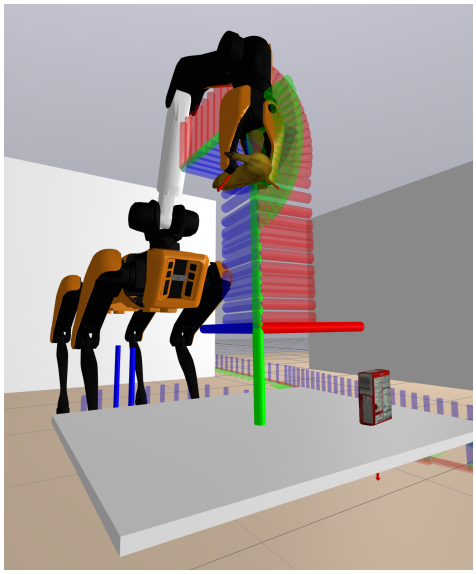


Fig. 5: Spot grasping the target object.

2) Probabilistic inverse graphics: Let $\mathcal{I} = \mathbb{R}^{k \times \text{width} \times \text{height}}$ denote the space of images with k channels. (If k = 4, these can be RGBD images.) In probabilistic inverse graphics, belief states are generated relative to a probabilistic model P on state space $\mathcal{S}$ and observation space $\mathcal{O} = \mathcal{I}^T$ for $t \in \mathbb{N}$, ie. where the observations are sequences of T images. P is a joint distribution on the scene s and a sequence of images which may be captured of the scene s by a noisy camera with known intrinsics, at T known poses in the environment. This distribution decomposes into a prior on the world state, $P_{\mathcal{S}}$, and a probability distribution $P_{\mathcal{I},X_{WC}}(I = \cdot; s)$ describing the distribution on possible noisy images I which may be captured of scene s by a camera at pose X_{WC}. Each noisy image is treated as independent from the others, conditional upon s. That is, if $P_{\mathcal{S}}$ has probability density $p_{\mathcal{S}}$ and $P_{\mathcal{I},X_{WC}}(I = \cdot; o)$ has density $p_{\mathcal{I},X_{WC}}(I; o)$, then the density $p_{T,\vec{X}_{WC}}$ for the joint distribution $P_{T,\vec{X}_{WC}}$ on the state and a sequence of images taken at each pose $\vec{X}_{WC}[t]$ for $t \in \{1, 2, \ldots, T\}$, is

$$p_{T,\vec{X}_{WC}}(s, (I_1, \ldots, I_t)) = p_{\mathcal{S}}(s) \prod_{t=1}^{T} p_{\mathcal{I},X_{WC}}(I_t; s)$$

In probabilistic inverse graphics, the distribution $P_{\mathcal{I},X_{WC}}$ is typically defined in terms of a deterministic renderer $R : \mathcal{S} \to \mathcal{I}$. The generative process corresponding to $P_{\mathcal{I},X_{WC}}(I = \cdot; s)$ is a process which adds stochastic noise to the deterministic rendered image $R(s)$. The goal of inverse graphics is to approximate the optimal belief state $P(s = \cdot; I_1, \ldots, I_t)$ over the world state, given images $(I_t)_{t=1}^{T}$.

B. Probabilistic inverse graphics in SPONANA

The SPONANA probabilistic perception system extends Bayes3D [3], building on the low-level inverse-graphics functionality the Bayes3D library provides, adding support for

perception in multi-room environments, and support for belief state queries about the probabilities as-of-yet unseen objects will be visible from given camera poses, given the objects' presence in the scene.

Our current SPONANA prototype makes the following assumptions to simplify the perception problem. We assume known models of the objects which may be encountered in the environment. (See Sec. VII-A for commentary on this.) We use YCB object models [30] but note that [3] supports online learning of new object models.

To simplify the inference algorithm while maintaining efficiency, we give Spot knowledge of a super-set of the object categories which may be present on each table, and knowledge of the number of objects on each table other than the target object. (The objects are still unknown in that the number of objects of each type is unknown within these constraints.) We believe it is a very straightforward extension to enable SPONANA to automatically infer the number of objects. Efficiently supporting more open-ended object libraries may require initially parsing scenes in terms of simpler representations, as suggested in Sec. VII-A, to enable faster object-parsing to occur on a much more concise representation of the scene than raw RGB-Depth images.

