使不可行任务变为可行：利用可移动物体重构不连通三维环境的规划

Samarth Kalluraya，Yiannis Kantaros

arXiv:2601.02645v1 [cs.RO] 6 Jan 2026

摘要——**已有多种规划器被开发出来，用于计算机器人从初始构型到目标构型的动态可行、无碰撞路径。这些研究的一个关键假设是目标区域可达；但在实际中，当环境不连通时，这一假设往往不成立。受此局限启发，我们研究已知的三维环境，其中包含形成不同可通行支撑面（平面）的物体（也称为方块），这些物体可以是不可移动的（例如桌子），也可以是可移动的（例如箱子）。这些平面可能因高度差、孔洞或横向间隔而彼此不连通。我们关注这样一类任务：机器人必须到达位于高处平面上的目标区域，而该区域无法到达。与其判定此类任务不可行，一种有效策略是让机器人与环境交互，重新布置可移动物体以创建新的可通行连接；这类问题称为可移动物体导航（NAMO）。现有的 NAMO 规划器通常针对二维环境，在其中将障碍物推开以清出一条路径。这些方法无法直接处理所研究的三维场景；在此类情况下，必须策略性地放置障碍物，以跨越这些物理断连。为应对这一挑战，我们开发了 BRiDGE（断连三维几何环境中的基于块体的重构），这是一种基于采样的规划器，它在机器人与物体构型上逐步构建树，从而计算可行规划，具体说明要移动哪些物体、将其放置在何处以及按什么顺序操作，同时考虑可移动物体数量有限的情况。为加快规划，我们引入了非均匀采样策略。我们证明了该方法具有概率完备性，并通过大量数值实验和硬件实验验证了其有效性。**

I. 引言

许多 MOTION 规划问题着重于计算从初始构型到目标构型的无碰撞、动力学可行的机器人路径 [1]–[3]。尽管许多经典方法——例如势场／向量场 [4]–[6],搜索式规划器 [7]–[9],以及基于采样的算法 [10]–[14]——假设在固定障碍物布局下目标可达，但现实世界中的环境往往不满足这一假设。在这种情况下，机器人不能只报告任务失败，而必须主动与环境交互，重新摆放可移动物体，以连接彼此隔绝的区域。为此，需要规划算法来确定要移动哪些障碍物、移动顺序以及放置位置。这个问题称为可移动物体导航(NAMO) [15]，并且已被证明是 NP 难问题 [16]。

已有若干规划器被提出用于解决 NAMO 问题 [15]–[29]。这些工作主要关注二维／平面环境，在这类环境中，只需将可移动物体推到一旁，为通往目标的路径清出空间。换句话说，移开障碍物后，目标区域便可到达。

S. Kalluraya 和 Y. Kantaros 任职于圣路易斯华盛顿大学电气与系统工程系，地址为美国密苏里州圣路易斯市，邮编 63130。本研究获得 NSF CAREER奖项 CNS #2340417 的支持。{k.samarth,ioannisk@wustl.edu}

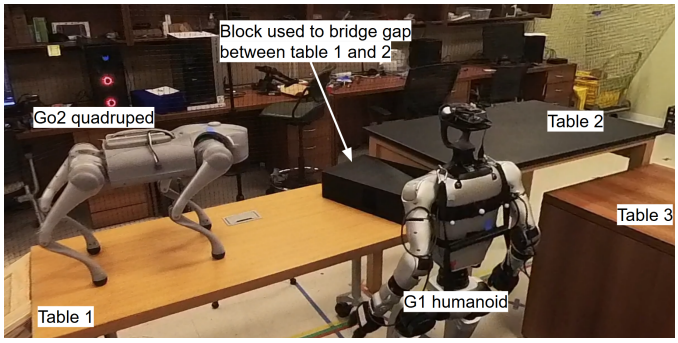


图 1. 四足机器人需要到达桌子 3 上的目标区域；参见第 VI-E 节。然而，这需要穿越由桌子 2 构成的高处平面。由于高度差，四足机器人的行走策略不允许它从桌子 1 爬到桌子 2。人形机器人没有让任务失败，而是巧妙地将一个块放在桌子 1 上，为四足机器人架起连接两张桌子的通道。

因此，这些方法无法直接处理具有多个高架平面的三维环境中的 NAMO 问题；由于环境的几何结构（即平面之间的物理间隙，例如台阶高度、地面上的孔洞或横向间隔）以及机器人动力学，此类环境本质上是不连通的。在这类情形下，要到达目标，可能不仅需要移开障碍物，还需要策略性地放置障碍物，使其充当平面之间的桥梁。因此，仅仅移除可移动物体并不能保证形成通路。一般而言，要到达目标区域，可能需要在多个平面之间架设桥梁；而当可移动物体数量有限时，可能需要重复利用某些物体来搭建这些桥梁。

本文研究三维环境中 NAMO 的一种搭桥变体。具体而言，我们考虑由物体（称为方块）构成的环境，这些物体分为可移动和不可移动两类（例如地板、桌子、床、凳子）。假设环境已知，即所有方块的位置及其可移动性均已给定。这些方块的顶面构成高架平面，它们在高度、方向和布局上各不相同。机器人的任务是到达位于其中一个平面上的目标区域。由于动力学约束，这类区域可能无法直接到达；例如，四足机器人可能无法从地板跳到书桌上以接近所需物体。在这种情况下，机器人必须将可移动物体用作资源（而不仅仅是需要移开的障碍物），通过重新摆放这些物体来连接各个平面，从而使目标区域变得可达；另见图1。这一问题设定的一项特殊挑战是，用于跨越这些间隙的物体本身可能无法触及。例如，机器人可能需要一张凳子才能爬上书桌，但凳子可能位于另一个高架平面上。在这种情况下，机器人必须首先

考虑如何利用其他可触及物体来接近凳子。当可触及物体数量较少时，问题会更加困难，因为必须策略性地使用同一批物体，并且可能需要多次使用它们来跨越不同的间隙。这些由障碍物和间隙引发的依赖关系，使得长时域抓取与放置计划成为必要。

为应对这一挑战，我们提出了**BRiDGE**（基于块的非连通三维几何环境重构），这是一种基于采样的规划框架，通过在机器人与方块配置上逐步构建树，计算可行的长时域操作规划，明确需要移动哪些方块、将其放置在何处以桥接平面，以及移动的顺序。为加速规划，**BRiDGE**采用非均匀采样策略，优先向潜在有希望的方向扩展树。这些方向由高层规划器确定，该规划器生成符号规划，指定方块抓取与放置动作的序列，但不推理这些动作的几何或动力学可行性，也不确定方块的精确放置位姿。在实现中，我们采用广度优先搜索（BFS）规划器和大语言模型（LLM）来生成这些高层规划，但也可以使用 A* 等其他规划器。需要强调的是，符号规划可能不可行或无法执行，例如，因为高层规划器本身并不完备（如大语言模型），或者因为它无法考虑几何与动力学约束（如 BFS 或大语言模型）。此类错误规划可能会将树引向不可行的方向，从而可能减慢规划速度。为缓解这一问题，我们提供了用于细化符号规划的机制：重新调用高层规划器进行重规划，并移除已证明不可行的高级动作。我们证明，**BRiDGE**具有概率完备性，这意味着如果存在解，那么随着基于采样的规划器迭代次数增加，找到解的概率将趋近于一。无论采用何种符号规划器来引导采样策略，这一保证都成立。我们进行了大量实验，证明我们的算法能够解决复杂的长时域规划问题，即使物体数量有限，且通常需要重复使用方块来桥接多个间隙。此外，我们通过实验证明，与均匀采样相比，我们的非均匀采样策略能够显著加快规划。最后，我们开展了硬件实验，展示了**BRiDGE**最初是为单个机器人设计的，但也可以应用于双机器人团队（由一台四足机器人和一台人形机器人组成）。二者具备互补的导航与操作能力，能够协作使目标区域对其中一台机器人可达；见图1。

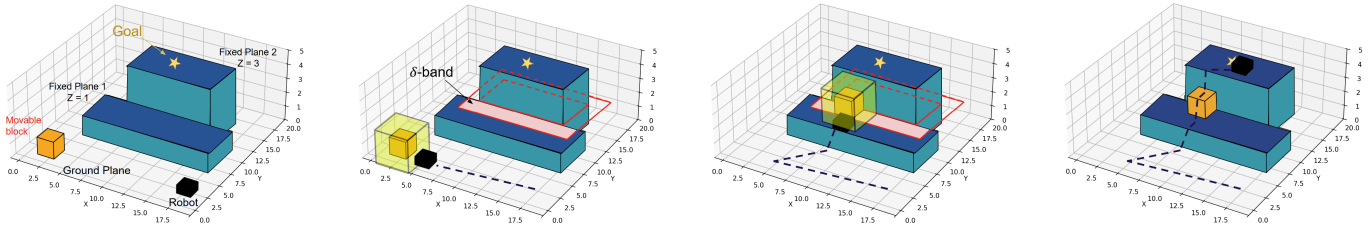
相关工作：接下来，我们从四个方面回顾相关工作：(i) NAMO；(ii) 重排规划；(iii) 多机器人互利协作；以及 (iv) 非均匀采样策略。

(i) NAMO：已有多种基于搜索和采样的规划器被开发出来，用于解决 NAMO 问题 [15], [17]–[23]。针对未知环境的扩展已在 [24]–[29] 中进行了研究。这些工作聚焦于二维平面导航，因此，可移动物体主要被建模为障碍物，机器人应将其推开，以重新打通单一支撑面（例如地板）上的受阻路径。相比之下，我们的设定将物体视为在多个支撑面之间创建可通行连接的资源；我们利用方块构建台阶或桥梁，从而改变三维环境中不同平面之间的连通性。与我们最为相关的 NAMO 问题在 [30] 及其扩展 [31] 中得到了研究。这些论文通过在表面约束环境中进行反应式工具使用来处理导航障碍：当遇到不可避免的障碍物（例如地板上的孔洞）时，机器人通过逆向可供性识别合适的物体，并部署该物体以恢复连通性。这些工作的一项关键假设是，障碍物可以独立地解决，且每个障碍物都有一个始终可用的单一物体。因此，修复机制可以以纯局部的方式反复调用，而无需协调针对多个障碍物的动作。在我们的设定中，该假设并不成立，因为障碍物（即高架平面之间的物理间隙）可能相互依赖，并且由于可用方块数量有限，必须协调处理。换言之，在 [31], 中，每个障碍物都可以通过一次抓取与放置动作解决，而我们的设定则需要长时域的抓取与放置规划。例如，到达目标区域可能需要架设跨越间隙 X 的桥梁，但间隙 X 只能使用位于不可到达平面 Y 上的方块 A 来跨越。解决这种情况需要机器人推理如何利用其可到达的方块（可能多次使用），先进入平面 Y，取回方块 A，然后部署该方块以跨越间隙 X。 [31] 并未发展出此类规划能力；相反，在前述示例中，该工作会假设用于跨越间隙 X 的方块 A 对机器人来说是可到达的。此外，与 [31], 不同，我们的规划器具有概率完备性保证。

(ii) 重排规划（*RP*）：重排规划领域也有相关工作，其中机器人会与可移动物体交互，以实现特定的物体—目标配置 [32]–[35]。与 NAMO 不同，重排规划问题通常不指定机器人的目标，而是着重于实现预先定义的物体最终排列。相比之下，在我们的问题设定中，机器人必须到达指定的目标区域，而障碍物的最终构型并未预先设定。机器人必须推理如何重排障碍物，以便自己能够通行至目标，而不是仅仅按照固定的目标配置排列物体。

(iii) 多机器人系统中的互利共生：另一类相关工作探讨如何利用机器人团队中的异质性，完成单个机器人无法独立完成的任务 [36]–[40]。例如，设想一个地面机器人必须到达一个目标区域，而该区域与其初始位置之间隔着一片水域。这些工作并不会判定任务不可行，而是设计协调框架，让机器人相互协助——例如，由水陆两栖机器人运送地面机器人——以完成任务。然而，在这些方法中，机器人不会主动改变环境结构。在前述例子中，如果没有水陆两栖机器人，任务就会被视为不可行。相比之下，我们的框架允许机器人主动与环境交互并重排环境，从而连接原本互不连通的区域；例如，搭建一座“桥”跨越水域。

(iv) 非均匀采样策略：若干研究设计了非均匀采样策略以加速基于采样的



(a) (b) (c) (d)

图 2. 包含三个固定（不可移动）平面的示例环境：地面（z=0）、平面 1（z=1）和平面 2（z=3），以及一个高度为 $h_1 = 1$ 单位的可移动块 b_1。机器人（黑色立方体）最初位于地面平面上，任务是到达平面 2 上的目标状态（黄色星形）。根据 NavigReach，机器人最多可以跨越 1 个单位的垂直高度差。可以看到，机器人无法从图 2(a) 所示的初始位置到达目标区域。虽然从地面可以到达平面 1，但由于两者的高度差为 2 个单位，平面 2 无法从平面 1 到达；此外，由于两者的高度差为 3 个单位，也无法从地面直接到达。因此，机器人首先靠近并抓取块 b_1，如图 2(b) 所示。半透明的绿色方框显示由 ManipReach 生成的可抓取区域。 δ带（粉色区域）标示了平面 1 上的一个区域：若将 b_1 放置于此处，便可能从平面 1 到达平面 2（见第 IV-B 节）。抓取 b_1后，机器人将其搬运到平面 1 并放置在粉色区域内（图 2(c)）；此时，绿色方框显示由 ManipReach 确定的可行放置区域。机器人以放置的块为台阶，爬上该块，再登上平面 2，最终到达目标（图 2(d)）。黑色虚线表示机器人的路径。

规划器 [41]–[49]。然而，这些方法针对的是不同的规划问题，重点是在假设存在无障碍路径的前提下，寻找初始机器人构型与目标机器人构型之间的无障碍路径，因此无法直接应用于我们的场景。 [23], 考虑了 NAMO 问题中的非均匀采样，但该研究聚焦于二维环境，因此不能直接推广到我们的问题。

贡献： 第一，我们提出了不连通的三维环境中的一种新型 NAMO 问题，其中将可移动物体视为用于连接不连通平面的资源，而不仅仅是需要移除的障碍物。第二，我们提出了一种新颖的基于采样的算法，该算法联合探索机器人和物体的构型空间，以确定需要移动哪些物体、将其放置在何处以及以何种顺序移动，从而实现目标可达性。第三，我们设计了非均匀采样策略，优先沿有希望的方向进行树扩展。第四，我们证明了该算法是概率完备的。最后，我们提供了大量数值实验和硬件实验结果，验证了该方法的有效性和效率。

II. 问题定义

A. 环境

我们考虑一个位于 $\mathbb{R}^3$ 的环境，其中包含一组 $\mathcal{B} = \{b_1, \ldots, b_{N_b}\}$ 个 $N_b \geq 1$ 已知物体，称为 方块。为简单起见，我们将这些方块建模为直棱柱（见备注 2.3）。每个方块的几何构型 b_i 由其 $\mathbf{q}_i^j$ 个顶点的坐标$j \in \{1, \ldots, s\}$及其底面高度$h_i > 0$确定。我们将 b_i 的几何构型简洁地表示为一个元组：

$$\mathbf{b}_i = (\mathbf{Q}_i, h_i), \tag{1}$$

其中$\mathbf{Q}_i = [\mathbf{q}_i^1, \mathbf{q}_i^2, \ldots\ldots, \mathbf{q}_i^s] \in \mathbb{R}^{s\times 3}$是收集底面 s 顶点坐标的矩阵。

方块分为两类：(i) 不可移动方块，收集于 $\mathcal{B}^{\text{nm}} \subseteq \mathcal{B}$，代表环境中的固定结构（例如地板、桌子或床），其构型 $\mathbf{b}_i$不会随时间改变；以及 (ii) 可移动方块， $\mathcal{B}^{\text{m}} \subseteq \mathcal{B}$，即其位置和方向（由其底面在 <style id='b'>b $\mathcal{B}^{\text{nm}} \subseteq \mathcal{B}$ 中的顶点表示）可由机器人改变的物体。

假设机器人可以在每个方块的顶面上移动。我们将这些表面称为平面。形式上，与方块 b_i 关联的平面定义为

$$\mathcal{P}_i = \{\mathbf{q} \in \mathbb{R}^3 \mid g_j(\mathbf{q}) \leq 0, \forall j \in \{1, \ldots, s\}\}, \tag{2}$$

其中 $g_j(\mathbf{q})$是由块顶面多边形的边界确定的内向半空间的线性不等式。[1]为简化表述，本文还假设可移动块的顶面 b_i （因此其高度 h_i）保持不变。这意味着，尽管可移动块b_i 的位置和方向（即$\mathbf{Q}_i$）可以改变，但该块不能翻转。[2]

不可移动块对应的平面收集在集合 $\mathcal{P}^{\text{nm}} = \{\mathcal{P}_i \mid b_i \in \mathcal{B}^{\text{nm}}\}$中。在集合 $\mathcal{P}^{\text{nm}}$ 中，我们还包含地面平面，记为 $\mathcal{P}_G$，该平面并非由任何块 $b_i \in \mathcal{B}$诱导得到。同样，我们将 $\mathcal{P}^{\text{m}} = \{\mathcal{P}_i \mid b_i \in \mathcal{B}^{\text{m}}\}$ 定义为与可移动块对应的平面集合。时间 t时由所有块 b_i确定的环境结构记为 $\Omega(t)$。图2展示了此类环境的一个示例。

B. 机器人动力学

我们考虑一个全向机器人，其构型完全由其位置$\mathbf{x} \in \mathcal{X} \subseteq \mathbb{R}^3$决定，其中 $\mathcal{X}$表示机器人的构型（位置）空间，由所有平面上的可通行区域组成（$\mathcal{P}^{\text{nm}}$ 和 $\mathcal{P}^{\text{m}}$）。其底层运动动力学通过一个函数进行抽象，该函数用于检查两个位置之间是否存在动力学上可行且无碰撞的路径，定义如下。

