用于交互设备原型设计的成本感知贝叶斯优化

Thomas Langerak
阿尔托大学
赫尔辛基，芬兰

Renate Zhang†
阿尔托大学
赫尔辛基，芬兰

Ziyuan Wang
阿尔托大学
赫尔辛基，芬兰

Per Ola Kristensson
剑桥大学
英国剑桥

Antti Oulasvirta
阿尔托大学 & ELLIS研究所
芬兰赫尔辛基

arXiv:2602.01774v1 [cs.HC] 2 Feb 2026

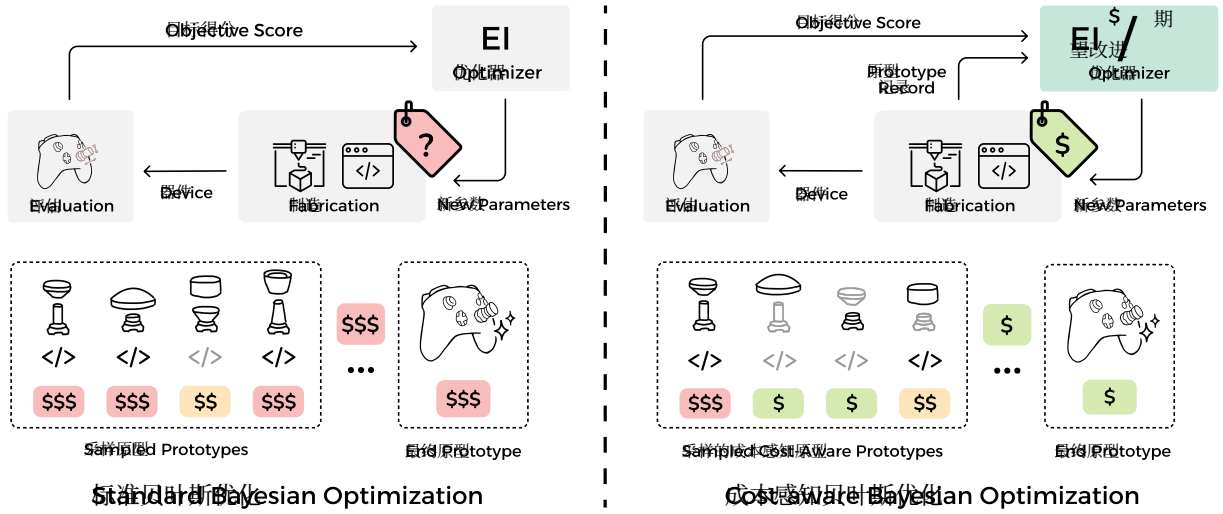


图 1：我们扩展了用于交互设备原型制作的成本感知贝叶斯优化方法。该方法考虑了硬件和软件实验设计的不同成本，同时复用记录中存储的现有原型。通过将成本模型嵌入采集函数，优化器可以权衡期望改进（EI）与成本。由于我们的方法仅作用于采集函数层面，因此不会影响建模过程和代理模型，具有广泛的适用性。

摘要

在迭代式设计中，决定哪些创意值得制作原型是一项核心考量。当预期改进较大且成本较低时，就应制作原型。然而，这很难判断，因为成本可能差异极大：简单的参数微调可能只需几秒，而制造硬件则会消耗材料和能源。这种不对称性可能会阻碍设计师探索设计空间。本文提出对成本感知贝叶斯优化方法的扩展，以纳入多种原型制作成本。该方法基于贝叶斯优化的优势，只需对采集函数进行最小限度的修改。其核心思路是利用设计师估计的成本，引导采样朝着更具成本效益的原型进行。在技术评估中，该方法取得了与成本无关基线相当的效用同时仅需 ≈70% 的成本；在严格预算下，其表现是基线的三倍。一项包含 12 名参与者的现实操纵杆设计任务被试内研究也显示出类似收益。这些结果表明，将原型制作成本纳入考量可以使贝叶斯优化更契合现实世界设计项目。

CCS 概念

• **以人为中心的计算** → **人机交互 (HCI)**；界面设计原型制作；
•**计算理论** → 数学优化。

关键词

成本感知、贝叶斯优化、原型制作、交互设备

ACM 参考文献格式：
Thomas Langerak、Renate Zhang、Ziyuan Wang、Per OlaKristensson 和 Antti Oulasvirta。2026 年。《面向交互设备原型制作的成本感知贝叶斯优化》。载于《*2026* 年 *CHI* 人机交互系统中的人为因素会议论文集》（*CHI '26*），*2026* 年*4*月*13*日至*17*日，西班牙巴塞罗那。ACM，纽约，纽约州，美国，23 页。
https://doi.org/10.1145/3772318.3791024

†工作完成于在阿尔托大学实习期间。同时隶属于维也纳工业大学（维也纳，奥地利）。

本作品采用知识共享署名 4 .0 国际许可协议授权 .
CHI '26，西班牙巴塞罗那© 2026 版权所有者/作者保留版权。ACM ISBN
979-8-4007-2278-3/2026/04
https://doi.org/10.1145/3772318.3791024

1 引言

在原型制作中，成本并非众多因素之一；它会决定性地影响哪些东西会被制作成原型，进而深刻影响最终结果。制作原型时，一切都有标价：时间、劳力、材料和外部服务。以游戏控制器为例，成本会在制造、硬件采购、电子设计、软件开发和集成工程等环节不断累积。根据保真度的不同，费用可能从几百美元到数万美元不等。硬件成本尤其高昂，也是制造[52], 普适计算 [53], 实体计算 [19], 辅助技术 [18], DIY 社群 [37], 和交互设备[30, 31]等人机交互研究领域关注的核心问题。制作实体物品——外壳、致动器、传感器和显示器——需要耗费材料、时间和精力。正如 Holmquist 感叹的那样：“比特很便宜，原子很昂贵” [19]。这些成本影响重大。它们不仅限制了人们会探索哪些原型，也限制了团队会尝试多少次迭代，因此团队承受着越来越大的压力，必须以尽可能低廉且可持续的方式把事情做好。

本文探讨一个与现有研究互为补充的方向。人机交互领域降低原型制作成本的努力通常分为三类：通过抽象设计选择来提高迭代效率的工具包 [3]；降低材料或时间成本的制造技术 [48]；以及在制造设计候选方案之前评估其可行性或性能的计算方法 [7, 12]。总的来说，这些方法旨在降低实现特定原型构想的成本。然而，它们往往没有探讨成本如何影响哪些构想值得进行原型制作。

我们研究并扩展了一种专门针对原型制作中探索–利用问题的成本感知方法。在进行原型制作时，设计师必须不断决定是尝试新想法（探索），还是改进一个已有希望的候选方案（利用）。贝叶斯优化近来作为一种有原则地平衡这些选择的方法，在人机交互研究中日益受到关注 [11, 22, 25, 36]。它具有样本高效的特点，能够考虑不确定性、支持多目标，并允许采用不同的引导技术 [14]。虽然使用优化器显然有助于设计师更好地探索设计空间 [34], , 但现有研究忽略了成本。在每个原型的费用大致相同的情境下，这种简化是可以接受的。然而，在现实项目中，成本各不相同且差异很大，忽视成本的方法可能毫无用处。它忽略了这样的情况：制作一个昂贵的原型可能会比微调现有设计更严重地耗尽预算。我们认为，关键决策——下一步值得尝试什么——应被视为一个资源分配问题。

CABOP是面向交互设备原型设计的贝叶斯优化的成本感知扩展（图1）。它通过最大化单位成本期望改进来选择下一个实验设计，从而在现实预算考量下实现更高效的探索。与现有的贝叶斯优化人机交互应用（例如，[11, 25, 36]）一样，我们的方法假定预先提供了参数化设计空间。我们的主要贡献是针对高效原型制作，对采集函数 进行适配与情境化，如图2所示。采集函数是贝叶斯优化的“核心”：它根据迄今已学到的信息决定下一步尝试哪个样本。在标准贝叶斯优化中，采集函数选择下一个实验设计，方式是最大化期望效用；通常是性能上的期望改进。成本感知贝叶斯优化[32] 将此目标修改为最大化单位成本期望改进：

$$\alpha(\mathbf{x}) = \frac{\mathrm{EI}(\mathbf{x})}{c(\mathbf{x})}, \tag{1}$$

其中，$\mathrm{EI}(\mathbf{x})$ 是实验设计$\mathbf{x}$ 的期望改进，而 $c(\mathbf{x})$ 是估算成本。尽管这种形式在优化文献中已有先例，我们的贡献在于将其调整并应用于人机交互原型设计。

我们通过两项技术贡献，拓展了原型制作中的成本感知。首先，尽管原型制作成本差异很大，设计师通常可以提前估算这些成本。为使采集函数能够利用这一点，我们提出了成本模型。该模型支持灵活地对人机交互项目中的主要原型制作成本进行建模。设计空间中的每个参数都可以分配成本。例如，调整操纵杆增益函数的成本可能相对较低（如 10 美元），而修改操纵杆的按钮或外壳则可能需要进行 3D打印，因此成本高出一个数量级（如 100 美元）。尽管该模型允许任意分配成本，为了指导设计师，我们通过引入成本分类，将成本分为三类：**微调**（低成本调整）、**交换**（重复使用现有子设计）以及**创建**（生成新方案）。其次，我们的模型纳入了已经制作出的原型，也就是我们所说的原型记录。在许多情况下，可以利用过去的设计来降低未来成本，例如重复使用已制作的组件、已实现的软件模块或经过测试的模板。这些成果代表了沉没成本，使后续迭代的成本更低。从采集函数的角度来看，成本并非固定不变，而是取决于原型记录；也就是说，取决于已经实现并制作完成、因而可以重复使用的现有原型集合。尽管该模型未能涵盖可能影响成本的组件间交互，但它提供了一种实用方法，可用于考虑主要成本来源。

我们既通过仿真，也通过一项采用真实硬件与软件成本的用户研究来评估我们的方法。结果表明，与标准贝叶斯优化相比，我们的方法能够在不损害结果质量的情况下大幅降低成本。在预算约束下，它能识别出比标准方法性能更好的解，并能以累积成本的 ≈70% 找到性能相近的解。此外，我们发现我们的方法能够动态适应不断变化的成本景观，并在可能时优先选择更便宜的替代方案。在用户研究中，参与者以原始成本的 ≈67% 实现了性能相当的实验设计。