1) The SPONANA state space: In a SPONANA environment with m rooms, a world-state $S \in \mathcal{S}$ is a tuple $S = (\bar{S}_1, \bar{S}_2, \ldots, \bar{S}_m)$, where $\bar{S}_i$ is the state of the ith room. Each $\bar{S}_i$ is a set describing the configuration of the entities in the ith room. [1] Specifically, $\bar{S}_i = \{e_1^i, e_2^i, \ldots, e_n^i\}$ where n is the number of objects in the room, and each e_j^i is a tuple $(c_j^i, X_{We_j^i})$ of the category c_j^i of the jth entity in the ith room, and the object's pose $X_{We_j^i}$ expressed in the world frame. [2]

2) The prior on scenes: In SPONANA, our scene prior is the uniform prior over a subset $\mathcal{S}'$ of the space $\mathcal{S}$ of all scenes. $\mathcal{S}'$ contains all scenes in which each room has N or fewer objects with categories in the category superset prespecified to SPONANA, and where each object is in flush contact with the table. Future extensions to this prior, for instance following [11], could enable SPONANA to infer full scene graphs of object contact relationships from observed images.

3) The observation model: Following [3], SPONANA uses a renderer-based likelihood, based on the NVDiffrast depth renderer [31]. We expect that the SPONANA system would be robust across a variety of different depth-image noise models, such as those proposed in [11] and [3]. In our experiments we experimented with a new noise model suggested to us by the authors of [3]; as we expect SPONANA is robust across these noise models, we elide the details for brevity.

4) Inference: Following [3], we perform pose estimation using sequential Monte Carlo through a sequence of distributions over increasingly large scenes. (The first distribution only supports scenes with one object, the second supports scenes with two objects, and so on.) Each sequential Monte Carlo proposal distribution fits a new object using GPU-accelerated coarse-to-fine enumeration over possible object poses. As this is a highly accurate proposal distribution, it suffices to run sequential Monte Carlo with only one particle. The result is a belief state like that in Figure 2d, containing objects with known poses. From this, the perception system can output the flag $1_{\text{TargetFound}}$, as well as do the calculations described in the following paragraph.

5) Estimating the probability of seeing the target object at a new pose: After fitting all the known object poses, the SPONANA belief-state estimator must output a vector $\vec{p}_{\text{visibility}}$ as described in Section IV-B. To do this, if the target object has not been seen, the perception system generates a set of samples of possible target locations in the environment. It accepts or rejects each sample based on the likelihood of the observed image given a scene containing the target object at that location. This produces a belief state like that illustrated in Figure 2b, containing, in addition to the known object poses, a set of samples of possible locations for the target image which would be consistent with the images observed so far. SPONANA can then consider the fraction of these possible poses which would be visible from any given camera pose X_{WC}, to populate the visibility probability vector $\vec{p}_{\text{visibility}}$. Each time the SPONANA `Perception` module is called, it performs this calculation for each looking pose in the current room Spot is in (ie. for each element of $\vec{X}_{TC}$). It then returns this information to the task controller.

VI. Results

A. Comparing the efficiency of probability-aware and unaware task-level controllers.

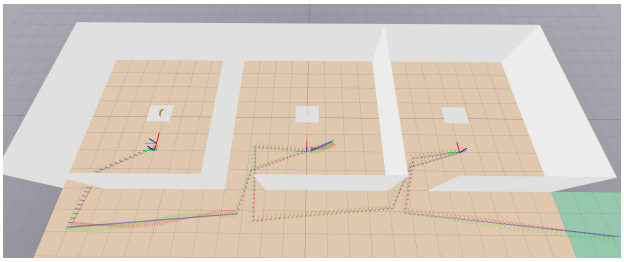
Our primary research objective in developing SPONANA was to test the viability of using probabilistic inverse graphics for perception in mobile manipulation robots. As the motivation for probabilistic perception orients around the need for uncertainty-awareness, the key property we wished to ensure the SPONANA system had was that its use of uncertaity-awareness improved its efficiency at its task-level goal. To verify this, we developed an ablation to the belief-aware SPONANA controller from Section IV-B, and compared the efficiency of the two controllers (Table I). This ablated task-level controller receives no information about the belief state except whether or not it has seen the target object. Specifically,

