arXiv:2304.07357v2 [cs.RO] 5 Sep 2024

凸几何体间高效的增量式穿透深度估计

Wei Gao

摘要—穿透深度（PD）在机器人学中至关重要，因为它广泛应用于动态仿真、运动规划、触觉渲染等领域。扩展多面体算法（EPA）是解决这一问题的事实标准，它通过扩展隐式集合的内部多面体近似来估计穿透深度。本文提出一种基于优化的新算法，可增量式地估计最小穿透深度及其方向。我们的方法一大优势是能够利用时空相干性进行热启动。这种相干性在许多机器人应用中自然出现（例如，相邻仿真时间节点之间的时间连贯性）。因此，我们的算法实现了显著的加速比——在多个基准测试中，其速度比 EPA 快 5–30 倍。此外，我们的方法与 EPA 基于相同的隐式几何表示，因此能够轻松集成到现有软件栈中。代码和补充文档见：https://github.com/weigao95/mind-fcl。

I. 引言

穿透深度（PD）是一种距离度量，用于表征两个重叠形状相互穿入的程度。穿透深度对各种机器人应用都非常重要。例如：1）在动态仿真中，几乎所有刚体接触模型都使用穿透深度来计算接触力 [24], [16], [22] ； 2）在运动规划中，许多规划器 [23], [20], [10]旨在调节（最小化）穿透深度，以减轻并避免碰撞；3）在触觉渲染中，穿透深度 [7], [14], [12], 用于处理物体间的相互作用。穿透深度估计通常与二元碰撞检测结合使用，后者以重叠形状对作为输入 [2], [19]。一种常见的加速碰撞检测和穿透深度估计的策略，是将计算分为两个阶段 [4]。第一阶段是“宽相位”，用于排除距离过远的形状对。第二阶段是“窄相位”，用于检查通过宽相位的形状对是否确实发生碰撞，并计算发生碰撞的形状对的穿透深度。本文重点讨论窄相位中的穿透深度估计，具体如下。

问题表述。 我们研究两个闭凸形状 $A_1, A_2 \subset R^n$。通常，空间维度 $n = 2$ 或 $n = 3$对应二维（2D）或三维（三维）设置。非凸形状可以通过计算其凸包 [15] 或进行凸分解 [13]来处理。如果两个形状A_1, A_2 相互碰撞，当 $A_1 \cap A_2 \neq \emptyset$。对于两个$A_1, A_2$处于碰撞状态的形状，

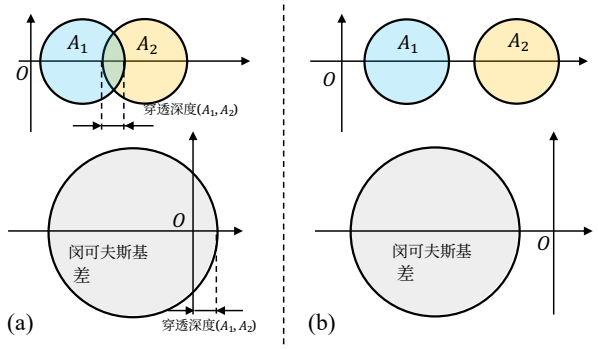

图 1： 穿透深度和闵可夫斯基差示意图。(a) 对于两个发生碰撞的形状，它们的闵可夫斯基差包含原点 O。穿透深度 PD(A_1, A_2) 也是从原点 O 到它们的闵可夫斯基差边界的最短距离。(b) 对于两个未重叠的形状，原点 O 位于它们的闵可夫斯基差之外，因此 PD 未定义。

穿透深度 PD(A_1, A_2) 定义为

$$\begin{aligned} \text{PD}(A_1, A_2) = \ & \min_{d_{1,2} \in R^n} \ \|d_{1,2}\|_2 \\ & \text{subject to:} \ \ \text{interior}(A_1 + d_{1,2}) \cap A_2 = \emptyset \end{aligned} \tag{1}$$

其中， $\|\cdot\|_2$ 为 L2 范数；$A_1 + d_{1,2}$ 表示对A_1中的每个点施加平移$d_{1,2} \in R^n$ 后得到的集合。问题 (1) 的最优解位移 $d^*_{1,2} \in R^n$ 称为最小穿透位移。单位向量$n^*_{1,2} =$ ，即 normalized($d^*_{1,2}$)，通常称为最小穿透方向。显然，$d^*_{1,2} = n^*_{1,2} \times \text{PD}(A_1, A_2)$。示意图见图 1。

相关工作。 目前最广泛使用的穿透深度估计方法是 EPA [25], ，它与用于碰撞检测的 Gilbert-Johnson-Keerthi（GJK）算法 [6] 配合使用。GJK 和 EPA 都基于闵可夫斯基差 [6],进行运算；闵可夫斯基差是由原始形状A_1和A_2 构造出的凸集（闵可夫斯基差的详细解释见第 II 节）。直观而言，EPA 算法通过构建闵可夫斯基差的内接多面体近似来估计穿透深度。EPA 算法需要以一个内接于闵可夫斯基差的四面体作为输入，该四面体由 GJK 算法针对每对发生碰撞的形状生成。基于闵可夫斯基差的表述，GJK 和 EPA 能够处理任意凸形状，例如凸多面体和基本几何体（即球体、长方体等）。这些优点使 EPA 成为穿透深度估计的事实标准。然而，使用 EPA 进行精确的穿透深度估计，可能需要构建包含大量顶点和面的内接多面体近似。本文采取了不同的