定义 *2.1*（函数`NavigReach`）：函数 NavigReach : $\mathcal{X} \times \mathcal{X} \times \Omega \rightarrow \{\text{True}, \text{False}\}$ 以两个机器人位置$\mathbf{x}_i$, $\mathbf{x}_j \in \mathcal{X}$ 和当前环境状态 $\Omega(t)$ 作为输入。如果存在一条

[1]每个$g_j(\mathbf{q})$的形式为$g_j(\mathbf{q}) = \mathbf{a}_j \cdot (\mathbf{q} - \mathbf{q}_0) + \mathbf{b}_j$，其中$\mathbf{a}_j$是边 j的平面内法向量，$\mathbf{q}$ 是测试点，而$\mathbf{q}_0$ 是平面上的一个参考点。

[2]通过允许所提出的基于采样的规划器不仅在块位姿的连续空间（由 $\mathbf{Q}_i$ 表示）中搜索，还在块高度的离散空间中搜索，可以放宽这一假设。

从 $\mathbf{x}_i$ 到 $\mathbf{x}_j$ 在 $\Omega(t)$ 中的动态可行无碰撞路径；否则，它返回 False。

机器人还配备了一个机械臂，使其能够 Pick 和 Place 方块。我们将低层级的机械臂动力学抽象为以下高层级函数：

定义 *2.2*（函数*ManipReach*）：函数 ManipReach：$\mathcal{B}^{\mathrm{m}} \times \Omega \to \mathcal{X}$ 以一个可移动块 $b_i \in \mathcal{B}^{\mathrm{m}}$ 和环境状态 $\Omega(t)$ 作为输入，并返回机器人构型的集合 $\mathcal{X}_{\mathrm{manip}} \subseteq \mathcal{X}$；在 $\Omega(t)$ 中满足运动学和动力学约束的前提下，机器人可以从这些构型出发与 b_i 进行可行交互。因此，给定 $\Omega(t)$ 中方块 b_i 的构型 $\mathbf{b}_i$，该函数返回机器人可以从中抓取 b_i 或将其放置 b_i 到同一构型 $\mathbf{b}_i$的所有机器人构型。

NavigReach 和ManipReach的实现细节见第 VI-A节。

C. 任务规范

机器人最初位于平面 $\mathcal{P}_i$ 之一，并被要求到达位于平面 $\mathcal{P}_j$（$\mathcal{P}_i \neq \mathcal{P}_j$）上的指定目标状态 $\mathbf{x}_{\mathrm{goal}} \in \mathcal{X}$。我们关注这样的情况：由于机器人动力学不允许其直接移动或跳跃到所需平面，初始机器人位置 $\mathbf{x}_{\mathrm{start}}$无法到达$\mathbf{x}^{\mathrm{goal}}$；即，NavigReach($\mathbf{x}_{\mathrm{start}}$,$\mathbf{x}^{\mathrm{goal}}$,$\Omega(0)$) = False。此时，机器人不能报告任务失败，而必须确定如何重新配置环境。

为对此进行形式化，我们定义操作动作 Move($\mathbf{b}_i$,$\mathbf{b}'_i$)，它将块 b_i 从当前构型 $\mathbf{b}_i$ 移至新构型 $\mathbf{b}'_i$。执行此类动作需要机器人导航至 $\mathcal{X}_{\mathrm{manip}}$= ManipReach($b_i$,$\Omega(t)$) 中的某个构型，使其能够抓取 b_i，将其运送至 $\mathbf{b}'_i$并放置在那里。为保留所有方块的可移动性，我们要求可移动方块只能放置在不可移动平面上，且可移动方块能够完全置于不可移动方块的顶部。目标是计算操作计划 $\tau = \langle \tau(1), \ldots, \tau(k), \ldots, \tau(H) \rangle$，其中每个 $\tau(k)$ = Move($\mathbf{b}_i$, $\mathbf{b}'_i$) 表示第 k 个操作动作，$k \in \{1, \ldots, H\}$，其中时域为 $H \geq 1$。执行 τ 后，机器人到达状态 $\mathbf{x}'$，该状态属于 $\Omega(T)$，且从该状态可以到达 $\mathbf{x}^{\mathrm{goal}}$，即对于某个 $T > 0$，NavigReach($\mathbf{x}'$, $\mathbf{x}^{\mathrm{goal}}$, $\Omega(T)$) = True。请注意，$k \in \{1, \ldots, H\}$ 表示规划 τ中的动作索引，而 $t \in [0, T]$ 表示连续时间。例如，执行第 H 个动作时，已经过了 T 个时间单位。

问题*1*：给定 NavigReach、ManipReach、一个初始位置 $\mathbf{x}_{\mathrm{start}}$、一个初始环境状态 $\Omega(0)$ 和一个目标状态 $\mathbf{x}_{\mathrm{goal}} \in \mathcal{X}$，计算一个操作计划 τ，使得执行后，机器人能够到达 $\mathbf{x}^{\mathrm{goal}}$。

注 *2.3*（复杂块结构）：只要NavigReach和ManipReach能够处理相关情况，我们的方法就能支持具有任意几何形状的方块（例如，具有非平坦顶面或非垂直侧面的方块）。具体而言，NavigReach 必须能够推理复杂表面和地形的可通行性，而ManipReach 必须能够评估放置在不平坦地形上的物体的静态稳定性。对于函数候选函数（在第III-A节中引入）也有类似要求。扩展我们的框架，使其适用于更多非结构化环境，需要设计这些高层抽象；这超出了本文的研究范围，将留待未来研究。

III. 环境重构的交互式规划器

本节介绍 **BRiDGE**，这是我们用于解决问题 1 的基于采样的规划框架。首先，在第 III-A 节中，我们介绍 **BRiDGE** 所使用的关键定义和效用函数。然后，在第 III-B 节中，我们描述用于生成操作计划的核心基于采样的规划器 τ，以解决问题1。

A. 规划器的设置

我们首先介绍将在 **BRiDGE** 中使用的关键定义和效用函数，以补充定义 2.1 和 2.2。具体而言，我们定义两个平面 $\mathcal{P}_i$ 和 $\mathcal{P}_j$ 何时两两不可达，即机器人无法从$\mathcal{P}_i$移动到 $\mathcal{P}_j$，而不违反动力学约束或进入障碍区域。

定义 *3.1*（成对不可达平面）：令 $\mathcal{P}_i$ 和 $\mathcal{P}_j$ 为任意两个不同的平面。若对于所有机器人位置对 $\mathbf{x}_i$、$\mathbf{x}_j \in \mathcal{X}$（其对应的点分别位于 $\mathcal{P}_i$ 和 $\mathcal{P}_j$ 上），NavigReach($\mathbf{x}_i$、$\mathbf{x}_j$、$\Omega(t)$) =False，则称 $\mathcal{P}_j$ 从 $\mathcal{P}_i$ 不可访问/不可达。

为简化表述，我们假设这一关系是对称的，即如果 $\mathcal{P}_j$ 无法从 $\mathcal{P}_i$ 到达，则反之亦然。如果 $\mathcal{P}_j$ 无法从 $\mathcal{P}_i$ 到达，则称它们之间存在一个“间隙”。我们关注的只是由不可移动平面形成的结构布局中的间隙。也就是说，我们考虑 $\mathcal{P}_i \in \mathcal{P}^{\mathrm{nm}}$ 和 $\mathcal{P}_j \in \mathcal{P}^{\mathrm{nm}}$ 之间将使用可移动方块桥接的间隙。不可移动方块之间的间隙通过以下函数进行识别：

定义 *3.2*（函数 *Gap*）：函数 Gap： $\mathcal{P}^{\mathrm{nm}} \times \mathcal{P}^{\mathrm{nm}} \times \Omega \to \{$True、False$\}$ 当且仅当 $\mathcal{P}_i \in \mathcal{P}^{\mathrm{nm}}$ 和 $\mathcal{P}_j \in \mathcal{P}^{\mathrm{nm}}$ 之间存在间隙（如定义 3.1所述）时返回 True；否则返回False。

接下来，我们介绍一个函数，用于确定哪些可移动方块可以通过重新放置，潜在地用于跨越 $\mathcal{P}_i$ 和 $\mathcal{P}_j$之间的间隙。

定义 *3.3*（函数候选函数）：函数 Candidate：($\mathcal{P}^{\mathrm{nm}}$, $\mathcal{P}^{\mathrm{nm}}$) $\to \{(b_n, \mathcal{P}_k) \mid b_n \in \mathcal{B}^{\mathrm{m}}, \mathcal{P}_k \in \mathcal{P}^{\mathrm{nm}}\}$ 以一对彼此不可达的不可移动平面 ($\mathcal{P}_i$, $\mathcal{P}_j \in \mathcal{P}^{\mathrm{nm}}$) 作为输入，并返回所有满足以下条件的元组 ($b_n$, $\mathcal{P}_k$) 的集合：存在至少一种将 b_n 放置在 $\mathcal{P}_k$ 上的放置配置；若实现该配置，则可能使机器人（由 NavigReach 判定）能够从 $\mathcal{P}_i$ 上的某个构型到达 $\mathcal{P}_j$ 上的某个构型，并将 b_n 的顶面用作中间平面。

请注意，候选函数不会返回可移动块的具体放置构型 b_n 位于不可移动平面上 $\mathcal{P}_k$。该函数也不必做到完美：它可以返回元组(b_n, $\mathcal{P}_k$)，但实际上不存在任何将 b_n 放置在 $\mathcal{P}_k$ 上并真正桥接所需间隙的方式。这些错误候选项之后会通过可行性检查筛除（见算法2）。我们对该函数作出如下假设：

假设*3.4*（候选集过近似）：Candidate($\mathcal{P}_i$, $\mathcal{P}_j$) 返回的集合必须包含所有能够在 ($\mathcal{P}_i$, $\mathcal{P}_j$) 之间实现间隙桥接放置的元组 (b_n, $\mathcal{P}_k$)。换言之，Candidate 可以包含假阳性，但不得排除任何真正可用于桥接间隙的选项。

一种保守的设计候选函数以满足假设3.4 是返回所有可能的元组 (b_n、$\mathcal{P}_k$)。在第 VI-A节中，我们将介绍一种不那么保守的间隙和候选函数实现，该实现利用了所有方块的右多棱柱结构。

B. 基于采样的规划器

为了计算 τ 解决问题1的规划，我们提出了一种基于采样的方法，该方法逐步构建一棵树，以探索机器人和方块构型的空间。所提出的算法总结于算法 1中。

下面我们直观地介绍 BRiDGE规划器算法 1 的各个步骤。该树定义为 $\mathcal{T}=(\mathcal{V}_T, \mathcal{E}_T)$，其中 $\mathcal{V}_T$ 和 $\mathcal{E}_T \subseteq \mathcal{V}_T \times \mathcal{V}_T$ 分别是节点集和有向边集。节点 $v \in \mathcal{V}_T$ 写作 $v = [\Omega^v$、$\mathbf{x}^v]$，其中 Ω^v 表示环境结构（即所有不可移动和可移动方块的构型），而 $\mathbf{x}^v \in \mathcal{X}$ 是与该节点关联的机器人位置。如果对某个方块 b_i 应用一次操作动作 Move($\mathbf{b}_i$, $\mathbf{b}_i'$)，使 Ω^v 转换为 $\Omega^{v'}$，则存在有向边 $(v, v') \in \mathcal{E}_T$。略微滥用记号，我们将该重构写作 $\Omega^{v'} = \Omega^v \oplus$Move($\mathbf{b}_i$, $\mathbf{b}_i'$)，其中 $\oplus$ 表示环境更新：将 b_i's 的构型 $\mathbf{b}_i$ 替换为 $\mathbf{b}_i'$。该树以 $v_{\text{root}} = [\Omega(0)$、$\mathbf{x}_{\text{start}}]$ 为根，并以初始集合 $\mathcal{V}_T = \{v_{\text{root}}\}$ 和 $\mathcal{E}_T = \emptyset$ 开始（算法 1 第 1 行）。该树通过交替执行采样和扩展步骤进行迭代生长。在每次迭代 κ 中，我们（i）采样一个现有的树节点 $v_{\text{rand}} \in \mathcal{V}_T$；（ii）为某个可移动方块 b_i 采样一个操作动作 Move($\mathbf{b}_i$, $\mathbf{b}_i'$)；（iii）尝试将该动作应用于 v_{rand}，以获得一个候选节点 v_{new}。如果该动作可行（稍后讨论），则将 $v_{\text{new}} = (\Omega^{v_{\text{new}}}, \mathbf{x}^{v_{\text{new}}})$ 添加到 $\mathcal{V}_T$，并将有向边 $(v_{\text{rand}}, v_{\text{new}})$ 添加到 $\mathcal{E}_T$。每次成功扩展并生成 v_{new} 后，我们都会测试目标是否可访问，即 NavigReach($\mathbf{x}^{v_{\text{new}}}, \mathbf{x}_{\text{goal}}, \Omega^{v_{\text{new}}}$) = True。请注意，我们并不要求在最终环境中目标能够从根节点 $\mathbf{x}_{\text{start}}$ 到达，而是要求它能够从包含于 v_{new} 中的当前机器人位置到达。

接下来，我们将更详细地描述 Alg. 1中使用的采样和扩展步骤。图 3 展示了正在构建的树的可视化结果。

采样树节点： 我们按照离散质量函数 $v_{\text{rand}} = [\Omega^{v_{\text{rand}}}$、$\mathbf{x}^{v_{\text{rand}}}] \in \mathcal{V}_T$ 对树节点进行采样 $f_{\mathcal{V}}: \mathcal{V} \to (0, 1)$（第 6 行，Alg. 1）。质量函数可以任意选择，但必须为每个节点分配非零概率（例如，均匀采样）。

采样操作动作： 接下来，我们采样一个动作 Move($\mathbf{b}_i$, $\mathbf{b}_i'$) 并将其应用于节点 v_{rand}，生成一个新节点 v_{new} 和边 (v_{rand}, v_{new})（第 7 行，Alg. 1）。该动作的采样过程是分层进行的，如下所述；参见 Alg.2。

(i) 采样一个高层三元组：首先，我们采样一个三元组

$$s = (b_i, \mathcal{P}_k, g), \tag{3}$$

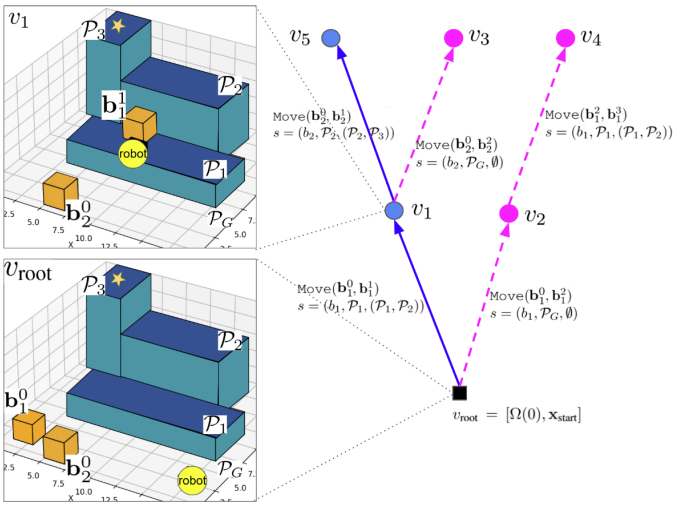


图 3. 增量式树构建的图示；树节点用 v_i 表示。每个节点表示机器人和方块的联合状态，即 [Ω, x],；每条边对应一个操作动作 Move(b_i, b_i')。因此，子节点与其父节点的区别在于单个块的位置和机器人构型。图的右侧展示了由所提出的规划器构建的树。实线蓝色边遵循规划 Π（参见第 IV-A 节）；其余边为虚线粉色边。在发现 v_5 之前，$\mathcal{V}_\Pi^{\max}$ 仅包含 v_1，因为它是唯一一个使用 Π 实现的节点。v_5 被添加到树后，$\mathcal{V}_\Pi^{\max}$ 立即仅包含节点 v_5。左侧展示了 v_{root} 和 v_1 处的环境状态和机器人状态。环境由四个不可移动平面组成：地面平面 $\mathcal{P}_G$ 以及由三个不可移动（蓝色）块形成的平面 $\mathcal{P}_1$、$\mathcal{P}_2$ $\mathcal{P}_3$ 和。目标（黄色星形）位于 $\mathcal{P}_3$ 上。有两个可移动的（橙色）块可供使用，记为 $\mathbf{b}_i^d$；上标 d 表示该块的第 d 个放置。

其中，$b_i \in \mathcal{B}^{\text{m}}$ 是待重新定位的可移动块，$\mathcal{P}_k \in \mathcal{P}^{\text{nm}}$是放置平面（即用于放置 b_i 的不可移动平面），而 $g \in (\mathcal{P}^{\text{nm}} \times \mathcal{P}^{\text{nm}}) \cup \varnothing$ 则编码了此次重新定位的意图。具体而言，$g = (\mathcal{P}_a, \mathcal{P}_b)$ 表示将 b_i放置在 $\mathcal{P}_k$ 上，以弥合 $\mathcal{P}_a$ 与 $\mathcal{P}_b$之间的间隙；而 $g = \varnothing$ 表示不具有跨隙作用的重新定位。

为生成这样的三元组 s，我们首先构造集合$\mathcal{S}$，该集合收集所有形如（3）的可能三元组。该集合包括：(i) 对于每个 $b_i \in \mathcal{B}^{\text{m}}$、$\mathcal{P}_k \in \mathcal{P}^{\text{nm}}$以及不可移动平面 $\mathcal{P}_a$、$\mathcal{P}_b$，满足 Gap($\mathcal{P}_a, \mathcal{P}_b, \Omega(0)$) = True 且 $(b_i, \mathcal{P}_k) \in$候选函数($\mathcal{P}_a, \mathcal{P}_b$) 的 $s = (b_i, \mathcal{P}_k, (\mathcal{P}_a, \mathcal{P}_b))$；以及 (ii) 对于每个$b_i \in \mathcal{B}^{\text{m}}$ 和所有 $\mathcal{P}_k \in \mathcal{P}^{\text{nm}}$，$s = (b_i, \mathcal{P}_k, \varnothing)$。集合 $\mathcal{S}$ 在树的初始化阶段构造（第 2 行，算法 1）。