Environment type	Belief Aware	Belief Unaware
Mostly Empty	**44s**	69s
Partially Occluded	**56s**	75s
Highly Occluded	71s	72s

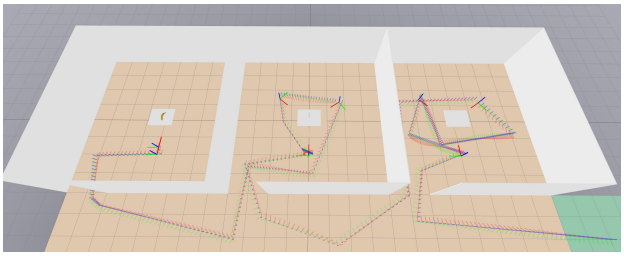
TABLE I: In-simulation time from initialization until Spot finds the target object, in the 3 environment variants illustrated in Figure 6. In many environments, the belief-aware controller is significantly more efficient than the belief-unaware variant.

[1] Note that we use the term "object" and "entity" interchangeably throughout this report. We use the letter e to refer to objects/entities, as we use o to refer to observations (images).

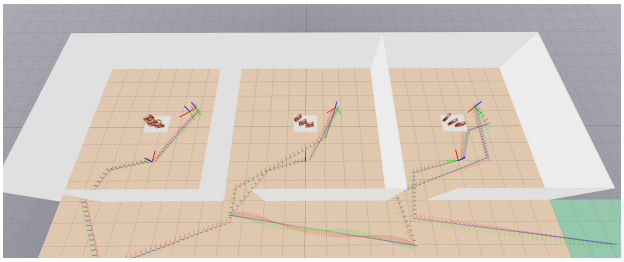
[2] In this project, we assumed the map is known, and Spot can be tracked exactly, so we describe the state space without including these values; in settings where these assumptions do not hold, the state space considered by the perception system must also contain the map and robot state.



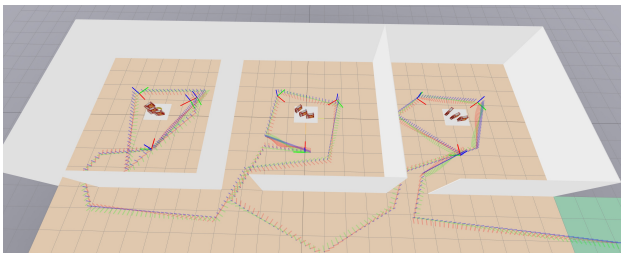
(a) Belief-aware controller; mostly empty environment.



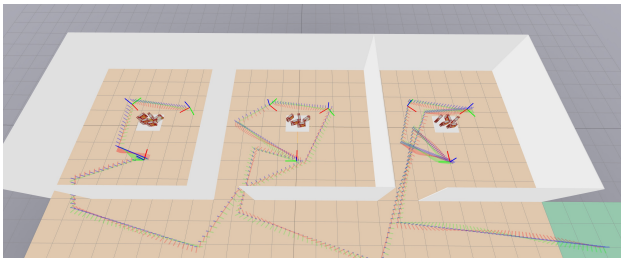
(b) Belief-unaware controller; mostly empty environment.



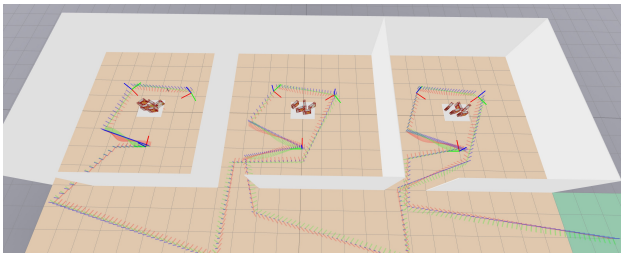
(c) Belief-aware controller; partly occluded environment.



