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

Teaching Robots Where To Go And How To Act With Human Sketches via Spatial Diagrammatic Instructions

Qilin Sun[1,*] Weiming Zhi[1,*] Tianyi Zhang [1] Matthew Johnson-Roberson [1]

***Abstract*— This paper introduces *Spatial Diagrammatic Instructions* (SDIs), an approach for human operators to specify objectives and constraints that are related to spatial regions in the working environment. Human operators are enabled to sketch out regions directly on camera images that correspond to the objectives and constraints. These sketches are projected to 3D spatial coordinates, and continuous *Spatial Instruction Maps* (SIMs) are learned upon them. These maps can then be integrated into optimization problems for tasks of robots. In particular, we demonstrate how Spatial Diagrammatic Instructions can be applied to solve the Base Placement Problem of mobile manipulators, which concerns the best place to put the manipulator to facilitate a certain task. Human operators can specify, via sketch, spatial regions of interest for a manipulation task and permissible regions for the mobile manipulator to be at. Then, an optimization problem that maximizes the manipulator's reachability, or coverage, over the designated regions of interest while remaining in the permissible regions is solved. We provide extensive empirical evaluations, and show that our formulation of Spatial Instruction Maps provides accurate representations of user-specified diagrammatic instructions. Furthermore, we demonstrate that our diagrammatic approach to the Mobile Base Placement Problem enables higher quality solutions and faster runtime.**

I. Introduction

We envision a future where human operators work jointly with robots. To achieve this, human operators need to be able to readily deliver instructions to the robot. These inputs can take many modalities, including language instructions or kinesthetic demonstrations. Each of these modalities has its benefits and limitations. Language is a natural and humanesque way to provide commands to a robot, but would require brittle pipelines to combine the language input with semantic information from its perception system. Kinesthetic demonstrations can be obtained when the user is co-located with a fixed-based manipulator. However, the physical handling of a mobile manipulator can be challenging, if not impossible.

In this paper, we develop a *Spatial Diagrammatic Instructions* (SDIs) framework, where the user directly sketches over images from camera views of the environment to identify regions that are relevant for an instruction. We highlight that humans have the exceptional ability to make inferences from diagrammatic sketches over images and relate instructions with spatial regions in the physical environment. We seek to endow collaborative robots with this same capability. We propose *Spatial Instruction Maps* (SIMs) which take user-specified sketches to learn differentiable and probabilistic models that identify the corresponding 3D spatial regions.

Fig. 1: Spatial Diagrammatic Instructions enable users to sketch on camera images of the environment. Top: Red indicates regions where the user sketches as regions of interest, while green indicates regions the user sketches as permissible regions. Bottom: Continuous spatial representations of these sketched regions can then be incorporated as objectives and constraints within optimization problems to find optimal positions for mobile manipulator base placement.

In particular, we leverage diagrammatic instructions to enable robot-human collaboration to solve for the optimal placement of the mobile base. That is, where should a mobile manipulator navigate to, such that it can effortlessly reach objects in the regions of interest? We demonstrate how our proposed spatial representations are amenable to integration into optimization problems, with the differentiability of the representations facilitating efficient optimization.

Concretely, the technical contributions of this paper include:

1) The *Spatial Diagrammatic Instructions* framework for human operators to specify spatial regions for optimization problems;
2) *Spatial Instruction Maps*, a *probabilistic* and *continuous* representation of user-specified regions as energy-based models;

[*] Equal Contribution
[1] Robotics Institute, Carnegie Mellon University, PA, USA.
Code is available at `https://github.com/PtSamuel/diagrammatic_instructions`

3) An inverse kinematic-free method that leverages user diagrammatic specifications to solve base placement for a mobile manipulator.

The remainder of the paper is as follows: we shall first give a bird's-eye view summary of the past work that is relevant to this paper in Section II. Then, we introduce the concept of *Spatial Diagrammatic Instructions*, which is central to our contributions, along with the problem formulation in Section III. Next, we describe the contributed methodology of building *Spatial Instruction Maps*, along with the integration of the maps into an optimization problem to solve the mobile base placement problem in Section IV. We subsequently report empirical evaluation of our contributed methodology in Section V and conclude in Section VI.

II. Related work

Robot Learning from Human Input: Humans can provide robots a wealth of information around how to act. One of the most well-studied problems where robots learn from human inputs is imitation learning [1], [2], [3]. In these setups, a human expert will provide demonstrations of the desired movement and the robot seeks to mimic the motion patterns. Frameworks around learning and then importantly generalizing the motion have also been proposed [4], [5]. Other extensions to imitation learning that consider human preference, where desirable behaviors are strengthened while undesirable ones are diminished, have also been explored [6]. Specifying robot motions via natural language has also been an area of research [7]. Diagrammatic Learning [8] has been proposed as a medium to teach robots motion by sketching the desired motion. Similarly, this work seeks to enable users to provide input via sketches, but of desired spatial regions rather than motion trajectories.

Building Spatial Representations: Representing the environment where a robot operates is central to robotics. Early approaches in this area looked at grid maps [9], [10]. Various approaches have also been explored to build continuous environment representations for robotics. These include models of occupancy [11], [12], motion [13], distance [14], and photo-realistic models [15], [16], [17]. In this work, we seek to construct continuous representations that are amenable to optimization, but the properties we wish to embed into the representation are spatial instructions that can be used for planning [18], [19] and optimization problems [20], [21], [22].

