Sampling-Based Motion Planning With Scene Graphs Under Perception Constraints

Qingxi Meng, *Student Member, IEEE*, Emiliano Flores, *Student Member, IEEE*, Thai Duong, *Member, IEEE*, Vaibhav Unhelkar, *Member, IEEE*, and Lydia E. Kavraki, *Fellow, IEEE*

***Abstract*—It will be increasingly common for robots to operate in cluttered human-centered environments such as homes, workplaces, and hospitals, where the robot is often tasked to maintain perception constraints, such as monitoring people or multiple objects, for safety and reliability while executing its task. However, existing perception-aware approaches typically focus on low-degree-of-freedom (DoF) systems or only consider a single object in the context of high-DoF robots. This motivates us to consider the problem of perception-aware motion planning for high-DoF robots that accounts for multi-object monitoring constraints. We employ a scene graph representation of the environment, offering a great potential for incorporating long-horizon task and motion planning thanks to its rich semantic and spatial information. However, it does not capture perception-constrained information, such as the viewpoints the user prefers. To address these challenges, we propose MOPS-PRM, a roadmap-based motion planner, that integrates the perception cost of observing multiple objects or humans directly into motion planning for high-DoF robots. The perception cost is embedded to each object as part of a scene graph, and used to selectively sample configurations for roadmap construction, implicitly enforcing the perception constraints. Our method is extensively validated in both simulated and real-world experiments, achieving more than $\sim 36\%$ improvement in the average number of detected objects and $\sim 17\%$ better track rate against other perception-constrained baselines, with comparable planning times and path lengths.**

***Index Terms*—Constrained motion planning, motion and path planning, vision-based navigation.**

I. Introduction

AUTONOMOUS robot systems have become more prevalent in a variety of human-centered environments, such as workplaces [1], hospitals [2], urban areas [3] and homes [4]. A key challenge in such environments is that the robot often has to plan a collision-free trajectory to finish its own tasks, while monitoring other humans or objects of interest along the trajectory. For example, a household robot may need to deliver an item while keeping a person's face or gestures in view, or a museum patrol robot may navigate around visitors while maintaining visibility of multiple paintings or sculptures. Such perception constraints are essential for monitoring the surrounding objects [5], [6], improving the robot's state estimation [7], [8], enabling safe navigation [9], [10], and mapping and exploration of the environment [11]. Therefore, we aim to address the problem of motion planning under perception constraints in this letter.

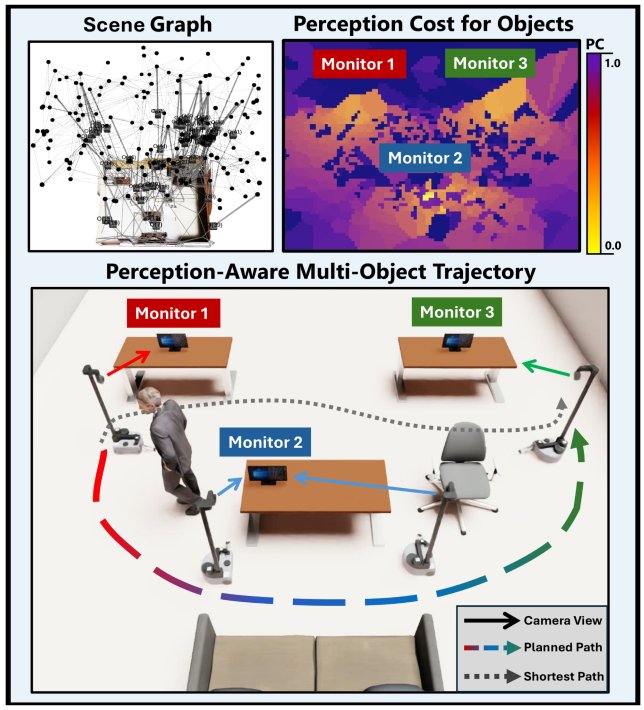

Fig. 1. An illustration of our perception-aware motion planner that leverages a scene graph embedded with perception costs to generate a trajectory from a start to a goal, while monitoring three objects of interest. The screen of the monitor is the preferable viewpoint in this scenario.

A common example of perception-aware constraints in motion planning is object tracking and monitoring, where the robot maintains visibility of a single object [5], [6], [12] or multiple objects [13], [14], that can be either static [5], [12] or dynamic [6], [13], [14] in the environment. Accurate tracking of multiple objects in the environment is particularly useful for collision

Received 15 October 2025; accepted 18 February 2026. Date of publication 13 March 2026; date of current version 23 March 2026. This article was recommended for publication by Associate Editor T. Kiyokawa and Editor O. Stasse upon evaluation of the reviewers' comments. This work was supported by NSF under Grant 2326390 and Grant CCF-2336612. *(Qingxi Meng and Emiliano Flores contributed equally to this work.) (Corresponding author: Qingxi Meng.)*

Qingxi Meng, Emiliano Flores, and Thai Duong are with the Department of Computer Science, Rice University, Houston, TX 77005 USA (e-mail: qm15@rice.edu).

Vaibhav Unhelkar and Lydia E. Kavraki are with the Department of Computer Science, Rice University, Houston, TX 77005 USA, and also with the , Department of Computer Science, Rice University, Houston, TX 77005 USA (e-mail: vaibhav.unhelkar@rice.edu; kavraki@rice.edu).

This article has supplementary downloadable material available at https://doi.org/10.1109/LRA.2026.3674013, provided by the authors.

Digital Object Identifier 10.1109/LRA.2026.3674013

avoidance, especially with aggressive quadrotor flights [5], [9], [10], or in cluttered and dynamic environments [6], [13], [15]. Besides objects, several works aim to maintain visibility of visual features for accurate state estimation [7], [8], such as visual-inertial odometry (VIO) [16], which is crucial for robot operations in the wild. Another exciting research direction is active mapping or exploration [11], where the robot trajectory is planned to explore the unmapped regions of the environment [17], [18], [19], either by choosing the next best goal state [17] or by maximizing the information gain of future sensor observations [18], [19].

The perception-aware constraints are commonly integrated as a cost, heuristic, or reward function in a motion planning problem, which is in turn, solved by an optimization solver [5], [13], a search-based [8] or sampling-based planner [7], [12], [20], or by reinforcement learning [10], [15]. Although successful in navigation with mobile robots, existing work on perception-aware motion planning largely focuses on settings with simplified robot models or with limited degrees of freedom (DoF). Closely related to our approach, PS-PRM [12] considers a perception-aware motion planning problem for a high-DoF robot but only monitors a known single object. Extending from monitoring a single object to multiple objects is nontrivial, as the planner must determine how to prioritize and achieve the correct viewpoints of the objects along the trajectory to maximize the overall user-defined perception score. Our work departs significantly from prior work on planning under perception constraints by developing a perception-aware sampling-based motion planner for high-DoF robots, e.g., mobile manipulators, that allows the robot to monitor multiple static objects of interest, stored in a scene graph built from sensor observations, while satisfying the kinematic constraints on the robot configuration.