Mech-Mind Robotics。邮箱：gaowei19951004@hotmail.com

方法，通过求解一个优化问题来增量估计穿透深度。我们方法的一项关键优势是能够利用许多机器人应用中的时空相干性，为优化过程提供热启动。例如，在动态仿真中，物体对在连续时间步中的穿透方向和深度往往非常相似。利用前一时间结点的信息初始化我们的增量式穿透深度算法，可以显著提升速度。

在 DEEP [11],中也探索了增量式穿透深度估计的思路，该方法通过遍历闵可夫斯基差的顶点来寻找穿透深度的局部最优解。与 [11],相比，我们的方法能够处理任意凸形状，而 [11] 仅适用于凸多面体。具体而言，所提出的算法能够计算两个基本形状之间或基本形状与多面体之间的穿透，而 [11] 不支持此类计算。此外，第 IV 节的实验结果表明，所提出的方法比 [11]快约 2 倍。在非凸几何体穿透计算这一更广泛的领域中，已有多项贡献 [9], [21], [17] 提出迭代计算闵可夫斯基差的切空间并向其投影的方法，这在概念上与我们的方法相关。与这些方法相比，我们方法的关键技术区别在于：1）一种为凸几何体构造闵可夫斯基差切空间的新方法（第 III-B 节）；2）一种用于减少迭代次数并提升性能的提前终止机制（第III-C节）。非凸穿透算法 [9], [21], [17] 与凸穿透算法 [25],[11]在问题复杂度和计算性能之间存在不同的权衡。这两类算法在实际应用中互为补充。研究人员还提出了显式构造闵可夫斯基差 [3] 以进行穿透深度估计的方法。然而，构造过程往往计算开销较大。

贡献。基于奠基性研究 [6], [25],[8], ，本文提出了一种用于一般凸形状之间的增量式穿透深度估计算法。具体而言，

- 我们将穿透深度估计表述为一个“差分凸”问题，通常采用序列二次规划（SQP）求解 [1]。将序列二次规划应用于穿透深度估计的主要挑战在于几何表示是隐式的（闵可夫斯基差和支撑函数 （第 II-B 节中对此进行了说明）。为了解决这一问题，我们提出了一种新颖的 SQP 实例化方法，该方法将经过修改的闵可夫斯基门户细化（MPR）算法 [8]用作子程序。
- 我们提出了一种新颖的快捷机制，可减少原始SQP程序在穿透深度估计中的计算量。此外，我们证明，尽管采用了该快捷机制，算法仍能收敛到局部最优解。
- 我们在多个基准测试上对所提方法进行了实验评估。与 EPA 相比，在准确性相当的情况下，我们的方法实现了 5–30 倍的加速。
- 我们的方法可以轻松集成到现有的软件堆栈中，并且我们提供了一个开源实现——

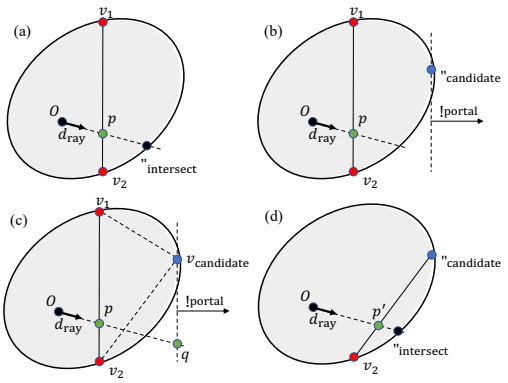

图 2： 改进后的 MPR 算法示意图，该算法用于计算原点射线与闵可夫斯基差边界的交点（即 (a) 中的 $v_{\text{intersect}}$ ）。算法以射线方向 d^{ray} 为输入。在每次迭代中，算法都会维护并更新一个“门户”；在二维情况下，门户是一条线段，在三维情况下则是一个三角形，且该门户内接于闵可夫斯基差。从 (a) 到 (c) 的示意图展示了一次 MPR 迭代：门户从 (a) 中的 (v_1,v_2) 更新为 (c) 中的新门户 $(v_2,v_{\text{candidate}})$。

定以便于在机器人应用中使用。

本文结构如下：第 II 节介绍预备知识。第 III 节阐述穿透深度算法。第 IV 节展示结果。第 V 节总结全文。

II. 预备知识

A. 闵可夫斯基差与穿透深度

如第 I 节所述，我们研究两个闭凸形状$A_1,A_2 \subset R^n$。通常维度$n = 2$ 或$n = 3$。,2 对A_1 和A_2 的闵可夫斯基差$\mathscr{D}_1$定义为

$$\mathscr{D}_{1,2} = A_1 - A_2 = \{v = v_1 - v_2 | v_1 \in A_1, v_2 \in A_2\} \quad (2)$$

根据 [25]中的证明，以下性质适用于$\mathscr{D}_{1,}2$：

- $\mathscr{D}_{1,}2$ 是一个闭凸集。
- $A1$ 和 $A2$ 重叠，当原点 $O \in \mathscr{D}_{1,}2$。
- 对于两个重叠形状 $A1$ 和 $A2$，问题 (1) 中定义的穿透深度 PD($A1$,$A2$) 对应于从 O 到 $\mathscr{D}_{1,}2$ 的边界的最小距离。

图 1 展示了一个示例。利用这些性质，我们可以将第 I 节中的问题 (1) 重新表述为：

$$\begin{aligned} \text{PD}(A_1,A_2) = \ & \min_{v \in R^n} \quad ||v||_2 \\ & \text{subject to:} \quad v \in \text{boundary}(\mathscr{D}_{1,2}) \end{aligned} \quad (3)$$