Base Placement for Mobile Manipulators: Positioning mobile manipulators has received notable interest in the context of industrial manufacturing. A painting robot, for example, usually paints a large work-piece portion by portion. A trajectory for the end-effector pose is given for each portion, and the base position needs to be optimized accordingly [23]. Under scenarios like this, multiple performance indices that serve as the optimization object have been proposed. One intuitive way is to maximize the proportion of target points that are achievable [24]. Besides, [23] proposed optimizing for the manipulability and dexterity of joints, [25] incorporated the manipulator stiffness into the objective, and [26] aimed at minimizing the energy consumption for completing a certain task. Within the robotics community, two methodologies have been explored: using a reachability map vs. not using one. For the former, [27] prepared a reachability map offline, by discretizing the space of end-effector poses. [28] extended this methods by applying a clustering algorithm on the potential base poses to find the best ones. For the other approach that does not use a reachability map, [29] proposed a co-evolutionary method to optimize the base positions for a pick-and-place task, combining grasp quality, manipulability, and collision avoidance into the performance metric. Unlike our proposed approach, these methods require an *a priori* representation of the environment layout and do not allow the user to specify desired objectives and constraints onto images.

III. Problem Formulation

In this section, we begin by introducing the concept of *Spatial Diagrammatic Instructions* (SDIs). Then, we define the *Mobile Base Placement Problem* (MBPP) and outline how the SDIs enable humans to collaborate with the robot to solve the MBPP.

A. Spatial Diagrammatic Instructions

We are assumed to have access to an RGB-D camera with known intrinsics and pose. We begin by providing the human operator with an RGB image of the environment, then prompt the human operator to provide spatial instructions by outlining a region on the image. Given the camera parameters, we project the instruction-specified regions into the environment, i.e. $\mathcal{X}_1, \mathcal{X}_2, \ldots, \mathcal{X}_m \subseteq \mathbb{R}^3$, where m is the number of instructions. Then, we can leverage the specified regions within optimization problems in robotics. These take the general form:

$$\text{Optimize} \quad F(x|\mathcal{X}_1, \mathcal{X}_2, \ldots, \mathcal{X}_m) \tag{1}$$

$$\text{s.t.} \quad G(x|\mathcal{X}_1, \mathcal{X}_2, \ldots, \mathcal{X}_m) = 0, \tag{2}$$

where F is an objective function and G denotes a set of constraint functions.

B. Mobile Base Placement Problem

A particular optimization problem that can benefit from diagrammatic instructions is the *Mobile Base Placement Problem* (MBPP). Consider the placement of a mobile base with a mounted manipulator, such that it has maximum coverage over a region of interest (ROI). This ROI should include regions where manipulation tasks are likely to begin or end. For example, in the task of moving objects from one table to another, the ROI would be the 2 tabletops. Additionally, the mobile base can also be constrained to be placed within a designated region.

The motivation for this problem is described as follows. When a robot manipulator mounted on a mobile platform performs a task like picking and placing an object, both the robot manipulator and the platform may move. Allowing motion of both, however, increases the difficulty in planning and potentially leads to greater actuation error. Where possible, it is therefore desirable to keep the mobile platform

fixed in place, and let the manipulator complete the whole task on its own. In this work, we attempt to find the base placement that best facilitates downstream tasks, especially those that require the manipulator to move long ranges. We now introduce the notations used throughout the paper. Let $\mathcal{X}_{ROI}$ denote the ROI that the mobile manipulator seeks to cover, and let the mobile base configuration be $\mathcal{C}$. In this paper, we consider $\mathcal{C}$ to have 3D coordinates $\mathbf{x} \in \mathbb{R}^3$ and yaw angle ω as states, i.e. $\mathcal{C} = (\mathbf{x}, \omega)$, but $\mathcal{C}$ can also be defined more generally. Let $\mathcal{B}$ denote the set of all allowable base configurations $\mathcal{C}$. We also need the forward kinematics of the manipulator $\mathbf{x}_e = f_e(\mathbf{q} \mid \mathcal{C})$, where $\mathbf{x}_e \in \mathbf{SE}(3)$ is the end-effector pose, which is a function of both $\mathbf{q}$, the manipulator's joint configurations, and the base configuration $\mathcal{C}$. We view the optimal base placement as one that maximizes the coverage of some ROI, and we quantify the coverage by the probability $\mathbf{P}(f_e(\mathbf{q} \mid \mathcal{C}) \in \mathcal{X}_{ROI})$. Therefore, we define the *Mobile Base Placement Problem* (MBPP) as:

$$\textbf{[MBPP]:} \qquad \arg\max_{\mathcal{C}} \mathbf{P}(f_e(\mathbf{q} \mid \mathcal{C}) \in \mathcal{X}_{ROI}), \tag{3}$$

$$\text{s.t.} \qquad \mathcal{C} \in \mathcal{B}. \tag{4}$$

Note that we leave $\mathbf{q}$ as a free variable, which will be discussed in Section IV-C.

IV. Optimization on Spatial Instruction Maps

After the human operator is prompted to sketch Spatial Diagrammatic Instructions over camera images, the instructions are projected into 3D space (outlined in Section IV-A). Then, we construct concise and continuous Spatial Instruction Maps using *energy-based models* (Section IV-B). These models are subsequently integrated into the mobile base placement problem to be solved. Solving the optimization is detailed in Section IV-C and the enforcement of constraints is elaborated in Section IV-D.

A. Projecting Spatial Diagrammatic Instructions into 3D

The human operator is prompted to sketch instructions directly onto images of the environment captured by an RGB-D camera and enclose regions on the image relevant to an instruction. Though these diagrams are 2D, they relate to relevant spaces within the 3D environment, and form the basis of optimization objectives (e.g. coverage of the selected space) and constraints (e.g. regions to stay in or stay out of).