Recently, metric-semantic maps [21], such as scene graphs [22], have emerged as powerful representations that unify geometric, semantic, and topological information for large-scale environments. Scene graphs organize semantic and metric information in a hierarchical structure, capturing relationships across abstraction layers. Scene graphs have been used for task planning and high-level reasoning, in combination with language models, e.g., SayPlan [23] and AutoGPT+P [24], or semantic instructions, e.g., GRID [25] and ConceptGraphs [26], or for explorations, e.g., RoboEXP [27].

As low-level motion planning requires geometric information and kinematic constraints to ensure the feasibility of a motion plan, recent work has explored the use of scene graphs for both task and motion planning [28], [29], [30] in a hierarchical manner. A "coarse" task plan is generated at the higher abstraction levels, such as buildings, rooms, or objects, and is then used to guide a local geometric planner at the occupancy level [28], [30], or to generate a heuristic function for a multi-heuristic A$*$ geometric planner [29]. However, these works focus on low-dimensional robot systems, e.g., robot or camera poses, without considering kinematic constraints.

Instead, we develop a sampling-based perception-aware probabilistic roadmap (PRM) planner for high-DoF robots, that integrates the robot's kinematic, geometric, and perception constraints, e.g., multi-object monitoring. The perception constraints are embedded with each object of interest in a scene graph as a perception cost function, which is used to inform the construction of a PRM. Given a robot configuration, the perception cost function describes the perception score of all objects of interest, that can be predefined or approximated by a neural network, pretrained to fit the confidence score of an object detection algorithm such as YOLOE [31]. For example, a high perception score or low perception cost is given if the camera pose, calculated via forward kinematics, leads to a clear view of multiple objects or humans in the camera image. We develop a perception-aware PRM graph construction by biasedly sampling robot configurations with low perception cost, i.e., high perception score. Given a start and a goal, an A$*$ search algorithm with our consistent heuristic design returns a robot path on the PRM that balances between the motion cost, representing the path's length or energy, and the perception cost, representing how well the robot can monitor the objects of interest along the path. We extensively validate our approach in both simulation and real-robot experiments. In summary, we propose a **M**ulti-**O**bject **P**erception-aware **S**cene-graph-based **P**robabilistic **R**oad**M**ap (MOPS-PRM) that:

- augments each object of interest in a scene graph with a learned perception costmap, specifying the preferable configuration regions for multi-object monitoring.
- constructs a perception-informed PRM on the configuration space of a high-DoF robot by selectively sampling nodes with low perception cost.
- generates a perception-aware trajectory with an A$*$ search on the perception-aware PRM.

II. Problem Statement

We consider a robot with configuration $\mathbf{q} \in \mathcal{C} = \mathcal{C}_{\text{free}} \cup \mathcal{C}_{\text{occupied}} \subseteq \mathbb{R}^n$, where n denotes the total number of degrees of freedom, including both the robot's base and its joints, and $\mathcal{C}_{\text{free}}$ and $\mathcal{C}_{\text{occupied}}$ are the free and occupied spaces, respectively. The robot operates in a workspace $\mathcal{W} \subseteq \mathbb{R}^3$ and is equipped with an onboard steerable RGB-D camera, controlled via the robot joints to observe the environment.

The goal of motion planning is to find a collision-free path $\boldsymbol{\pi} : [0, 1] \to \mathcal{C}_{\text{free}}$, from the start configuration $\boldsymbol{\pi}(0) = q_{\text{start}}$ to a goal region $\boldsymbol{\pi}(1) \in \mathcal{C}_{\text{goal}}$. Along a path $\boldsymbol{\pi}$, the robot also aims to monitor a set $\mathcal{O}$ of N objects of interest.

A motion cost $c_m(\boldsymbol{\pi})$, defined over the space of all possible paths Π, assigns a non-negative real number to each path $c_m : \Pi \to \mathbb{R}_{\geq 0}$, e.g., its length, energy or control effort. To model the object monitoring constraints, we introduce a perception cost function $f : \mathcal{C}_{\text{free}} \times \mathcal{O} \to \mathbb{R}_{\geq 0}$ that assigns each pair of configuration $\mathbf{q} \in \mathcal{C}_{\text{free}}$ and object $o \in \mathcal{O}$ a scalar value, measuring the quality of the observation of o by the onboard camera when the robot is at configuration $\mathbf{q}$. A lower perception quality implies a higher perception cost. We next define the overall perception cost at a configuration $\mathbf{q}$ as the weighted sum of the object-wise cost:

$$p(\mathbf{q}) = \sum_{o \in \mathcal{O}} w_o \, f(\mathbf{q}, o), \tag{1}$$

where the weight w_o is a user-defined importance of monitoring object o. The cumulative perception cost of a path is:

$$c_p(\boldsymbol{\pi}) = \int_0^1 p(\boldsymbol{\pi}(t))\, dt, \tag{2}$$

which aggregates the perception cost along the path $\boldsymbol{\pi}$. Our goal is to find the optimal path $\boldsymbol{\pi}^*$ that minimizes the weighted combination of motion and perception cost, *i.e.*,

$$\begin{aligned} \boldsymbol{\pi}^* &= \arg\min_{\boldsymbol{\pi}\in\Pi} c_m(\boldsymbol{\pi}) + \alpha c_p(\boldsymbol{\pi}), \\ \text{s.t.}\quad & \boldsymbol{\pi}(t) \in \mathcal{C}_{\text{free}} \quad \forall t \in [0,1], \\ & \boldsymbol{\pi}(0) = q_{\text{start}}, \boldsymbol{\pi}(1) \in \mathcal{C}_{\text{goal}}, \end{aligned} \tag{3}$$

where the weighting factor $\alpha \geq 0$ controls the trade-off between motion and perception cost.

III. Perception-Aware Planner With Multi-Object Monitoring Using Scene Graphs

An overview of MOPS-PRM is provided in Section III-A with details on how we develop our perception-informed PRM construction Section III-B, and how we augment a scene graph with the perception cost in Section III-C.

A. MOPS-PRM Planning

While finding the optimal trajectory in (3) is challenging due to the high-dimensional configuration space of high-DoF robots, our approach instead constructs a probabilistic roadmap (PRM) in the free space $\mathcal{C}_{\text{free}}$ and searches for an optimal path $\boldsymbol{\pi}$ on the PRM from the start to the goal.