(d) Belief-unaware controller; partly occluded environment.



(e) Belief-aware controller; highly occluded environment.



(f) Belief-unaware controller; highly occluded environment.

Fig. 6: Comparison between uncertainty aware and unaware SPONANA controllers. The belief-unaware controller inspects each table in the environment from three vantage points, until it finds the banana (leftmost table). The belief-aware controller only checks the tables from other viewpoints if it thinks there is sufficient probability that the banana could be visible from there, given its observations so far. Only in highly cluttered environments (e-f) is the uncertainty aware controller unable to find more optimal trajectories than the baseline.

the belief-unaware controller is obtained by modifying lines 11-18 in Algorithm 1. These lines in the algorithm compute the next location Spot should search for the target object in the current room, based on probability estimates for where the object may be, and command Spot to proceed to the next room when it is sufficiently improbable that the target object would be visible from another location in the room. Our belief-unaware controller variant replaces lines 11-18 with code that simply has Spot follow a fixed trajectory through the environment, inspecting each table from several vantage points, until it finds the target object.

Figure 6 illustrates the two controllers, run in 3 environment variants with different objects on the tables. Panels (b), (d), and (f) illustrate that the belief-unaware controller follows roughly the same trajectory in each environment variant, viewing each table from 3 viewpoints. Panels (a), (c), and (e) illustrate the more efficient trajectories taken by the belief-aware controller. When there is little clutter on the tables, the belief-aware controller recognizes that they are empty after only inspecting each table from one viewpoint, and proceeds to the next room (panel (a)). In environments where there is some clutter, and one side of the table appears more likely to have good visibility than the other, the belief-aware controller steers Spot to this optimal vantage point (panel (c)) rather than proceeding along an observation-unaware trajectory (panel (d)). Table I quantifies the superior performance of the belief-aware controller, in the time taken for Spot to find the target object in each environment.

VII. Discussion

Our primary objective in developing SPONANA was to integration-test modern systems for probabilistic inverse graphics, in the context of open-world mobile robotics. We found that systems like [3] have indeed become efficient and robust enough to integrate into controllers for mobile robots comprising multiple sub-modules. We conducted our experiments in simulation, but note that [11] presents compelling evidence regarding the robustness of probabilistic inverse graphics to noisy real-world camera images. Our experiments in Section VI also confirm that robotic systems based on these

probabilistic perception algorithms can be more efficient at common open-world robotics tasks than probability-unaware baselines.

A. Desiderata for future probabilistic perception systems

While SPONANA supports the effectiveness of constrained uses of probabilistic inverse graphics for perception in small-scale mobile manipulation tasks, it also sheds light on ways the scope of probabilistic perception may need to expand to support the development of robust mobile manipulators operating in even more open-ended environments. We wish to highlight two key expansions of scope our experience with SPONANA indicates may be necessary.

1) Explaining matter without known object meshes: Our experience indicates that probabilistic perception systems may need to expand their state representations, to enable them to explain visual matter which cannot be well-understood by any of the object models known to the perception system. One approach is to use systems like [3] which support few-shot object learning in real time, and develop a robotic system which can scan novel objects upon encountering them, to add the object shape to its library of known objects. We anticipate that this is a good approach for robots which must manipulate unknown objects. In other situations, it is unimportant and inefficient to understand an object model, but important for the robot to be able to reason about the potential for collision with, and occlusion by, matter in the environment. An example is SPONANA environments with unknown objects or clutter occluding the target object. One approach to solving this problem is to develop probabilistic inverse graphics systems where latent scenes are represented as containing both objects with known meshes, and coarser descriptions of other matter in the environment, perhaps using occupancy grids [32] or 3D Gaussians [2]. [3]

2) Richer interfaces between probabilistic perception and belief-state control: In SPONANA, the belief-state controller queries the probabilistic perception system in two ways: (1) asking whether the target object has been spotted yet, and (2) querying the probability that the target object would be visible from a given camera pose in the world. Because the type of the target object is known ahead of time, the SPONANA perception can eagerly compute the response to these queries each time it processes a new image from the camera sensor. However, generating a high-precision estimate of the probability that the target object will be found at a given location requires generating a large collection of samples of possible target object locations consistent with the observed images. This is an expensive operation, and in even simple extensions to the SPONANA task, this eager-evaluation approach would become intractable. For instance, consider the SPONANA variant in which the target object's type is not known ahead of time, but can instead be specified online. Eager evaluation would require generating large sample sets of possible target object locations for every object category. Further, consider that in practice, only very crude estimates of these object-finding probabilities may suffice (as small differences in probability will not cause significantly sub-optimal performance, except in the highest-stakes scenarios).