After the operator has sketched the diagrammatic instructions, we then obtain all the pixels that are enclosed in the sketch and find their projection into the 3D environment. Here, we denote each pixel projected into 3D as $\mathbf{x}_s \in \mathcal{S} \subseteq \mathbb{R}^3$, where $\mathcal{S}$ denotes the set of 3D points corresponding to the region enclosed by the sketch. We then have, for each enclosed pixel (u, v), the transformation:

$$\mathbf{x}_s = R_C \begin{bmatrix} \frac{u-c_x}{f_x} z \\ \frac{v-c_y}{f_y} z \\ z \end{bmatrix} + t_C,$$

where z is the measured depth, or the distance from the pixel to the camera's sensor plane, R_C, t_C are respectively the camera's rotation and translation in the world frame, and f_x, f_y, c_x, c_y are the focal lengths and center points along x and y axes of the camera. These parameters can be obtained from the camera's manufacturer. After this is done, all of the 3D points obtained, $\mathbf{x}_s \in \mathcal{S}$, are scattered in the regions outlined by the human operator. Next, we shall explore the construction of models defined over these regions.

B. Spatial Representations via Energy-Based Models

Diagrammatic instructions are sketched onto images with discrete pixels which are subsequently projected as a discrete point set $\mathcal{S}$ into 3D space, but optimization objectives and constraints defined over the corresponding specified regions require a continuous domain. We therefore need to develop continuous and probabilistic representations of the specified regions. Here, we model the probability of any coordinate $\mathbf{x}$ being in the space specified by the user in the sketches, $\mathbf{P}(\mathbf{x} \in \mathcal{S} \mid \mathbf{x})$, as a continuous function over $\mathbf{x} \in \mathbb{R}^3$, which we call the *Spatial Instruction Map* (SIM).

We choose to use *energy-based models*, which are highly flexible and can model complex distributions efficiently. Energy-based models (EBMs) are probabilistic models with densities over data $\mathbf{x}$ given as:

$$p_\theta(\mathbf{x}) = \frac{\exp(E_\theta(\mathbf{x}))}{Z_\theta}, \qquad Z_\theta = \int \exp(E_\theta(\mathbf{x}))\mathrm{d}\mathbf{x}, \tag{5}$$

where $E_\theta(\mathbf{x}) \in \mathbb{R}$ is known as the *energy function*, which is here represented as a neural network with parameters θ, and Z_θ is the normalizing constant (or *partition function*).

Noise-contrastive approaches provide a way to estimate parameters of the EBM, θ, without direct access to Z_θ by learning a classifier to identify data from generated negative samples. In particular, *Noise Contrastive Estimation (NCE)* [30] is an approach that assumes that there are negative examples $\tilde{\mathbf{x}}$ drawn from a noise distribution $q(\tilde{\mathbf{x}})$. The energy estimation is then cast as a binary classification problem, and a logistic regressor is applied to minimize the binary cross-entropy loss:

$$\mathcal{L}_{\mathrm{NCE}}(\theta) = \sum_{i=1}^{N} [\log \sigma(E_\theta(\mathbf{x}_i)) + \log(1 - \sigma(E_\theta(\tilde{\mathbf{x}}_i)))], \tag{6}$$

where $\sigma(\cdot)$ is the sigmoid function. The loss function can be efficiently optimized via gradient descent methods. Consequently, we can train $E_\theta(\cdot)$ as a classifier that identifies points in $\mathcal{S}$ from generated negative samples, then the trained model will represent $\mathcal{S}$ as a continuous, differentiable distribution. And importantly, because $E_\theta(\cdot)$ is trained as a classifier, we have

$$\mathbf{P}(\mathbf{x} \in \mathcal{S} \mid \mathbf{x}) = \sigma(E_\theta(\mathbf{x}_i)). \tag{7}$$

$E_\theta(\cdot)$ is trained as follows: $E_\theta(\cdot)$ is modeled by a 2-layer fully connected neural network with width 256. Positive samples are uniformly drawn from $\mathcal{S}$, and negative samples by uniformly sampled from a larger space surrounding $\mathcal{S}$. All training is conducted with an AdamW optimizer [31] with learning rate 10^{-3} and weight decay 10^{-4}.

C. Solving the MBPP

After preparing the SIMs based on EBM, we gain a continuous representation of the ROI and can now proceed

to solving the MBPP. A commonly used approach to find the joint configurations and base configuration to reach a target pose is by an inverse kinematics (IK) solver. In the current setting, however, IK-based methods have a fundamental limitation. The unconstrained end-effector orientation leaves significantly more freedom for the solver, which is compounded by the kinematic redundancies introduced by the mobile base. Consequently, to achieve an end-effector position, there could be a wide ranges of allowable base placements, and selecting the base placement that maximizes the coverage of the ROI could be difficult, if not intractable.

To address this problem, we model the reachable positions by the end-effector as a probability distribution over all the allowable joint configurations, $\mathcal{Q}$, and let $p_{\mathbf{q}}(\cdot)$ be the joint p.d.f. of $\mathbf{q}$. Recall from Eq. (3) that given a base configuration $\mathcal{C}$, the probability of end-effector lying in the ROI is:

$$\mathbf{P}(f_e(\mathbf{q} \mid \mathcal{C}) \in \mathcal{X}_{ROI}) = \mathop{\mathbb{E}}_{\mathbf{q}\in\mathcal{Q}}(\mathbf{P}(f_e(\mathbf{q} \mid \mathcal{C}) \in \mathcal{X}_{ROI})), \quad (8)$$

which is the expectation over all $\mathbf{q} \in \mathcal{Q}$. By Eq. (7), we have

$$\mathbf{P}(f_e(\mathbf{q} \mid \mathcal{C}) \in \mathcal{X}_{ROI}) = \mathop{\mathbb{E}}_{\mathbf{q}\in\mathcal{Q}}(\sigma(E_\theta(f_e(\mathbf{q} \mid \mathcal{C})))). \quad (9)$$