总之，我们的贡献如下：

（1）将成本感知贝叶斯优化应用于交互设备领域，展示如何在成本各异的设计空间中进行导航。（2）一种适用于人机交互项目的成本模型，在考虑可重复使用的原型的同时，刻画软件与硬件生产之间的差异。（3）仿真和用户研究的结果，系统比较成本感知优化与标准方法之间的权衡。

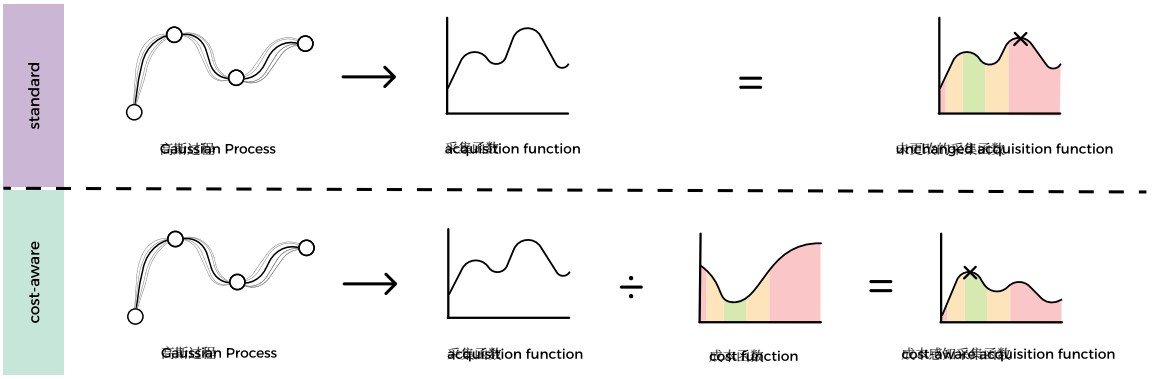


图 2：上：在标准贝叶斯优化中，代理模型（例如高斯过程）会预测效用，而 期望改进 (EI) 则直接定义采集函数。这种方法 不考虑成本；它将所有评估视为成本相同。下：成本感知贝叶斯优化通过将 EI 除以估算成本 $c(\mathbf{x})$，纳入成本模型，从而计算单位成本期望改进 [32]。这使重点从单纯追求样本效率转向追求成本高效，优先进行成本低、改进幅度大的评估。颜色对应于 微调 ($)、交换 ($$) 或 创建 ($$$) 参数。

2 相关工作

在人机交互领域，成本往往是决定某样东西是否会被实际构建的关键因素。它影响着一个原型是否值得制作 [52], 、一项设计在DIY辅助技术等资源受限的环境中是否可行 [18], ，以及普适计算等领域能否实现规模化发展 [19]。除了影响个体决策之外，成本还决定了谁能够切实参与研究与开发，尤其是在资源匮乏的环境中 [9], ，也决定了公共创客空间等场所是否拥有开展教育的机会 [37]。这些成本并非理论上的概念。它们直接决定了什么会被制造出来、谁有机会制造，以及哪些想法会被探索或放弃。

2.1 成本降低

HCI 领域以三种主要角度探讨了原型制作成本这一问题。首先，工具包研究通过抽象化或简化设计决策来提升迭代效率 [3, 15]。工具包通常针对特定的输入或输出模态（例如低成本眼动追踪 [54]），或面向学生等特定受众进行设计 [24]。另一方面，制造方法的进步旨在减少材料用量和生产时间，例如通过更高效的实体构建技术 [1, 48], ，这些技术通常针对创客空间等场景进行定制 [6]。最后，计算工具通过在投入制造之前预测候选设计的可行性或性能来辅助设计师 [7, 12]。总体而言，这些工作主要致力于降低实现特定原型配置的成本。我们的方法与这些工作互为补充，关注的不是降低单个构建的成本，而是提升设计迭代的效率。我们并非降低制作成本，而是在现实预算限制下处理<style id='7'>接下来要制作什么这一决策。

2.2 人机交互中的贝叶斯优化

设计师在迭代原型时，常常需要在探索与利用之间取得平衡。早期草图和低保真模型样件有助于以较低成本进行广泛探索，而需要投入更多工作量的构建则留待用于完善有前景的方向。这种权衡体现了贝叶斯优化的逻辑：它利用概率模型，引导对设计空间进行样本高效的探索 [14]。贝叶斯优化已应用于人机交互领域，用于优化评估成本高、存在噪声或带有主观性的交互系统。该方法使用代理模型估算性能，并通过采集函数选择下一项配置，在不确定性与性能之间取得平衡。

应用包括动画 [5], 可穿戴设备设计 [23],视觉美学 [26, 27], 照片增强 [58], 空中手势 [58], 打字 [35], 和字体排印 [49]。这些系统受益于贝叶斯优化能够以较少的评估次数搜索大型空间。“贝叶斯优化作为助手”进一步探索了这一技术如何支持创意工作流程，同时让设计师保留决策控制权 [25]。

尽管这些示例展示了贝叶斯优化在指导设计迭代方面的有效性，但大多数先前研究并未考虑成本。相比之下，交互系统的原型制作会带来显著的成本差异。有些更改只需快速进行软件编辑，而另一些则需要制造时间、材料资源，或重复使用现有组件。这些不对称且依赖具体情境的成本在 人机交互实践中很常见，但标准贝叶斯优化方法并未对此加以考虑。

贝叶斯优化的相关扩展方法旨在应对复杂的评估空间，但并未直接解决这一挑战。例如，多目标优化 [20] 可以将成本视为一项额外指标，但这种方法假设成本是一个未知目标，必须与性能一同建模。实际上，设计师往往了解成本或能够估算成本；这一点正是我们在本研究中加以利用的。类似地，多保真优化 [21] 利用准确度和开销各不相同的评估，但其重点是学习低保真度近似与真实值之间的偏差。相比之下，此处所述的原型制作成本通常是已知的，且与保真度不同；因此，需要一种基于明确且不对称的资源成本来分配评估的优化策略，而不是对未知目标或保真度相关性进行建模。

2.3 成本感知贝叶斯优化

在人机交互领域之外，研究者已探索成本感知贝叶斯优化，通过将成本直接纳入采集函数 [17, 29, 33]来扩展标准形式。成本感知方法不再选择期望改进最高的配置，而是力求最大化单位成本期望改进，从而更高效地利用有限预算，并优先进行成本低、价值高的评估。这种方法已应用于成本存在差异的多个领域。在机器人学中，它通过结合实体试验与仿真评估 [59], , 减少对高成本实体试验的依赖，并有助于避免不安全的配置 [4]。在化学领域，它能减少试剂消耗和设置时间 [43]。在环境监测中，采集函数经过调整，以尽量缩短路径长度或降低能源使用量 [38]。在机器学习中，它通过考虑训练时间和数据集大小的差异 [45]，改进超参数调优。

然而，这些方法通常假设成本是输入参数的固定函数。在交互式系统设计中，这一假设并不成立，因为成本取决于具体情境，例如当前配置、组件复用情况和原型记录。例如，如果已经制作了合适的组件，修改硬件参数的成本可能很低；但如果必须从头开始，成本就会很高。这些动态成本和原型记录成本是人机交互原型制作工作流的核心因素，却未被传统的成本感知贝叶斯优化技术建模。我们的工作通过引入一种模块化成本模型来填补这一空白，该模型能够捕捉异质且依赖情境的成本。虽然 *EI/C* 公式本身沿用了成熟的成本感知优化方法，但我们针对具体情境的创新之处在于成本项的结构，以及它对原型复用和设计情境的依赖。我们将这一成本模型融入贝叶斯优化过程，以支持在迭代式原型制作场景中进行高效探索；在这些场景中，成本具有可变性、不对称性，并会产生重要影响。

3 背景：成本感知贝叶斯优化

贝叶斯优化是一种广泛用于黑箱函数全局优化的方法 [14]。当目标函数没有闭式表达式且评估成本高昂时，它尤其有价值——这在设计与工程领域十分常见，因为评估往往需要实体制造或用户测试。

标准贝叶斯优化会根据以往的评估结果 $\mathrm{D}_t = \{(\mathbf{x}_i, y_i)\}_{i=1}^{t}$，构建目标函数的概率代理模型（通常为高斯过程）。基于该模型，它会计算一个采集函数 $\alpha(\mathbf{x})$，用于评估候选配置$\mathbf{x}$的效用。随后，优化器选择使该采集函数最大化的候选项：$\mathbf{x}_{t+1} = \arg\max_{\mathbf{x}} \alpha(\mathbf{x})$。一种经典选择是期望改进（*EI*） [14], , 它优先考虑预期表现超过当前观测到的最佳值的点 y^+：

$$\mathrm{EI}(\mathbf{x}) = \mathbb{E}_{p(y|\mathbf{x})}[\max(y - y^+, 0)] \tag{2}$$

这种方法平衡了利用 （在模型预测性能较高的地方进行采样）和探索 （在模型不确定的地方进行采样）。

然而，EI的标准表述隐含地假设每次评估的成本都相同（例如，成本 $c(\mathbf{x}) \approx \text{const}$）。这一成本盲假设在物理原型制作中存在问题，因为微调软件参数与制造新的硬件组件之间的成本差异可能达到几个数量级。忽略这些差异可能导致资源分配效率低下，使优化器将预算浪费在信息增益仅略有提升、但成本高昂的配置上。

为解决这一问题，成本感知贝叶斯优化会调整采集策略，以考虑异质评估成本。尽管存在多种形式，例如加性成本惩罚或多目标优化，一种稳健且有效的方法是优化单位成本期望改进 [32, 45]:

$$\mathbf{x}^* = \arg\max_{\mathbf{x}} \quad \frac{\mathrm{EI}(\mathbf{x})}{\tilde{c}(\mathbf{x})} \tag{3}$$