We define the PRM $G = (V, E)$, where V is the set of P collision-free configurations and E the set of edges connecting them. The edges in E are checked for collisions with obstacles by a validity checker, e.g., [32], during our PRM construction. The nodes are generated by a "perception-aware" sampling scheme (Section III-B), connected to their k-nearest neighbors and checked for collision, creating a set of PRM edges (see lines 14-19 of Algorithm 1). Each edge $(\mathbf{q}_u, \mathbf{q}_v)$ is represented by a local motion $\boldsymbol{\pi}_{uv}(t)$, $\boldsymbol{\pi}_{uv}(0) = \mathbf{q}_u$, $\boldsymbol{\pi}_{uv}(1) = \mathbf{q}_v$, which depends on the kinematic constraints of the robot, e.g., a Reeds-Shepp curve for the base of a non-holonomic mobile manipulator. Each edge $(\mathbf{q}_u, \mathbf{q}_v)$ is assigned a cost:

$$c(\mathbf{q}_u, \mathbf{q}_v) = c_m(\boldsymbol{\pi}_{uv}) + \alpha \cdot c_p(\boldsymbol{\pi}_{uv}), \tag{4}$$

where $c_m(\boldsymbol{\pi}_{uv})$ is the edge's motion cost, e.g., the length or total control efforts of $\boldsymbol{\pi}_{uv}$, and $c_p(\boldsymbol{\pi}_{uv})$ is the edge's perception cost from (2):

$$c_p(\boldsymbol{\pi}_{uv}) = \int_0^1 \sum_{o\in\mathcal{O}} w_o\, f(\boldsymbol{\pi}(t), o)\, dt. \tag{5}$$

However, the perception cost $f(\mathbf{q}, o)$ for each pair $(\mathbf{q}, o)$ is typically unknown in advance for an arbitrary configuration $\mathbf{q}$. Therefore, we approximate the perception cost by a neural costmap $f_{\boldsymbol{\theta}}(\mathbf{q}, o)$ with parameter $\boldsymbol{\theta}$, trained on supervised data from an object detector and augmented to each object $o \in \mathcal{O}$ in a scene graph (see Section III-C). As a result, the edge's perception cost is approximated as:

$$c_p(\boldsymbol{\pi}_{uv}) \approx \sum_{k=0}^{K-1} \sum_{o\in\mathcal{O}} w_o\, f_{\boldsymbol{\theta}}(\boldsymbol{\pi}(t_k), o)\, \delta t, \quad K \geq 2, \tag{6}$$

where $t_0, t_1, \ldots, t_K$ are discrete times sampled along the edge with time steps $\delta t = \frac{1}{K}$.

After the PRM is constructed, MOPS-PRM connects the start and goal to the roadmap and applies an A$*$ search [33] with a consistent heuristic to find a path from q_{start} to $\mathcal{C}_{\text{goal}}$ for the robot to follow. Following [12], we define a consistent "hop"-based heuristic function that lower-bounds the remaining path cost to the goal:

$$h(\mathbf{q}) \;=\; H_{\min} \cdot c^{\min}, \tag{7}$$

where $H_{\min}(\mathbf{q})$ denotes the hop distance, i.e., the minimum number of edges required to reach the goal from the configuration $\mathbf{q}$ obtained via a shortest-path search, and $c^{\min}$ represents the minimum edge cost over the entire roadmap:

$$c^{\min} = \min_{(u,v)\in E} c(\mathbf{q}_u, \mathbf{q}_v). \tag{8}$$

For any $(u, v) \in E$, a feasible path from u to the goal is to take (u, v) and then follow a shortest path from v to the goal. Thus, we have: $H_{\min}(\mathbf{q}_u) \leq 1 + H_{\min}(\mathbf{q}_v)$.

Therefore, our heuristic function $h(\cdot)$ is consistent:

$$\begin{aligned} h(\mathbf{q}_u) &\leq (1 + H_{\min}(\mathbf{q}_v)) \cdot c^{\min} \\ &\leq c(\mathbf{q}_u, \mathbf{q}_v) + h(\mathbf{q}_v). \qquad \text{by Eqs. (7) and (8)} \end{aligned}$$

This formulation only requires nonnegativity of edge costs, without assuming a specific form such as Euclidean distance for motion cost. It applies broadly, e.g., when c_m is defined as the trajectory length, energy, or control effort, and c_p is a non-negative cost derived from neural perception scores. The weight α controls the tradeoff between the motion and perception costs, e.g., the higher the weight α is, the longer path the A$*$ search might return and vice versa.

The solution returned solves (3) only for trajectories that lie on the PRM. However, as the number of nodes increases, the solution asymptotically converges to the true optimal trajectory. MOPS-PRM is illustrated in Algorithm 1 with implementation details provided in Section IV-A.

B. Perception-Aware Sampling

An important subroutine in MOPS-PRM is to sample a set of nodes for roadmap construction. As the configuration space is high-dimensional, it is beneficial to bias the sampling process towards regions with low perception cost. Given a sampled configuration $\mathbf{q}_0 \in \mathcal{C}_{\text{free}}$, we would like to find a nearby $\mathbf{q}^*$ that minimizes the perception cost:

$$\mathbf{q}^* = \arg\min_{\mathbf{q}\in\mathcal{C}_{free}} \left(p(\mathbf{q}) + \lambda \|\mathbf{q} - \mathbf{q}_0\|_2^2 \right), \tag{9}$$

where $p(\mathbf{q})$ is the perception cost in (1) and the distance $\|\mathbf{q} - \mathbf{q}_0\|_2^2$ is a regularization term with coefficient λ to penalize large deviation from $\mathbf{q}_0$. However, solving (9) exactly is challenging

Algorithm 1: MOPS-PRM Construction.

Input: Set of objects of interest $\mathcal{O}$, neural costmap $f_{\boldsymbol{\theta}}$, scene graph S, number of nodes P
Output: Our perception-aware PRM $G = (V, E)$

1 $V, E \leftarrow \emptyset$
2 **while** $|V| < P$ **do**
  // Stage 1: Sampling & projection
3 $\mathbf{q}_0 \leftarrow \textsc{Sample}(\mathcal{C}_{\text{free}})$
4 $\mathcal{C}_{\mathbf{q}_0} \leftarrow \emptyset$
5 $\mathcal{X}_{\mathcal{O}} \leftarrow \textsc{ExtractCentroids}(\mathcal{O})$
6 **foreach** c *in* $\mathcal{X}_{\mathcal{O}}$ **do**
7 $\mathbf{q}_0^c \leftarrow \textsc{Project}(\mathbf{q}_0, c)$ (Eq. (12))
  // Stage 2: Local sampling
8 $\mathcal{C}^c_{\text{local}} \leftarrow \textsc{SampleLocal}(\mathbf{q}_0^c)$
9 **foreach** $\mathbf{q}^c_{\text{loc}} \in \mathcal{C}^c_{\text{local}}$ **do**
  // Check field of view (FOV)
10 **if** $c \in \text{FOV}(\mathbf{q}^c_{\text{loc}})$ **then**
11 $\mathcal{C}_{\mathbf{q}_0} \leftarrow \mathcal{C}_{\mathbf{q}_0} \cup \{\mathbf{q}^c_{\text{loc}}\}$
  // Neural costmap & selection
12 $\mathbf{q}_{\text{node}} \leftarrow \arg\min_{\mathbf{q} \in \mathcal{C}_{\mathbf{q}_0}} p(\mathbf{q})$ (Eq. (10))
13 $V \leftarrow V \cup \{\mathbf{q}_{\text{node}}\}$
  // Edge connection
14 **foreach** $\mathbf{q}_{\text{near}} \in \textsc{NearestNeighbors}(\mathbf{q}_{\text{node}})$ **do**
15 $\boldsymbol{\pi}_{loc} \leftarrow \textsc{LocalMotion}(\mathbf{q}_{\text{node}}, \mathbf{q}_{\text{near}})$
16 **if** $\textsc{IsCollisionFree}(\boldsymbol{\pi}_{loc}, S)$ **then**
17 $c_p \leftarrow \textsc{PerceptionCost}(f_{\boldsymbol{\theta}}, \mathcal{O})$ (Eq. (6))
18 $c \leftarrow c_m(\boldsymbol{\pi}_{loc}) + \alpha\, c_p(\boldsymbol{\pi}_{loc})$ (Eq. (4))
19 $E \leftarrow E \cup \{(\mathbf{q}_{\text{node}}, \mathbf{q}_{\text{near}}, c)\}$
20 **return** (V, E)