设边界上的点v^* $\mathscr{D}_{1,}2$ 是问题 (3) 的一个最优解。第 I 节中的最小穿透位移$d^*_{1,}2 = -v^*$（换言之，interior($A_1 - v^*$)$\cap A_2 = /0$）。第 I 节中的最小穿透方向可计算为 $n^*_{1,2}$ =normalized($-v^*$)。

B. 凸形状的支撑函数

GJK/EPA [6], [25] 使用一种称为支撑函数的隐式函数几何表示。对于给定的凸集A，我们可以将其支撑函数 $\text{supp}_A(\cdot)$ 定义为

$$\text{supp}_A(d) = \text{argmin}_{x \in A} \ \text{dot}(x,d) \ \subset A \quad (4)$$

其中，dot(·, ·)表示点积，d 是一个单位向量，记作支撑方向。直观地说，支撑函数会计算A 中沿支撑方向最远的点。对于某些d，满足公式（4）的点可能不止一个。按照已有研究的做法，我们假设 $\text{supp}_A(\cdot)$ 对每个d都会返回一个点，该点可以从满足公式（4）的点中任意选取。支撑函数与凸集的支撑超平面 [1]密切相关。对于给定的方向d 和$v = \text{supp}_A(d)$，平面 $\{x|\text{dot}(d, x - v) = 0\}$ 是凸形状A 在点$v \in A$处的支撑平面。

如 [6], 所示，我们可以利用两个形状A_1, A_2 的支撑函数 $\text{supp}_{A_1}(\cdot)$ 和 $\text{supp}_{A_2}(\cdot)$，构造它们的闵可夫斯基差的支撑函数 $\text{supp}_{\mathscr{D}_{1,2}}(\cdot)$。具体而言，

$$\text{supp}_{\mathscr{D}_{1,2}}(d) = \text{supp}_{A_1}(d) - \text{supp}_{A_2}(-d) \tag{5}$$

换言之，我们可以利用 $\text{supp}_{A_1}(d)$ 和 $\text{supp}_{A_2}(-d)$，从 $\text{supp}_{\mathscr{D}_{1,2}}(d)$ 中得到一个点。本文中的算法将通过支撑函数访问闵可夫斯基差。

C. 原点射线与闵可夫斯基门户细化

我们考察两个重叠形状A_1 和A_2，因此它们的闵可夫斯基差 $\mathscr{D}_{1,2}$ 包含原点。给定方向d_{ray} 作为输入，我们定义一条原点射线，它从原点O 出发并沿d_{ray} 延伸，定义如下：

$$\text{origin_ray}(d_{\text{ray}}) = \{x | x = t\ d_{\text{ray}},\ \text{for } t \geq 0\} \tag{6}$$

MPR [8] 算法最初是为二元碰撞检测提出的。本小节介绍一种经过修改的 MPR，用于计算原点射线与$\mathscr{D}_{1,2}$ 边界的交点：

$$v_{\text{intersect}}(d_{\text{ray}}) = \text{origin_ray}(d_{\text{ray}}) \cap \text{boundary}(\mathscr{D}_{1,2}) \tag{7}$$

如图 2 (a) 所示，MPR 会维护并更新一个门户，它是内接于闵可夫斯基差 $\mathscr{D}_{1,2} \subset R^n$ 且与原点射线相交的 n-单形。对于二维情形，门户是内接于 $\mathscr{D}_{1,2}$ 的线段。例如，图 2 (a) 中的线段 (v_1, v_2) 是一个门户，它与原点射线交于p。对于三维情形，门户是一个三角形。门户通过“查找门户”过程进行初始化， [8], 如算法 1 所示。

在每次迭代中，MPR 都会计算门户法向量d_{portal}。然后，以d_{portal}为参数调用支撑函数 $\text{supp}_A(\cdot)$，得到一个点$v_{\text{candidate}}$。由于原点射线与门户 (v_1, v_2) 相交，因此它必定与线段 $(v_1, v_{\text{candidate}})$ 和 $(v_2, v_{\text{candidate}})$ 中的一条相交。与射线相交的那条线段将成为下一次迭代的新门户。这就是算法 1 中的门户更新步骤。迭代会持续进行，直到 p 在门户法向量方向上的误差不超过某个容差 Δ，从而成为$v_{\text{intersect}}$的良好近似，如图 2 所示。

算法 1 第 II-C 节中用于原点射线相交的 MPR

要求： $\mathscr{D}_{1,2}$ 具有支撑函数 $\text{supp}_{\mathscr{D}_{1,2}}(\cdot)$
要求： 射线方向d_{ray}
要求： 容差 Δ
门户$_0 \leftarrow$查找门户($\mathscr{D}_{1,2}$)
当 $k = 0, 1, 2, \ldots$ **执行**
  $d_{\text{portal}_k} \leftarrow$门户法向量(门户$_k$)
  $v_{\text{candidate}} \leftarrow$补充$_{\mathscr{D}_{1,2}}$(门户$_k$)
  $p_k \leftarrow$ 求交（原点射线(d_{ray}), $d_{\text{portal}\ k}$) -
  ▷ 平面由其法向量及平面上的一个点定义
  $q_k \leftarrow$ 求交（原点射线(d_{ray}), 平面($d_{\text{portal}\ k}$, $v_{\text{candidate}}$))
  如果 |点积($q_k - p_k$, 归一化($d_{\text{portal}\ k}$))| $\leq \Delta$ **则**
    返回 (p_k, $v_{\text{candidate}}$, $d_{\text{portal}\ k}$)-
  结束如果
  门户$_{k+1} \leftarrow$更新门户(portal$_k$, $v_{\text{candidate}}$)
结束 while