此处， $\tilde{c}(\mathbf{x})$表示评估$\mathbf{x}$的预估成本。通过用期望改进除以成本，采集函数实际上将搜索转化为投资回报率问题。它会优先选择相对于其花费能带来高信息增益的候选项，鼓励先收集“低成本”信息（例如通过简单的参数微调），再着手进行“高成本”探索（例如完整重建），除非高成本选项有望带来显著更大的性能突破。

在这项工作中，我们采用这种单位成本改进形式。我们的具体贡献不在于优化策略本身，而在于针对交互设备原型制作领域定义成本函数 $\tilde{c}(\mathbf{x})$，相关定义将在以下各节中介绍。

4 理解交互设备迭代式原型制作中的成本

原型制作涉及各种成本。例如，根据项目的不同，软件更改可能成本低廉，而硬件修改则可能需要材料和数小时的制造工作 [19]。这些成本通常产生于系统组件层面：例如支架、滤波器或外壳等独立元件，或软件中的模块或参数。在本文中，“成本”指评估原型配置所需的任何可衡量资源，例如时间、金钱、劳动力或能源。成本还取决于已完成的工作、需要修改的内容，以及先前原型组件的可复用程度。虽然可以使用任何成本单位，但在适当情况下，不同资源可以通过简单的传递函数进行换算（例如将劳动力或时间折算为货币），不过有些项目（如工作量）可能难以准确换算。为了支持更切实高效的设计迭代，优化系统必须将成本视为与既往工作相关、不断变化的结构化约束，而不是固定的标量。

本节将探讨制作交互系统原型所涉及的成本。首先，我们通过一个具体案例说明实际中的这些成本。接着，我们将原型制作成本进行分类，并在下一节将其形式化为用于贝叶斯优化的成本模型。最后，我们扩展该模型，纳入原型记录，以反映重复使用现有原型的机会。

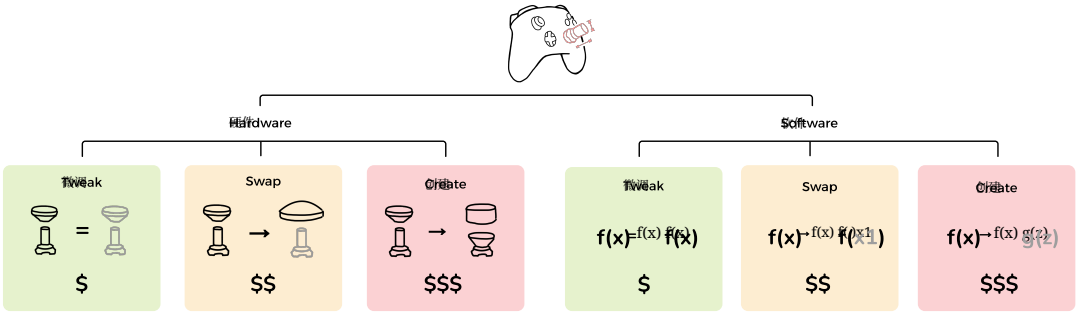


图3：一种考虑已制作原型的迭代式原型制作成本分类。微调组件自上次迭代以来保持不变，几乎不产生成本。交换组件曾用于先前的配置，可以以中等成本重复使用。创建组件此前从未制作过，必须从零开始制造或实现，因而产生最高成本。设计师为成本感知贝叶斯优化填写这些成本类型。

4.1 示例案例：操纵杆原型制作

设想为游戏玩家迭代设计一款 Xbox 风格的操纵杆。设计师确定了三个需要改进的主要组件：杆、顶帽和软件滤波器。每个组件都可以独立配置：例如，杆长范围为 3–20 毫米；顶帽的宽度和曲率各不相同；滤波器则可以在不同的平滑策略之间切换（例如，固定增益、自适应或不使用滤波）。

假设设计师已经制造了直径为 6 毫米和 9 毫米的杆、宽度为 20 毫米且曲率分别为 4.0 毫米和 5.0 毫米的顶帽，并实现了一个固定增益滤波器。如果新配置选用 9 毫米的杆、宽度为 20 毫米且曲率为 5.0 毫米的顶帽，以及同一个滤波器，那么所有组件都与之前制作的版本相同——无需制造新组件。

如果设计师改为选用 6 毫米的杆（之前已制作）、尚未制造的曲率为 22 毫米的顶帽，以及尚未实现的自适应滤波器，那么只有杆可以重复使用。顶帽和滤波器必须从头开始创建、设计和实现，成本会显著增加。这反映了原型制作中的一种常见模式：迭代通过有选择地利用现有成果来推进。

成本取决于设计师的优先事项和具体情况；例如，使用丙烯腈-丁二烯-苯乙烯塑料（ABS）打印一次可能只需 1 美元，但也可能需要等待 3 小时。软件也存在类似的权衡：熟练的开发人员可能只需相对较短的时间就能实现一个滤波器（<1小时），但为过时的控制器重新编译固件可能要花费长得多的时间（>1 周）。

4.2 成本分类

我们提出一种成本分类（图 3），将成本分为三类，涵盖与硬件和软件相关的各个方面。成本可采用任何计量单位，例如金钱、时间或能量。为对此进行形式化，我们在参数组件层面分配成本：参数组件是定义原型功能元素的一组变量，例如杆或滤波器。随后，根据每个组件与此前构建的配置之间的关系对其进行分类：

- 微调 **($)**：该组件与上一次迭代相比没有变化，可以以极低的成本重复使用。
- 交换 **($$)**：该组件与最近一次迭代不同，但曾在更早的迭代中出现过，可以以适中的成本取回或重新组装。
- 创建 **($$$)**：该组件此前从未制造过，因此需要以高成本进行完整制造或实现。

这种分类反映了原型制作的实际过程：设计师会避免完整重建，除非确有必要；尽可能重复使用已验证可行的配置；并倾向于进行与先前原型相符的最小改动。在第 5.1 节中，我们将这一分类形式化为一种可供优化器使用的模型。

4.3 将以往原型纳入考量：原型记录

实验设计成本并非固定不变，而是会随着迭代而变化。随着评估的配置越来越多，可复用组件的数量也会增加，从而降低后续改动的边际成本。为捕捉这一效应，我们引入原型记录，用于追踪哪些参数组件已制造，以及它们最近一次被使用的时间。

将每种配置与先前的迭代进行比较，以确定各组件是未改变、重复使用还是新增。这为优化器引入了记忆机制，使其能够利用以往原型的结构。系统不会将所有变更一视同仁，而是可以优先考虑那些单位成本效用高的变更。

我们的方法捕捉了真实世界原型制作的关键特性：模块化设计、选择性复用和渐进式改进。该方法不受单位成本定义的限制，因此可以灵活地集成到现有设计工具和优化流程中。

4.4 更多示例

我们展示另外三个案例，以说明如何应用成本分类；详情请参阅附录 A。

Dexmo [16] 是一种用于虚拟现实中手指追踪和力反馈的机械外骨骼。其组件结构——手指连杆机构、力反馈单元（力反馈单元）和控制器——自然对应我们的成本分类：滤波等软件更新属于低成本（微调），连杆组件通常可以重复使用（交换），而力反馈单元的改动则需要重新校准和重新制造（创建）。相关成本包括制造时间和材料。

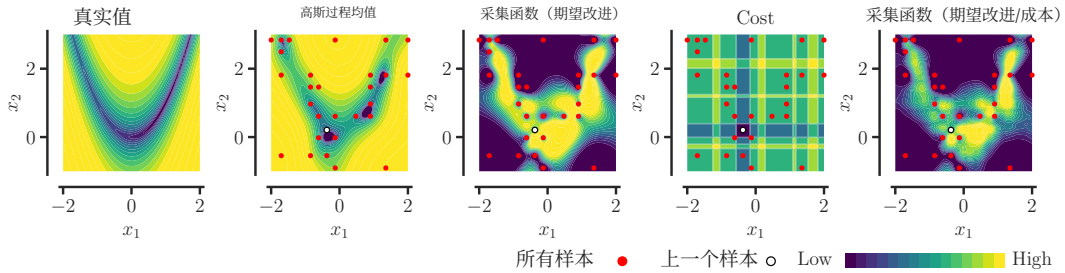


图 4：单个时间步中迭代优化过程的快照。我们的方法持续循环执行采样、测试和更新模型；此处，我们根据先前样本（红点）的历史记录，将用于选择下一个样本的组件可视化。 x_1 和 x_2 表示原型参数。左：Rosenbrock真实值。左中：根据当前样本对真实值进行近似的高斯过程（GP）代理模型。中：由 GP 得出的期望改进（EI）。右中： x_1 和 x_2的成本景观；先前创建的配置形成低成本的垂直带和水平带（使搜索偏向于 微调 和 交换 操作）。右：最终的成本感知采集函数，该函数使用成本模型对 EI 进行缩放。公差σ_g 已设置为较高值，以提高可见性。

Omni [30, 60] 是一种集成传感与驱动的体积式触觉触控笔。核心电磁铁的制造和仿真成本高昂（创建），而触控笔改造和传感器重新配置的成本较低（交换），滤波器调节则微不足道（微调）。除时间和材料外，人工劳动也可能是一项显著成本。

Back-Hand-Pose [55] 使用腕戴式摄像头和深度学习来估计手部姿态。关键瓶颈是重新训练：摄像头位置或数据表示的变化往往会令模型失效（创建），而对预处理的微调操作（交换）或学习率（微调）则允许部分复用。该场景特有的成本包括数据采集和GPU训练时间。

5 方法

我们的方法将面向交互设备的成本感知优化付诸实施。尽管优化框架（式3）是标准的，我们的主要贡献在于成本模型——即对 $\tilde{c}(\mathbf{x})$ 的定义。首先，将原型参数分组为组件（例如，杆或顶帽等硬件元件，或滤波器等软件功能）。其次，为每个组件分配一个复用类别，以反映其与先前工作的关系：未改变（微调）、复用自先前配置（交换），或新创建（创建）。最后，将这些类别映射为任意单位的数值成本（例如时间、金钱、能量），用以量化每个组件变更所需的工作量。