for our PRM construction, as the perception cost is approximated by:

$$p(\mathbf{q}) \approx \sum_{o \in \mathcal{O}} w_o\, f_{\boldsymbol{\theta}}(\mathbf{q}, o), \tag{10}$$

with a nonlinear neural costmap $f_{\boldsymbol{\theta}}(\mathbf{q}, o)$. Instead, we introduce a perception-aware local sampling scheme that empirically approximates $\mathbf{q}^*$ in two stages, as outlined in Algorithm 1. The first stage (lines 1–7 of Algorithm 1) projects the sampled configuration $\mathbf{q}_0$ on a constrained manifold, where the camera pose points toward the objects. The second stage (lines 8–12) performs local sampling around each projection and selects the configuration with the lowest perception cost.

In the first stage, given the sample $\mathbf{q}_0$, we calculate the camera pose via forward kinematics, and generate viewpoint candidates by projecting the camera optical axis toward a set $\mathcal{X}_{\mathcal{O}}$ of $(N + \binom{N}{2} + 1)$ desired centroids, consisting of the centroid of each object $o \in \mathcal{O}$, the centroid of each object pair, and the centroid of all objects collectively. This captures common cases where potentially the best viewpoint either focuses on observing a single object, all objects, or the transition between a pair of objects.

After experimentation, we observed that lower perception costs are obtained when the camera's optical axis is aligned with the object centroid (as illustrated in the "Multi-Object Constrained Sampling" block in Fig. 2). Let $\mathbf{m}(\mathbf{q}) \in \mathbb{R}^3$ denote the camera center, $\mathbf{z}(\mathbf{q}) \in \mathbb{S}^2$ the unit optical axis, and $\mathbf{x}_c \in \mathbb{R}^3$ the 3-D coordinates of the desired centroid $c \in \mathcal{X}_{\mathcal{O}}$. We define the lateral (image-plane) projection residual as

$$\boldsymbol{\phi}(\mathbf{q}, c) := \left(\mathbf{I}_3 - \mathbf{z}(\mathbf{q})\mathbf{z}(\mathbf{q})^\top\right)\left(\mathbf{x}_c - \mathbf{m}(\mathbf{q})\right) \in \mathbb{R}^3, \tag{11}$$

where $\mathbf{I}_3$ is the 3×3 identity matrix [34]. The matrix $\mathbf{P}(\mathbf{q}) = \mathbf{I}_3 - \mathbf{z}(\mathbf{q})\mathbf{z}(\mathbf{q})^\top$ is the orthogonal projector onto the tangent plane of $\mathbb{S}^2$ at $\mathbf{z}(\mathbf{q})$, i.e., $\mathbf{z}(\mathbf{q})^\top \boldsymbol{\phi}(\mathbf{q}, c) = 0$ and the residual $\boldsymbol{\phi}(\mathbf{q}, c)$ has only two degrees of freedom corresponding to the lateral error in the image plane. We then project $\mathbf{q}_0$ onto a configuration, whose camera pose aligns with the centroid $c \in \mathcal{X}_{\mathcal{O}}$ by solving:

$$\mathbf{q}_0^c = \arg\min_{\mathbf{q} \in \mathbb{R}^k} \|\boldsymbol{\phi}(\mathbf{q}, c)\|^2 + \lambda \|\mathbf{q} - \mathbf{q}_0\|_2^2$$
$$\text{s.t.} \quad \mathbf{q} \in \mathcal{C}_{\text{free}},\ \|\mathbf{q} - \mathbf{q}_0\|_2 \le \rho, \tag{12}$$

where $\lambda \ge 0$ is a regularization weight that encourages the solution to be close to $\mathbf{q}_0$, and $\rho > 0$ is an optional trust-region radius restricting the projection to a ball around $\mathbf{q}_0$. The parameters λ and ρ allow us to balance between the PRM coverage of the configuration space and biased sampling towards regions with low perception cost. For a large λ/small ρ, the projected point $\mathbf{q}_0^c$ stays close to the uniformly sampled $\mathbf{q}_0$, encouraging more even coverage of the configuration space. For a small λ/large ρ, the projected point $\mathbf{q}_0^c$ tends to be biased towards regions with low perception cost. The projection problem (12) can be solved efficiently via gradient descent, e.g., using an L-BFGS-B solver [35]. If the optimization does not converge within the iteration limit, we discard the sample $\mathbf{q}_0$ and obtain a new one.

In the second stage, we sample M configurations $\{\mathbf{q}^c_{0(i)}\}_{i=1}^M$ around each projected $\mathbf{q}_0^c$ by adding a zero-mean Gaussian noise $\mathbf{n} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma})$ so that the corresponding camera's field of view (FOV) will still capture the centroid $c \in \mathcal{X}_{\mathcal{O}}$. This process generates a set of $(N + \binom{N}{2} + 1)(M + 1)$ candidates: $\mathcal{C}_{\mathbf{q}_0} = \{\bigcup_{c \in \mathcal{X}_{\mathcal{O}}} \{\mathbf{q}^c_{0(i)}\}_{i=1}^M \cup \{\mathbf{q}_0^c\}\}$. The perception cost of all candidates in $\mathcal{C}_{\mathbf{q}_0}$ is calculated via (10) efficiently in parallel using the neural costmap $f_{\boldsymbol{\theta}}(\mathbf{q}, o)$ (see Section III-C). The candidate with the lowest perception cost: $\mathbf{q}_{\text{node}} = \arg\min_{\mathbf{q} \in \mathcal{C}_{\mathbf{q}_0}} p(\mathbf{q})$, is added to our perception-aware PRM (lines 12-13 in Algorithm 1 and "Local Sampling" block in Fig. 2).