Optimization on $E_\theta(\mathbf{x})$ is more robust than on $\sigma(E_\theta(\mathbf{x}))$, so we instead estimate the expected energy,

$$\mathcal{E}(\mathcal{C}) = \mathop{\mathbb{E}}_{\mathbf{q}\in\mathcal{Q}}(E_\theta(f_e(\mathbf{q} \mid \mathcal{C}))).$$

As $p_{\mathbf{q}}(\cdot)$ is unknown, we assume the distribution of each joint position q_j in $\mathbf{q}$ is independent and follows a uniform distribution within its physical limits, $[q_{j_{min}}, q_{j_{max}}]$. We then estimate the distribution of end-effector positions by sampling joint positions in $\mathcal{Q}$. That is, we estimate $\mathcal{E}(\mathcal{C})$ by

$$\hat{\mathcal{E}}(\mathcal{C}) = \frac{1}{N}\sum_{i=1}^{N} E_\theta(f_e(\mathbf{q}_i \mid \mathcal{C})), \quad (10)$$

where N is the number of samples, and the $\mathbf{q}_i$'s are uniformly sampled from $\mathcal{Q}$. Consequently, we reframe the MBPP into the following optimization problem

$$\textbf{[MBPP]:} \qquad \arg\max_{\mathcal{C}} \hat{\mathcal{E}}(\mathcal{C}), \quad (11)$$

$$\text{s.t.} \qquad \mathcal{C} \in \mathcal{B}. \quad (12)$$

We solve this problem by stochastic gradient ascent on $\hat{\mathcal{E}}$, where the gradient is computed as

$$\frac{\partial \hat{\mathcal{E}}}{\partial \mathcal{C}} = \frac{1}{N}\frac{\partial}{\partial \mathcal{C}}\sum_{i=1}^{M} E_\theta(f_e(\mathbf{q}_i \mid \mathcal{C})) \quad (13)$$

$$= \frac{1}{N}\sum_{i=1}^{M} \frac{\partial E_\theta(f_e)}{\partial f_e}\frac{\partial f_e(\mathbf{q}_i \mid \mathcal{C})}{\partial \mathcal{C}}. \quad (14)$$

$\frac{\partial E_\theta(f_e)}{\partial f_e}$ can be easily computed by automatic differentiation, as $E_\theta(\cdot)$ is a neural network model. The derivative of the forward kinematics $\frac{\partial f_e(\mathbf{q}_i|\mathcal{C})}{\partial \mathcal{C}}$ is also easily attainable by the package PyTorch Kinematics.

D. Constraints

The discussion above derives an objective for the MBPP, where we wish the coverage of the manipulator over the regions of interest to be maximized. We now describe the imposition of the constraint $\mathcal{B}$ on the base pose. Specifically, constraints are separately applied to the 3D coordinates of the base $\mathbf{x} = (x, y, z)$ and its yaw angle ω. We optimize with projected gradient methods [32] to enforce constraint satisfaction, taking the following constraints into consideration:

1) **Diagrammatic instruction constraints on the x, y coordinates of the base position:** If the human operator opts to specify a permissible region, $\mathcal{X}_p$, the same method discussed in IV-A is used to find the coordinates of the region in 3D, and their x, y components are extracted.
2) **Box constraint on z:** The height of a mobile base could be variable, e.g. on a quadruped, so we restrict $z \in [z_{min}, z_{max}]$.
3) **Box constraint on ω:** Likewise, we restrict $\omega \in [\omega_{min}, \omega_{max}]$.

During gradient ascent, if a step along the gradient results in an infeasible solution, the current solution is updated to be nearest feasible solution, i.e. the height z is clamped to $[z_{min}, z_{max}]$, the yaw angle z is clamped to $[\omega_{min}, \omega_{max}]$,

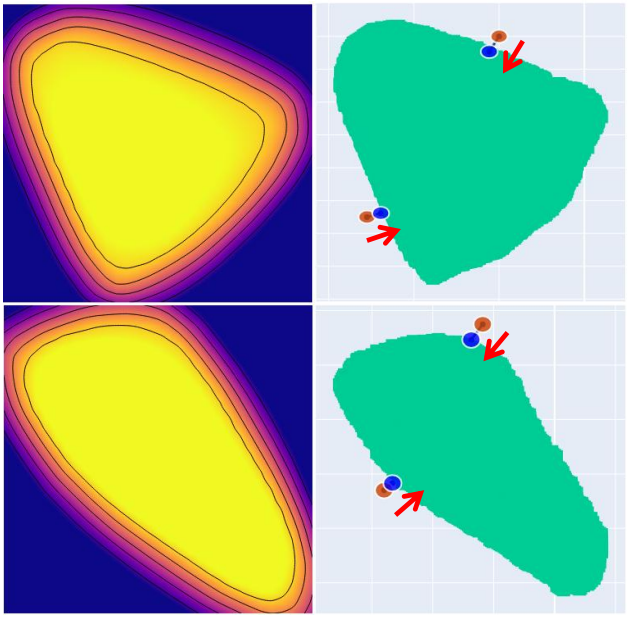

Fig. 2: Left: Examples of constraint SIMs. Contours indicate equal output values. Right: Examples of applying Algorithm 1 to project infeasible points (brown) back into the feasible region. Red arrows show the direction of projection. The resulting points on the boundary are shown in blue.