然后，我们根据质量函数 $s \in \mathcal{S}$ 对一个三元组进行采样 $f_{\mathcal{S}}$：$\mathcal{S} \to (0,1)$。函数 $f_{\mathcal{S}}$ 可以任意选择，只要它为每个$s \in \mathcal{S}$（例如，均匀分布）分配非零概率（第1行，算法2）。

(ii) 采样下一个区块配置：采样得到的三元组 s 决定用于扩展树节点 v_{rand} 的块 b_i。给定意图 g 和放置平面 $\mathcal{P}_k$，我们对 b_i 采样一个候选配置 $\mathbf{b}_i'$，以定义动作 Move($\mathbf{b}_i$, $\mathbf{b}_i'$)。具体而言，给定 v_{rand} 和三元组 s，我们最多进行 $N \geq 1$ 次独立试验，以生成 $\mathbf{b}_i'$。只有当由此产生的动作 Move($\mathbf{b}_i$,$\mathbf{b}_i'$) 可由机器人执行（依据定义 2.1 和 2.2），且满足意图 g 时，该次试验才算成功。采样 $\mathbf{b}_i'$ 并验证可行性的过程包括以下三个步骤。

(ii.1) 采样机器人位置并检查可行性。我们

首先计算抓取集 $\mathcal{X}_{\text{grasp}} = \text{ManipReach}(b_i, \Omega^{v'})$，即机械臂能够在构型 $\mathbf{b}_i$下抓取 b_i 的机器人构型集合。随后，根据分布 $f_{\mathcal{X}_{\text{grasp}}}$:$\mathcal{X}_{\text{grasp}} \rightarrow \mathbb{R}\geq 0$ 采样一个抓取构型 $\mathbf{x}^{\text{grasp}}$（算法 2 第 3 行）。接着，我们检查 NavigReach($\mathbf{x}^{v_{\text{rand}}}$, $\mathbf{x}^{\text{grasp}}$,$\Omega^{v_{\text{rand}}}$) = True 是否成立，即从当前机器人位置 $\mathbf{x}^{v_{\text{rand}}}$ 到抓取位姿 $\mathbf{x}^{\text{grasp}}$ 是否存在可行路径（算法 2 第 4 行）。如果不满足此条件，则丢弃本次尝试；否则，继续下一步。

(ii.2) 采样下一个区块配置并检查意图实现。其次，我们在放置平面 $\mathcal{P}_k$ 上采样块的下一个构型 $\mathbf{b}'_i$ 。令 $\mathcal{C}^{v_{\text{rand}}}\text{block}(b_i, \mathcal{P}_k, \Omega^{v_{\text{rand}}})$ 表示块 b_i 的物理有效位姿构型空间，其中块的底面完全位于环境 $\Omega^{v_{\text{rand}}}$ 中的 $\mathcal{P}_k$ 上（将 b_i 视为被携带物体进行评估）。我们从分布 $f_{\mathcal{C}^v_{\text{block}}}$ 中采样 $\mathbf{b}'_i$: $\mathcal{C}^{v_{\text{rand}}}\text{block} \rightarrow \mathbb{R}_{\geq 0}$（算法 2 第 5 行），并通过定义的更新构造新环境: $\Omega^{\text{temp}} = \Omega^{v_{\text{rand}}} \oplus \text{Move}(\mathbf{b}_i, \mathbf{b}'_i)$（算法 2 第 6 行）。请注意，$\Omega^{\text{temp}}$ 与 $\Omega^{v_{\text{rand}}}$ 之间唯一的区别在于 b_i 的构型。给定 Ω^{temp}，我们验证动作 $\text{Move}(\mathbf{b}_i, \mathbf{b}'_i)$ 是否实现意图 g。具体而言，如果 $g = (\mathcal{P}_a, \mathcal{P}_b)$，我们检查 $\text{Gap}(\mathcal{P}_a, \mathcal{P}_b, \Omega^{\text{temp}}) = \text{False}$，即检查 $\mathcal{P}_a$ 与 $\mathcal{P}_b$ 之间的间隙是否已被连通（算法 2 第 7 行）。如果 $g = \varnothing$，则意图自然得到满足。如果意图 g 未实现，则舍弃该次尝试；否则，继续第三步。

(ii.3) 对放置位置进行采样并检查搬运轨迹可行性。第三，我们计算所有可使机械臂在 $\mathbf{b}'_i$处放置 b_i 的机器人构型集合 $\mathcal{X}_{\text{drop}}$，即 $\mathcal{X}_{\text{drop}}$ =ManipReach(b_i,Ω^{temp})。然后，我们从分布 $f_{\mathcal{X}_{\text{drop}}}$ 中采样 $\mathbf{x}^{\text{drop}}$： $\mathcal{X}_{\text{drop}} \rightarrow \mathbb{R}_{\geq 0}$（算法 2 第 8 行）。接着，我们检查 NavigReach($\mathbf{x}^{\text{grasp}}$,$\mathbf{x}^{\text{drop}}$,$\Omega^{\text{temp}}$) = 是否为 True，也就是检查是否存在一条从块的抓取位置 $\mathbf{x}^{\text{grasp}}$ 到放置位置 $\mathbf{x}^{\text{drop}}$ 的可行路径（算法 2 第 9 行）。如果满足此条件，则当前试验成功，随后进行扩展步骤；否则，重复上述过程。

树扩展：一旦某次试验成功，我们便构造新节点 $v_{\text{new}} = [\Omega^{v_{\text{new}}}, \mathbf{x}^{v_{\text{new}}}]$，其中 $\Omega^{v_{\text{new}}}$ =Ω^{temp}，且 $\mathbf{x}^{v_{\text{new}}} = \mathbf{x}^{\text{drop}}$（算法 1 第 9 行）。然后，我们将节点集和边集分别更新为 $\mathcal{V}_T \rightarrow \mathcal{V}_T \cup \{v_{\text{new}}\}$ 和$\mathcal{E}_T \rightarrow \mathcal{E}_T \cup \{(v_{\text{rand}}, v_{\text{new}})\}$（算法 1 第 9 行）。将边 $(v_{\text{rand}}, v_{\text{new}})$ 添加到树后，我们会存储相应的操作动作 $\text{Move}(\mathbf{b}_i, \mathbf{b}'_i)$，以及使其能够执行的机器人抓取位置和放置位置 $\mathbf{x}^{\text{grasp}}$ 与 $\mathbf{x}^{\text{drop}}$。具体而言，我们通过迭代构造函数 $\mathcal{A}$，将 $\mathcal{E}_T$ 中的每条边映射到其对应的动作及相关机器人位置（算法 1 第 10 行）。接下来，我们检查 NavigReach($\mathbf{x}^{v_{\text{new}}}$, $\mathbf{x}^{\text{goal}}$, $\Omega^{v_{\text{new}}}$) = 是否为 True，即检查在 $\Omega^{v_{\text{new}}}$ 中能否从 $\mathbf{x}^{v_{\text{new}}}$ 到达目标状态（算法 1 第 11 行）。若可以，算法 1 即终止并返回规划（其提取方式将在下一段中说明）（算法 1 第 12 行）；否则，迭代索引递增 $\kappa \leftarrow \kappa + 1$, 并重复该过程。如果所有 N 次用于采样有效动作 $\text{Move}(\mathbf{b}_i, \mathbf{b}'_i)$的试验均失败，则迭代索引仍会递增$\kappa \leftarrow \kappa + 1$，但不更新树。

计划提取： 一旦在 v_{new}处通过目标测试，我们就通过从 τ 回溯至1 中的问题 v_{new}，沿着 v_{root} 到 $\mathcal{E}_T$，并恢复相应的

算法1：BRiDGE：基于采样的规划器

输入： 初始状态（$\mathbf{x}_{\text{start}}$，$\Omega(0)$），目标状态 $\mathbf{x}^{\text{goal}}$
输出： 动作序列 τ，用于到达$\mathbf{x}^{\text{goal}}$

$\mathcal{V}_T \leftarrow \{ v_{\text{root}} = [\Omega(0), \mathbf{x}_{\text{start}}] \}$; $\mathcal{E}_T \leftarrow \emptyset$; $\mathcal{A} \leftarrow \emptyset$;
构建 $\mathcal{S}$;
$\kappa \leftarrow 0$;
当 $\kappa < K_{\max}$ **do**
  $\kappa \leftarrow \kappa + 1$;
  采样 $v_{\text{rand}} = [\Omega^{v_{\text{rand}}}, \mathbf{x}^{v_{\text{rand}}}] \in \mathcal{V}_T$通过 $f_{\mathcal{V}}$;
  $[\Omega^{v_{\text{new}}}, \mathbf{x}^{\text{grasp}}, \mathbf{x}^{v_{\text{new}}}, (\mathbf{b}_i, \mathbf{b}'_i), \text{成功}]$ = 移动(v_{rand}, $\mathcal{S}$);
  如果 成功=为真 **则**
    $v_{\text{new}} \leftarrow [\Omega^{v_{\text{new}}}, \mathbf{x}^{v_{\text{new}}}]$; 将 v_{new} 添加到 $\mathcal{V}_T$ 和边 $(v_{\text{rand}}, v_{\text{new}})$ 到 $\mathcal{E}_T$;
    $\mathcal{A}[(v_{\text{rand}}, v_{\text{new}})] \leftarrow$ (移动($\mathbf{b}_i$, $\mathbf{b}'_i$), $\mathbf{x}^{\text{grasp}}$, $\mathbf{x}^{v_{\text{new}}}$);
    如果 *NavigReach*($\mathbf{x}^{v_{new}}$,$\mathbf{x}^{\text{goal}}$,$\Omega^{v_{new}}$) **则**
      返回回溯规划 τ 使用 $\mathcal{A}$;

算法 2： 采样移动

输入： 树节点 v_{rand}，三元组集合 $\mathcal{S}$
输出： Ω^{temp}，$\mathbf{x}^{\text{grasp}}$，$\mathbf{x}^{\text{drop}}$，$(\mathbf{b}_i, \mathbf{b}'_i)$，成功标志

通过 $f_{\mathcal{S}}$ 对 $s = (b_i, \mathcal{P}_k, g)$ 进行采样;
对 $\ell = 1$ **到** N **执行**
  在 $\mathbf{x}_{\text{grasp}} \sim f_{\mathcal{X}_{\text{grasp}}}$ 上采样 $\mathcal{X}_{\text{grasp}}$=ManipReach($b_i$, $\Omega^{v_{\text{rand}}}$);
  如果 ¬*NavigReach*($\mathbf{x}^{v_{rand}}$, $\mathbf{x}^{\text{grasp}}$, $\Omega^{v_{rand}}$) **则继续**;
  在有效位姿上对$\mathbf{b}'_i \sim f_{\mathcal{C}^v_{\text{block}}}$ 进行采样，范围为 $\mathcal{P}_k$;
  $\Omega_{\text{temp}} \leftarrow \Omega^{v_{\text{rand}}} \oplus$ 移动($\mathbf{b}_i$, $\mathbf{b}'_i$);
  如果$(g \neq \emptyset) \wedge$(间隙($\mathcal{P}_a$, $\mathcal{P}_b$, Ω^{temp})) **则继续**;
  采样 $\mathbf{x}_{\text{drop}} \sim f_{\mathcal{X}_{\text{drop}}}$ on $\mathcal{X}_{\text{drop}}$=ManipReach($b_i$, Ω^{temp});
  如果 ¬*NavigReach*($\mathbf{x}^{\text{grasp}}$, $\mathbf{x}^{\text{drop}}$, Ω^{temp}) **则继续**;
  返回 Ω^{temp},$\mathbf{x}^{\text{grasp}}$,$\mathbf{x}^{\text{drop}}$,$(\mathbf{b}_i,\mathbf{b}'_i)$,真;
返回 $\emptyset, \emptyset, \emptyset, \emptyset$, 假;

通过函数 $\mathcal{A}$ 获取操作动作。请注意，τ 指定了动作序列 $\tau(k)$，但不包括执行这些动作所需的机器人轨迹。然而，在步骤 (ii.1)–(ii.3) 中进行的可行性检查保证存在连接抓取和放置位置（可通过 $\mathcal{A}$ 恢复）的可行轨迹，以执行每个动作 $\tau(k)$。

IV. 非均匀采样策略

本节讨论如何构造用于实现**BRiDGE**（算法1）采样策略的质量函数和分布。一个直接的选择是对 $f_{\mathcal{V}}$、$f_{\mathcal{S}}$、$f_{\mathcal{X}_{\text{grasp}}}$、$f_{\mathcal{C}^v_{\text{block}}}$和 $f_{\mathcal{X}_{\text{drop}}}$采用均匀质量函数和分布。然而，这可能会减慢规划过程，进而降低构造可行计划的速度，尤其是方块/平面的数量增加时；详见第VI节。为缩短首次求解时间，我们设计了

非均匀采样策略，分别用于：(i) 节点选择 $f_{\mathcal{V}}$、(ii) 三元组选择 $f_{\mathcal{S}}$和 (iii) 块位姿采样 $f_{\mathcal{C}_{\text{block}}^{v}}$；其他分布均采用均匀分布。在第IV-A节中，我们联合设计了用于 $f_{\mathcal{V}}$和 $f_{\mathcal{S}}$ 的非均匀采样策略；在第IV-B 节中，我们讨论 $f_{\mathcal{C}_{\text{block}}^{v}}$的设计。在第IV-C节中，我们讨论如何在非均匀采样策略偏向不可行方向时，对其进行在线调整。

A. 设计 $f_{\mathcal{V}}$ 和 $f_{\mathcal{S}}$

我们用于 $f_{\mathcal{V}}$ 和 $f_{\mathcal{S}}$ 的非均匀采样策略由符号计划 Π 决定，该计划定义为一系列高层导航和操作动作，且不会在每一步验证其可行性。

1) BFS规划器（Π）：规划 Π 将使用广度优先搜索（BFS）规划器构建。该规划器可在任意迭代 κ 和树节点$v_\Pi = (\Omega_\Pi, \mathbf{x}_\pi) \in \mathcal{V}$处调用。最初，它在迭代$\kappa = 0$ 从根节点 $v_\Pi = v_{\text{root}}$ （即 $\Omega_\Pi = \Omega_{\text{start}}$且 $\mathbf{x}_\pi = \mathbf{x}_{\text{start}}$）调用，以寻找规划 Π 。该规划定义为一系列三元组，即 $\Pi = \langle s_1, \ldots\ldots, s_M \rangle$，其中 $s_m = (b_i, \mathcal{P}_k, g)$，若执行该规划，目标区域便可能对机器人变得可达。执行 $s_m = (b_i, \mathcal{P}_k, g)$是指机器人将可移动块 b_i 放置到不可移动平面 $\mathcal{P}_k$ 上，以实现意图 g。请注意，此符号计划既未指定 b_i的确切放置位姿，其存在也不能保证存在将 b_i 放置到 $\mathcal{P}_k$ 上以实现意图 g的可行放置，或保证机器人必然能够到达 $\mathcal{P}_k$ 。事实上，构建 Π时会忽略所有平面/块几何和机器人坐标；几何执行的可行性（可达性、无碰撞放置和路径存在性）则留待算法 1 处理。计算 Π的目的是确定有希望的高层方向，具体而言，是确定应移动哪些块以及可将它们放置到哪些平面上的顺序，以用于构建非均匀分布 $f_{\mathcal{V}}$ 和 $f_{\mathcal{S}}$。由于其符号性质，最终得到的规划 Π 可能不可行且不可执行；后文还将讨论 BFS 规划器如何在后续迭代 $\kappa > 0$ 中从节点 $v_\Pi \in \mathcal{V}$ 重新规划，以修订 Π，进而改进 $f_{\mathcal{V}}$ 和 $f_{\mathcal{S}}$。

接下来，我们将讨论 BFS规划器如何生成规划 Π，即在迭代 $\kappa = 0$ 时。从节点 v_Π 出发，它会搜索符号层面的高层动作序列，以求解问题1。与算法1不同，后者将每个 $\texttt{Move}(\mathbf{b}_i, \mathbf{b}_i')$ 视为一个单独的高层（拾取—放置）动作，BFS规划器采用更精细的动作抽象：每一步只执行一个基本动作——*GoTo*⟨*Plane*⟩（移动到相邻平面）；*Pick*⟨*Block*⟩（拾取当前平面上的一个块）；或*Place*⟨*Block*⟩（将手持的块放置在当前平面上）。这种分解使其能够在不验证几何关系的情况下，对较长的动作序列进行推理。