C. Embedding Perception Costs in Scene Graphs

Many perception constraints can be characterized by a scalar score s, such as the confidence value output $s \in [0, 1]$ of an object detection model, *i.e.*, YOLOE [31], which can be converted to a perception cost l, e.g., $\ell = 1 - s$ or $\ell = 1/s$. To efficiently query the cost during PRM construction, we train a neural network $f_{\boldsymbol{\theta}}(q, o)$ that predicts the perception cost of a pair of configuration $\mathbf{q} \in \mathcal{C}_{\text{free}}$ and object $o \in \mathcal{O}$. The network $f_{\boldsymbol{\theta}}(q, o)$ is pre-trained on a wide range of common objects, e.g., representative objects in a home or a hospital, and can be used across different environments. We consider the robot's forward kinematics as a non-trainable first layer of the neural network, calculating the camera pose from the configuration $\mathbf{q}$. The input of the second layer is the relative pose between the robot's

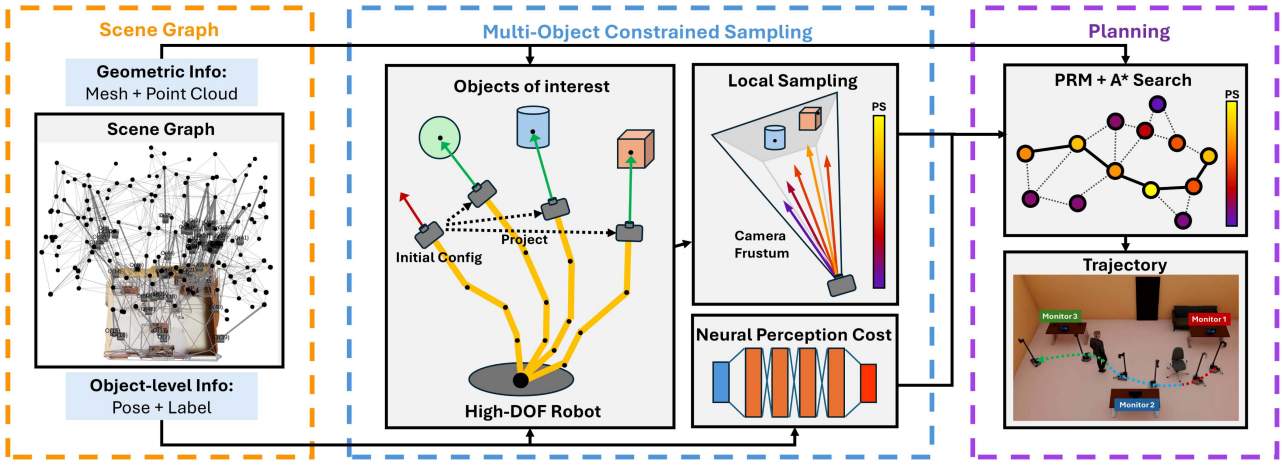

Fig. 2. This figure presents the pipeline of our planner. The planner takes the scene graph as input, combining geometric and object-level information with a neural perception cost function to perform multi-object constrained sampling. Sampling is performed (see Section III-B) to construct a PRM, which is searched using A$*$ to generate a trajectory that effectively accomplishes perception tasks involving multiple objects along the path.

onboard camera and the object o, together with an encoding of the object's semantic class, such as *monitor* or *human*. This is followed by a neural network, such as a multi-layer perceptron (MLP), that outputs an estimate of the perception cost $f(\mathbf{q}, o)$.

The training dataset $\mathcal{D} = \{(\mathbf{q}_i, o_i, s_i)\}_{i=1}^{D}$ is generated using a task-specific perception model, which provides the perception score s_i for each robot-object pair (q_i, o_i). The neural costmap $f_{\boldsymbol{\theta}}$ is trained via supervised learning to fit the dataset $\mathcal{D}$, enabling batched parallel evaluation of perception costs. In practice, not all objects in the scene graph are assigned a perception costmap; only those important to monitor are given this attribute.

IV. Experimental Results

In our experiments, we verify the effectiveness of our multi-object perception-aware scene-graph-based PRM with simulated and real-robot experiments using the Hello Robot's Stretch 2 [36] and Isaac Sim [37] for simulation and visualization. The Stretch 2 is a high-DoF mobile manipulator: in our experiments, we control its differential-drive, non-holonomic base (3 DoF) together with the pan–tilt joints of the onboard camera, introducing nontrivial kinematic constraints, leading to a challenging perception-aware planning problem. All experiments were conducted on an Intel i7-12700 K CPU and an NVIDIA GeForce RTX4090 GPU.

A. Implementation Details

While MOPS-PRM planner can admit different motion and perception costs, and different forms of scene graphs, we present the specific implementation choices that we used.

The motion cost $c_m(\boldsymbol{\pi})$ of a path $\boldsymbol{\pi}$ is computed as the sum of the Euclidean distances of all individual edges along the path. For the A$*$ heuristic in (7), we use the Euclidean distance between the current configuration $\mathbf{q}$ and the goal as the motion component. Meanwhile, the perception cost label ℓ, used to train our neural costmap in Section III-C, is chosen as a quadratic function $(1-s)^2$, where s is the confidence score provided by the object detector YOLOE [31]. We chose this perception cost to emphasize on higher confidence scores, guiding the planner to favor views that yield more reliable detections. As the motion cost and the perception cost have different units and ranges, we normalize both costs to the range [0,1] using their minimum and maximum values over the entire roadmap, easing parameter tuning for the weight α.

For perception-aware sampling in Section III-B, an L-BFGS-B solver [35] is used to solve (12) with parameters $\rho = 0.05$, $\lambda = 0.3$, and the maximum iterations set to 100. For the local sampling function in the second stage, we choose $M = 5$ and use a Gaussian noise $\mathbf{n} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$. For the PRM nearest-neighbor selection in line 14 of Algorithm 1, the number of neighbors is set to be 5, empirically balancing graph connectivity and computational efficiency.

We build our scene graph from camera images using the Khronos framework [38]. At the lowest layer, the scene graph contains a semantically annotated mesh of the environment geometry, representing obstacles in the environment, which we convert into a parallelization-friendly CAPT point cloud [39], allowing us to perform collision checking efficiently using fine-grained parallelism.

The neural cost function in Section III-C is implemented as a multilayer perceptron (MLP) [40] with five fully connected layers of 256 units and ReLU activations. For the training dataset, we uniformly sample 50000 robot-independent camera poses in Isaac Sim that keep the object in view. These viewpoints are not tied to a specific robot configuration and can be used with any high-DoF platform via forward kinematics. We then render the corresponding images and evaluate perception costs using the YOLOE [31] model. The neural cost function is trained on these perception costs across diverse objects and humans from the COCO dataset [41], ensuring applicability to both real and simulated experiments.