Algorithm 1 Projection to Boundary

Input: constraint Spatial Instruction Map $\mathcal{G}(\cdot)$, coordinates $\mathbf{p} = (x, y)$, threshold probability τ, maximum number of iterations T_{Project}, precision ϵ
function PROJECT
  for $t \leftarrow 1$ to T_{Project} **do**
    if $|\mathcal{G}(\mathbf{p}) - \sigma^{-1}(\tau)| > \epsilon$ **then**
      $\mathbf{g} \leftarrow \nabla_{\mathbf{p}}\mathcal{G}(\mathbf{p})$
      $\mathbf{p} \leftarrow \mathbf{p} - \frac{\mathbf{g}}{\|\mathbf{g}\|^2}(\mathcal{G}(\mathbf{p}) - \sigma^{-1}(\tau))$ ▷ Eq. (15)
    else
      break
    end if
  end for
end function
Output: $\mathbf{p}$

Algorithm 2 MBPP Optimizer

Input: step size α, initial base placement $\mathcal{C}_0 = (x_0, y_0, z_0, \omega_0)$, height limits z_{min}, z_{max}, yaw angle limits $\omega_{min}, \omega_{max}$, number of joint position samples N, allowable joint positions $\mathcal{Q}$, forward kinematics $f_e(\cdot)$, energy function $E_\theta(\cdot)$, number of iterations T_{MBPP}, constraint Spatial Instruction Map $G(\cdot)$, threshold τ.
Initialize $\mathcal{C} \leftarrow \mathcal{C}_0$
for $t \leftarrow 1$ to T_{MBPP} **do**
  Uniformly sample N joint configs $\mathbf{q}_{1\cdots N} \in \mathcal{Q}$
  $\hat{\mathcal{E}} \leftarrow \frac{1}{N}\sum_{i=1}^{N} E_\theta(f_e(\mathbf{q}_i \mid \mathcal{C}))$
  $\mathbf{g} \leftarrow \frac{1}{N}\sum_{i=1}^{N} \frac{\partial E_\theta(f_e)}{\partial f_e}\frac{\partial f_e(\mathbf{q}_i|\mathcal{C})}{\partial \mathcal{C}}$
  $(x, y, z, \omega) \leftarrow \mathcal{C} + \alpha\mathbf{g}$ ▷ Gradient ascent on $\hat{\mathcal{E}}$
  $z \leftarrow clamp(z, z_{min}, z_{max})$
  $\omega \leftarrow clamp(z, \omega_{min}, \omega_{max})$
  $\mathbf{p} \leftarrow (x, y)$
  if $G(\mathbf{p}) < \tau$ **then**
    $\mathbf{p} \leftarrow \text{PROJECT}(\mathbf{p}, \mathcal{G}, \tau)$
  end if
  $(x, y) \leftarrow \mathbf{p}$
  $\mathcal{C} \leftarrow (x, y, z, \omega)$
end for
Output: optimized base placement $\mathcal{C}$

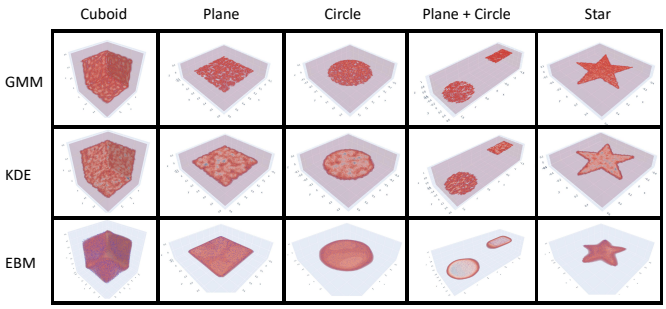

Fig. 3: KDE, GMM, and our energy-based SIM fitted on different dataset of spatial points.

and the x, y coordinates are projected to the nearest feasible solution in $\mathcal{X}_p$, which can be efficiently handled by leveraging the method discussed in IV-B. Specifically, we train another constraint SIM $G(\cdot)$ to predict the probability $\mathbf{P}(\mathbf{p} \in \mathcal{X}_p \mid \mathbf{p})$, for any $\mathbf{p} \in \mathbb{R}^2$, by contrastive loss. The boundary of the region is then an equal-probability contour corresponding to a threshold τ. If a gradient update steps out from $\mathcal{X}_p$, the x, y coordinates are projected back to the contour corresponding a $P = \tau$. For added numerical stability, we take $\mathcal{G}(\cdot)$ as the output of $G(\cdot)$ prior to the final sigmoid layer, and instead solve for $\mathcal{G}(\mathbf{p}) = \sigma^{-1}(\tau)$, where $\sigma^{-1}(\cdot)$ is the inverse of sigmoid function. We use Newton's method towards the feasible region by considering the residual,

$$\mathbf{p} \leftarrow \mathbf{p} - \frac{\nabla_{\mathbf{p}}\mathcal{G}(\mathbf{p})}{\|\nabla_{\mathbf{p}}\mathcal{G}(\mathbf{p})\|^2}(\mathcal{G}(\mathbf{p}) - \sigma^{-1}(\tau)). \tag{15}$$

The projection of infeasible solutions to constraint satisfaction is described in Algorithm 1, and the entire optimization procedure is outlined in Algorithm 2. Additionally, the SIMs and projection of a number of points outside the constraint area is shown in Fig. 2.