This suggests that the development of efficient belief-space controllers may benefit from research developing improved interfaces by which controllers can communicate queries to a probabilistic perception system, to trigger the lazy, and possibly highly approximate, evaluation of the belief state query.

B. SPONANA Limitations and Next Steps

Above, we articulated two ways probabilistic perception systems could be expanded to enable approaches like SPONANA to be applied to a much wider range of open-world mobile manipulation tasks. Here, we articulate several limitations of our current SPONANA implementation.

1) Perception: For limitations and next steps for the probabilistic perception system, see the commentary in Section V-B and Section VII-A.

2) Improving the robustness of object manipulation: The least robust system in the SPONANA implementation is the grasping system. The current grasping system currently fails to grasp the target object in the unit test when the simulation is run with realistic friction parameters. [4] We believe the problem is twofold. First, the current antipodal grasp selection algorithm generates gripper poses optimized for parallel grippers like the Weiss robotics WSG. These selected grasps often perform poorly when using the Spot's angular gripper, and we have had found that hard-coding grasp locations can work better than than the automatically-generated grasps optimized for parallel grippers. While grasps from the antipodal grasp selector have succeeded in unit tests, in our full SPONANA simulation we use manually chosen grasp poses. These poses were specified relative to the target object, and then automatically executed in simulation using the inferred target object pose returned by perception. Future SPONANA implementations will need to develop a new grasp selection procedure, optimized for the Spot gripper. Second, the current arm controller uses position control, rather than force control. We believe adding force control will be necessary to apply enough pressure to the target object to pick it up when the simulation's friction parameters are not artificially inflated. Our understanding is that the real Boston Dynamics Spot ships with a built-in grasping procedure; deploying SPONANA on a real Spot robot may be able to utilize this procedure rather than requiring the manual implementation of force control.

3) Improved motion planning collision checking: Our current RRT implementation occasionally results in Spot nicking

[3]We thank the CHISight team at the MIT Probabilistic Computing Project for ongoing conversations about extending probabilistic inverse graphics systems to enable the representation of matter poorly explained by known object models, and for the idea of using 3D Gaussians for this.

[4]For our initial experiments we set friction parameters to be artificially inflated, to enable end-to-end testing of the SPONANA prototype with successful banana pickup. With regular friction parameters, the banana slips out of the current gripper's hand. Switching from position to force control should fix this problem.

the walls in the environment. Before deploying SPONANA in the real world, it will be necessary to improve the consistency of the collision-checker used in motion planning.

4) Belief space planning: Task-level control in SPONANA is managed by a finite state machine with transitions determined by the current system belief state (Section IV-B). To extend SPONANA to support a more open-ended array of tasks, this belief-state controller should be replaced by a more task-general belief space planner [5, 6].

5) SLAM: Our prototype SPONANA controller was given ground-truth access to Spot's pose in the world frame, and was also given access to the environment map. (It was not, however, given ground-truth knowledge of the objects in the enviornment.) To enable Spot to find bananas (or other target objects) outside simulation and in environments with unknown maps as well as unknown objects, it will be necessary to extend the SPONANA perception system to perform simultaneous localization and mapping [33].

VIII. Conclusion

In this report, we described SPONANA, a prototype of an uncertainty-aware mobile manipulation system built on top of a probabilistic inverse graphics perception system. We applied SPONANA to the task of retrieving bananas from unknown banana lounges, and demonstrate that uncertainty awareness enabled behavior up to 1.5x more efficient than that of a belief-unaware controller. We described two desiderata we believe may be important in future probabilistic perception systems and belief-state controllers: the capacity to understand visual matter not readily parseable as objects, and richer interfaces for communication between belief-state control and probabilistic perception modules.