B. Multi-Object Detection in a Simulated Office

We evaluate our approach in a simulated environment containing objects commonly found in an office, such as tables, chairs,

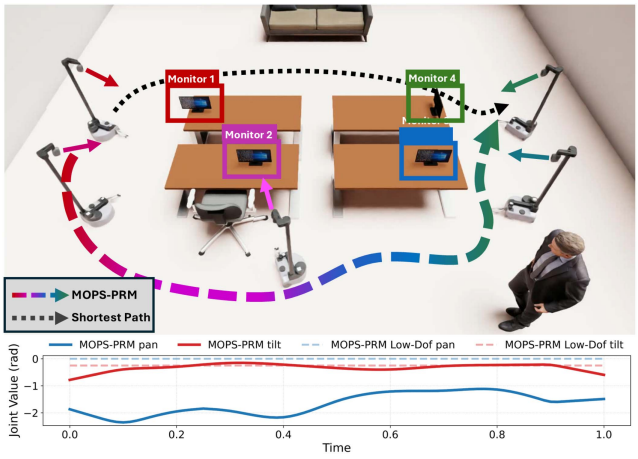

Fig. 3. In our simulated benchmarks, the robot moves from a start to a goal in an office environment while monitoring the four screens of the monitors placed on the table. The robot takes the longer path to observe the monitors, where the arrows illustrate the camera orientations. The bottom plot shows the camera pan-tilt joint angles along the trajectory.

humans, and monitors (an example is shown in Fig. 3). This setting reflects typical scenarios faced by robotic assistants in office environments, where the robot must perform navigation or delivery tasks while monitoring multiple task-relevant objects, such as screens or humans. The task is to plan collision-free motions from a start to a goal while ensuring that the robot maintains visibility of monitors placed around the environment. To generate test cases, we sample 100 motion planning problems by selecting random collision-free start and goal configurations on opposite sides of the room, ensuring that the robot must traverse the environment while balancing motion and perception costs. As illustrated in Fig. 3, we place the objects of interest in physically plausible locations (e.g., resting on a surface rather than floating in the air) to create realistic and meaningful scenarios for evaluation.

We compare MOPS-PRM against three baselines: "Closest-Object Low-DoF", "Closest-Object", and "Lowest-Cost-Object". At a configuration $\mathbf{q}$, "Closest-Object Low-DoF" and "Closest-Object" always monitor the nearest object by projecting the camera view toward it, while "Lowest-Cost-Object" selects the object with the lowest perception cost as evaluated by the same neural cost function used in MOPS-PRM. In "Closest-Object Low-DoF", planning is restricted to the non-holonomic base with all other joints fixed, resembling perception-aware planning for aerial or ground robots. A comparison on the movement of camera joints is included in Fig. 3, with sample paths from the same environment. Both "Closest-Object Low-DoF" and "Closest-Object" define the perception cost as the distance to the selected object, whereas "Lowest-Cost-Object" instead uses the neural cost function of MOPS-PRM. We set time limits for PRM construction, adapted from OMPL [42], to ensure similar number of nodes across all methods: 5 seconds for "Closest-Object Low-DoF" and "Closest-Object", and 30 seconds for "Lowest-Cost-Object" and MOPS-PRM.

Perception performance is evaluated using YOLOE [31] for object detection and Deep SORT [43] for tracking, whose "track rate" metric describes the benefits of continuously monitoring multiple objects beyond detection. One key metric is the average number of objects detected per frame, $\overline{D}$, which measures frame-to-frame visual coverage by averaging the number of detected objects across all frames along a trajectory. Detection confidence is measured in two forms: the average confidence $\overline{C}$, computed as the mean confidence score s_i over all successful detections in the set S, and the scaled average confidence $\overline{C}_{\text{sc}} = \overline{D}\,\overline{C}$, which emphasizes the ability to maintain both high-confidence detections and consistent multi-object coverage along the trajectory.

Table I summarizes the results across the 100 planning problems. Both "Closest-Object Low-DoF" and "Closest-Object" incur lower computational overhead, as reflected in their significantly faster PRM construction and planning times. However, their strict focus on the nearest object leads to reduced coverage, evident from a lower average number of objects detected per frame $\overline{D}$. Their confidence metrics also lag behind, since they do not account for accurate perception cost estimates from each robot configuration. Meanwhile, the 'Lowest-Cost-Object" baseline achieves confidence scores comparable to MOPS-PRM by leveraging the neural cost function, but its inability to consider multiple objects simultaneously results in our method achieving more than $\sim 36\%$ improvement in the average number of detected objects per frame and a $\sim 17\%$ higher track rate. For this "track rate" metric, MOPS-PRM achieves the highest performance clearly surpassing all baselines. This highlights the advantage of continuously monitoring multiple objects beyond mere single-frame detection. While the baselines may achieve slightly shorter planning times or path lengths by focusing on an object at a time, MOPS-PRM explicitly accounts for multi-object monitoring, and hence, substantially improves perception performance.

Fig. 5 illustrates how our planner's performance scales with the PRM size and the number of objects. With the number of objects fixed at 5, increasing the PRM size increases roadmap construction time, while planning time remains low, typically around or below one second. The average number of detections per frame also increases, indicating that a denser roadmap supports more stable perception quality. When the PRM size is fixed at roughly 300 nodes, adding more objects drives up construction time and modestly increases planning time. At the same time, perception performance improves as the number of PRM nodes or objects increases, as reflected in higher average detections per frame. While the PRM construction is time-consuming, it only occurs once, and can be reused multiple times for path generation. The results suggest that our approach remains practical as the problem size increases, with most of the overhead concentrated in the one-time construction stage.

C. Real Robot Experiments

As shown in Fig. 4, the robot is placed in an indoor environment and is tasked to move from a starting position shown in red to the corner of the room shown in green. To reach this goal, the robot must pass through a narrow passage created by an intervening chair, resulting in a challenging scenario with multi-modal solutions and high collision risks.

TABLE I

THIS TABLE COMPARES THE PERFORMANCE OF OUR METHOD WITH THREE BASELINES INTRODUCED IN SECTION IV. FOR PLANNING METRICS, *BUILD TIME*, *PLAN. TIME*, AND *PATH LEN.* DENOTE THE AVERAGE PRM CONSTRUCTION, PLANNING TIME, AND THE AVERAGE PATH LENGTH, RESPECTIVELY. FOR PERCEPTION METRICS, *AVG. DET. OBJ.*, *TRACK RATE*, *AVG. CONF.*, AND *SCALED AVG. CONF.* DENOTE THE AVERAGE NUMBER OF DETECTED OBJECTS PER FRAME, THE TRACKER UPDATE SUCCESS RATE, THE AVERAGE DETECTION CONFIDENCE SCORE, AND THE CONFIDENCE SCORE SCALED BY THE AVERAGE NUMBER OF DETECTED OBJECTS, RESPECTIVELY

Method	Motion Planning				Perception			
	Cost Fun.	Build Time (s)	Plan. Time (s)	Path Len.	Avg. Det. Obj. (0–4)	Track Rate (0–1)	Avg. Conf. (0–1)	Scaled Avg. Conf. (0–4)
Closest-Object Low-DoF	Distance	5.0	0.22	16.63	0.44	0.72	0.45	0.20
Closest-Object	Distance	5.0	0.08	15.66	0.50	0.65	0.42	0.21
Lowest-Cost-Object	Neural	30.0	0.95	19.07	1.12	0.76	0.48	0.54
MOPS-PRM (Ours)	Neural	30.0	0.98	20.42	**1.53**	**0.89**	**0.49**	**0.75**