BFS规划器会探索一个图 $\mathcal{G} = (\mathcal{W}, \mathcal{E})$，其中$\mathcal{W}$ 和 $\mathcal{E}$ 分别表示该图的节点集和边集。具体而言，$\mathcal{W}$ 由符号状态 $w = (\mathcal{P}_{\text{cur}}, b_h, L) \in \mathcal{W}$ 组成，其中 $\mathcal{P}_{\text{cur}} \in \mathcal{P}^{\text{nm}}$ 是机器人在该符号状态下所处的不可移动平面；$b_h \in \mathcal{B}^{\text{m}}$是机器人当前携带的块，或者当机器人未携带任何物体时为 $\mathcal{P}_k$；而 L 是一个函数，用于为任意块 b 赋予符号状态。形式化地说，$L: \mathcal{B}^{\text{m}} \to \{\text{held}\} \cup \{(\mathcal{P}_k, g) \mid \mathcal{P}_k \in \mathcal{P}^{\text{nm}}, g \in \{\emptyset\} \cup \{(\mathcal{P}_a, \mathcal{P}_b)\}\}$。换言之，如果块 b 放置在 $\mathcal{P}_k$ 上，且意图为 g，那么 $L(b) = (\mathcal{P}_k, g)$；如果块 b 正由机器人携带（即 $b = b_h$），那么 $L(b) = \text{held}$。我们定义初始节点 $w_0 = (\mathcal{P}_{\text{start}}, \emptyset, L)$，它使用 v_Π构建；因此，在 $\kappa = 0$处，$\mathcal{P}_{\text{start}}$ 是 Ω_Π 中包含 $\mathbf{x}_\Pi$ 的不可移动平面。要确定未被机器人携带的块 b 的状态 L，只需检查移除 b 是否会影响任意两个不可移动平面之间的可达性。对于当前满足 $\text{Gap}(\mathcal{P}_i, \mathcal{P}_j, \Omega_\Pi) = \text{False}$ 的任意一对 $\mathcal{P}_i$ 和$\mathcal{P}_j$ ，我们评估从块 b 当前所在的平面 $\mathcal{P}_k$移除该块后，是否会导致 $\text{Gap}(\mathcal{P}_i, \mathcal{P}_j, \Omega_{\Pi-b}) = \text{True}$，其中 $\Omega_{\Pi-b}$ 是完全移除块 b后的环境 Ω_Π 。若结果为是，则认为 b对于确保 $\mathcal{P}_i$ 与 $\mathcal{P}_j$ 之间的可达性至关重要，并将 $L(b) = (\mathcal{P}_k, (\mathcal{P}_i, \mathcal{P}_j))$；否则，将 $L(b) = (\mathcal{P}_k, \varnothing)$。对所有块 b重复此过程，以构建与初始节点 w_0关联的映射函数。请注意，随着机器人执行前文介绍的 GoTo、Pick 和 Place 动作，该函数 L 可能会在不同状态 $w \in \mathcal{W}$ 之间发生变化，下一节将对此进行讨论。

边集合（w，w'）$\in \mathcal{E}$ 由上述基本动作（***GoTo***、***Pick***、***Place***）诱导而成。接下来，我们将详细描述符号状态 $w = (\mathcal{P}_{\text{cur}}, b_h, L)$ 下可用的动作，以及对应的下一状态 w'。在任意状态w 下，以下基本动作都是可行的：

- ***GOTO***$(\mathcal{P}')$。此动作指示机器人从当前平面 $\mathcal{P}_{\text{cur}}$移动到平面 $\mathcal{P}'$ 。只有当 $\mathcal{P}'$ 可从 $\mathcal{P}_{\text{cur}}$到达时，才能在节点 w 应用此动作。我们认为，当且仅当满足以下条件之一时，此操作才可行：(i) $\text{Gap}(\mathcal{P}_{\text{cur}}\mathcal{P}', \Omega(0)) = \text{False}$；或 (ii) 存在某个块 b ，使得 $L(b) = \hat{} (\mathcal{P}_k, (\mathcal{P}_{\text{cur}}\mathcal{P}'))$。在 (i) 中，$\Omega(0)$ 表示完全移除所有可移动块后的环境。因此，若满足 (i)，则意味着 $\mathcal{P}_{\text{cur}}$ 与 $\mathcal{P}'$ 无需可移动块的“帮助”即可成对可达（即它们在结构上彼此可达）。条件 (ii) 表明，所考虑的平面通过可移动块 b相连。给定 w 和 ***GOTO***$(\mathcal{P}')$，后继状态为 $w' = (\mathcal{P}', b_h, L)$。
- ***Pick***(b)。如果符号状态 $(b_h = \emptyset)$ 中没有携带任何块，则可以拾取当前平面 $\mathcal{P}_{\text{cur}}$ 上的任意块；也就是说，对于每个满足 $L(b) = (\mathcal{P}_{\text{cur}}\ g)$ 的 b ，以及某个 g，都可以选择动作 ***Pick***(b)。给定w 和 ***Pick***(b)，后继状态为 $w' = (\mathcal{P}_{\text{cur}} b, L)$，其中持有$L(b)$。
- ***Place***$(b_h, \mathcal{P}_{\text{cur}}\ g)$。如果正在携带一个块（$b_h \neq \emptyset$），则可以将其放置在当前平面上，既可以临时放置（$g = \emptyset$），也可以采用任何可接受的桥接意图 $g = (\mathcal{P}_a, \mathcal{P}_b)$，使得 $(b_h, \mathcal{P}_{\text{cur}}) \in$候选函数$(\mathcal{P}_a, \mathcal{P}_b)$。给定 w 和 ***Place***$(b_h, \mathcal{P}_{\text{cur}}\ g)$，后继状态为 $w' = (\mathcal{P}_{\text{cur}}\ \emptyset, L)$，其中 $L(b) = (\mathcal{P}_{\text{cur}}\ g)$。

应用广度优先搜索算法，旨在寻找一条位于 $\mathcal{G}$ 上的路径，将初始节点 $w_0 = (\mathcal{P}_{\text{start}}, \emptyset, L)$ 与满足 $\mathcal{P}_{\text{cur}} = \mathcal{P}_{\text{goal}}$ 的目标节点 $w_g = (\mathcal{P}_{\text{cur}}, b_h, L)$ 连接起来，其中 $\mathcal{P}_{\text{goal}}$ 是

[3]如果某个块对于确保多个对之间的可达性至关重要，然后，我们随机选取一对 $(\mathcal{P}_i, \mathcal{P}_j)$。

包含 $\mathbf{x}_{\text{goal}}$ 的不可移动平面。令 $w_0, \ldots, w_g$ 为 $\mathcal{G}$ 中的最短节点路径。该路径对应于一系列 *GoTo*/*Pick*/*Place* 动作；然而，为了获得 $\Pi = \langle s_1, \ldots, s_M \rangle$ $(s_m = b_i, \mathcal{P}_k, g)$，我们按顺序仅导出 *Place* $(b_i, \mathcal{P}_k, g)$ 动作。

2）非均匀质量函数 $f_{\mathcal{V}}$：接下来，我们设计一个由规划Π引导的非均匀质量函数 $f_{\mathcal{V}}$。为了构造 $f_{\mathcal{V}}$，我们首先需要引入函数 PlanStep：$\mathcal{V} \to \mathbb{N}$。对于任意节点 $v \in \mathcal{V}_T$，当且仅当从 v_Π 到 v 的路径恰好包含 m条边，且这些边依次实现 s_1、……、s_m的步骤时，该函数返回 PlanStep(v) $= m$当且仅当。也就是说，用于对实现从 v_{root} 到达 v 的移动动作进行采样的三元组序列，与 Π中的三元组完全相同，块、放置平面和意图均相匹配 g。否则，我们令 PlanStep(v) $= \varnothing$。

我们还需要引入集合 $\mathcal{V}_\Pi^{\max}$，其定义如下。

$$\mathcal{V}_\Pi^{\max} = \left\{ v \in \mathcal{V}_T \ : \ \text{PlanStep}(v) = \max_{v' \in \mathcal{V}} \text{PlanStep}(v') \right\}.$$

直观地说，$\mathcal{V}_\Pi^{\max}$包含那些沿着进展最远 Π的节点（另见图 3）。

给定 $\mathcal{V}_\Pi^{\max}$，我们定义如下非均匀质量函数：

$$f_{\mathcal{V}}(v) = \begin{cases} \dfrac{p_{\text{plan}}}{|\mathcal{V}_\Pi^{\max}|}, & v \in \mathcal{V}_\Pi^{\max}, \\ \dfrac{1 - p_{\text{plan}}}{|\mathcal{V}_T \setminus \mathcal{V}_\Pi^{\max}|}, & v \in \mathcal{V}_T \setminus \mathcal{V}_\Pi^{\max}, \end{cases} \tag{4}$$

对于某个 $p_{\text{plan}} \in (0,1)$。换言之，以概率 $p_{\text{plan}} \in (0,1)$ 从 $\mathcal{V}_\Pi^{\max}$ 中均匀采样，以概率 $1 - p_{\text{plan}}$ 从 $\mathcal{V}_T \setminus \mathcal{V}_\Pi^{\max}$ 中均匀采样。请注意，p_{plan} 的值越高，$f_{\mathcal{V}}$ 就越倾向于选择节点 $v \in \mathcal{V}_\Pi^{\max}$。选择 $p_{\text{plan}} = \frac{|\mathcal{V}_\Pi^{\max}|}{|\mathcal{V}_T|}$ 即可恢复均匀采样，此时 $f_{\mathcal{V}}$ 在 $\mathcal{V}_T$ 上保持恒定。

3) 非均匀质量 $f_{\mathcal{S}}$：给定所选节点 $v = v_{\text{rand}}$，我们将所有三元组的集合 $\mathcal{S}$ 划分为三个分组。(i) 符号计划 Π推荐的下一个三元组：如果 PlanStep(v) $= m$ 且 Π 中的下一个符号步骤为$s_{m+1} = (b_i, \mathcal{P}_k, g)$，则令 $\mathcal{S}_{\text{plan}}(v) = \{ s_{m+1} \}$；否则，如果 PlanStep($v$) $= \varnothing$，则 $\mathcal{S}_{\text{plan}}(v) = \emptyset$。(ii) 跨越间隙的三元组：$\mathcal{S}_{\text{gap}}(v) = \{(b_i, \mathcal{P}_k, (\mathcal{P}_a, \mathcal{P}_b)) \in \mathcal{S}\}$。(iii) 不跨越间隙的三元组：$\mathcal{S}_{\text{nogap}}(v) = \{(b_i, \mathcal{P}_k, \emptyset) \in \mathcal{S}\}$。注意，$\mathcal{S} = \mathcal{S}_{\text{plan}}(v) \cup \mathcal{S}_{\text{gap}}(v) \cup \mathcal{S}_{\text{nogap}}(v)$。

令 p_1、p_2、$p_3 > 0$，其中 $p_1 + p_2 + p_3 = 1$。给定节点 v，该三元组根据以下质量函数进行采样

$$f_{\mathcal{S}}(s \mid v) = \begin{cases} p_1, & s \in \mathcal{S}_{\text{plan}}(v), \\ \dfrac{p_2}{|\mathcal{S}_{\text{gap}}(v)|}, & s \in \mathcal{S}_{\text{gap}}(v) \\ \dfrac{p_3}{|\mathcal{S}_{\text{nogap}}(v)|}, & s \in \mathcal{S}_{\text{nogap}}(v) \end{cases} \tag{5}$$

注意，如果 $\mathcal{S}_{\text{plan}}(v) = \emptyset$，那么我们通过设置 $p_1 = 0$ 并在剩余分组上重新归一化来重新分配概率质量，使得 p_2，$p_3 > 0$，以及 $p_2 + p_3 = 1$。请注意，如果 $\mathcal{S}_{\text{plan}}(v) \neq \emptyset$，那么 p_1的值越高，$f_{\mathcal{S}}$ 就越倾向于选择遵循规划 Π的三元组。通过设置 $p_1 = |S_{\text{plan}}(v)|/|\mathcal{S}|$，$p_2 = |S_{\text{gap}}(v)|/|\mathcal{S}|$，$p_3 = |S_{\text{nogap}}(v)|/|\mathcal{S}|$，即可恢复均匀三元组采样。

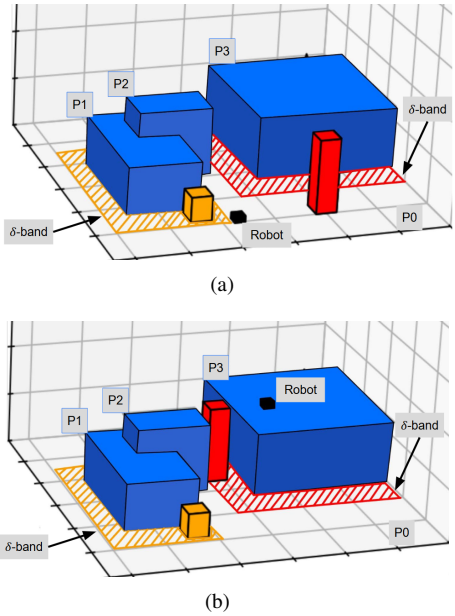


图 4. 展示环境中两个相互独立的间隙（$\mathcal{P}_G$，$\mathcal{P}_1$）和（$\mathcal{P}_2$，$\mathcal{P}_3$）的 δ-带示例。在图 4(a) 和图 4(b) 中，黄色阴影区域表示间隙（$\mathcal{P}_G$，$\mathcal{P}_1$）的 δ-带；这里，黄色方块的放置平面为 $\mathcal{P}_k = \mathcal{P}_G$，它与其中一个间隙平面重合，因此将 $\mathcal{P}_1$ 的边界投影到 $\mathcal{P}_G$ 上并进行膨胀，从而得到黄色 δ-带。红色阴影区域表示间隙（$\mathcal{P}_2$，$\mathcal{P}_3$）的 δ-带；在这种情况下，红色方块的放置平面 $\mathcal{P}_k = \mathcal{P}_G$ 既不是 $\mathcal{P}_2$，也不是 $\mathcal{P}_3$，因此将 $\mathcal{P}_3$ 的边界（也可以选择 $\mathcal{P}_2$）投影到 $\mathcal{P}_G$ 上并进行膨胀，形成红色 δ-带。图 4(b) 展示了位于其对应 δ-带内的红色方块位姿，该方块用于跨越间隙。

B. 设计 $f_{\mathcal{C}_{block}^v}$

本节中，我们设计一种非均匀分布 $f_{\mathcal{C}_{\text{block}}^v}$，用于生成选定块 b_i在给定平面 $\mathcal{P}_k$上的放置位置。令 $\mathcal{C}^v_{\text{块}}(b_i, \mathcal{P}_k, \Omega^v)$ 表示一个集合，其中收集了在给定与树节点 v 相关联的环境 Ω^v 时，块 b_i 在环境 $\mathcal{P}_k$ 的自由空间中的所有无碰撞位姿。为简洁起见，令 $\mathcal{C}^v_{\text{块}} \equiv \mathcal{C}^v_{\text{块}}(b_i, \mathcal{P}_k; \Omega^v)$。分布 $f_{\mathcal{C}^v_{\text{block}}}(\mathbf{b}'_i \mid b_i, \mathcal{P}_k, g, \Omega)$ 取决于所选块 b_i的意图 g。具体而言，我们考虑以下两种情况：

临时放置 ($g = \varnothing$)：如果 $g = \varnothing$，即没有搭桥跨越间隙的意图，那么块 b_i 可以放置在$\mathcal{P}_k$上的任意位置。因此，我们将 $f_{\mathcal{C}^v_{\text{block}}}$ 设计为在 $\mathcal{C}^v_{\text{块}}$上的均匀分布，即$f_{\mathcal{C}^v_{\text{block}}}(\mathbf{b}'_i \mid b_i, \mathcal{P}_k, g = \emptyset, \Omega) = $均匀分布的$_{\text{块}}$。

桥接（$g = (\mathcal{P}_i, \mathcal{P}_j)$）：如果 $g = (\mathcal{P}_i, \mathcal{P}_j)$，这意味着应将$b_i$ 放置在 $\mathcal{P}_k$ 上，以跨越平面 $\mathcal{P}_i$ 和 $\mathcal{P}_j$之间的间隙。因此，在这种情况下，我们设计一种分布，使 b_i 在 $\mathcal{P}_k$ 上的放置偏向于相关支撑平面的正交投影周围宽度为 δ的带状区域。具体而言，如果 $\mathcal{P}_i \neq \mathcal{P}_k$ 和 $\mathcal{P}_j \neq \mathcal{P}_k$，我们将 $\mathcal{P}_i$ 或 $\mathcal{P}_j$ 投影到 $\mathcal{P}_k$ 上（见图 2(b)–2(c)；另见图 4）。如果 $\mathcal{P}_k = \mathcal{P}_j$，则将 $\mathcal{P}_i$ 投影到 $\mathcal{P}_k$上。在两种情况下，都会将所得投影扩展 δ，形成 δ带状区域（见图 2(b)–2(c)，其中平面 2 的边界投影到平面 1 上，随后经膨胀得到 δ带状区域；另见图 4）。令 $\mathcal{C}^\delta_{\text{gap}} \subseteq \mathcal{C}^v_{\text{block}}$ 为收集 b_i 在此宽度为 δ的带状区域内所有无碰撞位姿的集合。然后，我们将 $f_{\mathcal{C}^v_{\text{block}}}$ 定义为

如下:

$$f_{\mathcal{C}^{v}_{\text{block}}}(\mathbf{b}'_i \mid b_i, \mathcal{P}_k, (\mathcal{P}_i, \mathcal{P}_j), \Omega) = p_{\text{gap}} \cdot \text{Uniform}(\mathcal{C}^{\delta}_{\text{gap}}) + (1 - p_{\text{gap}}) \cdot \text{Uniform}\big(\mathcal{C}^{v}_{\text{block}} \setminus \mathcal{C}^{\delta}_{\text{gap}}\big), \tag{6}$$

使用 $0 < p_{\text{gap}} < 1$。这会使$\mathbf{b}'_i$ 偏向（由 $\mathcal{C}^{\delta}_{\text{gap}}$捕获的）最有可能弥合间隙的区域，同时确保 $\mathcal{C}^{v}_{\text{block}}$ 中的每个无碰撞位姿仍具有非零概率。将 $\delta = 0$ 设为相应值即可恢复均匀采样，从而得到 $\mathcal{C}^{\delta}_{\text{gap}} = \emptyset$。