我们将这些信息整合到贝叶斯优化所用的采集函数中。我们不再仅根据期望改进来选择下一个配置，而是选择单位成本带来最高改进的配置（公式 3）。这样可以鼓励优化器优先考虑成本低但信息丰富的改动，只有在预计能带来显著改进时才提出成本高昂的改动。由此得到的方案可即插即用，替代标准贝叶斯优化，从而提高现实世界中的原型制作任务的效率。

5.1 成本建模

在原型制作中，成本不仅取决于参数值，还取决于这些参数如何对应功能组件（第 4 节）。有些组件必须制造或实现，而另一些则可以从之前的迭代中重复使用。我们的成本模型通过同时纳入组件结构和原型记录，明确体现了这一区别。

设$\mathbf{x}$ 表示一个完整的原型，例如具有特定物理尺寸和软件设置的操纵杆。我们将原型划分为若干组件，即形式化定义的不相交组， $G = \{g_1, g_2, \ldots\ldots, g_k\}$，其中每个组件 c 都对应一个有意义的硬件或软件模块。每个组件包含一个或多个相关参数，例如所有与实验设计、传感器有关的参数，或一组与输入滤波相关的软件参数。在每次迭代 t中，我们会跟踪两项内容：当前原型$\mathbf{x}_t$，以及原型记录 $\mathrm{H}_t = \{G_0, \ldots\ldots, G_{t-1}\}$，其中记录了迄今为止已制造或实现的所有 k组件。通过这份记录，我们可以识别提议的原型是在重复使用先前的原型，还是引入了全新的内容。我们通过检查每个组件 $g \in G$的状态，来定义评估提议配置$\mathbf{x}$ 的成本 $c(\mathbf{x})$：

$$c_g(\mathbf{x}_g) = \begin{cases} c^g_{\text{tweak}} & \mathbf{x}_g = \mathbf{x}_{g,t} \quad \text{(reuse previous prototype)} \\ c^g_{\text{swap}} & \mathbf{x}_g \in \mathcal{H}_t, \mathbf{x}_g \neq \mathbf{x}_{g,t} \quad \text{(reuse from record)} \\ c^g_{\text{create}} & \mathbf{x}_g \notin \mathcal{H}_t \quad \text{(create new prototype)} \end{cases} \tag{4}$$

评估$\mathbf{x}$ 的总成本是各组件成本之和：

$$c(\mathbf{x}) = \sum_{g \in G} c_g(\mathbf{x}_g). \tag{5}$$

直观来说，这意味着原型的总成本是其所有单个组件成本之和，无论这些组件是硬件还是软件，都取决于它们是已配置、需要交换还是需要创建。这种表述支持任意组件。例如，操纵杆可能由底座、手柄和盖帽等独立硬件组件，以及用于滤波或控制策略的软件组件构成。在组件级对成本进行建模，能够实现复用感知优化，使其符合真实原型制作工作流中的成本分布方式。

成本的平滑松弛。为了选择下一个配置，贝叶斯优化必须优化采集函数。这种优化通常使用基于梯度的方法进行，而这要求所有组件——包括成本函数——都可微。然而，我们原始的成本模型是离散的：每个组件都被标记为微调、交换或创建，并对应相应的标量成本。这使其无法与基于梯度的优化兼容。

为此，我们引入了一种对离散代价模型进行平滑松弛的方法。其核心思想是根据当前配置与先前使用过的配置之间的相似程度，为每个组件分配一个软性成本。这样，成本便可以作为连续且可微的函数来处理。

首先，我们定义参数值之间的相似度函数。对于每个组件 $g \in G$（例如握持形状、传感器布局），我们使用径向基函数（RBF）核来量化当前$\mathbf{x}_g$设计与先前$\mathbf{x}'_g$设计之间的相似度：

$$K(\mathbf{x}_g, \mathbf{x}'_g) = \exp\left(-\frac{\|\mathbf{x}_g - \mathbf{x}'_g\|^2}{2\sigma_g^2}\right) \tag{6}$$

这里，σ_g是一个可调参数，用于控制灵敏度：较小的σ_g值要求两者高度相似时必须非常接近，而较大的值则允许更广泛的泛化。我们使用 RBF 核，是因为它能提供一种平滑、可调且随距离增加而衰减的相似度度量，从而能够控制对重复使用的限制有多严格。其带宽决定了只有近乎相同的配置才会被视为可重复使用（较小的 σ），还是允许更广泛的泛化（较大的 σ）。此外，RBF 核具有无穷阶可微性，因此非常适用于基于梯度的优化。

我们将提出的配置 $\mathbf{x}_g$ 与两个参照进行比较：1）当前配置 $\mathbf{x}_{g,t}$，用于确定组件是否保持不变（微调）；2）原型记录 $\mathcal{H}_g = \{\mathbf{x}_{g,0}, \ldots, \mathbf{x}_{g,t-1}\}$，其中存储了组件 g 的所有先前配置（从而支持交换）。

我们使用径向基函数核计算软相似度权重：

$$w_{\text{tweak}} = K(\mathbf{x}_g, \mathbf{x}_{g,t}) \tag{7}$$

$$w_{\text{swap}} = \sum_{\mathbf{x}'_g \in \mathcal{H}_g} K(\mathbf{x}_g, \mathbf{x}'_g) \tag{8}$$

对于 创建，w_{create}，我们将其视为可调参数，因为无法根据相似度计算得出。这样可以确保无论核相似度如何，模型始终保留非零的基线概率来提出新配置。将 w_{create} 设为较高值，会通过优先选择制造成本较高的未见原型来鼓励探索。较低值则会让优化器更加保守，侧重于重复使用和小幅调整。实际上，此参数控制着优化器偏离现有配置的幅度。

我们通过使用径向基函数核定义基于相似度的权重，为每个组件计算软成本。这些权重经过归一化后，形成关于重复使用类型的概率分布，并据此对组件成本进行插值：

$$\tilde{c}_g(\mathbf{x}_g) = \frac{w_{\text{tweak}} \cdot c^g_{\text{tweak}} + w_{\text{swap}} \cdot c^g_{\text{swap}} + w_{\text{create}} \cdot c^g_{\text{create}}}{w_{\text{tweak}} + w_{\text{swap}} + w_{\text{create}}} \tag{9}$$

总成本是所有组件成本之和：

$$\tilde{c}(\mathbf{x}) = \sum_{g \in G} \tilde{c}_g(\mathbf{x}_g) \tag{10}$$

我们的公式为离散代价模型提供了平滑且可微的松弛，同时保留了其核心直觉：与现有或当前实验设计越接近的配置，成本越低。可调权重和带宽参数 σ_g 可用于控制对重复使用的灵敏度。窄核（较小的 σ_g）要求几乎完全重复使用，而宽核（较大的 σ_g）则允许更灵活地衡量相似性。

这种松弛使成本感知采集函数能够兼容基于梯度的优化。它可以高效地集成到现代贝叶斯优化流程中，同时仍遵循原始基于复用的模型的结构和意图。

总之，所提出的成本分类处于离散空间中（包括微调、交换或创建）。为了优化采集函数，我们需要确保梯度平滑。为此，我们将这个离散空间转换为连续空间。具体而言，我们基于所提议原型与原型记录之间的相似度，使用径向基函数核近似每种离散代价。因此，这种相似度加权是对所提出成本结构的松弛版本。

更新原型记录。评估新的配置$\mathbf{x}_t$后，我们更新原型记录 $\mathrm{H}t$，以纳入新创建的任何组件。形式上，对于每个组件 $g \in G$，如果$\mathbf{x}_{g,\ t} \notin \mathrm{H}_g$，则将其追加到记录中：

$$\mathcal{H}_g \leftarrow \mathcal{H}_g \cup \mathbf{x}_{g,t} \quad \text{if } \mathbf{x}_{g,t} \notin \mathcal{H}_g. \tag{11}$$

此更新确保未来的成本计算能够识别可复用组件的可用性。因此，原型记录会随着优化过程不断演进，使系统能够避免重复制造，并优先选择利用既有工作成果的配置。

算法 1采用平滑松弛的成本感知优化

要求： 带有分组的参数空间 $\mathrm{X}\,G$，高斯过程先验 GP_0，初始原型记录 H_0，成本级别 $\{c^g_{\text{微调}}, c^g_{\text{swap}}, c^g_{\text{创建}}\}$，每个组 g

$t \leftarrow 0$, $\mathrm{D}_0 \leftarrow \varnothing$, $\mathrm{H}_t \leftarrow \mathrm{H}_0$, 高斯过程$_t \leftarrow$ 高斯过程$_0$

当未收敛**时**

  定义采集函数：

  $a_{\text{cost}}(\mathbf{x})$ = 期望改进$_{\mathcal{GP}_t}(\mathbf{x})$ / $\tilde{c}(\mathbf{x},\ \mathrm{H}_t)$

  优化采集：

  $\mathbf{x}^* \leftarrow$ 参数最大化$_{\mathbf{x}\in\mathcal{X}}\ a_{\text{cost}}(\mathbf{x})$

  执行并评估：

  实现配置 $\tilde{\mathbf{x}}$（例如，调整至公差范围内）

  计算离散代价 $c^* \leftarrow c(\tilde{\mathbf{x}},\ \mathrm{H}_t)$

  观察效用 $y^* = f(\tilde{\mathbf{x}})$

  $\mathrm{D}_{t+1} \leftarrow \mathrm{D}_t \cup \{(\tilde{\mathbf{x}}\ y^*)\}$

  $\mathrm{H}_{t+1} \leftarrow \mathrm{H}_t \cup$ 新组件$(\tilde{\mathbf{x}},\ \mathrm{H}_t)$

  高斯过程$_{t+1} \leftarrow$ 更新高斯过程(高斯过程$_t$, D_{t+1})