	KDE	GMM-10	GMM-50	GMM-100	Ours
Cuboid	3.851	6.034	6.182	6.177	**8.039**
Plane	4.909	7.213	7.217	7.119	**9.194**
Circle	5.169	7.490	7.506	7.389	**9.381**
Plane + Circle	2.859	5.596	5.612	5.465	**6.979**
Star	4.753	7.023	7.095	7.081	**8.656**

TABLE I: Our SIMs which leverage EBMs outperform KDE and GMM models, based on log-likelihood metrics. Higher log-likelihood values indicate better performance.

V. Empirical Evaluation

In this section, we provide empirical evaluations on both the quality of the EBM representation of SIMs, as well as the quality of our optimization methodology. The specific questions that we seek to answer include: (1) How does the energy-based model (EBM) representation of Spatial Instruction Map (SIM) compare against alternative methods? (2) How does the optimization perform against baseline methods that solve for mobile base placement? (3) How does our approach handle multi-modality, or multiple specified regions which cannot all be covered?

A. EBMs Represent SIMs Well

We seek to investigate how our EBM representation of Spatial Diagrammatic Maps compares against other common approaches, in constructing continuous probability density estimates. We evaluate the quality of estimating a distribution over datapoints from the following datasets: **Cuboid**, **Plane**, **Circle**, **Plane + Circle** and **Star**, as shown in Fig. 3. For each dataset, we fit the model on the dataset, and report the per-sample log-likelihood on another dataset sampled from the same distribution.

In particular, we compare against the following baseline methods:

- **Gaussian Mixture models (GMMs)** [33]: We fit mixtures of Gaussians with full covariances. We evaluate on three variants of the GMM, each with 10, 50 and 100 Gaussians in the mixture.
- **Kernel Density Estimator (KDE)** [34]: We fit KDEs with a Gaussian kernel with a length-scale parameter tuned to maximize the log-likelihood.

Table I summarizes the values of log likehood for KDE, GMM, and our SIMs that are built on EBMs. KDE yields worse results than GMM, while GMM-50 outperforms GMM-10 and also GMM-100 by a narrow margin. This suggests that increasing the number of Gaussians beyond a limit leads to overfitting and degrades the performance. On the other hand, the log likelihoods of EBMs are consistently above those of KDEs and GMMs, indicating that EBMs perform better at approximating those distributions and can generalize to spatial regions where observations are relatively sparse.

B. Solving the MBPP with Spatial Diagrammatic Maps

We evaluate our approach of formulating and solving optimization problems which build upon our SIMs on multiple real-world scenes, where images are taken by a Vzense time-of-flight RGB-D camera. Diagrammatic instructions to specify ROIs for manipulation and permissible regions for

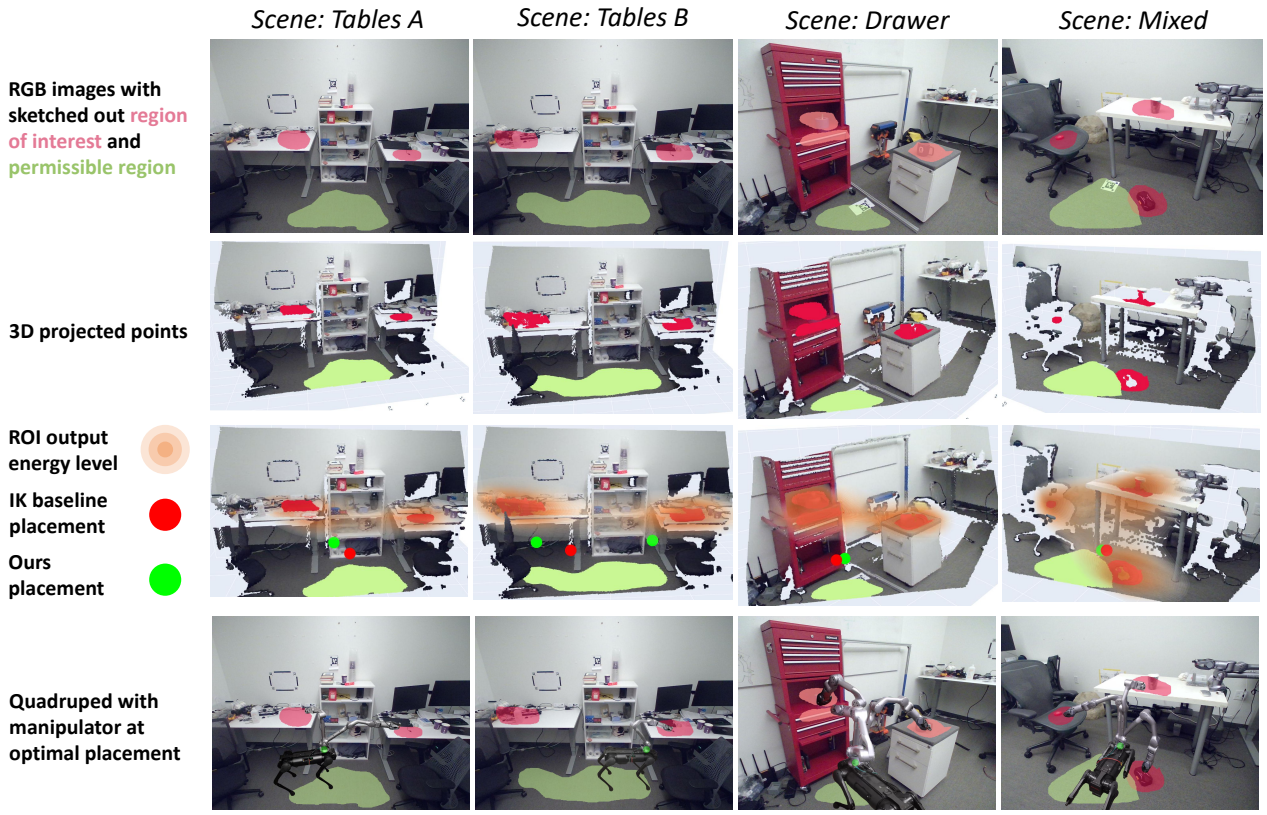

Fig. 4: Row 1: Scenes with annotated SDIs. Row 2: SDIs projected into 3D. Row 3: Energy output levels from SIMs fitted on sketched ROIs. Row 4: Quadruped with manipulator posited at the optimal placements (indicated by green dot).