C. 因不可行的建议而修订 Π

如前所述，符号计划 Π可能包含无法执行的三元组。回顾一下，三元组 s_m =(b_i, $\mathcal{P}_k$, g) 表示应将块 b_i 放置在平面$\mathcal{P}_k$ 上，以实现意图 g。规划Π中包含的此类三元组 s_m 可能因以下原因而不可行／无法执行。首先，规划 Π 的构建依赖于定义 3.3 中引入的 Candidate函数，该函数可能生成无法弥合目标间隙的方块。因此，如果 s_m 中的 g 要求弥合间隙，那么b_i 可能无法做到这一点。请注意，构建 Π时不包含任何低层几何可行性测试，因为其目标只是快速构建一系列高层三元组s_m ，以期完成任务。其次，无论块 b_i 在 $\mathcal{P}_k$上的意图 g 为何，即使在构建 Π 期间执行 GoTo 动作前进行了可行性检查， $\mathcal{P}_k$ 仍可能无法被机器人到达。这可能是因为与 GoTo 动作相关的可行性测试并不充分。例如，如果满足 GoTo 的条件 (i)（即环境中不存在可移动块时，两个平面由结构连接），那么由于其他可移动块的存在，这些平面在与树节点 v 关联的环境中最终可能彼此断开。我们再次强调，此类低层几何可行性测试留待算法 2 执行（见第 4、7 和 9 行），因为算法 2 负责为实现采样意图 g的方块生成可行的机器人位姿和放置位置。如果算法 2 未能在 N 次尝试内验证规划 Π中所含三元组 s_m 的可行性，那么我们就将其降级 s_m。具体而言，我们将其从 $\mathcal{S}_{\text{plan}}(v)$ 中移除，并将其保留在 $\mathcal{S}_{\text{gap}}(v)$ 中（该节点上不赋予特殊权重）。在这种情况下， $f_{\mathcal{S}}$ 的构造将不再受Π 驱动。

一旦发生这种情况，在下一次外层迭代（κ+1）中，采样一个节点 $v_{\text{rand}} \sim f_{\mathcal{V}}$后，我们会像之前一样重新计算一个新的符号计划 Π′ ，并设置 $v_{\Pi} = v_{\text{rand}}$ （可以选择 $\mathcal{V}$ 中的任意节点）。在重新计算期间，我们会明确排除符号动作 $Place$（s_m），以避免广度优先搜索过程生成一个新的规划 Π′ ，再次推荐算法 2 未能验证的三元组 s_m 。由此得到的规划 Π′ 随后会取代 Π，作为指导非均匀采样策略的符号计划。请注意，由于 Π′ 是在节点 $v_{\Pi} \neq v_{\text{root}}$处调用的，因此对于所有无法从 v_{Π} 到达的节点（例如 v_{root}），我们都有 PlanStep(v) = 0；这些节点位于增量构建的有向树 $\mathcal{T}$中。对于每个被判定为不可行的符号动作，我们都会重复这一过程。请注意，尽管 s_m 已从符号计划中移除，但由于它存在于 $\mathcal{S}_{\text{gap}}$（v）中，算法 1 仍可能对其进行采样。

注*4.1*（生成 Π 的高层规划器）：用于计算规划 Π（基于图 $\mathcal{G}$ ）的广度优先搜索规划器可以替换为其他规划器，例如 A*。此外，由于 Π不一定可行，因此也可以使用启发式规划器（例如大语言模型（LLM））来生成，这类规划器对用户更友好。在这种情况下，不可行性的另一个来源是此类规划器无法保证其结果正确。在第VI节中，我们将说明如何提示大语言模型构造 Π ，并讨论相关的权衡。

V. 概率完备性

在本节中，我们将证明我们的算法具有概率完备性，即，随着迭代次数 τ（若可行计划存在），我们的算法找到可行计划的概率1趋近于1 当算法 κ Alg.1 的迭代次数趋于无穷时。为正式陈述这一结果，我们首先引入计划间隙的概念（该概念也用于 [21], [23]）。考虑任意一个规划

$$\tau = \langle \tau(1), \ldots, \tau(k), \ldots, \tau(H) \rangle$$

由算法1 返回且解决问题1的规划，其中每个动作形式如下

$$\tau(k) = \texttt{Move}\big(\mathbf{b}_{i_k}, \mathbf{b}^{\star}_{i_k}\big)$$

移动块 b_{i_k} 至构型 $\mathbf{b}^{\star}_{i_k}$。如果对于任意替代规划， τ 具有间隙 $\varepsilon > 0$ ，则称其满足

$$\tilde{\tau} = \langle \tilde{\tau}(1), \ldots, \tilde{\tau}(k), \ldots, \tilde{\tau}(H) \rangle$$

且对所有 $k \in \{1, \ldots, H\}$，满足以下两个条件:

(i) $\tilde{\tau}(k)$ =移动$\big(\mathbf{b}^{i_k}, \tilde{\mathbf{b}}_{i_k}\big)$ 移动的是与 $\tau(k)$ 相同的块 b_{i_k}（步骤 k 处的块标识不变）；

(ii)$\| \tilde{\mathbf{b}}^{i_k} - \mathbf{b}^{\star}_{i_k} \| < \varepsilon$ 相对于方块位姿空间上的固定范数,

规划 $\tilde{\tau}$ 也是可行的。下文中，我们将第 k步满足(i)–(ii)简洁地记为 $\|\tau(k) - \tilde{\tau}(k)\| \leq \varepsilon$。

定理5.1（概率完备性）：假设存在一个可行规划 $\tau = \langle \tau(1)\text{.....}, \tau(H) \rangle$，其具有间隙$\varepsilon > 0$ （如上文所定义），且假设3.4成立。令采样分布 $f_{\mathcal{V}}$、 $f_{\mathcal{S}_{\text{Cand}}}$和 $f_{\mathcal{C}^{v}}$块为第IV节中定义的分布，并令 p_{plan}、 p_1、 p_2、 p_3、 $p_{\text{gap}} \in (0, 1)$。当迭代次数 $\kappa \to \infty$时，算法1–2返回可行规划的概率收敛于 1。

证明：为证明这一结果，我们首先引入以下记号和定义。令 $\mathcal{T}^{\kappa} = \{\mathcal{V}^{\kappa}_{T}, \mathcal{E}^{\kappa}_{T}\}$ 表示算法 1 在第 κ次迭代结束时构造的树。由于规划 τ 的间隙为 $\varepsilon > 0$, 因此存在替代可行规划 $\tilde{\tau}$，使得 $\|\tau(k) - \tilde{\tau}(k)\| \leq \varepsilon$对所有 k均成立。在整个证明中，凡提及 $\tilde{\tau}(k)$ ，均假设其与 ε的距离不超过 τ。

我们定义两个随机变量，用于对采样过程和可行计划的计算进行建模。首先，我们定义伯努利随机变量 X_{κ} ：如果在迭代 κ 时，对于某个 $k \in \{1, \ldots, H\}$，满足以下条件，则其值为 1：(i) 通过一条边（u, v）将节点 v 添加到树中，该边实现了 k 的 $\tilde{\tau}$ 步骤，即该边执行动作 $\tilde{\tau}(k)$ = 移动 ($\mathbf{b}^{i_k}$, $\mathbf{b}^{i_k}$)，其中 $\|\mathbf{b}^{i_k} - \mathbf{b}^{\star}_{i_k}\| \leq \varepsilon$，且其前驱节点 u对应于步骤 $k-1$ 的实现

（对于 $k=1$，取 $u=v_{\text{root}}$）；以及 (ii) 在迭代 κ 之前，树中不存在实现步骤 k 的节点 v'，即不存在一条边（u', v'）满足其与 τ_k ε 接近。此外，一旦在某次迭代 κ 中添加了一个对应于步骤 H 且同时满足 (i) 和 (ii) 的节点，今后的任何迭代都不可能再对任何 k 满足条件 (ii)；在这种情况下，按照约定，对于所有 $\kappa'>\kappa$，设 $X_{\kappa'}=1$。

其次，我们定义随机变量 Y_κ，用于统计伯努利变量$X_1,\dots,X_\kappa$中的成功次数；即，$Y_\kappa=\sum_{i=1}^{\kappa}X_i$。根据 X_i的构造（以及上述约定），当且仅当与动作 $\tilde{\tau}(H)$对应的节点已添加到树 $\mathcal{V}_T^\kappa$中时，$Y_\kappa\ge H$ 当且仅当，也就是说，已找到一个解。因此，要证明概率完备性，只需证明 $\mathbb{P}(Y_\kappa\ge H)\to 1$ 随着$\kappa\to\infty$而趋于$\kappa\to\infty$。

利用切比雪夫不等式，我们将失败概率（在 κ 次迭代中成功扩展次数少于 H 次，即未达到目标）界定如下：

$$\begin{aligned}\mathbb{P}(Y_\kappa<H)&=\mathbb{P}(\mathbb{E}[Y_\kappa]-Y_\kappa\ >\ \mathbb{E}[Y_\kappa]-H)\\&\le\mathbb{P}(\,|Y_\kappa-\mathbb{E}[Y_\kappa]|\ \ge\ \mathbb{E}[Y_\kappa]-H\,)\\&\le\frac{\mathrm{Var}(Y_\kappa)}{\big(\mathbb{E}[Y_\kappa]-H\big)^2},\end{aligned}\tag{7}$$

其中，第二个不等式是切比雪夫界。接下来，我们用伯努利变量X_i表示 $\mathbb{E}[Y_\kappa]$ 和$\mathrm{Var}(Y_\kappa)$。令 $p_i:=\mathbb{P}(X_i=1)$。于是

$$\mathbb{E}[Y_\kappa]=\sum_{i=1}^{\kappa}\mathbb{E}[X_i]=\sum_{i=1}^{\kappa}p_i,\tag{8}$$

$$\mathrm{Var}(Y_\kappa)=\sum_{i=1}^{\kappa}\mathrm{Var}(X_i)+2\sum_{1\le i<j\le\kappa}\mathrm{Cov}(X_i,X_j)\tag{9}$$

$$=\sum_{i=1}^{\kappa}p_i\,(1-p_i)+2\sum_{1\le i<j\le\kappa}\Big(\mathbb{E}[X_iX_j]-p_ip_j\Big).\tag{10}$$

接下来我们证明：**(a)** $\lim_{\kappa\to\infty}\mathbb{E}[Y_\kappa]=\infty$，以及 **(b)**$\mathrm{Var}(Y_\kappa)\ \le\ 3\,\mathbb{E}[Y_\kappa]$ 对所有 κ。利用 **(b)**，我们可以将（7）中的界改写为

$$\mathbb{P}(Y_\kappa<H)\ \le\ \frac{\mathrm{Var}(Y_\kappa)}{\big(\mathbb{E}[Y_\kappa]-H\big)^2}\ \le\ \frac{3\,\mathbb{E}[Y_\kappa]}{\big(\mathbb{E}[Y_\kappa]-H\big)^2}.\tag{11}$$

将 **(a)** 与（11）结合可得

$$\lim_{\kappa\to\infty}\mathbb{P}(Y_\kappa<H)=0.$$

等价地，

$$\lim_{\kappa\to\infty}\mathbb{P}(Y_\kappa\ge H)=1,$$

由此完成了概率完备性的证明。

a) ***(a)*** **的证明。**：我们证明 $\lim_{\kappa\to\infty}\mathbb{E}[Y_\kappa]=\infty$。令 $p_\kappa:=\mathbb{P}(X_\kappa=1)$。由于 $\mathbb{E}[Y_\kappa]=\sum_{i=1}^{\kappa}p_i$，只需给出 $\sum_{i=1}^{\kappa}p_i$ 的一个发散下界。

固定一个尚未采样的步骤索引 $k\in\{1,\dots,H\}$。令 v_{k-1} 表示树中一个可实现步骤 k 的父节点（对于 $k=1$，$v_0=v_{\text{root}}$）。令 $s_k^\star$ 表示使步骤 $\tilde{\tau}(k)$ 在 v_{k-1}处得以实现的三元组。在迭代κ 期间，若发生以下情况，则事件 $\{X_\kappa=1\}$ 发生：(i) 选择合适的前驱节点 v_{k-1}；(ii) 选择三元组 $s_k^\star$；并且 (iii) 成功采样区块配置（ii.1-ii.3），使得所得动作满足 $\|\tilde{\tau}(k)-\tau(k)\|\le\varepsilon$。我们对每个阶段分别给出下界。

节点选择。 使用节点分布 $f_{\mathcal{V}}$，每个节点的质量均非零（因为 $p_{\text{plan}}\in(0,1)$ 且分组内的采样服从均匀分布）。算法 1 每次迭代最多添加一个节点，因此 $|\mathcal{V}_T^\kappa|\le\kappa+1$。因此，根据$f_{\mathcal{V}}$ 在 [4]的构造

$$\begin{aligned}\underbrace{f_{\mathcal{V}}\big(v_{k-1}\,\big|\,\kappa\big)}_{=:\,p_V(\kappa)}\ &\ge\ \frac{\min\{\,p_{\text{plan}},\,1-p_{\text{plan}}\,\}}{\kappa+1}\\&=\ \frac{c_V}{\kappa+1},\qquad c_V\in(0,1).\end{aligned}\tag{12}$$

其中 $c_V:=\min\{p_{\text{plan}},\,1-p_{\text{plan}}\}$

三元组选择。 在节点 v_{k-1}处，候选集 $\mathcal{S}$ 是有限的。根据 $f_{\mathcal{S}}$ 在 [5]中的构造（桶质量为 p_1，p_2，$p_3\in(0,1)$且各分组内均匀分布），可得

$$\begin{aligned}\underbrace{f_{\mathcal{S}}\big(s_k^\star\,\big|\,v_{k-1}\big)}_{=:\,p_S(k)}\ &\ge\ \min\left\{p_1,\frac{p_2}{|\mathcal{S}_{\text{gap}}(v_{k-1})|},\frac{p_3}{|\mathcal{S}_{\text{nogap}}(v_{k-1})|}\right\}\\&=\ c_S(k)\ \in(0,1).\end{aligned}\tag{13}$$

其中 $c_S(k)=\min\left\{p_1,\frac{p_2}{|\mathcal{S}_{\text{gap}}(v_{k-1})|},\frac{p_3}{|\mathcal{S}_{\text{nogap}}(v_{k-1})|}\right\}$。注意到 $c_S(k)>0$，因为 $p_1,p_2,p_3>0$，并且假设 3.4 保证了 $s_k^\star\in\mathcal{S}$；否则 $c_S(k)$ 可能为零。由于 $k\in\{1,\dots,H\}$ 取自有限集合，我们可以固定一个与步数无关的均匀下界

$$c_S\ :=\ \min_{1\le k\le H}c_S(k)\ \in(0,1).\tag{14}$$

区块配置选择。 令 $\mathbb{B}_k:=\{\mathbf{b}:\ \|\mathbf{b}-\mathbf{b}_{i_k}^\star\|<\varepsilon\}$表示以 $\tau(k)$ 给出的目标配置为中心、半径为 ε 的球。由于根据 $f_{\mathcal{C}_{\text{block}}^v}$ [6] 的构造，该分布在两个可行区块配置子集上均匀分布，其权重分别为 $p_{\text{gap}}\in(0,1)$ 和 $1-p_{\text{gap}}$；又因为 $\varepsilon>0$，我们得到严格为正的单次抽取成功概率 $p_{\mathbb{B}}(k)\gtrsim 0$，即选中一个位于 ε 球 $\mathbb{B}_k$内的区块配置 $\mathbf{b}_{i_k}$的概率。每次迭代进行 $N\ge1$ 次独立同分布尝试时，至少成功一次的概率为

$$p_{\mathbb{B},N}(k)\ :=\ 1-\big(1-p_{\mathbb{B}}(k)\big)^N\ \in(0,1).\tag{15}$$

由于 H 是有限的，我们可以定义步长均匀常数

$$c_{\mathbb{B},N}\ :=\ \min_{1\le k\le H}p_{\mathbb{B},N}(k)\ \in(0,1).\tag{16}$$

每次迭代成功概率的下界。结合上述三个阶段，我们得到

$$\begin{aligned}p_\kappa\ &=\ \mathbb{P}(X_\kappa=1)\\&\ge p_V(\kappa)\cdot p_S\cdot p_{\mathbb{B},N}\\&=\ \frac{c_V}{\kappa+1}\ \cdot\ c_S\ \cdot\ c_{\mathbb{B},N}=\ \frac{C}{\kappa+1},\end{aligned}\tag{17}$$

其中，$C=c_V\cdot c_S\cdot c_{\mathbb{B},\ N}\in(0,\ 1)$。因此，

$$\mathbb{E}[Y_\kappa]=\sum_{i=1}^{\kappa}p_i\ \ge\ \sum_{i=1}^{\kappa}\frac{C}{i+1}\ \xrightarrow[\kappa\to\infty]{}\ \infty,$$

由此完成（**a**）。

b) （***b***）的证明。： 从（10）开始，

$$\mathrm{Var}(Y_\kappa) = \sum_{i=1}^{\kappa} p_i(1-p_i) + 2 \sum_{1\le i<j\le\kappa} \Big(\mathbb{E}[X_iX_j] - p_ip_j\Big).$$

对于第一项， $p_i(1-p_i) \le p_i$，因此

$$\sum_{i=1}^{\kappa} p_i(1-p_i) \;\le\; \sum_{i=1}^{\kappa} p_i \;=\; \mathbb{E}[Y_\kappa].$$

对于协方差项，由于 $X_iX_j \le X_i$,

$$\mathbb{E}[X_iX_j] - p_ip_j \;\le \mathbb{E}[X_i] - p_ip_j \;\le\; p_i(1-p_j) \;\le\; p_i,$$

因此

$$2 \sum_{1\le i<j\le\kappa} \Big(\mathbb{E}[X_iX_j] - p_ip_j\Big) \;\le\; 2\sum_{i=1}^{\kappa} p_i \;=\; 2\,\mathbb{E}[Y_\kappa].$$

结合这两个界可得

$$\mathrm{Var}(Y_\kappa) \;\le\; \mathbb{E}[Y_\kappa] + 2\,\mathbb{E}[Y_\kappa] \;=\; 3\,\mathbb{E}[Y_\kappa], \tag{18}$$