  $t \leftarrow t + 1$

结束 while

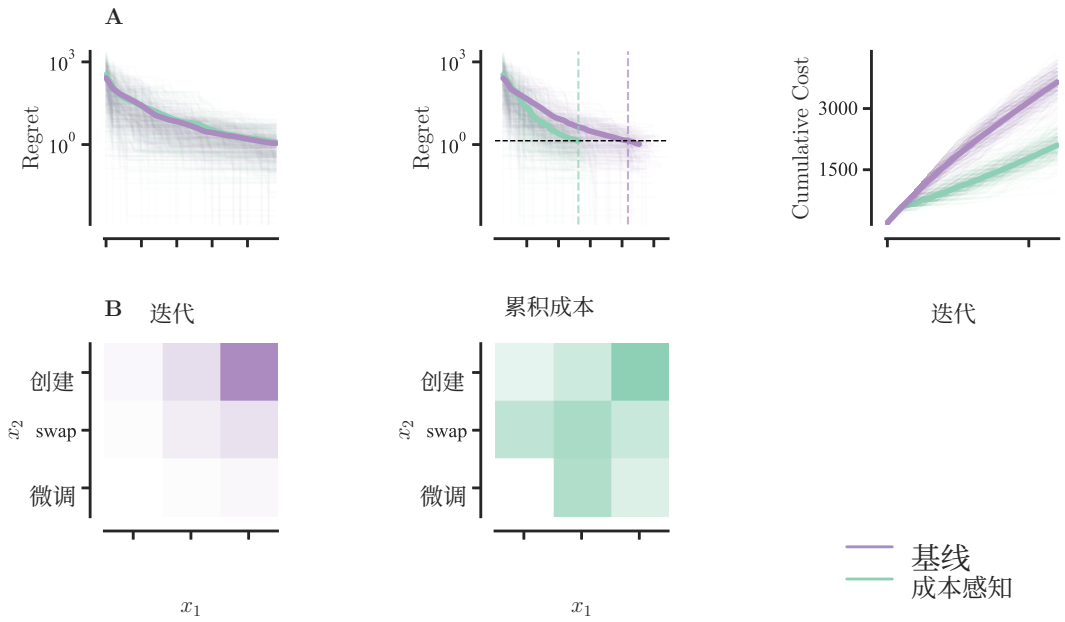


图 5：研究1。固定迭代次数下的成本感知与基线贝叶斯优化。A：性能指标。左：基线收敛到略低的遗憾值。中：成本感知以 ~2/3 的成本实现了相近的效用。竖线标示成本感知方法的最低成本。右：基线累积成本的速度显著更快。（均值不包括 < 95% 试验）。B：每种硬件（x_1）和软件（x_2）操作的样本数量。基线对成本高昂的创建编辑操作过度采样，而成本感知方法则在各种成本结构之间均匀分配样本。

5.2 算法详解

该算法遵循标准贝叶斯优化循环，但会修改采集函数以考虑成本。在每次迭代中（算法 1），系统使用上文所述的平滑松弛方法，选择相对于估计成本而言能带来最佳期望改进的配置。这鼓励重复使用和小幅调整，除非预测某个新配置能够带来显著改进。选定配置后，系统会将其映射到最接近的可制造原型（例如，以满足制造约束），对其进行评估，并将其添加到用于更新高斯过程的数据集中。原型记录也会随之更新，以反映新创建的组件，并影响未来的成本估算。随着时间推移，这一过程有助于更高效地分配原型制作工作量，优先考虑成本低但信息丰富的改动。

6 技术评估

为了评估我们的成本感知优化方法能否在反映现实世界原型制作的情境中提高效率，我们开展了五项仿真研究。这些实验分离并考察了迭代设计中的关键挑战：成本累积、预算约束、非对称成本结构、参数复杂度以及动态资源可用性。总的来说，这些测试评估了在现实条件下对成本进行显式建模能否改善优化效果。

6.1 基线与实验设置

我们将我们的方法与人机交互应用中常见的一种标准贝叶斯优化方法进行比较 [34]。基线通过期望改进来选择候选项，而不考虑成本。我们的方法则修改采集函数，转而优先考虑单位成本期望改进。

为实现系统化基准测试，我们使用二维 Rosenbrock 函数作为真实效用曲面（见图4）：

$$f(x_h, x_s) = (1 - x_1)^2 + 100(x_1 - x_2^2)^2 \tag{12}$$

其中，x_1 和 x_2 是不同的原型参数，例如硬件和软件。Rosenbrock 函数因其狭窄的弯曲谷地而常用于优化 [44]。它也可作为人机交互原型设计的替代模型，因为 x_1 和 x_2 之间的相互依赖关系反映了软硬件耦合。我们在附录C中评估了另外三个真实函数（Ackley、Goldstein-Price 和 Levy）。

为了模拟人在回路评估噪声，我们在函数输出中同时加入加性高斯噪声和乘性高斯噪声：

$$y = f(x) * \epsilon_m + \epsilon_a, \quad \epsilon_a \sim \mathcal{N}(0, 0.1), \quad \epsilon_m \sim \mathcal{N}(1, 0.1) \tag{13}$$

加性噪声用于建模固定的评估变异（例如手部抖动），而乘性噪声则反映测量不确定性如何随输出幅度变化而缩放。

成本在组件和操作之间具有结构化分布。每个参数都属于某个硬件或软件组件。根据操作类型，变更会产生相应成本：微调 = 1、交换 = 10、创建 = 100，适用于这两类组件。

优化运行通过 Sobol 采样抽取三个随机配置作为开始。两种方法使用相同的高斯过程模型、采集函数优化器（L-BFGS-B）和停止准则。所有参数在建模时都归一化为 $[0, 1]$，执行时再进行反归一化。

性能通过遗憾值来衡量——即全局最优值与已观测到的最佳值之间的差异——以此反映

收敛速度和解质量。我们报告随迭代变化的遗憾值，以评估样本效率；并报告随累积成本变化的遗憾值，以评估实际效率。

实现。 我们的实现采用 Python 3.12 编写，使用 Scikit-learn [40] 进行高斯过程建模，并使用 SciPy [51] 进行数值优化。所有实验均在 Apple Macbook M3 上进行，并实时运行。为确保鲁棒性，所有实验均包含以随机种子初始化的多次试验。源代码可在 https://github.com/aalto-ui/CABOP 获取。

6.2 研究1：无限预算下的性能

设计师通常会不断迭代，直到设计达到足够好的程度 [56]。在这种情况下，成本仍然重要：它决定了获得最终设计所需的整个流程成本。本研究旨在检验，即使不设定显式预算，我们的成本感知方法能否以较低的累积成本取得相近的结果。

设置。 我们在二维 Rosenbrock 函数上进行了 25 次迭代，每种方法进行了 250 次独立试验。两种方法均从三个随机 Sobol 样本开始。在每一步中，我们记录了后悔值和操作成本类型（微调、交换、创建）。迭代次数固定，但累积成本不固定，因此我们可以比较样本效率（步数）和成本效率（所用资源）。由于分布不服从正态，统计分析采用 Mann–Whitney U 检验。

结果。结果如图 5 所示。在 25 次迭代中，两种方法都降低了遗憾值，基线的最终遗憾值（$M = 1.01, SD = 1.18$）略低于成本感知方法（$M = 1.14, SD = 1.17$）。然而，这一差异在统计上并不显著（$p = 0.12$）。我们的方法以较低的成本（$M = 1529, SD = 542$）达到最佳遗憾值，而基线则需要更高的成本（$M = 2785, SD = 890$），$p < .01$。我们的方法最终成本也明显低于基线（成本： $M = 2101.39, SD = 277.91$ $M = 3642.76$ $SD = 290.41$ ；基线: , ），且这一差异具有统计显著性（$U = 29.50, p < .01$）。采样行为存在显著差异：基线过度使用了成本高昂的创建操作，而我们的方法更倾向于成本较低的编辑操作。卡方检验证实了这一模式（$\chi^2(5, N = 26000) = 2956.87, p < .01$），标准化残差较大，表明基线明显过度选择高成本操作，而我们的方法则使用成本较低的

微调并创建，而不会降低结果质量。这些结果表明，我们的方法利用成本更低的选项来降低累积成本，同时不影响质量。

6.3 研究2：预算约束下的性能

大多数真实世界原型制作都受资源限制 [50]。设计师很少能够自由地进行优化；相反，当资金、时间或材料耗尽时，他们就会停止。本研究评估了当可用总预算受到上限限制时，我们的方法能否取得更好的结果。

设置。 我们模拟了成本预算固定在 600 至 7000 个单位之间的优化过程。每种方法都用尽全部预算，并在预算耗尽后终止。对于每个预算水平，我们记录了最终遗憾值。每个水平下，每种方法均进行了 25 次优化运行，

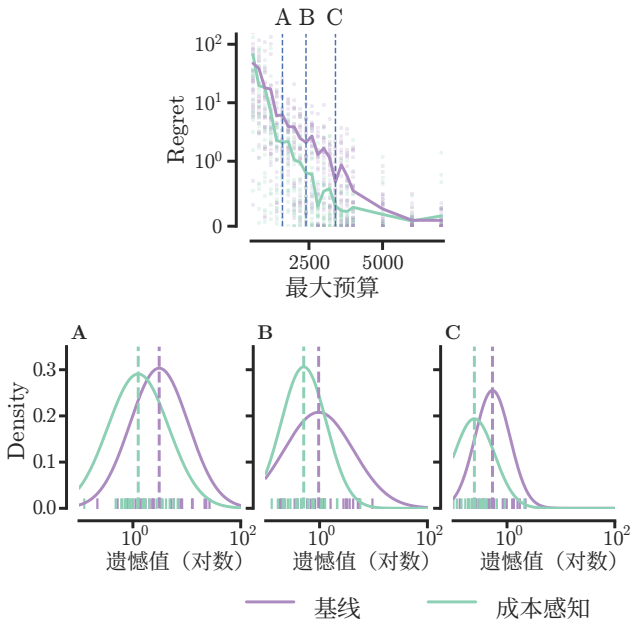


图6：研究2。Tp[]：固定成本预算从600到7000个单位时的最终遗憾值和遗憾值分布。其他：特定预算下的横截面，对应于左图中的垂直线。成本感知方法在中间区间显著优于基线；在该区间内，总预算并非微不足道，但也没有大到可以忽略不计。