III. 穿透深度估计算法

如第 II 节所示，穿透深度估计可表示为

$$\begin{aligned} &\min_{v \in R^n} \quad ||v||_2 \\ &\text{subject to:} \quad v \in \text{boundary}(\mathscr{D}) \end{aligned} \tag{8}$$

其中，$\mathscr{D}$ 是包含原点的闵可夫斯基差。为简化记号，省略了$\mathscr{D}_{1,2}$ 的下标。如第 II-A 节所示，我们只能通过其支撑函数 $\text{supp}_{\mathscr{D}}(\cdot)$访问凸集 $\mathscr{D}$ 。这种隐式几何表示意味着：

1) 对于给定点 $v \in$边界($\mathscr{D}$)，很难计算一个经过v的 $\mathscr{D}$ 支撑超平面（或等价地，法向量为法向量的超平面）。许多优化过程都需要这一操作，例如第 III-A 节中的 SQP。2) 支撑函数 $\text{supp}_{\mathscr{D}}(\cdot)$ 实际上是操作 1) 的“逆运算”：对于给定的法向方向d，我们可以计算一点 $v = \text{supp}_{\mathscr{D}}(d)$。这一点与法向量$d$结合后，能够为 $\mathscr{D}$定义一个支撑超平面。

本节将解决这一挑战，并提出一种基于优化的 PD 估计算法。

为清晰起见，我们首先在第 III-A 节和第 III-B 节介绍两种“概念性”算法。第一种算法着重说明 SQP 过程，并假设我们能够计算$v \in$边界($\mathscr{D}$)的支撑超平面。正如上文所述，这一假设对我们的问题并不成立。为此，我们提出一种新颖的 SQP 实例化方法，将经过修改的 MPR 算法用作子程序。这是第 III-B 节中的第二种“概念性”算法。最后，我们介绍实际的 PD 算法，该算法采用捷径机制来减少计算量并显著加快速度。

A. 表述为差分凸问题

在本小节中，我们研究以下优化问题，它与问题 (8) 非常相似：

$$\begin{aligned} &\min_{v \in R^n} \quad ||v||_2 \\ &\text{subject to:} \quad v \in \text{boundary}(\mathscr{E}) \end{aligned} \tag{9}$$

其中，$\mathscr{E}$ 也是R^n 中的一个闭凸集，且包含原点。与问题 (8) 不同，我们假设 $\mathscr{E}$ 具有显式表示，因此可以为每个点 $v \in \text{boundary}(\mathscr{E})$ 计算一个支撑超平面。

算法 2 用于问题 10 的序列二次规划

要求： $\mathscr{E}$ 支持计算支撑超平面(·)
要求： $v_{\text{init}} \in$ 边界$(\mathscr{E})$

$v_0 \leftarrow v_{\text{init}}$
当 $k = 0, 1, 2, \ldots$ **执行**
  $n_{\text{plane } k} \leftarrow$ 计算支撑超平面$(\mathscr{E}, v_k)$
  ▷ 由法向量 $n_{\text{plane } k}$ 和其上的一点 v_k 定义的平面
  $z_k \leftarrow$将原点投影到平面(平面$(v_k, n_{\text{plane } k})$)
  $v_{k+1} =$ 边界交点$(O$ 到 z_k，边界$(\mathscr{E}))$
结束循环
返回 v_k

由于闵可夫斯基差 $\mathscr{D}$ 不满足这一假设，本小节提出的PD算法仅为“概念性”算法，具体实现将在第 III-B 节给出。

举一个具体例子，凸形状 $\mathscr{E}$ 可以表示为连续凸函数$g(v)$的水平集。换言之，$\mathscr{E} = \{v|g(v) \leq 0\}$ 且边界为$\{v|g(v) = 0\}$。对于一点 $v \in$ 边界$(\mathscr{E})$，我们可以计算一个支撑超平面，其法向量为 $\nabla g(v)$，其中 ∇ 是次梯度算子。

问题（9）中决策变量的定义域仅包含 $\mathscr{E}$ 的边界。由于凸集 $\mathscr{E}$ 包含原点，我们可以将输入域扩大到 $\mathscr{E}$ 内部以外的所有位置，并将优化问题改写为

$$\begin{aligned} &\min_{v \in R^n} \quad ||v||_2 \\ &\text{subject to:} \quad v \in (R^n \setminus \text{inner}(\mathscr{E})) \end{aligned} \tag{10}$$

其中，$R^n \setminus$inner$(\mathscr{E})$ 是 inner$(\mathscr{E})$ 的补集。问题（10）可以使用序列二次规划 [1] 算法求解。此外，问题（10）是一个差分凸问题。因此，可以利用其他性质简化序列二次规划：

- 序列二次规划通常需要一个信赖域。对于差分凸问题，该信赖域可以是无穷大 [1]。
- 约束 $v \in (R^n \setminus$inner$(\mathscr{E}))$ 的线性化是支撑超平面外侧的一个半空间。因此，序列二次规划的内部二次规划步可以通过将原点投影到支撑超平面上来实现。

序列二次规划算法见算法 2，该算法在以下步骤间交替进行：1）计算支撑超平面；2）将原点投影到该超平面上；3）计算投影线与边界的交点。详细推导见补充材料。

算法 2 无法直接应用于闵可夫斯基差 $\mathscr{D}$，因为它需要显式的几何表示，才能为边界点计算支撑超平面。第 III-B 节将解决这一难题。

B. 使用 *MPR* 子程序进行实例化

在本小节中，我们将算法 2 中的 SQP过程应用于闵可夫斯基差 $\mathscr{D}$，它由其支撑函数 supp$_{\mathscr{D}}$(·) 表示。对于边界$(\mathscr{D})$上的每个点 $v \in$，我们都可以计算其方向 $d_v =$ normalized(v)。我们建议将d_v 用作决策变量