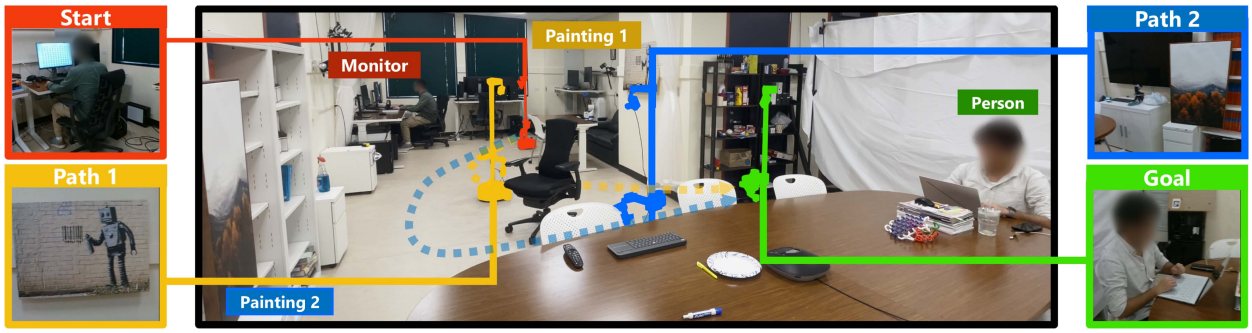

Fig. 4. In this real-robot experiment, the robot plans two different paths with the same start (shown in red) and goal (shown in green) based on a user-specified importance of the two paintings: the yellow path prioritizes painting 1, while the blue path prioritizes painting 2. Both paths start by observing a human with a monitor and end by looking at another human sitting at the table while the middle sections of the paths differ as they prioritize observing different paintings.

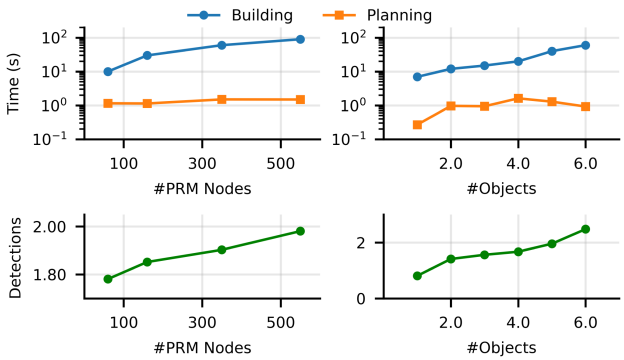

Fig. 5. Performance of MOPS-PRM under varying number of objects and PRM sizes. In the first column, the number of objects is fixed at five. In the second, PRM size is approximately 300 nodes. We report the planning and PRM construction times, and the average number of detections per frame.

Unlike the simulation experiments, this setup introduces a different challenge, since the objects are farther apart and facing different directions. While monitoring multiple objects simultaneously, the robot must transition its focus between different objects of interest while maintaining smooth motion and maximizing perception scores along the trajectory.

The robot is tasked to consider the monitor near its starting position and a person near the end position, while detecting one of two objects, either a robot painting (painting 1) or a landscape painting (painting 2) placed on a cabinet, as shown in Fig. 4. The priority of monitoring each object is encoded by user-defined weights, as described in Section III-A.

Fig. 4 illustrates how MOPS-PRM generates trajectories based on which painting is prioritized. The two resulting paths are shown in yellow and blue, each highlighting a representative robot configuration along the path corresponding to the case where the respective painting is given higher weight. Averaged over 100 runs, our planner takes around 1.64 seconds to generate each plan and takes around 30.0 seconds to build the PRM. The path length for this experiment, measured as the Euclidean distance at all 5-DoF edges of the path, is around 19.13 for the yellow path and around 18.10 for the blue path. The average number of objects detected in each frame is around 0.84 on the yellow path and 0.81 on the blue path. This experiment demonstrates the planner's ability to monitor multiple objects while respecting the assigned weights of each object of interest.

V. DISCUSSION

We develop MOPS-PRM, a roadmap-based perception-aware motion planner for high-DoF robots tasked with multi-object monitoring. Perception awareness is modeled via a costmap anchored to objects of interest in a scene graph, guiding the perception-aware PRM construction and A$*$ search to produce paths that balance motion efficiency with perception quality. This enables applications such as museum patrol, patient monitoring, or industrial inspection where robots must move

efficiently while maintaining visibility of key objects. As scene graphs have shown tremendous potential for task and motion planning in semantically rich environments, our work serves as a first step towards perception-aware task and motion planning for high-DoF robots, and can be further extended to leverage the scene graph's topology for perception-aware task-level reasoning. Besides, we also aim to extend the framework to include tree-based planners (e.g., RRT variants), handle map uncertainty for more robust planning, explore perception-aware motion planning in dynamic and interactive environments, and handle previously unseen objects, e.g., by incorporating open-set object detection [44].

REFERENCES

[1] A. Bolu and Ö. Korçak, "Adaptive task planning for multi-robot smart warehouse," *IEEE Access*, vol. 9, pp. 27346–27358, 2021.

[2] P. Qian et al., "ASTRID: A robotic tutor for nurse training to reduce healthcare-associated infections," in *Proc. Robot.: Sci. Syst.*, 2025.

[3] Y. Liao, J. Xie, and A. Geiger, "KITTI-360: A novel dataset and benchmarks for urban scene understanding in 2D and 3D," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 45, no. 3, pp. 3292–3310, Mar. 2023.

[4] N. M. M. Shafiullah et al., "On bringing robots home," 2023, *arXiv:2311.16098*.

[5] D. Falanga, P. Foehn, P. Lu, and D. Scaramuzza, "PAMPC: Perception-aware model predictive control for quadrotors," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2018, pp. 1–8.

[6] H. Masnavi, A. K. Singh, and F. Janabi-Sharifi, "Differentiable-optimization based neural policy for occlusion-aware target tracking," *IEEE Robot. Automat. Lett.*, vol. 9, no. 12, pp. 11714–11721, Dec. 2024.

[7] B. Ichter, B. Landry, E. Schmerling, and M. Pavone, "Perception-aware motion planning via multiobjective search on GPUs," in *Robotics Research*, Cham, Switzerland: Springer International Publishing, 2020, pp. 895–912.

[8] L. Bartolomei, L. Teixeira, and M. Chli, "Perception-aware path planning for UAVs using semantic segmentation," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2020, pp. 5808–5815.

[9] A. Loquercio, E. Kaufmann, R. Ranftl, M. Müller, V. Koltun, and D. Scaramuzza, "Learning high-speed flight in the wild," *Sci. Robot.*, vol. 6, no. 59, 2021, Art. no. eabg5810.

[10] Y. Song, K. Shi, R. Penicka, and D. Scaramuzza, "Learning perception-aware agile flight in cluttered environments," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2023, pp. 1989–1995.

[11] J. A. Placed et al., "A survey on active simultaneous localization and mapping: State of the art and new frontiers," *IEEE Trans. Robot.*, vol. 39, no. 3, pp. 1686–1705, Jun. 2023.