与研究1一样，由于结果分布不服从正态分布，我们使用 Mann–Whitney U 检验来评估统计显著性。

结果。结果如图6所示。在大多数预算水平下，成本感知方法始终取得更低的最终遗憾值，其中最显著的提升出现在中等预算水平。当预算为1600时，我们的方法（$M = 2.11$， $SD = 1.83$）的遗憾值显著低于基线（$M = 6.21$， $SD = 7.48$），$p = .027$。在 2400, $p <$ 处，这 .01 3400, 一差距进一步扩大（$M = 0.80$，而基线为2．70）。即使在两种方法都接近收敛时，成本感知优化仍表现更好（$p = .02$）。这体现了一种平衡：如果预算几乎不存在，所有资源都必须花费在原型创建上，因为在早期阶段没有可供依赖的原型记录；而如果预算足够充足，两种方法都能够找到最优设计。我们的方法在中等预算范围内优势最大，在有限资源下，高效复用与选择性创建能够带来更好的结果。

6.4 研究3：对成本不对称性的敏感性

原型制作通常涉及成本差异很大的组件[8]。具体而言，与软件编辑相比，硬件变更通常需要更长的制造时间和更多的材料用量。本研究评估我们的方法是否能适应此类成本不对称性，而标准方法则对所有评估一视同仁。

设置。 我们分别改变了硬件和软件 创建 成本，取值为 {100, 400, 1600, 6400}，由此得到 16 种成本条件。对于每种

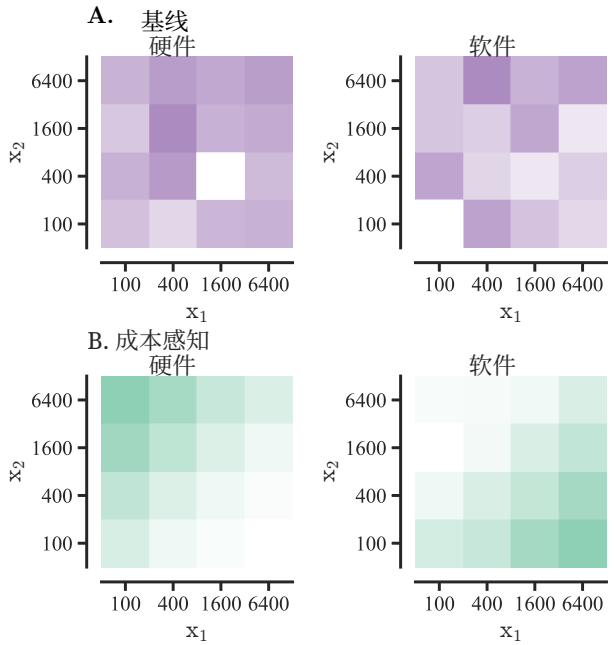


图7：研究3。按方法、组件类型和成本不对称性划分的创建样本数量，其他成本保持不变。每张热图显示硬件或软件样本数量随硬件和软件创建成本变化的情况。我们的方法会根据成本不对称性调整采样行为，减少对昂贵组件的选择；而基线则对每个组件进行均匀采样。为便于阅读，每张热图均单独归一化。

条件，我们在二维 Rosenbrock 函数上进行了 50 次优化试验，每次迭代 25 轮。我们记录了按组件和操作类型（微调、交换、创建）统计的采样数量。我们使用广义线性混合效应模型，评估这些方法是否会根据成本不对称性调整采样行为。

结果。我们的方法会根据成本变化调整采样策略（见图7）：随着硬件成本上升，执行的硬件$p < .01$操作显著减少，低成本软件编辑则有所增加（交互效应）。基线没有表现出这种适应性——无论成本如何变化，硬件和软件的采样率都几乎保持不变。事后对比证实，基线对硬件创建的采样过多（估计值为 = 0，对数尺度为0.62，$p < .01$），而对软件编辑的采样不足。相比之下，硬件成本较高时，我们的方法会将工作量转向成本较低的软件参数。这表明，只有成本感知方法会根据不对称性，在不同模态之间动态调整工作量，从而更好地适应现实世界约束。

6.5 研究4：随原型复杂度扩展

随着原型变得日益复杂，设计师面临的参数和组件数量也不断增加 [47]。有些系统存在密集耦合，修改任何参数都需要完整重建；另一些系统则被拆分为不同的组件，因此可以进行局部编辑操作。本研究评估我们的方法能否有效应对复杂度增加，并利用组件降低成本。

设置。 我们使用多维 Rosenbrock 函数评估了两种复杂度形式。首先，我们改变参数数量（{1, 2, 4, 8, 16}）；这些参数都位于单一组件中，因此任何改动都更有可能触发完整的创建 操作。其次，我们将硬件参数总数固定为 8，并改变独立制造的组件数量（{1, 2, 4, 8}），每次改动只影响相应的组件。对于这两种情况，我们进行了 50 次试验，每次包含 25 次迭代，并使用广义线性混合效应模型分析采样行为。

结果。 结果如图 8 所示。当单个组件中的参数数量增加时，两种方法都更加依赖成本高昂的 创建 操作。不过，在低到中等维度下，我们的方法更有效地限制了这一增长。例如，当参数数量为 1–2 时，我们的方法中 创建 操作显著少于基线（$p < .05$），但这一差异在更高维度下有所减小（$p > .5$）。这表明，两种方法都会受到参数耦合加剧的影响。相较之下，随着模块化程度提高，我们的方法持续减少了创建 采样，并越来越多地依赖低成本编辑操作。这一优势从 组件数量 = 2 开始显现（$p < .05$），并随着模块化程度进一步提高而扩大；在 创建 操作的平均次数上，最终差异超过 10 次，出现在 组件数量 = 8时。基线受到的影响较小。这些结果表明，我们的方法能够利用结构所带来的重复使用机会，而基线仅略微调整了其对系统结构的采样。

6.6 研究5：重新加权成本

真实世界原型制作是动态的：制造资源、材料获取或团队专业知识可能在过程中途发生变化 [8]。这些波动会改变设计操作的有效成本。本研究检验我们的方法在单次优化周期内成本发生变化时，是否会调整其采样行为。

设置。 我们在三次随机初始化下，对两种条件进行了24次迭代评估。在恒定条件下，成本权重保持固定。在动态条件下，硬件创建成本在第10次迭代时增加十倍（例如，工具不可用），随后在第17次迭代时降至十分之一（例如，发现更快的制造）。我们比较创建操作在三个成本阶段中的频率：第4–10次、第11–17次和第18–24次迭代。每种条件重复进行了50次优化试验。广义线性混合效应模型评估了条件、阶段与操作类型之间的交互作用，重点关注创建采样。

结果。 我们的方法能够很好地适应成本变化。在动态条件下，创建 采样在成本上升后显著减少（$p < .01$），并在成本再次下降时大幅增加（$p < .01$）。相比之下，恒定条件下没有明显变化。在不同条件下，动态运行在高成本阶段采样的创建 操作更少（$p = .01$），而在成本下降时采样更多（$p < .01$）。尽管效用曲面完全相同，仍出现了这些变化，这表明我们的方法追踪的是成本结构，而不仅仅是结果。这些结果表明，我们的方法会根据变化中的约束动态重新分配采样力度。

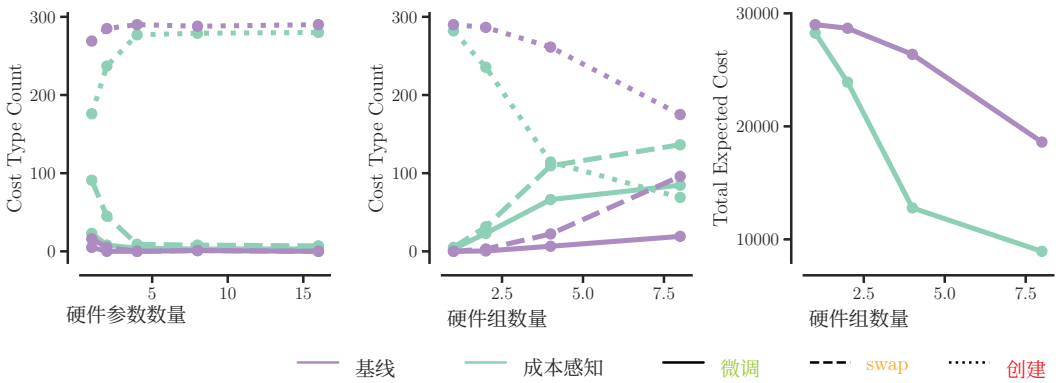


图8：研究4。硬件参数被微调、交换或创建的迭代次数与维度的关系。参数数量会对两种方法都产生负面影响；然而，随着组件数量增加，成本感知方法相较于基线获得的收益显著更大。本研究使用了 n维Rosenbrock函数。

6.7 研究6：认知成本估计不准确的影响

在最后一项实验中，我们考验该方法应对不准确的成本规格的鲁棒性——这在现实中很常见，设计师的估计值与实际资源消耗往往存在偏差。这带来了一个棘手的优化问题：该方法必须借助一张“错误的地图”——成本模型——在效用景观中探索，使优化器的内部逻辑与终止回合的严格外部预算约束相脱节。

设置。我们在认知成本（成本感知贝叶斯优化用于规划的成本）与真实成本（从有限预算中扣除的成本）之间引入差异。我们将认知成本定义为真实成本乘以一个系数 α，该系数在 10^{-1}（严重低估）到 10^{1}（严重高估）之间变化，采用对数尺度。优化器假定认知成本准确无误，并且在运行期间不会更新该成本。我们评估这种不匹配对最终遗憾值、迭代次数以及采样行为变化的影响（微调、交换和创建）。我们进行了50次随机试验。

结果。 图 9 显示，整体模式是一致的：提高对原型成本的认知成本会减少其使用，而降低认知成本则会增加采样。根据类别不同，这会影响迭代次数和最终遗憾值。

不同类别受到的影响各异。假设 微调 成本高昂时，优化器会转而选择交换；假设其成本较低时，影响很小，因为没有成本更低的替代选项。对交换 成本的估计错误则影响更大。低估会增加交换 的使用，取代创建 并增加迭代次数；高估则会使优化器转向创建，减少迭代次数并增加遗憾值。对于创建，影响是直接的：低估会导致过度使用并加快预算消耗，而高估则会将采样转向微调 和交换，从而增加迭代次数。