算法3 使用支撑函数的序列二次规划

要求： 支撑函数 supp$_{\mathscr{D}}$(·)
要求： 初始方向d_{init}
要求： MPR 子程序的容差 Δ

$d_0 \leftarrow d_{\text{init}}$
当 $k = 0, 1, 2, \ldots$ **执行**
  MPR 输出 $\leftarrow$ mpr(supp$_{\mathscr{D}}$, d_k, Δ) ▷算法 1
  $(p_k, v_{\text{candidate}}, d_{\text{portal } k}) \leftarrow$ MPR 输出
  支撑平面 $\leftarrow$ Plane$(v_{\text{candidate}}, d_{\text{portal } k})$
  $z_k \leftarrow$将原点投影到平面(支撑平面)
  如果 应终止$(p_k, z_k, d_{\text{portal } k})$ **则**
    返回 相交(原点射线(d_k),支撑平面)
  结束如果
  $d_{k+1} \leftarrow d_{\text{portal } k}$
结束循环

并基于以下观察，在问题（10）中通过d_v隐式表示v：

1）点 $v \in$boundary$(\mathscr{D})$ 可以通过 d_v估算（在某个容差 Δ 下可达到更高的准确性），并将第 II-C 节中的 MPR 算法用作子程序。2）更重要的是，收敛后，v 处的支撑超平面法向量可以用门户法向量 d_{portal}近似。需要强调的是，通常 $d_{\text{portal}} \neq d_v$。

基于这些观察，我们提出了用于闵可夫斯基差 $\mathscr{D}$的算法 3，该算法由算法 2 转换而来。由于该算法将在第III-C 节中被取代，我们在本小节中仅作直观分析。

算法3的一项关键性质是使用 $d_{\text{portal } k}$近似支撑超平面法向量，图 3 对此作了进一步说明。当容差为 Δ 的 MPR（算法 1）子程序收敛时，方向 d_v 上的真实交点v 必定位于门户（图 3 中的线段 $(v_{\text{portal 1}}, v_{\text{portal 2}})$）与$v_{\text{candidate}}$处支撑超平面之间。门户与支撑超平面相互平行（它们共享相同的法向量 d_{portal}），且由于 MPR 在本次迭代中终止，这两个平面之间的距离不超过 Δ。算法 1 使用 p 近似 v，并使用 d_{portal} 近似v处的支撑超平面法向量。这种近似等效于从 $\mathscr{D}$ 中移除被门户与原点隔开的点，如图 3(b) 所示。这是因为p 是图 3(b) 中形状的精确交点，而d_{portal} 是该交点处精确的支撑超平面。被移除的部分是一个凸区域，其沿d_{portal} 方向的“厚度”不超过 Δ。因此，容差 Δ 设得越小，近似就越准确。

算法3在实践中可能有效。然而，MPR子程序可能需要很多次迭代，计算开销很大。我们将在第III-C节讨论这一问题。

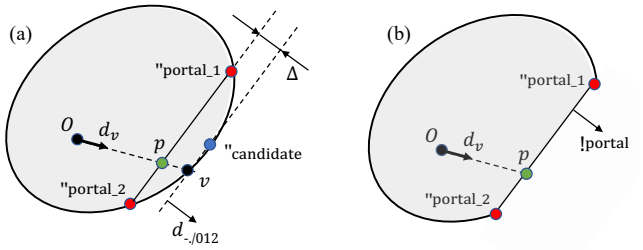

图 3： 第 III-B 节和算法3中使用的近似方案。(a) 展示了 MPR 以容差 Δ 收敛时，门户与支撑超平面的配置；此时 MPR 被用作算法 3 中的子程序。直观而言，第 III-B 节中的近似相当于移除最终门户“外部”的所有点，如 (b) 所示。详细说明见第 III-B 节。

C. 使用快捷方法估计穿透深度

在现有软件栈中， [19], [2], MPR通常采用较小的容差，以便进行精确的二元碰撞检测。然而，算法3中的MPR子程序主要用于通过支撑超平面法向量d_{k+1} 生成搜索方向d_{portal}。因此，我们提出一种提前终止机制（快捷方法）：一旦发现“足够好”的d_{portal} ，便减少MPR迭代次数。

图4 展示了 MPR 子程序中门户与支撑超平面的配置。门户与支撑超平面之间的距离可能远大于容差 Δ，因为这不一定是最后一次迭代。设点 p 为原点射线(d_v)与门户的交点。MPR 子程序使用 p 作为v 的当前估计值（即与 $\mathscr{D}$ 边界的交点），且 $|p| \leq |v|$。换言之，如果我们沿方向 d_v继续执行 MPR 子程序，那么我们求得的穿透深度将下 界于 $|p|$。

此外，我们可以将原点O 投影到支撑超平面上，得到z，如图4所示。令v_z 为线段（O,z）与 $\mathscr{D}$的边界的交点。显然， $|v_z| \leq |z|$。如果切换到新的搜索方向$d_z = \text{normalized}(z)$，则我们找到的穿透深度$upper$ 有上界 $|z|$。