这就证明了 （**b**）。

■

VI. 实验验证

本节在不同复杂度的设置中评估基于采样的规划器 BRiDGE。评估重点包括规划运行时间、综合规划的时域 H 、不同偏置参数的影响，以及算法在 Candidate函数的精确实现和过近似实现下的鲁棒性；这两种实现分别会产生初始可行或初始不可行的符号规划 Π。第 VI-A 节总结所有实用函数的实现，包括 NavigReach、ManipReach、Gap 和 Candidate，以及用于比较的基线策略。第 VI-B 节展示了在平面、方块数量以及所需规划时域逐渐增加的环境中的可扩展性结果，并着重说明符号偏置的作用，以及错误候选项对运行时间的影响。第 VI-C 节提供了与均匀采样和 LLM驱动的非均匀采样策略的对比实验。在第 VI-D 节中，我们讨论不同偏置参数对基于采样的规划器运行时间的影响。最后，第 VI-E 节报告了在由人形机器人和四足机器人组成的异构双机器人系统上进行的硬件实验，展示了我们的方法在真实多层环境中的实用性。所有仿真均使用 Python 3 实现，并在配备 Intel Core i7-8565U CPU（1.8 GHz）和 16 GB RAM 的计算机上运行。实验视频见 https://vimeo.com/1151395396?share=copy&fl=sv&fe=ci，代码见 https://github.com/kantaroslab/BRiDGE

A. 实现细节、评估指标与基线方法

下文将介绍 **BRiDGE** 所使用的实现细节，包括第II节和III-A节中介绍的效用函数（用于设置我们的规划器）、评估指标和基线方法。

机器人导航能力：本节中，我们考虑一台全向机器人，即能够沿任意期望路径行进的机器人。我们假设机器人具有一种攀爬策略，使其能够在两个平面 $\mathcal{P}_i$ 和 $\mathcal{P}_j$ 的高度差小于 $\delta_H = 1.2$ 个单位且横向间距小于 $\delta_L = 2$ 个单位时，从一个平面移动到另一个平面。根据这些设定，我们可以高效地判断两个位置 $\mathbf{x}_i$和 $\mathbf{x}_j$之间是否存在动力学可行路径，从而构建效用函数 NavigReach($\mathbf{x}_i$,$\mathbf{x}_j$,$\Omega(t)$)。为加快可达性检查，我们预先计算一个概率路图（PRM）[10]。该 PRM 是一个图 $\mathcal{G}_G = (\mathcal{V}_G, \mathcal{E}_G)$，其中节点集合包含无障碍的机器人构型，边集合则表示节点之间动力学可行的连接。节点可以位于静态平面（$\mathcal{P}^{\mathrm{nm}}$）上，也可以位于可移动方块（$\mathcal{P}^{\mathrm{m}}$）的表面上。随后，NavigReach($\mathbf{x}_i$,$\mathbf{x}_j$,$\Omega^v$) 只需判断在环境 Ω^v中定义的 $\mathcal{G}_G$ 里是否存在连接 $\mathbf{x}_i$ 和 $\mathbf{x}_j$ 的路径，并返回 True 或 False。该 PRM 最初在规划树的根节点处构建，并随着规划树的增长而不断更新。每个新生成的树节点都对应一种不同的可移动方块排列，因此需要对 PRM 进行局部更新：(i) 拾起一个块会移除所有构型位于该块顶面上的 PRM 节点 $\mathcal{P}_i$，并相应移除与这些节点相连的所有边；(ii) 还可能在附近节点之间（距离在 PRM 连接半径以内）新增边，因为移除该块后，这些边变为无碰撞；(iii) 放置一个块会移除直线线段与该块新占据体积相交的边；(iv) 并在该块顶面上的节点与附近那些因这次放置而变得可达的平面上的节点之间新增边。这些增量式修改确保 PRM 始终与不断变化的环境几何和连通性保持一致。需要强调的是，只要提供 NavigReach($\mathbf{x}_i$,$\mathbf{x}_j$,$\Omega(t)$) 函数，就可以考虑更复杂的机器人动力学；例如，可以使用运动学-动力学规划器 [50]来实现该函数。

机器人操作能力：我们还假设机器人配备了夹爪，使其能够操作可移动方块。在实现中，我们将 ManipReach(b_i, Ω^v) 近似为距 b_i 不超过 $\delta_M = 1.9$ 个单位的所有无碰撞机器人构型，这足以支持抓取和放置操作。

函数的实现 间隙($\mathcal{P}_i$, $\mathcal{P}_j$, Ω^v)**：**为了评估间隙($\mathcal{P}_i$, $\mathcal{P}_j$, Ω^v)，我们只需检查与节点 v 关联的概率路图中，是否存在一条从 $\mathcal{P}_i$ 上的某个节点通往 $\mathcal{P}_j$上的某个节点的路径；若不存在，间隙将返回 `True`。请注意，这是对定义 3.2 中所定义函数的保守实现3.2，因为它仅考虑位于 $\mathcal{P}_i$ 和 $\mathcal{P}_j$上的概率路图节点/构型，而非这些平面上的所有可能构型。

函数 候选函数($\mathcal{P}_i$, $\mathcal{P}_j$**)：**候选函数：我们通过轻量级的几何高度检查来实现此函数，该检查会返回一个保守集合（过度近似的集合，包含所有真正的候选项，也可能包含假阳性）。目标是通过剪枝明显不正确的选项来缩小搜索空间。具体来说，如果将

b_n 放置在 $\mathcal{P}_k$ 上的任意位置，都会使 b_n 的顶面处于机器人从 $\mathcal{P}_i$ 和 $\mathcal{P}_j$ 任一平面踏上或踏下该方块时的步高范围内（步高范围由 δ_H 和 δ_L决定），那么我们就将(b_n，$\mathcal{P}_k$)加入 Candidate(b_n，)的输出集合中。不正确的候选项可以接受，之后会在算法 1 中通过可行性检查将其筛除。请注意，由于所有方块都采用直棱柱结构，Candidate($\mathcal{P}_i, \mathcal{P}_j$)返回的一对($b_n, \mathcal{P}_k$)只有在 $\mathcal{P}_k$ 上没有足够空间放置b_n时才是不正确的候选项——这可能是因为 b_n 的底面大于 $\mathcal{P}_k$的顶面，也可能是因为$\mathcal{P}_k$上已放置多个方块，导致没有空间放置 b_n。关于在不可移动平面上放置可移动方块的要求，请参见第二节。

配置所提出的采样策略：除非另有说明，我们通过以下设置配置非均匀采样策略：(i) $p_{\text{plan}} = 0.9$ in $f_{\mathcal{V}}$; (ii) (p_1, p_2, p_3) =(0.85，0.15，0.05) in $f_{\mathcal{S}}$; 以及 (iii) $p_{\text{gap}} = 0.9$ in $f_{\mathcal{C}^v_{\text{block}}}$。此外，我们使用由 BFS规划器生成的符号计划 Π 。

基线方法：我们将**BRiDGE** 与两种基线方法进行比较，它们与我们的规划器仅在采样策略上有所不同；我们的方法和基线方法均运行 $K_{\max} = 10,000$ 次迭代，且 $N = 100$。我们考虑以下设置：[4]

(i) 均匀采样：算法 1 中的节点选择 在搜索树上采用均匀分布，从候选超集 $\mathcal{S}$中均匀抽取三元组（块、放置平面、意图）。我们仍使用有偏策略（$f_{\mathcal{C}^v_{\text{block}}}$）对块配置进行采样；除非另有说明，我们将 $p_{\text{gap}} = 0.9$设置为，以确保与我们提出的非均匀采样策略进行有意义的比较。此选择是有意为之，因为我们观察到，由于搜索空间过大，完全均匀的策略无法解决时域为 $H \geq 3$ 的规划问题。请注意，此设置完全不需要使用规划 Π 。我们强调，该框架在概率意义上是完备的，因为如第 IV节所述，只需适当设置 p_{plan}、p_1、p_2和 p_3，即可通过我们的方法恢复该框架，且不违反定理 5.1。

(ii) 非均匀*LLM*驱动采样：该框架采用与我们完全相同的采样策略（包括所有参数的取值）。唯一的区别在于，我们用大语言模型替换了广度优先搜索规划器，如注释 4.1所述，因此该框架同样具有概率完备性。具体而言，我们提示 GPT5.1 在平面和方块上生成符号计划 Π，使用广度优先搜索规划器可获取的相同高层信息：平面和方块集合、平面之间的间隙关系、可用的高层动作（GoTo、Pick、Place 及其放置意图），以及简单的任务约束。这些信息以结构化的类JSON提示提供，用于描述环境和允许执行的动作；大语言模型则返回一系列符号动作，其三元组格式与广度优先搜索生成的规划完全相同 Π；我们的提示示例见 https://github.com/kantaroslab/BRiDGE。由于输入—输出接口与广度优先搜索规划器一致，LLM引导的规划可以在我们的框架中直接替代

[4]我们还尝试使用大语言模型（GPT 5.1）作为端到端规划器（不涉及采样），直接求解所考虑的问题。虽然大语言模型能够生成高层规划（定义为三元组序列，见非均匀LLM驱动采样策略），但我们观察到，它们无法进行低层空间推理，因此无法生成包含可行机器人位姿和物体放置位置的低层规划。

表 I 使用基于广度优先搜索的偏置策略时的时域 $H_{\text{MIN}} = 2$ 结果。报告 BFS 偏置在以下情况下的结果：(初始可行计划 Π) 和 (初始不可行计划 Π：时间、查询次数)。

$\lvert\mathcal{P}^{\text{nm}}\rvert$	$\lvert\mathcal{B}^{\text{m}}\rvert$	**BFS 偏置** [初始 Π 可行] (时间（秒）)	**BFS 偏置** [初始 Π 不可行] (时间（秒），查询次数)
3	2	7.3	18，2 次查询
3	5	7.9	26，3 次查询
3	10	7.1	52，5 次查询
6	2	7.2	20，2 次查询
6	5	7.3	33，4 次查询
6	10	7.2	56，5 次查询
9	2	7.1	19，2 次查询
9	5	8.2	28，3 次查询
9	10	7.9	57，5 次查询

表 II 使用基于广度优先搜索的偏置策略时的时域 $H_{\text{MIN}} = 6$ 结果。报告了以下情况下的广度优先搜索偏置结果：(初始可行计划 Π) 和 (初始不可行计划 Π：时间、查询次数)。

$\lvert\mathcal{P}^{\text{nm}}\rvert$	$\lvert\mathcal{B}^{\text{m}}\rvert$	**广度优先搜索偏置** [初始 Π 可行] (时间（秒）)	**广度优先搜索偏置** [初始 Π 不可行] (时间（秒），查询次数)
6	3	7.6	24，3 次查询
6	6	7.5	35，4 次查询
6	10	7.9	49，5 次查询
9	3	8.1	26，3 次查询
9	6	7.7	38，4 次查询
9	10	7.9	54，6 次查询
12	3	7.8	26，3 次查询
12	6	8.3	39，4 次查询
12	10	8.1	48，5 次查询

基于采样的规划器。

评估指标：我们采用以下评估指标：（i）计算第一个可行规划所需的运行时间；（ii）所设计规划的时域H ；（iii）构造的树的大小；（iv）由于偏置到不可行方向而重新生成符号计划 Π 的次数（也称为向BFS规划器发起的查询次数）。报告的指标为3次试验的平均值。

B. 问题复杂度下的可扩展性分析

在本节中，我们评估**BRiDGE**在不可移动平面数量、可移动方块数量以及规划时域 H （即使目标区域变得可达所需的动作数量）方面的可扩展性。我们考虑最小可能规划时域记为 H_{min} 且为 $H_{\text{min}} \in \{2, 6\}$ 的情况。对于每个规划时域，我们通过排列三种平面数量和三种方块数量，生成九个环境。具体而言，当 $H_{\text{min}} = 2$ 时，我们考虑不可移动平面的数量为 $|\mathcal{P}^{\text{nm}}| \in \{3, 6, 9\}$，可移动方块的数量为 $|\mathcal{P}^{m}| \in 2$，5，10。类似地，当 $H = 6$ 时，我们考虑 $|\mathcal{P}^{\text{nm}}| \in \{6, 9, 12\}$ 和 $|\mathcal{P}^{m}| \in \{3, 6, 10\}$。

在所有案例研究中，我们观察到，当候选函数函数是精确的（即不会产生任何假阳性）时，Π 广度优先搜索规划器生成的初始高层规划一开始就是可行的。[5] 为了评估针对候选函数的不完善实现以及可能出现的情况的鲁棒性

[5]一般而言，规划 Π 即使候选函数是精确的，即它只返回有效方块，规划仍可能不可行；参见第IV-C节。

对于不可行规划，我们测试了两种设置：(i) 初始规划Π是可行的；(ii) 初始规划 Π 是不可行的，具体做法是在候选函数的输出中手动插入错误的候选方块，从而模拟该函数的过度近似实现。两种设置下的结果$H_{\min}=2$ 汇总于表I 中。当初始规划 Π 可行时，我们的方法计算第一个可行解所需的运行时间 τ在所有环境中几乎保持不变（约 7 秒），与环境的复杂程度无关。这是因为 BFS 规划 Π 始终将采样策略引向可行动作，同时仍允许进行一定程度的随机探索，以执行几何检查并生成未由 Π规定的轨迹。当 Π可能包含不可行动作时，这些动作会在检测到失败后被剪枝，并触发 Π的重新计算，如第IV-C节所述。表格第三列中的“查询次数”量化了这一开销，即 Π 被重新计算的次数。运行时间大致随重新规划次数增加而增长（例如，从查询 2 次时的 18 秒增加到查询 5 次时的 57 秒）。请注意，在所有情况下，所设计规划的时域均为 $H=2$，这意味着规划器找到了可能的最短时域规划。对于 $H_{\min}=6$的情况也呈现出类似趋势，结果汇总于表Π中；所有情况下，所设计规划的时域均为 $H=H_{\min}=6$。与 $H_{\min}=2$情况相比，主要区别在于运行时间略高，这是预期的，因为规划时域有所增加。

为了更清楚地展示本方法所解决的规划问题的复杂性，以及它所体现的长时域推理与规划行为，下面我们将详细介绍算法针对一个具有参数 $H_{\min}=6$,$|\mathcal{P}^{\mathrm{nm}}|=6$ （包括地面平面）和 $|\mathcal{B}^{\mathrm{m}}|=3$ 的代表性案例研究生成的规划；参见图 5。不可移动（非地面）平面 $\mathcal{P}_1$, $\mathcal{P}_2$, $\mathcal{P}_3$, $\mathcal{P}_4$ 和 $\mathcal{P}_5$ 的高度分别为 2、2、3、5 和 4。目标区域位于平面 $\mathcal{P}_4$ 上。机器人最初位于地面平面上，因此，鉴于第VI-A节讨论的导航能力，它无法到达目标区域。为了到达目标区域，机器人使用了三个可移动方块：高度为 $h_1=1$ 、最初位于 $\mathcal{P}_1$ 上的方块 b_1；高度为$h_2=1$ 、最初位于地面平面上的方块 b_2；以及高度为 $h_3=2$ 、最初位于 $\mathcal{P}_5$ 上的方块 b_3 。从图5可以看出，方块 b_3 是唯一可用于跨越 $\mathcal{P}_3$ 与 $\mathcal{P}_4$（目标区域所在处）之间间隙的方块，只需将其放置在 $\mathcal{P}_1$ 上即可。然而，机器人最初无法到达 b_3 。要到达那里，必须先到达 $\mathcal{P}_2$ ，再到达 $\mathcal{P}_5$；但这两个平面都无法从地面平面直接到达，而且从 $\mathcal{P}_2$ 也无法到达 $\mathcal{P}_5$ 。鉴于机器人的导航与操作能力，访问 $\mathcal{P}_5$ 的唯一办法是同时使用方块 b_1 和 b_2：一个用于跨越地面平面与 $\mathcal{P}_2$ 之间的间隙，另一个用于跨越 $\mathcal{P}_2$ 与 $\mathcal{P}_5$ 之间的间隙。然而，最初只有 b_2 可以到达（图5(b)）。因此，机器人首先将 b_2 移到 $\mathcal{P}_1$ 旁边，使其能够攀爬至 $\mathcal{P}_1$ （图5(c)），随后将 b_1 移至地面平面，以搭起连接地面与 $\mathcal{P}_2$ 的通道（图5(d)）。请注意，要将b_3 移至地面，必须先移除地面平面与 $\mathcal{P}_1$ 之间原有的桥梁（方块b_2），然后使用 b_2 搭建连接平面 $\mathcal{P}_2$ 和 $\mathcal{P}_5$ 的桥梁。这样，机器人便能到达 $\mathcal{P}_5$（图5(e)），并将 b_3 移至地面平面（图5(f)）。

表 III 时域 $=2$对比。报告以下方法的规划时间：BFS 偏置、均匀采样和大语言模型偏差。

$\lvert\mathcal{P}^{\mathrm{nm}}\rvert$	$\lvert\mathcal{B}^{\mathrm{m}}\rvert$	**BFS 偏置**（秒）	**均匀分布**（秒）	大语言模型偏差（秒）
3	2	7.3	8	16
3	5	7.9	12	20
3	10	7.1	18	28
6	2	7.2	26	21
6	5	7.3	50	25
6	10	7.2	80	31
9	2	7.1	46	25
9	5	8.2	75	32
9	10	7.9	110	38

表 IV 时域 $=6$的比较。报告的规划时间：BFS 偏置、均匀采样和大语言模型偏差。

$\lvert\mathcal{P}^{\mathrm{nm}}\rvert$	$\lvert\mathcal{B}^{\mathrm{m}}\rvert$	**BFS 偏置**（秒）	**均匀分布**（秒）	大语言模型偏差（秒）
6	3	7.6	N/A	107
6	6	7.5	N/A	121
6	10	7.9	N/A	205
9	3	8.1	N/A	144
9	6	7.7	N/A	156
9	10	7.9	N/A	272*
12	3	7.8	N/A	160
12	6	8.3	N/A	263*
12	10	8.1	N/A	209

此时，机器人和方块 b_1 以及 b_3 都位于地面上。机器人使用 b_1将 b_3 临时放置在地面上，以重新连接地面平面与 $\mathcal{P}_1$ 之间的间隙。具体而言，机器人将 b_1 推回到 $\mathcal{P}_1$ 旁边（图5(g)），抓住 b_3，然后带着 b_3爬上 b_1 ，再爬到 $\mathcal{P}_1$ 上，最后将 b_3 放置在 $\mathcal{P}_1$上，从而连接起 $\mathcal{P}_3$与 $\mathcal{P}_4$ 之间的间隙（图5(h)）。最后，机器人从 $\mathcal{P}_2$ 爬到 $\mathcal{P}_3$上，再爬到 b_3上，最终抵达 $\mathcal{P}_4$ 以进入目标区域（图5(h)）。