总体而言，该方法以可预测的方式调整采样，同时保持性能。其影响取决于算子及误差方向，这表明维持成比例的成本关系比精确数值更重要。

研究	标准成本感知	
S1：成本高效	X	✓
S2：在成本预算约束下有效	X	✓
S3:适应成本不对称性	X	✓
S4:利用模块化结构	~	✓
S5:响应周期中途改变成本	NA	✓
S6:对成本不确定性的灵敏度	NA	✓

表1：基线与我们的成本感知方法在六项研究中的比较。成本感知方法始终优于基线。 ~：标准方法对模块化结构产生了影响，但这是组件数量增加所固有的，并非该方法所致。

6.8 总结

结果汇总见表1。总体而言，我们发现，对成本结构进行建模有助于实现更明智的采样和更高效的资源利用。在研究1中，成本感知方法的性能与基线相当，但累积成本显著更低。研究2表明，在固定预算下，成本感知优化通过更高效地分配工作量，取得了更低的遗憾值。研究3证实，我们的方法能够适应成本不对称性，将工作量转向成本较低的编辑操作；这与设计师优先进行低工作量修改的做法相呼应。在研究4中，只有成本感知方法利用了模块化，通过减少昂贵的重建来降低成本，展现出基线所不具备的结构敏感性。研究5展示了动态适应能力：当周期中途成本发生变化时，我们的方法会相应调整采样，在整个过程中保持效率。最后，研究6表明，即使成本设定有误，该方法依然稳健；它会以合理的方向调整采样，即便估计值不准确，也能维持整体性能。

评估稳健性。 虽然我们使用了Rosenbrock函数，但并未声称它能全面体现人机交互设计空间的结构。不过，为了测试鲁棒性，我们在仿真中使用了随机种子和噪声。结果表明，性能提升对噪声或初始条件并不敏感，观察到的行为也不是特定随机运行造成的假象。使用其他真实值的补充结果见附录C。

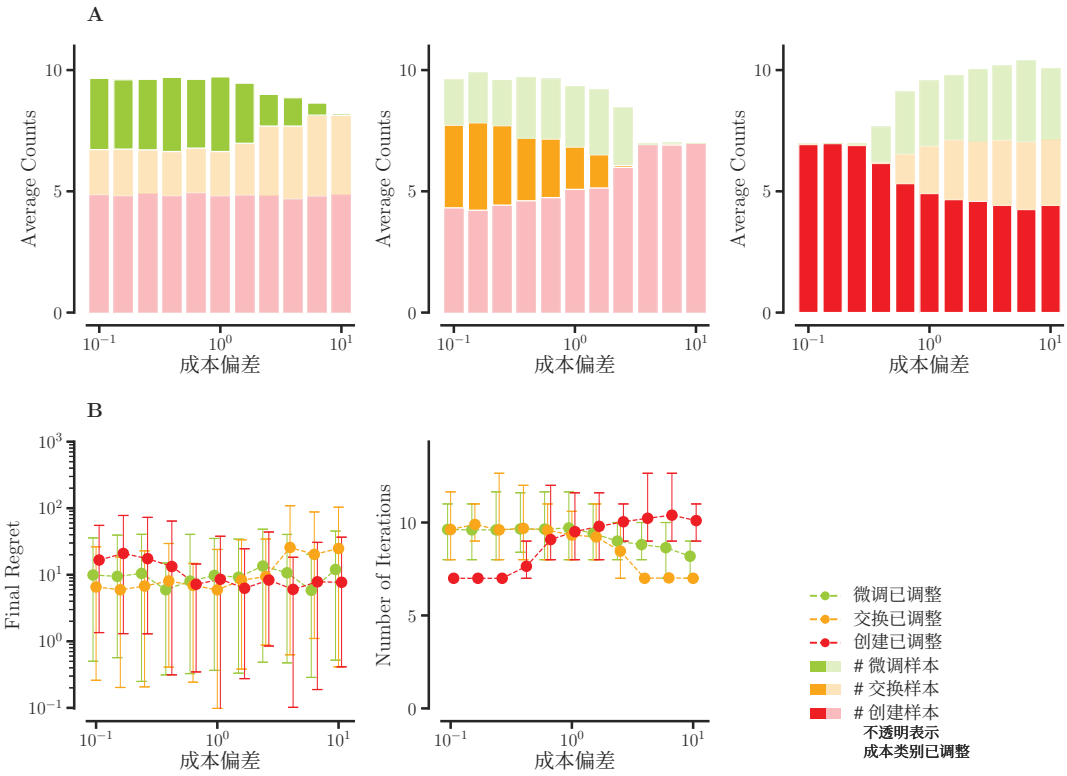


图9：研究6：不准确的成本估算对优化行为的影响。认知成本按某个系数缩放，相对于真实成本 α，范围从 10^{-1} 到 10^{1}。A：在不同成本类别存在偏差（不透明）时，微调、交换和创建操作的采样行为。B：图表展示错误设定的成本如何影响最终遗憾值（左）和迭代次数（中），总体体现了该方法的鲁棒性以及成本带来的复杂相互作用。误差线表示95%的置信区间。不透明的柱形表示经过调整的成本类别；浅色柱形表示未改变的类别。B中的每条线都对应A中的同一项成本调整。

7 用户研究

我们开展了一项被试内用户研究，评估与标准贝叶斯优化基线相比，成本感知的人在回路优化能否提高定制软硬件操纵杆配置的效率。我们选择这一测试平台，是因为它贴近现实，结合了耗时且成本高昂的实际硬件修改，以及快速且成本低廉的软件更改，从而形成了符合实际情况的评估工作量不对称性。

优化器结合了任务表现与参与者的舒适度评分。在开始优化之前，每位参与者都指定了性能与偏好之间的权重，并在整个环节中使用该权重计算效用。通过这一指标，我们得以评估该方法能否在不降低结果质量的情况下减少原型制作工作量。

参与者。 我们通过本地广告招募了12名参与者（5名女性，7名男性；年龄：$M = 23.3, SD = 3.3$）。所有参与者均有正常或矫正至正常的视力，并表示每天都会使用电脑。我们筛选了参与者的既往游戏经验，以确保他们具备足够的手眼协调能力。利手不作为筛选因素。所有参与者均完成了研究。未排除任何数据。每位参与者均获得一张可在当地餐厅使用的20欧元代金券作为报酬。

任务与材料。 操纵杆设计基于 Microsoft Xbox [57] 的 Adaptive Thumbstick Toppers。这些是可3D打印的标准控制器摇杆改装件。参与者可以定制硬件参数（杆长、摇杆帽凸度、摇杆帽宽度）和软件参数（灵敏度和反应性），汇总见表2。为了让任务能在实验时间内完成，我们预先制作了大量硬件参数组合（超过600种杆与顶帽的组合）。

灵敏度和反应性的成本结构均设为微调：1，交换：10。由于所有软件配置都已预先实现，可供运行时选择，创建成本不适用。两个软件参数均被视为独立组件。对于硬件，杆的成本设为微调：1，交换：10，创建：100；顶帽的成本则设为微调：1，交换：10，创建：1000。创建新顶帽的成本较高，反映了其更高的物理复杂度和更长的3D打印时间。这些数值旨在近似表示相对工作量，而非精确耗时——例如，软件更改通常只需浏览菜单；而硬件制造虽然人工劳动较少，却会因打印时间而造成长时间延误。我们的目标是对现实世界中设计迭代的不对称性和实际阻力进行建模，而不强行将其映射为绝对时长。

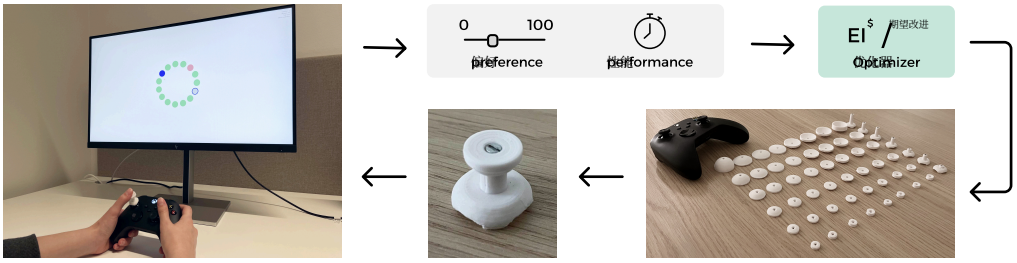


图 10：用户研究概览。参与者执行一项菲茨定律任务：使用操纵杆将光标移向高亮目标（蓝色实心圆），并通过 A键或其中一个扳机确认选择。成功选中后，目标会在环的另一侧重新出现，任务随即重复。每次试验结束后，参与者会对测试配置进行评分，分数范围为 0 到 100。优化器综合主观偏好与客观性能，提出新的配置。随后将建议的配置交换到器件中，并重复此过程，直至达到迭代次数。

参数	描述	范围
软件		
灵敏度	输入到屏幕位置的增益	0–1
反应性	低通滤波器（前一位置的权重）	0–1
硬件		
杆：长度	底座到顶帽的距离	3–21 毫米
顶帽：凸性	曲率（– 为凸，+为凹）	±0.66
顶帽：宽度	顶帽的直径	10–30 毫米

表 2：控制输入参数及其描述和范围。两个参数（摇杆帽凸度和宽度）属于同一个组件。硬件组件均为预制件，因为研究期间时间有限，无法自行制作。

该任务是往复式圆形菲茨定律任务，包含15个目标，分为两个难度指数（宽度：60px、距离：150px；宽度：80px、距离：250px）。这些参数经实证选定，旨在使任务具有挑战性，同时确保每个目标都能在2.5秒内完成。每次迭代中，如果15个目标里有超过2个未命中，就必须重新进行该次迭代。任务如图10所示。优化器的效用 y 是一个加权的偏好与性能评分，其中性能指每个目标的任务完成时间，且仅将命中的目标计入考量。