根据上述分析，在每次MPR子程序迭代中，我们都有：1）沿当前搜索方向d_v的穿透深度下界 $|p|$；以及2）一个新的搜索方向d_z 候选点，其沿该方向的穿透深度上界为 $|z|$。如图4所示，当沿新方向d_z 的上界z 小于沿原方向d_v的下界时，我们建议切换到新的搜索方向d_z 。

根据上述分析，整体穿透深度估计算法总结于算法4中，该算法使用算法5中的一个子程序。序列二次规划过程与算法3几乎完全相同，区别在于MPR子程序改为算法5。MPR子程序会在满足上述条件时提前终止，如算法5所示。尽管采用了新的捷径机制，该序列二次规划仍保证收敛到局部最优解。详细分析请参阅补充材料。

D. 实现细节：初始化

算法4需要一个初始方向d_{init}。这是对可能的最小位移方向的猜测

算法4 用于穿透深度估计的序列二次规划算法

需要： 支撑函数 $\text{supp}_{\mathscr{D}}(\cdot)$**需要：** 初始方向$d_{\text{init}}$在算法5中用 mpr 快捷方法替换算法3中调用的 MPR 子程序。算法3中的容差 Δ 不再需要。

算法5 第III-C节中带快捷方法的MPR子程序 要求：$\mathscr{D}_1$, 2，支撑函数 $\text{supp}_{\mathscr{D}_1,2}(\cdot)$ 要求：射线方向 d_{ray} $\text{portal}_0 \leftarrow$ find portal($\mathscr{D}_1$,2) while k $= 0, 1, 2, \ldots$ do d_{portal} k $\leftarrow$门户法向量(portal$_k$) $v_{\text{candidate}} \leftarrow \text{supp}_{\mathscr{D}_1,2}(\text{portal}_k)$ $p_k \leftarrow$ 相交(原点射线(d_{ray}), portal$_k$)▷ 平面由其法向量及平面上的一个点定义 $z_k \leftarrow$将原点投影到平面(Plane($v_{\text{candidate}}$, d_{portal} k)) if $|z_k| \leq |p_k|$ then ▷ 第III-C节中的快捷方法 return p_k, $v_{\text{candidate}}$, d_{portal} $_k$end if $\text{portal}_{k+1} \leftarrow$更新门户 (portal$_k$, $v_{\text{candidate}}$) end while

将形状A_1 移开，使其不再与形状A_2发生碰撞。对于具有较强空间或时间连贯性的应用，可能存在对最小穿透方向的良好应用特定猜测。例如，在动态仿真中， [24], [16], [22] 碰撞物体对在连续的仿真步中往往具有非常相似的穿透方向和深度。同样，在基于优化的运动规划中， [23], [20], [10] 我们可能会使用上一次优化迭代中的穿透方向作为d_{init}，尤其是在采用信赖域机制、以确保连续迭代之间差异较小的规划算法中。如果没有针对穿透方向的应用特定先验，另一种方法是使用两个形状的质心差作为d_{init} ，如 [11]所建议。

IV. 结果

在本节中，我们通过实验研究所提方法的鲁棒性、准确性和效率。首先，我们将该方法与 EPA [25] 和 DEEP [11]进行比较。随后，我们开展消融研究，以突出第III-C节中引入的捷径机制。我们对参数进行调整，使所有算法的准确性大致相同。源代码和补充文档可通过此链接获取。

A. 与EPA的比较

我们将该方法与 Libccd 库中实现的 EPA [25] 算法进行比较。Libccd [5], 是一款广泛用于机器人应用 [19], [22]的热门库。评估基于两组形状进行。第一组由简单的基本几何体组成，第二组则是顶点数量各不相同的凸多面体。

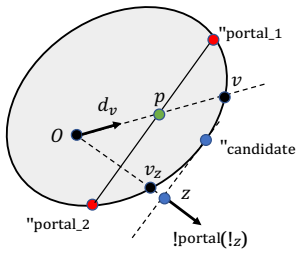

图4： 第III-C节中捷径机制的示意图。沿d_v方向的穿透深度下界为 $|p|$，而沿d_{portal}（d_z）方向的穿透深度上界为z。如第III-C节所述，当 $|p| \geq |z|$时，我们提出切换至d_z 作为新的搜索方向。

基本形状。 我们计算三种形状对之间的穿透深度：

- 球体与球体碰撞（球体与球体）
- 胶囊体与胶囊体碰撞（胶囊体 vs. 胶囊体）
- 球体与胶囊体碰撞（球体 vs. 胶囊体）

对于这些简单的基本形状，我们使用手工编写的穿透深度算法作为真实值。在本实验中，这些形状的特征尺寸（例如球体直径）为 1 米。所有统计结果均为针对形状对随机生成位姿进行 10000 次独立运行所得的平均值。详细设置请参阅补充材料。

我们的方法需要一个初始穿透方向猜测。为进行基准测试，我们采用以下方法获取初始猜测。给定手工编写的算法所提供的真实穿透方向，我们绕一个垂直于真实方向的随机轴旋转n 度。然后，将旋转后的方向用作初始猜测。

表 I 和表 II 展示了我们的方法在不同旋转扰动角度n下的准确性，其中n 范围为5° 至 45°。表 I 展示距离误差，表 II 展示方向误差。我们报告了 10000 次独立运行的平均偏差。EPA 和我们的方法都能准确估计穿透深度。尽管初始化误差较大，所提出的算法仍具有较高的准确性。

表 III 总结了我们方法的性能。与 EPA 相比，我们的方法可将速度提升 20 至 30 倍。此外，对于不同的形状类型以及初始猜测方向的偏差，性能提升均保持一致。