请注意，在此示例中，我们的方法计算出间接且不平凡的策略，例如：暂时搭桥跨越另一处间隙，以接触关键方块（例如，搭建中间桥梁以到达 $\mathcal{P}_5$）；进行临时放置，让机器人腾出手来执行其他动作，随后取回并重新部署该方块，以完成原始目标（例如，将 b_3 短暂放在地面上，然后用它连接 $\mathcal{P}_3$ 和 $\mathcal{P}_4$）；以及重复利用同一个方块，使其在不同时间发挥不同作用（例如， b_2 先连接地面与 $\mathcal{P}_1$ ，之后再连接 $\mathcal{P}_2$与$\mathcal{P}_5$）。

C. 与基线方法的对比实验

本节评估我们的算法在采用第VI-A 节讨论的均匀和非均匀 LLM 驱动的采样策略时，在第VI-B节介绍的案例研究中的表现。为简化分析，我们将评估范围限定于精确候选函数设定（即没有假阳性）；例如，将潜在的 LLM 错误与来自候选函数的不正确候选项结合起来，会使我们难以分离并评估 LLM 的具体影响。

比较 $H_{\min}=2$ 和 $H_{\min}=6$ 的结果分别如表 III 和IV所示。在这两张表中，可以看到一致的结果。对于 $H_{\min}=2$ （表III），

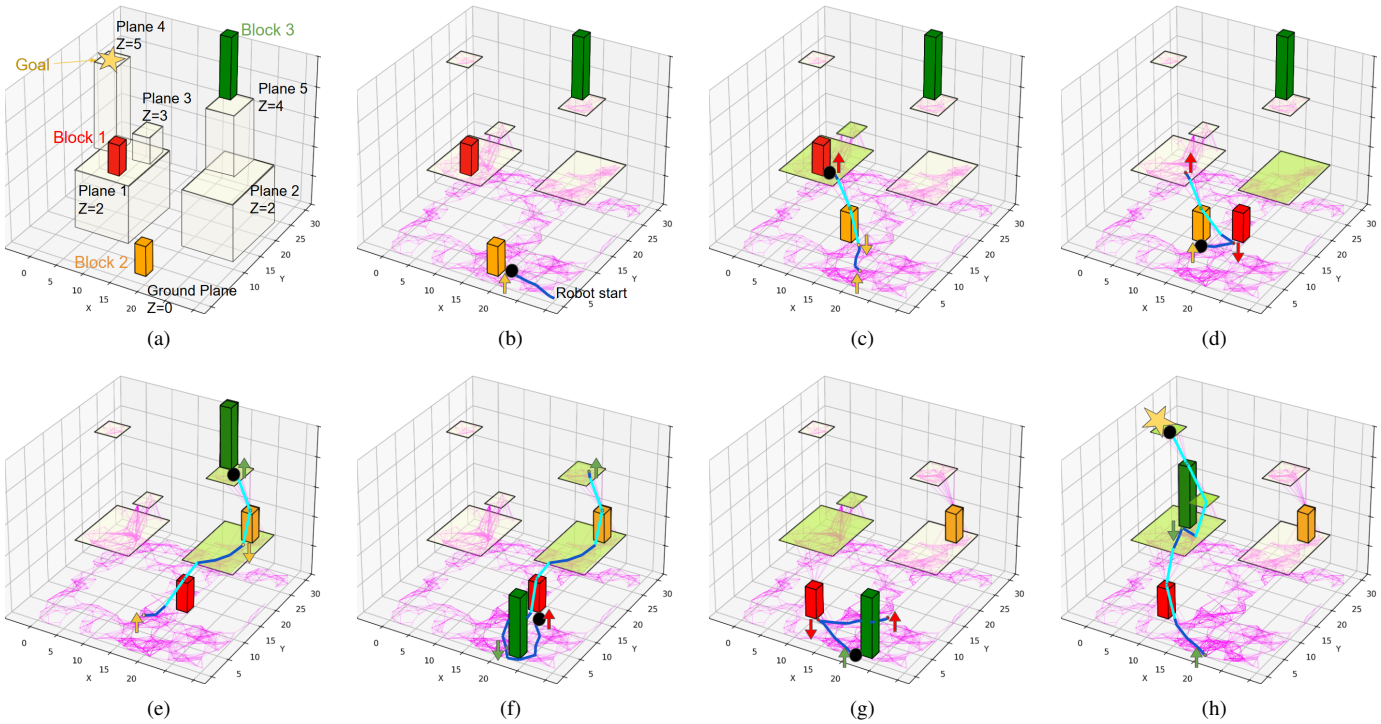


图5. 恢复的规划 τ 中每次 Move(b_i, b_i') 后的快照。平面上的机器人轨迹以蓝色显示；平面间轨迹以青色显示。向上箭头表示方块拾取动作；向下箭头表示方块放置动作。粉色线条表示概率路线图连通性（第6.1节））。从机器人位置可到达的平面以浅绿色突出显示。任务。图5(a)显示初始环境（地面加上平面 1–5，以及高度（Z））。方块1（红色）和方块2（橙色）高1个单位；方块3（绿色）高2个单位；机器人最多可上升/下降 $\delta_H = 1.2$ 个单位。起点位于地面；目标（⋆）位于平面 4。图5(b)显示初始概率路线图。图5(c)–图5(e)：使用方块1和2作为中间台阶，到达平面 5 并拾取方块3。图5(f)：将方块3暂时放置在地面上。图5(g)：重新利用方块1与平面 1 建立连通性。图5(h)：放置方块3以连接平面 3 和 4，使机器人能够到达目标区域。

均匀采样的运行时间会随着环境规模的增大而迅速增长（例如，在包含 3 个平面和 2 个可移动方块的环境中为 8 秒，而在包含 9 个平面和 10 个可移动方块的环境中则为 110 秒），尽管时域 $H_{\min}$ 是固定的。这是因为需要探索的空间，尤其是三元组 s的数量，会随着方块和平面数量的增加呈指数增长。因此，在没有任何指导来确定应优先处理哪些三元组的情况下，运行时间会显著增加。当不可移动平面和可移动方块的数量较少时，大语言模型偏置采样策略的表现不如均匀采样框架。例如，在包含 3 个平面和 2 个可移动方块的环境中，大语言模型偏置策略的运行时间为 16 秒，而均匀采样仅需 8 秒。然而，随着环境规模增大，大语言模型偏置策略的表现优于均匀采样：在包含 9 个平面和 10 个可移动方块的环境中，其运行时间为 38 秒，而均匀采样则为 110 秒。我们的方法优于两种基线方法，实现了显著更短的运行时间（$\approx$ 7秒），并且在所有案例研究中几乎保持不变。需要注意的是，大语言模型生成的规划 Π在全部九个案例中最初都是可行的，为采样策略提供了“可行”的方向（正如第 VI-B 节所述，BFS规划器也发挥了相同作用）；然而，由于需要额外查询 GPT 5.1 所产生的推理成本，其速度慢于我们的方法（以及小规模问题中的基于均匀采样的策略）。我们还注意到，所有方法返回的规划时域都完全相同：每种方法（BFS偏差、均匀采样和大语言模型偏差）均成功恢复了最短时域规划，所用时间为 $H = H_{\min} = 2$。由$\mathcal{T}$ 算法 1 构建的树中，广度优先搜索偏置规划器和大语言模型偏置规划器构建的树都较为紧凑——所有情况下节点数均在 3 到 5 之间，体现了它们强大的方向引导能力以及对探索的极少需求。相比之下，均匀采样规划器需要构建大得多的树，节点数 $|\mathcal{V}_T|$从最简单情况下的 4 个节点增至最大环境中的 57 个节点，这与运行时间的增长及搜索空间的指数级扩张相呼应。

对于包含 $H_{\min} = 6$ （表 IV）的案例研究，我们观察到，均匀采样在全部九个实例中均未能在迭代预算（$K_{\max}$）内找到解决方案。此外，基于大语言模型的采样策略的运行时间明显长于相应的广度优先搜索偏置运行–，范围从 107 秒到 272 秒–，这主要是因为大语言模型生成每个高层规划时固有的延迟 Π；即使其生成的符号计划是正确的，这一成本仍占主导地位。在某些情况下——表 IV中以星号标记的情况——大语言模型还在 Π内生成了错误的符号动作，候选函数却仅包含正确的元组。这类错误源于大语言模型在推理长时域符号任务时固有的易错性。当提出错误动作时，算法 2 中的几何检查会将其拒绝，而我们会重新查询大语言模型以获取修订后的规划，这与第 IV-C节中的失败刷新机制类似。这些重新规划事件引入了额外开销，

表 V H=6的偏差敏感性，6 个平面，3 个方块（仅正确候选项）。

$(p_1,\ p_2,\ p_3)$	p_{plan}；时间（秒），($\lvert\mathcal{V}_T\rvert$ 个节点)		
	0.2	**0.5**	**0.8**
(0.2, 0.4, 0.4)	14.1, (34)	11.4, (19)	9.7, (13)
(0.5, 0.25, 0.25)	10.7, (17)	9.6, (12)	8.75, (10)
(0.9, 0.05, 0.05)	9.5, (15)	8.7, (9)	8.5, (7)

表 VI H=2的偏差敏感性：3 个平面、2 个方块，包含一个不可行候选项，且高层规划固定 Π（不进行重新规划）。

$(p_1,\ p_2,\ p_3)$	p_{plan}；时间（秒），($\lvert\mathcal{V}_T\rvert$ 个节点)		
	0.2	**0.5**	**0.8**
(0.2, 0.4, 0.4)	7.8, (4)	12.8, (4)	22.1, (5)
(0.5, 0.25, 0.25)	10.6, (4)	15.3, (3)	28.5, (4)
(0.9, 0.05, 0.05)	16.6, (3)	35.2, (4)	57.0, (4)

解释了带星号条目的运行时间为何明显更长。相比之下，正如第 VI-B节所述，我们提出的方法结合广度优先搜索规划器后，在所有评估的问题规模下，运行时间始终稳定在约 7.5–8.3秒。对于所有评估实例，使用广度优先搜索驱动和大语言模型驱动方法计算出的规划时域均为 $H = H_{\min} = 6$。在不同问题规模下，我们的广度优先搜索偏置规划器和大语言模型偏置规划器均生成了规模相近且较小的树，其中 $|\mathcal{V}_T|$ 范围为 7 到 13 个节点，体现了两种方法中强烈的方向性偏置。

D. 偏置参数的影响与规划的正确性 Π

本节评估我们的算法对所提出的广度优先搜索驱动的非均匀采样策略中所用参数 p_1、p_2、p_3 和 p_{plan} 的敏感性。具体而言，我们考虑以下取值：$p_{\text{plan}} \in \{0.2, 0.5, 0.8\}$（用于式 (4)）以及 $(p_1, p_2, p_3) \in$ {(0.2, 0.4, 0.4)、(0.5,0.25, 0.25)、(0.9, 0.05, 0.05)}（用于式 (5)）。对于每种参数构型，我们报告计算出首个可行解所需的平均运行时间，以及算法 1 生成的相应树节点数量，记为 $|\mathcal{V}_T|$。

首先，我们考虑第 VI-B 节讨论的案例研究，其中 $|\mathcal{P}^{\text{nm}}| = 6$、$|\mathcal{B}^{\text{m}}| = 3$ 和 $H_{\min} = 6$。为了将偏置的影响与函数候选函数可能产生的不正确输出区分开来，我们的规划器仅使用该函数的正确输出进行配置，即确实能够跨越间隙的方块。在此案例研究中，使用精确的候选函数时，广度优先搜索生成的规划 Π 在所有试验中最初均为可行规划。[6]结果如表 V所示。我们观察到一个一致的趋势：增大对广度优先搜索规划 Π的偏置，即增大 p_{plan}和 p_1，会缩短规划时间，因为树会优先向由正确的 Π推荐的三元组/方向扩展。同时，$|\mathcal{V}_T|$ 会大幅下降，因为对 Π 的偏置增大，这与直觉相符：对正确的高层规划加大偏置Π 可能会加快规划合成。

其次，为补充上述案例——该案例隔离了在 Π 正确时偏置的影响——接下来我们考察一个[6]请注意，一般而言，即使使用“精确”的 Π，规划仍可能不可行；参见Candidate函数；见第IV-C节。

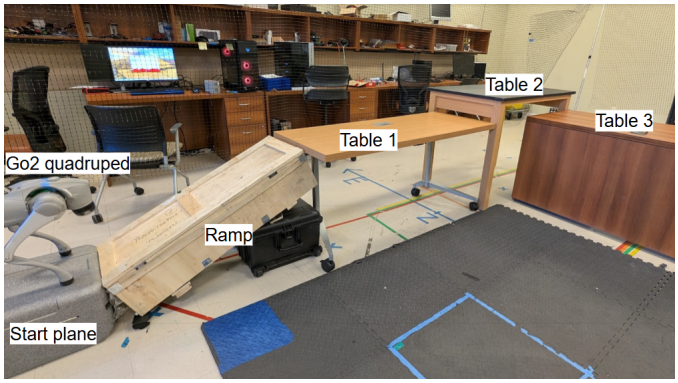


图 6. 硬件演示装置。Go2 从坡道底部出发，需要到达桌子 3。

对比情形，其中偏置与不可行的高层规划相互作用。具体而言，在此我们考虑一个时域较小的实例 $H_{\min}$=2，其中有三个不可移动平面和两个可移动方块。对于不可移动平面之间的某个间隙，只有一个方块能在物理上搭建桥梁，但 Candidate 函数会返回两个选项：一个可行，另一个不可行。因此，规划 Π包含一个不可行的三元组。我们有意禁用重新规划机制，使 BFS规划器仅调用一次；即使对应的三元组始终未能成功，所得规划 Π也保持固定。结果汇总于表 VI。与表 V 不同，在这里，增大对（不可行）规划 Π的偏置会持续增加规划时间：较高的 p_1 和较大的 p_{plan} 会使规划器将大部分精力用于扩展朝向不可行方向的树，导致探索不足，无法发现可行候选项。在最极端的设置（p_1、p_2、p_3）=(0.9, 0.05,0.05)和p_{plan}=0.8下，平均规划时间（不进行重新规划 Π）达到 57 秒。不过，如第 IV-C 节所述，如果允许即时修订 Π，相同的参数设置只需对 Π进行一次重新规划，就能在 12 秒内解决问题。我们注意到，$|\mathcal{V}_T|$ 在不同偏置设置下几乎保持不变，因为朝不可行方向进行偏置采样会产生大量被拒绝的样本，但成功的节点插入却很少。该实验表明，当高层规划正确时，激进偏置非常有益；但若候选项不正确，则可能适得其反，除非结合第 IV-C 节所述的修复机制。

备注 *6.1*（案例研究选择）：为完整起见，我们指出，上文使用的两种案例研究设置是有意选取的：时域为 $H_{\min}$=6 且使用精确的Candidate函数（没有假阳性），以及时域为 $H_{\min}$=2 且使用过近似候选函数。对于 $H_{\min}$=6 案例，如果引入了错误候选项且禁用重新规划，那么在所有偏置配置下，规划器都无法在 $K_{\max}$ = 10,000次迭代内找到解，因此偏置敏感性表将毫无参考价值。相反，对于 $H_{\min}$=2，当所有候选项都正确时，问题的时域太短，所有偏置设置的运行时间都相近（约为5–6 秒），且节点数量极少，因此无法呈现有意义的趋势。基于这些原因，我们仅展示上述有参考价值的案例。

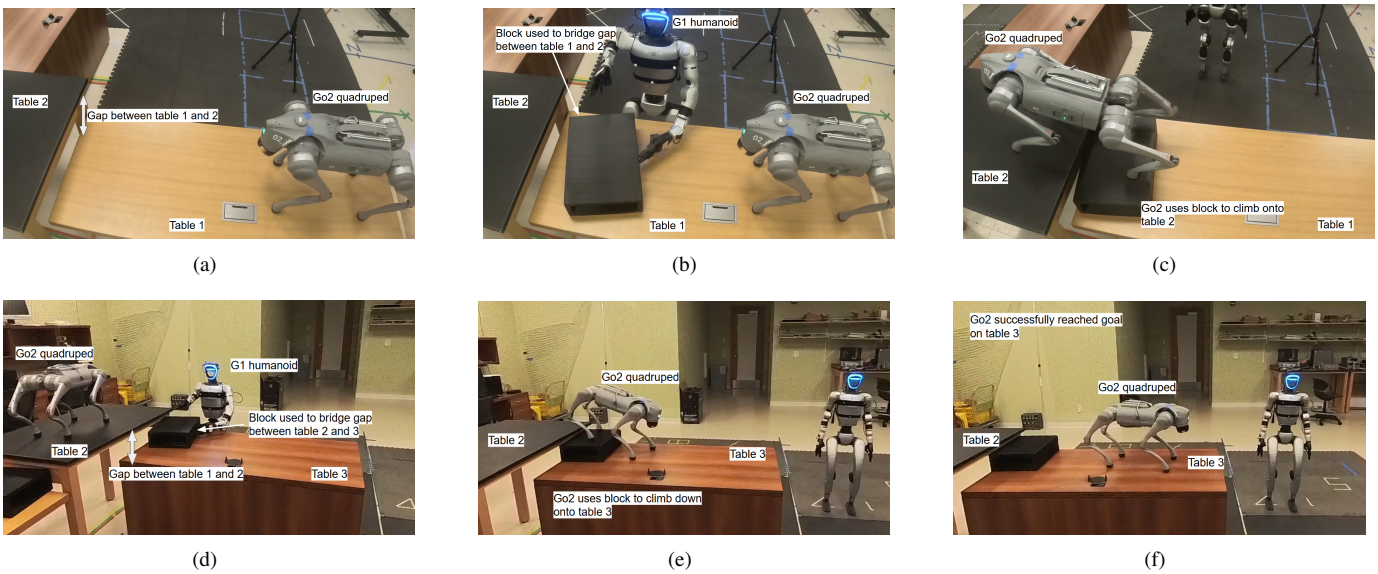


图 7. 硬件演示。图 7(a)：Go2 到达桌子 1，但无法爬上桌子 2。图 7(b)：随后，人形机器人放置第一个方块，搭起连接桌子 1 和桌子 2 的桥梁。图 7(c)：四足机器人利用该方块爬上桌子 2。图 7(d)-7(f)：类似地，随后人形机器人用第二个方块搭桥跨越桌子 2 与桌子 3 之间的间隙，使 Go2 能够到达桌子 3。