Hyperparameter	Value
N	1024
T_{Project}	20
T_{MBPP}	40
α	0.005
z_{min}, z_{max}	$0.15\,m, 0.42\,m$
$\omega_{min}, \omega_{max}$	$0, 2\pi$
$\mathcal{Q}, f_e(\cdot)$	Specified by URDF
τ	0.95

TABLE II: Hyperparameters used for the optimization.

Scene	*Tables A*	*Tables B*	*Drawer*	*Mixed*
Random reachability (%)	1.123	2.588	59.668	7.178
IK reachability (%)	1.660	2.832	70.263	15.039
Ours reachability (%)	**30.469**	**50.000**	**77.637**	**61.865**
IK runtime (s)	2.234	2.417	2.306	2.113
Ours runtime (s)	**0.350**	**0.424**	**0.440**	**0.451**

TABLE III: The performance of random baseline, IK baselines, and ours, evaluated through percent reachability and runtime. Higher reachability indicates better performance while lower runtime indicates greater efficiency.

the mobile base are sketched onto the images. This gives us 4 scenes, named *Tables A*, *Tables B*, *Drawer*, and *Mixed*.

We use the Unitree Z1 as the mobile manipulator the Unitree Aliengo quadruped as the mobile platform, which can elevate the manipulator base to a height between $0.15\,m$ and $0.42\,m$ above the ground. The quadruped is omnidirectional, i.e. $\omega \in [0, 2\pi)$. We use the joint limits specified in the URDF for Z1. We report the percentage of area of the ROI that is reachable via the simulator PyBullet [35]. Fig. 4 shows the specified ROI and constraint overlayed on the images, the 3D projected points for each region, and the optimized base placements. The hyperparameters used for the optimization in our experiments are tabulated in Table II.

We compare our method against a **Randomized Baseline** and an **Inverse Kinematics (IK) Baseline**. The randomized baseline randomly selects a base placement that satisfies all the constraints. Within the IK baseline, points are first sampled from the distribution learned by $E_\theta(\cdot)$, and for each point, Pybullet's IK solver is used to find, if any, a base placement and joint positions that make the end-effector reach that point. The output is the mean of all the base placements found, ensuring that the solution satisfies the constraints. Table III reports the percentage of points reachable for all three methods, along with the runtimes.

We observed that the IK baseline suffers from long runtime and suboptimal results. This is particularly the case in *Tables B*, where there exists multiple ROIs that are far apart. Additionally, the IK solution may often not produce significantly better results than the random baseline. This is because the inverse kinematics computes the base placement solely by a binary state indicating whether a position is reachable. That is, it does not incorporate the *probability* that a position is a region of interest.

C. Multi-modality in Diagrammatic Instructions

A mobile manipulator may have distinct ROIs that are separated by considerable distance. The SIM and the solution to the MBPP should be able to handle multiple distinct

modes over the space of the environment. The solution to the environment *Tables B*, shown in Figure 4, demonstrates the multi-modality. We observe that our approach to the MBPP can produce distinct solutions that provide coverage for each specific mode, while the inverse kinematics baseline is conflicted and tries unsuccessfully to cover both regions, resulting in a lower coverage percentage. On the other hand, when two ROIs can be simultaneously covered by a single placement of the manipulator, such as in *Tables A*, our approach finds the base position that best covers both regions.

VI. Conclusions and Future Work

In this work, we build a *Spatial Diagrammatic Instructions* framework, where the human operator provides Spatial Diagrammatic Instructions to convey to the robot regions that are relevant for an instruction. We propose *Spatial Instruction Maps*, which learn differentiable and probabilistic models that represent the relevant 3D regions. We demonstrate that our spatial representations faithfully capture the spatial regions. We also show that their differentiability empowers efficient optimization for downstream tasks by applying the representation on the problem of optimal placement of a mobile manipulator, and the result of optimization achieves a high coverage over the regions specified by the user. Therefore, we envision an avenue of future research to be adapting our Spatial Instruction Maps to be not only spatial but also *spatiotemporal*, enabling time dependent instructions to be incorporated.

References

[1] H. Ravichandar, A. S. Polydoros, S. Chernova, and A. Billard, "Recent advances in robot learning from demonstration," *Annual review of control, robotics, and autonomous systems*, 2020.

[2] A. Paraschos, C. Daniel, J. Peters, and G. Neumann, "Probabilistic movement primitives," in *Proceedings of the 26th International Conference on Neural Information Processing Systems*, 2013.

[3] J. Kober and J. Peters, "Learning motor primitives for robotics," in *IEEE International Conference on Robotics and Automation*, 2009.

[4] W. Zhi, T. Lai, L. Ott, and F. Ramos, "Diffeomorphic transforms for generalised imitation learning," in *Learning for Dynamics and Control Conference, L4DC*, 2022.

[5] M. A. Rana, M. Mukadam, S. R. Ahmadzadeh, S. Chernova, and B. Boots, "Towards robust skill generalization: Unifying learning from demonstration and motion planning," in *Proceedings of the 1st Annual Conference on Robot Learning*, 2017.