凸多面体。 第二项实验比较了我们的方法与 EPA 在凸多面体上的表现。我们采用类似的球体-胶囊体设置，但将球体替换为不同分辨率的凸多面体近似。更高的分辨率意味着更多的顶点和面，可提高近似的准确性，但代价是计算复杂度增加。

图5展示了所提出的算法在凸多面体顶点数量不同时的性能。所提出的方法始终优于 EPA 基线，加速比约为 6–10 倍。

	Ours (5°)	Ours (25°)	Ours (45°)	EPA
球体与球体	0.909	1.018	1.03	1.58
胶囊体对比娱乐注册胶囊体	1.092	1.12	1.12	0.39
球体与。胶囊体	1.255	1.31	1.30	1.72

表 I： 各种算法与真实值相比的穿透深度误差，单位为微米（10^{-6} 米）。结果误差为 10000 次独立运行的平均值。我们的方法在 5°、 25° 和 45° 处使用不同的角偏差（初始穿透方向）进行评估。几何体的特征尺寸（例如球体的直径）为 1 米。

	Ours (5°)	Ours (25°)	Ours (45°)	EPA
球体与球体	1.82	1.38	1.93	8.84
胶囊体与胶囊体	3.19	3.42	3.23	10.39
球体与胶囊体	2.25	2.31	2.30	10.72

表 II： 各种算法的穿透方向误差（单位：毫弧度，10^{-3} 弧度）与真实值的比较。结果误差为 10000 次独立运行的平均值。我们的方法在 5°、 25° 和 45° 处，针对初始穿透方向的不同角偏差进行评估。

B. 与 *DEEP [11]* 的比较

所提出的方法与 [11]，进行比较；该方法通过沿闵可夫斯基差表面移动，增量式地计算穿透距离。我们使用 C++ 实现基线算法，以及内部的凸多边形相交算法 [18]。我们采用建议的初始化方案：给定穿透方向的初始猜测，沿该方向分别取每个形状的极值顶点，以计算顶点对。如 [11] 只能处理凸多面体，我们采用与第 IV-A 节类似的球-球设置，但将球替换为其凸多面体近似。

图 6 展示了所提出的算法与基线之间的性能比较。初始方向偏差固定为 5° ，并将球体形状离散为不同的分辨率。所提出的方法优于基线 [11] ，加速比约为 2 倍。

	Ours (5°)	Ours (25°)	Ours (45°)	EPA
球体与球体	1.9	2.1	2.3	85.1
胶囊体对比胶囊体	1.6	1.9	2.0	39.5
球体对比胶囊体	2.0	2.2	2.4	61.5

表III： 所提出方法在初始方向猜测的不同角偏差下的性能。结果为 10000 次独立运行的平均耗时，单位为微秒。所提出的方法比 EPA 快 20 至 30 倍。

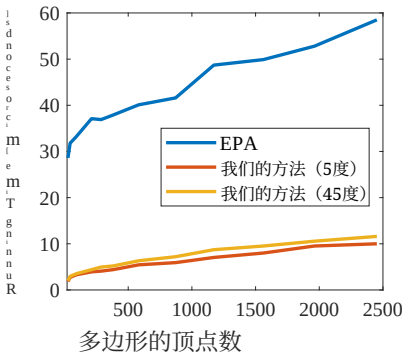

图5： 不同凸多面体顶点数下我们的方法与 EPA 的性能比较 [25] 。结果是基于 10000 次独立运行计算得到的平均时间，单位为微秒。我们的方法在 5° 和 45° 处采用不同的（初始穿透方向）角偏差进行评估。与 EPA 基线相比，所提出的方法实现了 6–10 倍的加速比。

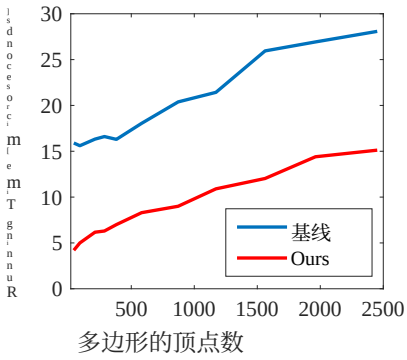

图 6： 不同凸多面体顶点数下我们的方法与 DEEP 的性能比较 [11] 。结果是平均时间，单位为微秒。两种方法均在 5° 处采用角偏差（初始穿透方向）进行评估。与基线相比，所提出的方法速度约快 2 倍。

C. 捷径机制的消融研究

第III-C节介绍了一种捷径机制，通过减少 MPR 迭代次数来提升性能。为突出其优势，我们通过移除所提出算法中的捷径机制进行消融研究。我们采用与第IV-A节相同的凸多面体设置。初始方向偏差固定为 45°。

结果如图7所示。我们使用支撑函数调用次数和运行时间来表征性能。从图中可以看出，所提出的捷径机制可将所需的支撑函数调用次数减半，并实现约 2–3 倍的加速比。

V. 结论

本文提出了一种用于估计凸形状之间穿透深度的新算法。为此，我们将穿透深度估计表述为差分凸优化。随后，我们提出了一种新颖的序列二次规划实现方法，该方法采用经过修改的 MPR 子程序来求解基于优化的穿透深度估计。我们还提出了一种捷径机制，可显著