E. 硬件演示

BRiDGE 面向单个机器人开发，其导航与操作能力由NavigReach和ManipReach函数编码。本节通过硬件实验展示如何将这些能力分配给两台机器人：（i）一台宇树 Go2四足机器人，能够穿越并攀爬满足 $\delta_H = 15$ cm的表面，但无法抓取物体；（ii）一台宇树 G1人形机器人，能够可靠地拾取和放置方块，但未配备多平面攀爬控制策略。我们在这对异构机器人上实例化规划器，由Go2执行所有导航动作，G1执行所有操作动作。机器人定位由*OptiTrack* 动作捕捉系统提供，以实现实时6自由度跟踪。

工作空间由四个平面组成：地面和三张不可移动的桌子；见图 6。桌子 1 通过坡道与地面相连，桌子 2 位于桌子 1 和桌子 3 之间，且高度最高。从桌子 1 到桌子 2，以及从桌子 2 到桌子 3 的高度差均约为 20厘米。两个 3D打印可移动方块 b_1 和 b_2，高度为 12厘米，放置在桌子 4 上，而桌子 4 位于 Go2 的工作空间之外。Go2 最初位于地面平面上，任务是到达位于桌子 3 上的目标区域。虽然 Go2 可以通过坡道从地面平面爬上桌子 1，但由于高度差，它无法从桌子 1 爬上桌子 2，也无法从桌子 2 爬上桌子 3，因为机器人能够可靠攀爬的垂直台阶高度最多为 $\delta_H = 15$ 厘米。因此，要到达目标区域，就必须使用两个可移动方块搭桥跨越桌子之间的间隙。请注意，Go2 不具备任何操作能力。不过，G1 可以抓取两个可移动方块，并将它们放置在任意不可移动平面上，从而让 Go2 能够到达起初无法抵达的表面。

为了将我们提出的单机器人规划器映射到这一双机器人设置中，我们定义：(i) ManipReach(b_i, Ω) 为完整工作空间；(ii) NavigReach 覆盖地面和桌子 1–3，并施加攀爬约束 $\delta_H = 15$。从概念上讲，这将 Go2（规划器中的主要机器人）建模为能够从当前构型抓取任意块，并将其放置在任意平面上。在实际执行中，这些操作动作由 G1 代替 Go2 完成。这使异构硬件组合（负责导航的 Go2 和负责操作的 G1）能够实现算法1中假定的单个抽象智能体的能力，而无需修改规划器本身。

我们运行算法1，不使用任何高层偏置：节点选择采用均匀分布，三元组也从候选集中均匀抽取。唯一的例外是块构型采样，在此过程中我们保留放置偏置并设置 $p_{\text{gap}} = 0.9$。规划器在 8.1 秒内返回一项可行计划，其中包含两个高层 移动($\mathbf{b}_i$, $\mathbf{b}_i'$) 动作，首先需要将b_1 从桌子 4 移至桌子 1，架起连接桌子 1 与桌子 2 的桥梁；其次将块 b_2 从桌子 4 移至桌子 3，架起连接桌子 2 与桌子 3 的桥梁。因此，在执行该计划期间，Go2 移至桌子 2（图 7(a)）；G1 使用b_1 架起连接桌子 1 与桌子 2 的桥梁（图 7(b)）；Go2 通过 b_1 爬上桌子 2（图7(c)）；G1 使用 b_2 架起连接桌子 2 与桌子 3 的桥梁（图7(d)）；Go2 通过 b_2 移至桌子 3（图7(e)），并到达其目标区域（图7(f)）。

实践经验与当前局限：我们的硬件演示也凸显了当前实现尚未解决的若干实际局限。首先，我们的系统假设所有方块和平面的位姿都已精确获知。在实践中，可靠的操作需要借助局部感知反馈，在抓取和放置过程中细化每个物体相对于机器人的相对位姿，从而实现闭环校正并提高执行的稳健性。其次，我们提出的框架假设每个采样得到的块放置位置在放置后都能保持静态稳定，但这一点在实践中未必成立，尤其是在环境

存在不平整且可能湿滑的地形时。放宽这一假设并非易事：稳定性取决于表面几何形状、摩擦力和质量分布，而要在线验证稳定性，则需要丰富得多的模型。近期研究将物理仿真器集成到基于采样的规划中，以评估候选放置位置的稳定性 [51]„ 将此类检查纳入我们的框架，是迈向现实世界部署的一个有前景的方向。最后，仅在一台人形机器人上实现我们的框架仍是一项重大挑战，主要原因是机器人必须能够可靠地攀上方块。执行此类行为需要对机器人与已放置方块之间的精确相对构型进行推理，同时生成可行的足步轨迹，以实现稳定攀爬。尽管我们的硬件实验将所有攀爬任务都交由 Go2 四足机器人完成，但要扩展系统以支持人形机器人攀爬，就需要集成全身运动规划和接触推理。这些局限体现了当前工作的自然扩展方向，而每一项都为未来工作指明了明确路径，以缩小我们已实现的规划能力与完全自主的现实世界能力之间的差距。

七、结论

本文研究了移动机器人的规划问题：机器人需要在由多个高处可通行平面、静态平面和可移动物体组成的环境中到达指定目标区域。我们重点关注初始状态下不可行的任务，即目标区域位于机器人无法到达的高处平面上。为使机器人能够到达此类目标区域，我们开发了一种基于采样的规划器，通过有策略地重新放置可移动物体，创建新的可通行连接，从而重新配置不连通的三维环境。该方法利用符号化高层规划器（广度优先搜索或大语言模型），引导采样倾向于有希望的方块放置位置。我们证明了所提出的方法具有概率完备性。实验表明，该方法能够高效解决复杂程度各异的长时域问题，即使高层符号化引导部分有误，也依然稳健。我们还通过人形机器人和四足机器人的硬件演示验证了该框架。未来工作将重点研究集成感知驱动的操作、在不平坦地形上进行考虑稳定性的方块放置，以及将系统扩展至更通用的多机器人能力。

参考文献

[1] S. M. LaValle，《规划算法》。剑桥大学出版社，2006年。[2] L. P. Kaelbling 和 T. Lozano-P´erez，“信念空间中的集成任务与运动规划”，《国际机器人研究杂志》，第32卷，第9-10期，第1194–1227页，2013年。[3] N. Matni、A. D. Ames 和 J. C. Doyle，“分层多速率控制的定量框架：迈向控制架构理论”，《IEEE 控制系统》，第44卷，第3期，2024年。[4] D. E. Koditschek 和 E. Rimon，“带边界流形上的机器人导航函数”，《应用数学进展》，第11卷，第4期，第412–442页，1990年。[5] V. Vasilopoulos、G. Pavlakos、K. Schmeckpeper、K. Daniilidis 和 D. E. Koditschek，“利用语义感知反馈在部分熟悉的平面环境中进行反应式导航”，《国际机器人研究杂志》，第41卷，第1期，第85–126页，2022年。

[6] V. Vasilopoulos、S. Castro、W. Vega-Brown、D. E. Koditschck 和 N. Roy，“面向部分已知环境中任务与运动规划的分层审议-反应式系统架构”，载于《2022年国际机器人与自动化会议（ICRA）》，2022年，第7342–7348页。[7] S. Koenig 和 M. Likhachev，“增量启发式搜索的新原理：理论结果”，载于 ICAPS，2006年，第402–405页。[8] B. J. Cohen、S. Chitta 和 M. Likhachev，“利用运动基元进行操作的基于搜索的规划”，载于《2010年IEEE国际机器人与自动化会议》。IEEE，2010年，第2902–2908页。[9] F. Nawaz、M. Sung、D. Gadginmath、J. D’sa、S. Bae、D. Isele、N. Figueroa、N. Matni 和 F. M. Tariq，“用于自主泊车的带动态避障的基于图的路径规划”，arXiv预印本 arXiv:2504.12616，2025年。[10] L. E. Kavraki、P. Svestka、J.-C. Latombe 和 M. H. Overmars，“高维构型空间中的概率路图路径规划”，《IEEE机器人与自动化汇刊》，第12卷，第4期，第566–580页，1996年。[11] S. Karaman 和 E. Frazzoli，“用于最优运动规划的基于采样的算法”，《国际机器人研究杂志》，第30卷，第7期，第846–894页，2011年。[12] A.-A. Agha-Mohammadi、S. Chakravorty 和 N. M. Amato，“Firm：运动不确定性与不精确测量下的基于采样的反馈运动规划”，《国际机器人研究杂志》，第33卷，第2期，第268–304页，2014年。[13] L. Janson、E. Schmerling、A. Clark 和 M. Pavone，“快速行进树：一种用于多维空间最优运动规划的快速行进采样方法”，《国际机器人研究杂志》，第34卷，第7期，第883–921页，2015年。[14] N. Wang 和 R. G. Sanfelice，“混合动力系统的运动规划：框架、算法模板与一种基于采样的方法”，《国际机器人研究杂志》，第02783649241312695页，2025年。[15] M. Stilman 和 J. J. Kuffner，“可移动障碍物间导航：复杂环境中的实时推理”，《国际类人机器人杂志》，第2卷，第04期，第479–503页，2005年。[16] G. Wilfong，“存在可移动障碍物时的运动规划”，载于《第四届计算几何年度研讨会论文集》，1988年。[17] M. Stilman、J.-U. Schamburek、J. Kuffner 和 T. Asfour，“可移动障碍物间的操作规划”，载于《2007年IEEE国际机器人与自动化会议论文集》。IEEE，2007年，第3327–3332页。[18] M. Stilman 和 J. Kuffner，“带人工约束的可移动障碍物间规划”，《国际机器人研究杂志》，第27卷，第11-12期，第1295–1307页，2008年。[19] S. K. Moghaddam 和 E. Masehian，“通过递归方法规划可移动障碍物间的机器人导航（NAMO）”，《智能与机器人系统杂志》，第83卷，第603–634页，2016年。[20] D. Nieuwenhuisen、A. F. van der Stappen 和 M. H. Overmars，“在可移动障碍物环境中的有效路径规划框架”，载于《机器人算法基础VII：第七届机器人算法基础国际研讨会精选论文》。施普林格，2008年，第87–102页。[21] J. Van Den Berg、M. Stilman、J. Kuffner、M. Lin 和 D. Manocha，“可移动障碍物间路径规划：一种概率完备方法”，载于《机器人算法基础 VIII：第八届机器人算法基础国际研讨会精选论文》。施普林格，2010年，第599–614页。[22] S. B. Bayraktar、A. Orthey、Z. Kingston、M. Toussaint 和 L. E. Kavraki，“利用因子化状态空间中的路径碎片整理求解重排谜题”，《IEEE机器人与自动化快报》，第8卷，第8期，第4529–4536页，2023年。[23] Y. Zhang 和 Y. Kantaros，“Namo-llm：大语言模型引导的高效可移动障碍物间导航”，《IEEE机器人与自动化快报》，2025年。[24] H.-n. Wu、M. Levihn 和 M. Stilman，“未知环境中的可移动障碍物间导航”，载于《2010年IEEE/RSJ智能机器人与系统国际会议》。IEEE，2010年，第1433–1438页。[25] M. Levihn、J. Scholz 和 M. Stilman，“可移动障碍物间导航的分层决策理论规划”，载于《机器人算法基础X：第十届机器人算法基础研讨会论文集》。施普林格，2013年，第19–35页。[26] ——，“具有不确定动力学的连续环境中的可移动障碍物规划”，载于《2013年IEEE国际机器人与自动化会议》

，2013，第 3832–3838 页。

[27] B. Renault、J. Saraydaryan 和 O. Simonin，《迈向 s-namo：在可移动障碍物间进行具有社会意识的导航》，载于《RoboCup 2019：机器人世界杯第二十三届 23》。施普林格，2019年，第 241–254 页。[28] K. Ellis、D. Hadjivelichkov、V. Modugno、D. Stoyanov 和 D. Kanoulas，《通过将多个物体推入存储区域，实现可移动障碍物间导航》，《IEEE Access》，第 11 卷，第 3174–3183 页，2023年。[29] S. Armleder、E. Dean-Leon、F. Bergner、J. R. Guadarrama Olvera 和 G. Cheng，《在存在可移动障碍物的非结构化环境中导航时，对未知物体进行基于触觉的协商》，Advanced Intelligent Systems，第 6 卷，第 3 期，文章编号 2300621，2024年。[30] M. Levihn、K. Nishi waki、S. Kagami 和 M. Stilman，《自主环境操作以辅助人形机器人运动》，载于 2014年 IEEE机器人与自动化国际会议（ICRA），2014年，第 4633–4638 页。[31] M. Levihn 和 H. Christensen，《在未知环境中将环境物体用作工具》，载于 2015年 IEEE-RAS 第 15 届人形机器人国际会议（Humanoids），2015年，第 160–165 页。[32] Y. Labb´e、S. Zagoruyko、I. Kalevatykh、I. Laptev、J. Carpentier、M. Aubry 和 J. Sivic，《用于高效视觉引导重排规划的蒙特卡洛树搜索》，IEEE机器人与自动化快报，第 5 卷，第 2 期，第 3715–3722 页，2020年。[33] A. Krontiris 和 K. E. Bekris，《应对棘手的物体重排问题实例》，载于《机器人学：科学与系统》，第 1123 卷，2015年。[34] J. Ahn 和 C. Mavrogiannis，《Relopush：使用非完整约束移动机器人推手在受限空间中进行多物体重排》，arXiv 预印本 arXiv:2409.18231，2024年。[35] K. Ren、L. E. Kavraki 和 K. Hang，《通过运动学-动力学规划和动态规划时域实现基于重排的操作》，载于 2022年 IEEE/RSJ智能机器人与系统国际会议（IROS）。IEEE，2022年，第 1145–1152 页。[36] A. A. Nguyen、F. Jabbari 和 M. Egerstedt，《异构多智能体系统中的互利相互作用》，载于 2023年 IEEE第62届决策与控制会议（CDC）。IEEE，2023年，第 411–418 页。[37] A. A. Nguyen、M. Rodriguez Curras、M. Egerstedt 和 J. N. Pauli，《将互利共生关系作为多机器人协作的框架》，Frontiers in Robotics and AI，第 12 卷，文章编号 1566452，2025年。[38] A. A. Nguyen、F. Jabbari 和 M. Egerstedt，《通过异构多机器人系统中的协作提升韧性》，IEEE控制系统开放期刊，2024年。[39] A. A. Nguyen、L. Guerrero-Bonilla、F. Jabbari 和 M. Egerstedt，《异构多机器人团队中可扩展的成对协作》，IEEE控制系统快报，第 8 卷，第 604–609 页，2024年。[40] M. Egerstedt，《机器人生态学：面向长时自主运行的基于约束的设计》。普林斯顿大学出版社，2021年。[41] X. Luo、Y. Kantaros 和 M. M. Zavlanos，《一种用于多机器人时序逻辑最优控制综合的无抽象方法》，IEEE机器人学汇刊，2021年。[42] J. Wang、W. Chi、C. Li、C. Wang 和 M. Q.-H. Meng，《神经RRT*：基于学习的最优路径规划》，IEEE自动化科学与工程汇刊，第 17 卷，1758，2020年4，1748 no. pp. – 。[43] B. Ichter、J. Harrison 和 M. Pavone，《学习用于机器人运动规划的采样分布》，载于 2018年 I EEE机器人与自动化国际会议（ICRA）。IEEE，2018年，第 7087–7094 页。[44] R. Liu、S. Li 和 X. Yin，《Nngtl：面向移动机器人的神经网络引导最优时序逻辑任务规划》，载于 2024年 IEEE机器人与自动化国际会议（ICRA），第 10 496–10 502 页。[45] J. J. Johnson、A. H. Qureshi 和 M. C. Yip，《利用变换器学习采样字典，实现高效且可泛化的机器人运动规划》，IEEE机器人与自动化快报，第 8 卷，第 12 期，第 7946–7953 页，2023 年。[46] Y. Kantaros 和 M. M. Zavlanos，《Stylus*：面向大规模多机器人系统的时序逻辑最优控制综合算法》，《国际机器人研究杂志》，第 39 卷，第 7 期，第 812–836 页，2020年。[47] A. H. Qureshi、Y. Miao、A. Simeonov 和 M. C. Yip，《运动规划网络：弥合基于学习的方法与经典运动规划器之间的差距》，IEEE机器人学汇刊，第 37 卷，66，2020年1，48no. pp. – 。[48] M. Feng、S. Li 和 X. Yin，《RRT* Former：利用变换器进行环境感知的基于采样的运动规划》，arXiv 预印本 arXiv:2511.15414，2025年。[49] S. Natraj、B. Sinopoli 和 Y. Kantaros，《用于加速基于采样的运动规划的保形化非均匀采样策略》，arXiv 预印本 arXiv:2511.04835，2025年。[50] I. A. Sucan 和 L. E. Kavraki，《通过内外部单元探索进行动力学运动规划》，载于机器人算法基础 *VIII*。施普林格，2009年，第449–464 页。[51] Y. Lee、W. Thomason、Z. Kingston 和 L. E. Kavraki，《基于仿真衍生的可行动作进行物体重构型》，载于*2023*年*IEEE*国际机器人与自动化会议 *(ICRA)*，2023年，第8104–8111页。