[12] Q. Meng et al., "Look as you leap: Planning simultaneous motion and perception for high-DoF robots," 2025, *arXiv:2509.19610*.

[13] J. Tordesillas and J. P. How, "PANTHER: Perception-aware trajectory planner in dynamic environments," *IEEE Access*, vol. 10, pp. 22662–22677, 2022.

[14] H. Zhang, J. Huo, Y. Huang, J. Cheng, and X. Li, "Perception-aware based UAV trajectory planner via generative adversarial self-imitation learning from demonstrations," *IEEE Internet Things J.*, vol. 12, no. 3, pp. 3248–3260, Feb. 2025.

[15] A. Singla, S. Padakandla, and S. Bhatnagar, "Memory-based deep reinforcement learning for obstacle avoidance in UAV with limited environment knowledge," *IEEE Trans. Intell. Transp. Syst.*, vol. 22, no. 1, pp. 107–118, Jan. 2021.

[16] A. I. Mourikis and S. I. Roumeliotis, "A multi-state constraint Kalman filter for vision-aided inertial navigation," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2007, pp. 3565–3572.

[17] A. Bircher, M. Kamel, K. Alexis, H. Oleynikova, and R. Siegwart, "Receding horizon "next-best-view" planner for 3D exploration," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2016, pp. 1462–1468.

[18] Z. Zhang, T. Henderson, S. Karaman, and V. Sze, "FSMI: Fast computation of Shannon mutual information for information-theoretic mapping," *Int. J. Robot. Res.*, vol. 39, no. 9, pp. 1155–1177, 2020.

[19] A. Asgharivaskasi and N. Atanasov, "Semantic OcTree mapping and Shannon mutual information computation for robot exploration," *IEEE Trans. Robot.*, vol. 39, no. 3, pp. 1910–1928, Jun. 2023.

[20] G. Costante, C. Forster, J. Delmerico, P. Valigi, and D. Scaramuzza, "Perception-aware path planning," 2016, *arXiv:1605.04151*.

[21] M. B. Alatise and G. P. Hancke, "A review on challenges of autonomous mobile robot and sensor fusion methods," *IEEE Access*, vol. 8, pp. 39830–39846, 2020.

[22] X. Chang, P. Ren, P. Xu, Z. Li, X. Chen, and A. Hauptmann, "A comprehensive survey of scene graphs: Generation and application," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 45, no. 1, pp. 1–26, Jan. 2023.

[23] K. Rana, J. Haviland, S. Garg, J. Abou-Chakra, I. Reid, and N. Suenderhauf, "SayPlan: Grounding large language models using 3D scene graphs for scalable robot task planning," in *Proc. Conf. Robot Learn.*, 2023, pp. 23–72.

[24] T. Birr, C. Pohl, A. Younes, and T. Asfour, "AutoGPT P: Affordance-based task planning with large language models," in *Proc. Robot.: Sci. Syst.*, 2024.

[25] Z. Ni et al., "GRID: Scene-graph-based instruction-driven robotic task planning," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2024, pp. 13765–13772.

[26] Q. Gu et al., "ConceptGraphs: Open-vocabulary 3D scene graphs for perception and planning," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2024, pp. 5021–5028.

[27] H. Jiang et al., "RoboEXP: Action-conditioned scene graph via interactive exploration for robotic manipulation," in *Proc. Conf. Robot Learn.*, 2025, pp. 3027–3052.

[28] A. Ray, C. Bradley, L. Carlone, and N. Roy, "Task and motion planning in hierarchical 3D scene graphs," in *Proc. Int. Symp. Robot. Res.*, 2024.

[29] Z. Dai et al., "Optimal scene graph planning with large language model guidance," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2024, pp. 14062–14069.

[30] V. K. Viswanathan, A. Patel, M. A. Saucedo, S. G. Satpute, C. Kanellakis, and G. Nikolakopoulos, "SPADE: Towards scalable path planning architecture on actionable multi-domain 3D scene graphs," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2025, pp. 17510–17517.

[31] A. Wang, L. Liu, H. Chen, Z. Lin, J. Han, and G. Ding, "YOLOE: Real-time seeing anything," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2025, pp. 24591–24602.

[32] W. Thomason, Z. Kingston, and L. E. Kavraki, "Motions in microseconds via vectorized sampling-based planning," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2024, pp. 8749–8756. [Online]. Available: https://ieeexplore.ieee.org/document/10611190

[33] P. E. Hart, N. J. Nilsson, and B. Raphael, "A formal basis for the heuristic determination of minimum cost paths," *IEEE Trans. Syst. Sci. Cybern.*, vol. 4, no. 2, pp. 100–107, Jul. 1968.

[34] Y. Ma, S. Soatto, J. Košecká, and S. Sastry, *An Invitation to 3D Vision: From Images to Geometric Models*, vol. 26. Berlin, Germany: Springer, 2004.

[35] C. Zhu, R. H. Byrd, P. Lu, and J. Nocedal, "Algorithm 778: L-BFGS-B: Fortran subroutines for large-scale bound-constrained optimization," *ACM Trans. Math. Softw.*, vol. 23, no. 4, pp. 550–560, Dec. 1997.

[36] C. C. Kemp, A. Edsinger, H. M. Clever, and B. Matulevich, "The design of stretch: A compact, lightweight mobile manipulator for indoor human environments," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2022, pp. 3150–3157.

[37] NVIDIA, "NVIDIA ISAAC Sim," 2022. Accessed: Aug. 08, 2024. [Online]. Available: https://developer.nvidia.com/isaac-sim

[38] L. Schmid, M. Abate, Y. Chang, and L. Carlone, "Khronos: A unified approach for spatio-temporal metric-semantic SLAM in dynamic environments," in *Proc. Robot.: Sci. Syst.*, 2024.

[39] C. W. Ramsey, Z. Kingston, W. Thomason, and L. E. Kavraki, "Collision-affording point trees: SIMD-Amenable nearest neighbors for fast collision checking," in *Proc. Robot.: Sci. Syst.*, 2024.

[40] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *Nature*, vol. 323, no. 6088, pp. 533–536, 1986.

[41] T.-Y. Lin et al., "Microsoft COCO: Common objects in context," in *Proc. 13th Eur. Conf. Comput. Vis.*, Zurich, Switzerland, Springer, 2014, pp. 740–755.

[42] I. A. Şucan, M. Moll, and L. E. Kavraki, "The open motion planning library," *IEEE Robot. Automat. Mag.*, vol. 19, no. 4, pp. 72–82, Dec. 2012.

[43] N. Wojke, A. Bewley, and D. Paulus, "Simple online and realtime tracking with a deep association metric," in *Proc. IEEE Int. Conf. Image Process.*, 2017, pp. 3645–3649.

[44] S. Liu et al., "Grounding DINO: Marrying DINO with grounded pre-training for open-set object detection," in *Proc. Eur. Conf. Comput. Vis.*, Springer, 2024, pp. 38–55.