References

[1] B. Mildenhall, P. P. Srinivasan, M. Tancik, J. T. Barron, R. Ramamoorthi, and R. Ng, "Nerf: Representing scenes as neural radiance fields for view synthesis," *Communications of the ACM*, vol. 65, no. 1, pp. 99–106, 2021.

[2] B. Kerbl, G. Kopanas, T. Leimkühler, and G. Drettakis, "3d gaussian splatting for real-time radiance field rendering," *ACM Transactions on Graphics (ToG)*, vol. 42, no. 4, pp. 1–14, 2023.

[3] N. Gothoskar, M. Ghavami, E. Li, A. Curtis, M. Noseworthy, K. Chung, W. T. Freeman, J. B. Tenenbaum, M. Klukas, and V. K. Mansinghka, "Bayes3d: fast learning and inference in structured generative models of 3d objects and scenes," *Under Review at ICRA 2023*, 2023. [Online]. Available: https://github.com/probcomp/bayes3d

[4] C. R. Garrett, C. Paxton, T. Lozano-Pérez, L. P. Kaelbling, and D. Fox, "Online replanning in belief space for partially observable task and motion problems," 2020.

[5] L. P. Kaelbling and T. Lozano-Pérez, "Integrated robot task and motion planning in the now," 2012.

[6] D. Hadfield-Menell, E. Groshev, R. Chitnis, and P. Abbeel, "Modular task and motion planning in belief space," in *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2015, pp. 4991–4998.

[7] A. Yuille and D. Kersten, "Vision as bayesian inference: analysis by synthesis?" *Trends in cognitive sciences*, vol. 10, no. 7, pp. 301–308, 2006.

[8] D. Kersten, P. Mamassian, and A. Yuille, "Object perception as bayesian inference," *Annu. Rev. Psychol.*, vol. 55, pp. 271–304, 2004.

[9] A. Yu, V. Ye, M. Tancik, and A. Kanazawa, "pixelnerf: Neural radiance fields from one or few images," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 4578–4587.

[10] M. D. Hoffman, T. A. Le, P. Sountsov, C. Suter, B. Lee, V. K. Mansinghka, and R. A. Saurous, "Probnerf: Uncertainty-aware inference of 3d shapes from 2d images," in *International Conference on Artificial Intelligence and Statistics*. PMLR, 2023, pp. 10 425–10 444.

[11] N. Gothoskar, M. Cusumano-Towner, B. Zinberg, M. Ghavamizadeh, F. Pollok, A. Garrett, J. Tenenbaum, D. Gutfreund, and V. Mansinghka, "3dp3: 3d scene perception via probabilistic programming," *Advances in Neural Information Processing Systems*, vol. 34, pp. 9600–9612, 2021.

[12] Wikipedia contributors, "Banana — Wikipedia, the free encyclopedia," 2023, [Online; accessed 11-December-2023]. [Online]. Available: https://en.wikipedia.org/w/index.php?title=Banana&oldid=1188727193

[13] T. the Beaver (Probably), "Random exclamation heard in the halls of mit."

[14] F. Manhardt, W. Kehl, N. Navab, and F. Tombari, "Deep model-based 6d pose refinement in rgb," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 800–815.

[15] X. Chen, R. Chen, Z. Sui, Z. Ye, Y. Liu, R. I. Bahar, and O. C. Jenkins, "Grip: Generative robust inference and perception for semantic robot manipulation in adversarial environments," in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2019, pp. 3988–3995.

[16] X. Deng, A. Mousavian, Y. Xiang, F. Xia, T. Bretl, and D. Fox, "Poserbpf: A rao–blackwellized particle filter for 6-d object pose tracking," *IEEE Transactions on Robotics*, vol. 37, no. 5, pp. 1328–1342, 2021.

[17] G. Zhou, N. Gothoskar, L. Wang, J. B. Tenenbaum, D. Gutfreund, M. Lázaro-Gredilla, D. George, and V. K. Mansinghka, "3d neural embedding likelihood for robust sim-to-real transfer in inverse graphics," *arXiv preprint arXiv:2302.03744*, 2023.