[6] S. Griffith, K. Subramanian, J. Scholz, C. L. Isbell, and A. L. Thomaz, "Policy shaping: Integrating human feedback with reinforcement learning," in *Advances in Neural Information Processing Systems*, C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Weinberger, Eds., vol. 26. Curran Associates, Inc., 2013.

[7] S. Tellex, N. Gopalan, H. Kress-Gazit, and C. Matuszek, "Robots that use language," *Annu. Rev. Control. Robotics Auton. Syst.*, 2020.

[8] W. Zhi, T. Zhang, and M. Johnson-Roberson, "Learning from demonstration via probabilistic diagrammatic teaching," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2024.

[9] A. Elfes, "Sonar-based real-world mapping and navigation," *IEEE Journal on Robotics and Automation*, 1987.

[10] S. Thrun, "Learning metric-topological maps for indoor mobile robot navigation," *Artificial Intelligence*, 1998.

[11] W. Zhi, L. Ott, R. Senanayake, and F. Ramos, "Continuous occupancy map fusion with fast bayesian hilbert maps," in *International Conference on Robotics and Automation (ICRA)*, 2019.

[12] F. Ramos and L. Ott, "Hilbert maps: Scalable continuous occupancy mapping with stochastic gradient descent," *International Journal Robotics Research*, 2016.

[13] W. Zhi, R. Senanayake, L. Ott, and F. Ramos, "Spatiotemporal learning of directional uncertainty in urban environments with kernel recurrent mixture density networks," *IEEE Robotics and Automation Letters*, 2019.

[14] J. J. Park, P. Florence, J. Straub, R. Newcombe, and S. Lovegrove, "Deepsdf: Learning continuous signed distance functions for shape representation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

[15] 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," in *ECCV*, 2020.

[16] T. Müller, A. Evans, C. Schied, and A. Keller, "Instant neural graphics primitives with a multiresolution hash encoding," *ACM Trans. Graph.*, 2022.

[17] T. Zhang, K. Huang, W. Zhi, and M. Johnson-Roberson, "Darkgs: Learning neural illumination and 3d gaussians relighting for robotic exploration in the dark," *arXiv preprint arXiv:2403.10814*, 2024.

[18] S. M. Lavalle, *Planning Algorithms*. Cambridge University Press, 2006.

[19] T. Lai, W. Zhi, T. Hermans, and F. Ramos, "Parallelised diffeomorphic sampling-based motion planning," in *Conference on Robot Learning (CoRL)*, 2021.

[20] N. Ratliff, M. Zucker, J. A. Bagnell, and S. Srinivasa, "Chomp: Gradient optimization techniques for efficient motion planning," in *IEEE International Conference on Robotics and Automation*, 2009.

[21] W. Zhi, I. Akinola, K. van Wyk, N. Ratliff, and F. Ramos, "Global and reactive motion generation with geometric fabric command sequences," in *IEEE International Conference on Robotics and Automation, ICRA*, 2023.

[22] W. Zhi, T. Lai, L. Ott, E. V. Bonilla, and F. Ramos, "Learning efficient and robust ordinary differential equations via invertible neural networks," in *International Conference on Machine Learning, ICML*, 2022.

[23] Q. Yu, G. Wang, X. Hua, S. Zhang, L. Song, J. Zhang, and K. Chen, "Base position optimization for mobile painting robot manipulators with multiple constraints," *Robotics and Computer-Integrated Manufacturing*, vol. 54, pp. 56–64, 2018.

[24] J. J. Yang, W. Yu, J. Kim, and K. Abdel-Malek, "On the placement of open-loop robotic manipulators for reachability," *Mechanism and Machine Theory*, vol. 44, no. 4, pp. 671–684, 2009.

[25] Q. Fan, Z. Gong, B. Tao, Y. Gao, Z. Yin, and H. Ding, "Base position optimization of mobile manipulators for machining large complex components," *Robotics and Computer-Integrated Manufacturing*, vol. 70, p. 102138, 2021.

[26] M. Gadaleta, G. Berselli, and M. Pellicciari, "Energy-optimal layout design of robotic work cells: Potential assessment on an industrial case study," *Robotics and Computer-Integrated Manufacturing*, vol. 47, pp. 102–111, 2017, sI: FAIM 2015.

[27] N. Vahrenkamp, T. Asfour, and R. Dillmann, "Robot placement based on reachability inversion," in *2013 IEEE International Conference on Robotics and Automation*, 2013, pp. 1970–1975.

[28] A. Makhal and A. K. Goins, "Reuleaux: Robot base placement by reachability analysis," in *2018 Second IEEE International Conference on Robotic Computing (IRC)*, 2018, pp. 137–142.

[29] D. Berenson, J. Kuffner, and H. Choset, "An optimization approach to planning for mobile manipulation," in *2008 IEEE International Conference on Robotics and Automation*, 2008, pp. 1187–1192.

[30] M. U. Gutmann and A. Hyvärinen, "Noise-contrastive estimation: A new estimation principle for unnormalized statistical models," in *International Conference on Artificial Intelligence and Statistics*, 2010.

[31] I. Loshchilov and F. Hutter, "Decoupled weight decay regularization," 2019.

[32] S. Bubeck, "Convex optimization: Algorithms and complexity," *Found. Trends Mach. Learn.*, 2015.

[33] C. M. Bishop, *Pattern recognition and machine learning, 5th Edition*, ser. Information science and statistics. Springer, 2007.

[34] B. W. Silverman, *Density Estimation for Statistics and Data Analysis*. Chapman & Hall, 1986.

[35] E. Coumans and Y. Bai, "Pybullet, a python module for physics simulation for games, robotics and machine learning," http://pybullet.org, 2016–2019.