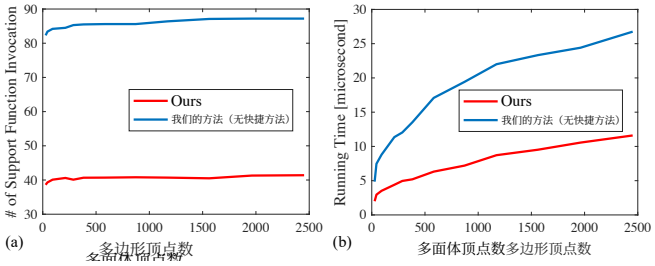

图7： 我们在第III-C节进行了消融研究，以突出所提出的快捷方法的优势。图中，(a) 展示了以支撑函数调用次数衡量的性能，(b) 展示了以微秒为单位的性能。所提出的捷径机制可将所需的支撑函数调用次数减半，并实现约 2–3 倍的加速比。

减少计算量。通过多项实验，我们证明，在准确性相当的情况下，所提方法相较于著名的 EPA 算法可实现 5–30 倍的加速比。

VI. 致谢

本研究是在作者受雇于 Mech-Mind Robotics 期间开展的。本文所表达的观点仅代表作者本人，并不代表资助机构的立场。

参考文献

[1] S. Boyd、S. P. Boyd 和 L. Vandenberghe。《凸优化》。剑桥大学出版社，2004年。[2] E. Coumans 和 Y. Bai。PyBullet：用于游戏、机器人学和机器学习物理仿真的 Python 模块。2016年。[3] D. Dobkin、J. Hershberger、D. Kirkpatrick 和 S. Suri。多面体交深度的计算。《Algorithmica》，9(6):518–533，1993年。[4] C. Ericson。《实时碰撞检测》。CRC出版社，2004年。[5] D. Fiser。Libccd：用于检测两个凸形状之间碰撞的库，2015年。[6] E. G. Gilbert、D. W. Johnson 和 S. S. Keerthi。一种快速计算三维空间中复杂物体间距离的过程。《IEEE机器人与自动化学报》，4:193–203，1988年。[7] A. Gregory、A. Mascarenhas、S. Ehmann、M. Lin 和 D. Manocha。多边形模型的六自由度触觉显示。载于《Visualization 2000论文集》，第139–146页。IEEE，2000年。[8] S. Jacobs。《Game Programming Gems 7》。2008年。[9] C. Je、M. Tang、Y. Lee、M. Lee 和 Y. J. Kim。Polydepth：使用迭代接触空间投影实时计算穿透深度。《ACM图形学汇刊（TOG）》，31(1):1–14，2012年。[10] D. Kappler、F. Meier、J. Issac、J. Mainprice、C. G. Cifuentes、M. W¨uthrich、V. Berenz、S. Schaal、N. Ratliff 和 J. Bohg。实时感知与反应式运动生成相结合。《IEEE机器人与自动化快报》，3(3):1864–1871，2018年。[11] Y. J. Kim、M. C. Lin 和 D. Manocha。使用对偶空间扩展增量估计凸多面体之间的穿透深度。《IEEE可视化与计算机图形学汇刊》，10(2):152–163，2004年。[12] V. Kumar 和 E. Todorov。MuJoCo Haptix：用于手部操作的虚拟现实系统。载于《2015 IEEE-RAS第15届人形机器人国际会议（Humanoids）》，第657–663页。IEEE，2015年。[13] K. Mamou 和 F. Ghorbel。一种简单高效的三维网格近似凸分解方法。载于《2009年第16届IEEE国际图像处理会议（ICIP）》，第3501–3504页。IEEE，2009年。[14] W. A. McNeely、K. D. Puterbaugh 和 J. J. Troy。使用体素采样的六自由度触觉渲染。载于《ACM SIGGRAPH 2005课程》，第42–es页。2005年。[15] A. A. Melkman。简单折线凸包的在线构造。《信息处理快报》，25(1):11–12，1987年。

[16] B. Mirtich。刚体接触：从碰撞检测到力的计算。载于《接触分析与仿真研讨会，IEEE国际机器人与自动化会议》，1998年。[17] G. Nawratil、H. Pottmann 和 B. Ravani。基于运动几何的广义穿透深度计算。《计算机辅助几何设计》，26(4):425–443，2009年。[18] J. O’Rourke、C.-B. Chien、T. Olson 和 D. Naddor。一种用于凸多边形相交的新型线性算法。《计算机图形学与图像处理》，19(4):384–391，1982年。[19] J. Pan、S. Chitta 和 D. Manocha。FCL：用于碰撞与邻近查询的通用库。载于《IEEE国际机器人与自动化会议》，第3859–3866页。IEEE，2012年。[20] J. Schulman、J. Ho、A. X. Lee、I. Awwal、H. Bradlow 和 P. Abbeel。利用序列凸优化寻找局部最优的无碰撞轨迹。Citeseer。[21] M. Tang 和 Y. J. Kim。使用物体范数对刚体与关节模型进行交互式广义穿透深度计算。《ACM图形学汇刊（TOG）》，33(1):1–15，2014年。[22] E. Todorov。具有接触与约束的凸且可解析求逆的动力学：MuJoCo中的理论与实现。载于《2014年IEEE国际机器人与自动化会议（ICRA）》，第6054–6061页。IEEE，2014年。[23] M. Toussaint、K. Allen、K. A. Smith 和 J. B. Tenenbaum。用于工具使用与操作规划的可微物理和稳定模式。[24] J. C. Trinkle、J.-S. Pang、S. Sudarsky 和 G. Lo。具有库仑摩擦的动态多刚体接触问题。《ZAMM应用数学与力学杂志/应用数学与力学杂志》，77(4):267–279，1997年。[25] G. Van Den Bergen。三维游戏物体的邻近查询与穿透深度计算。载于《游戏开发者大会》，第170卷，2001年。