[18] V. K. Mansinghka, T. D. Kulkarni, Y. N. Perov, and J. Tenenbaum, "Approximate bayesian image interpretation using generative probabilistic graphics programs," *Advances in Neural Information Processing Systems*, vol. 26, 2013.

[19] T. D. Kulkarni, P. Kohli, J. B. Tenenbaum, and V. Mans-

inghka, "Picture: A probabilistic programming language for scene perception," in *Proceedings of the ieee conference on computer vision and pattern recognition*, 2015, pp. 4390–4399.

[20] C. Li, R. Zhang, J. Wong, C. Gokmen, S. Srivastava, R. Martín-Martín, C. Wang, G. Levine, M. Lingelbach, J. Sun *et al.*, "Behavior-1k: A benchmark for embodied ai with 1,000 everyday activities and realistic simulation," in *Conference on Robot Learning*. PMLR, 2023, pp. 80–93.

[21] M. Savva, A. Kadian, O. Maksymets, Y. Zhao, E. Wijmans, B. Jain, J. Straub, J. Liu, V. Koltun, J. Malik *et al.*, "Habitat: A platform for embodied ai research," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2019, pp. 9339–9347.

[22] Y. Wan, J. Mao, and J. B. Tenenbaum, "Handmethat: Human-robot communication in physical and social environments," in *Thirty-sixth Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2022. [Online]. Available: https://openreview.net/forum?id=nUTemM6v9sv

[23] S. Yenamandra, A. Ramachandran, K. Yadav, A. Wang, M. Khanna, T. Gervet, T.-Y. Yang, V. Jain, A. W. Clegg, J. Turner *et al.*, "Homerobot: Open-vocabulary mobile manipulation," *arXiv preprint arXiv:2306.11565*, 2023.

[24] T. Gervet, S. Chintala, D. Batra, J. Malik, and D. S. Chaplot, "Navigating to objects in the real world," *Science Robotics*, vol. 8, no. 79, p. eadf6991, 2023.

[25] J. Wang, F. Adib, R. Knepper, D. Katabi, and D. Rus, "Rf-compass: Robot object manipulation using rfids," in *Proceedings of the 19th annual international conference on Mobile computing & networking*, 2013, pp. 3–14.

[26] A. Stone, T. Xiao, Y. Lu, K. Gopalakrishnan, K.-H. Lee, Q. Vuong, P. Wohlhart, B. Zitkovich, F. Xia, C. Finn *et al.*, "Open-world object manipulation using pre-trained vision-language models," *arXiv preprint arXiv:2303.00905*, 2023.

[27] R. Tedrake *et al.*, "Drake: Model-based design and verification for robotics," 2019.

[28] S. M. LaValle, "Rapidly-exploring random trees : a new tool for path planning," *The annual research report*, 1998. [Online]. Available: https://api.semanticscholar.org/CorpusID:14744621

[29] R. Tedrake, *Robotic Manipulation*, 2023. [Online]. Available: http://manipulation.mit.edu

[30] B. Calli, A. Singh, A. Walsman, S. Srinivasa, P. Abbeel, and A. M. Dollar, "The ycb object and model set: Towards common benchmarks for manipulation research," in *2015 international conference on advanced robotics (ICAR)*. IEEE, 2015, pp. 510–517.

[31] S. Laine, J. Hellsten, T. Karras, Y. Seol, J. Lehtinen, and T. Aila, "Modular primitives for high-performance differentiable rendering," *ACM Transactions on Graphics (TOG)*, vol. 39, no. 6, pp. 1–14, 2020.

[32] A. Hornung, K. M. Wurm, M. Bennewitz, C. Stachniss, and W. Burgard, "Octomap: An efficient probabilistic 3d mapping framework based on octrees," *Autonomous robots*, vol. 34, pp. 189–206, 2013.

[33] S. Thrun, W. Burgard, and D. Fox, "Probabilistic robotics (intelligent robotics and autonomous agents series)," *Intelligent robotics and autonomous agents, The MIT Press (August 2005)*, 2006.