实验流程。 每次实验开始时，参与者会收到标准化说明，并完成训练阶段：进行四轮迭代，每轮包含 15 个目标，且每轮使用不同的参数。正式研究以迭代方式进行：每一轮中，优化器会提出新的配置，随后重新配置操纵杆并更新软件参数。接着，参与者完成四轮迭代，每轮包含 15 个符合费茨定律的目标。之后，他们使用 0–100 的滑块对舒适度进行评分。优化器根据预先指定的效用权重更新其代理模型，以选择下一种配置。每次实验持续 ≈75 分钟。操纵杆的重新配置由实验人员执行，所用时间计入成本模型。所有实验均在受控的实验室环境中进行。

实验设计。 本研究采用被试内实验设计，并通过拉丁方进行平衡。因变量为最佳配置的客观表现以及参与者的主观偏好。两种方法均使用期望改进作为采集函数，并采用相同的高斯过程先验和 3 个随机初始化点。优化进行了 10 次迭代。参与者在参与研究前均已提供知情同意。根据当地法规，本研究无需进行伦理审批。数据已匿名化。

7.1 结果

我们的研究结果见图 11。为量化效率，我们分析了各方法达到最小遗憾值时的累积成本。我们采用配对样本 t 检验，比较成本方法与标准方法达到最大效用所需的迭代次数。差异具有统计显著性，$t(10) = 2.64$, $p = .03$，且效应量为中到大（$d = 0.80$，95% CI $[0.10, 1.47]$）。平均而言，成本方法以较低的成本达到最大效用（$M = 3422.64$，$SD = 1820.44$），而标准方法所需成本较高（$M = 5040.46$，$SD = 2034.20$）。我们的方法最终成本低于基线（成本：$M = 5086.64$，$SD = 977.33$；标准：$M = 7364.36$，$SD = 864.23$），曼–惠特尼检验也支持这一差异（$U = 6.00$, $p < .01$）。尽管噪声水平较高，且受样本量和贝叶斯优化的随机性影响，我们仍可得出结论：我们的方法能更早达到局部最优结果。

我们进行了配对样本t检验，以比较各方法在试验中取得的最佳结果。标准方法（$M = 0.37$，$SD = 0.16$）与成本方法（$M = 0.36$，$SD = 0.13$）之间没有显著差异，$t(10) = 0.30$, $p = .77$。这表明两种方法都能找到同样好的配置。

综合来看，这些结果表明，我们的成本感知方法仅需基线方法 67% 的原始成本，便能取得与之相当的成果。具体而言，如果设计师每小时收入为 20 美元，我们的方法约需 140 美元，而基线方法需 200 美元。这样的节省在多次迭代、多个项目或多个团队中会不断累积。

采样类型。 优化后的操纵杆的定性示例表明，对同一参与者而言，两种方法通常会得出相似的设计，而成本感知方法往往能以更低的成本实现这一点（图 12）。与此同时，不同参与者之间也存在一些差异，说明该方法能够支持定制。

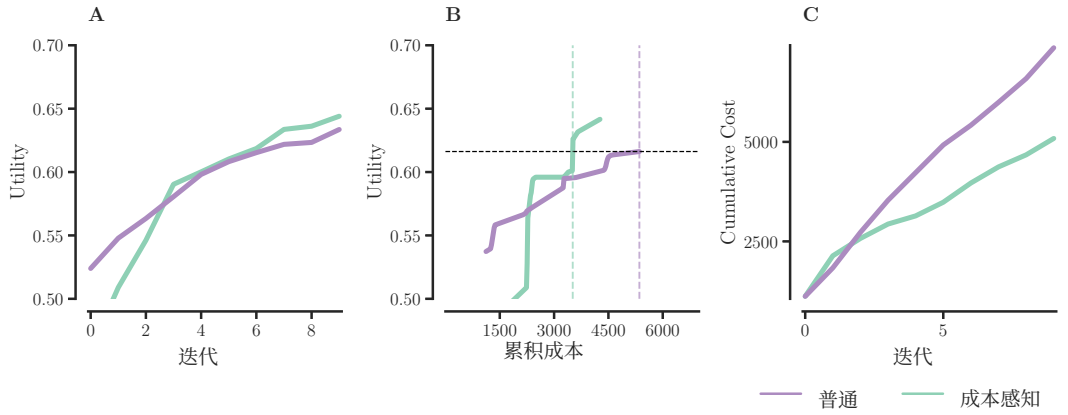


图 11：用户研究结果。A：各迭代的平均效用。B：效用与累积成本的关系。我们的方法仅以基线 ≈67% 的成本就达到了相近的质量结果。参与者各自不同的累积成本，在大多数参与者完成任务后产生了显著的视觉噪声。由于观测数量减少，此图已截断。C：与基线相比，我们的成本感知方法使累积成本的增幅更低。

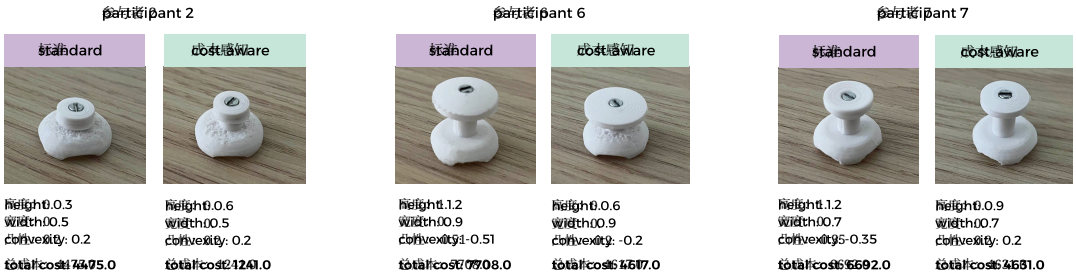


图 12：三位参与者在两种方法中选出的最佳操纵杆示例。从同一参与者的结果来看，两种方法得到的设计相似，而成本感知方法实现这些设计所需的成本更低。相比之下，不同参与者之间的设计存在差异，这表明我们的方法支持成本感知定制。总成本指整个过程累积的成本

8 总结与讨论

我们为贝叶斯优化引入了一种成本感知扩展，将原型制作所需的工作量纳入考量。在仿真和人在回路研究中，该方法的表现始终优于成本无关基线。在仿真中，该方法以低 30–40% 的成本找到了遗憾值相当的设计，能够适应设计空间中的结构性不对称性（如模块化、维度），并在减少支出的同时保持每次迭代的改进率（第 6 节）。在一项有 12 名参与者的用户研究中，参与者对软硬件操纵杆配置进行定制，该方法以原始成本的 ≈67% 实现了相当的任务表现（第 7 节）。这些发现表明，将成本纳入优化循环能够提高效率而不牺牲质量。

我们方法的核心洞见是，人机交互原型设计中的成本并非固定不变，而是取决于实验设计历史。重要的不仅是某个配置的参数值，还包括已经构建了什么、哪些内容可以重复使用，以及参数如何映射到实体或软件组件。我们的成本模型通过在组件级应用三种复用类别，并利用记录过往构建及其时间新近程度的原型记录，明确呈现了这种依赖关系。这一表述使成本感知优化在工作量差异显著的实验设计场景中既可行又有意义。重要的是，这一方法不会改变效用的定义或测量方式，只会改变对工作量的核算方式。

要在原型制作工作流中应用成本模型，研究人员需要定义组件、分配复用类别，并将其整合到优化循环中。该流程包含五个步骤：**(1)** 将实验设计参数划分为组件，以反映系统的构建和修改方式（例如外壳、传感器或控制滤波器）；**(2)** 为每个组件分配一个复用类别——微调 表示与上一次迭代相比未发生变化，交换表示重复使用早期原型中的组件，或创建 表示新制造或实现；**(3)** 将每个类别映射为一个标量成本，可以使用粗略取值（例如 1/10/100），也可以根据工作量估算；一致性比精确度更重要；**(4)** 在优化过程中计算配置成本，并将其纳入采集函数，引导优化器进行成本效益高、影响显著的编辑操作；以及**(5)** 随着迭代推进更新成本结构，使模型能够适应制造方法的演变或成本优先级的变化。这一工作流轻量且灵活，能够明确体现复用并考虑成本。

最终，这项工作使优化能够在以成本为核心关切的原型设计场景中得到更有效的运用。我们的

方法将关注重点从样本效率转向成本效率，使优化器的探索—利用平衡与设计师在决定制作何种原型时所面临的考量相一致。

8.1 局限性与未来工作

我们当前的模型假设各组件彼此独立地变化。实际上，许多物理和软件编辑操作彼此依赖：更换传感器可能需要重新设计外壳、调整电源布线并更新软件校准 [42]。采用更结构化的成本模型（例如树表示）或许能更好地捕捉这些影响。

成本也会随保真度而变化。设计师经常会先使用低保真打印件、模型样件或仿真，再决定是否制作高保真版本 [46]。我们的方法假设评估保真度一致，但保真度本身也是一项设计决策。将该方法扩展为多保真贝叶斯优化 [10]，或许有助于权衡可信度与成本，在支持更广泛的早期探索的同时，推迟成本更高的评估。

我们的方法完全在采集函数层面运行，因此能够广泛兼容不同形式的反馈。只要代理模型能够纳入这些反馈，采集函数就可以使用客观评分、主观评分、偏好或拒绝信息。这使该方法适用于人在回路工作流：设计师可以引导流程、排除不可行样本或调整优先级 [58]。

采集函数所做的不只是选择样本——它们还定义了优化的目的。我们的方法假设目标是通过最大化期望改进，快速最大化效用。但在早期原型制作中，目标可能是理解，而非性能。以信息增益或不确定性降低为目标的采集策略 [39] 或许能更好地支持这些阶段。未来的工具可能会随着时间动态调整采集行为，根据可用预算、设计师的输入或不断变化的目标，在探索与利用之间进行平衡 [13]。

进一步的方向是将成本模型扩展到单一指标之外。有些成本无法映射到共同的尺度，或许更适合被作为独立的预算维度处理。多预算形式将使设计师能够平衡异质资源，并沿帕累托前沿选择配置。将此类模型整合到贝叶斯优化中，并理解设计师如何处理这些权衡，是未来值得探索的方向。

对于贝叶斯优化在实验设计中的所有应用而言，也许最关键的问题在于，它要求预先定义参数化设计空间。推进能够在优化过程中推断、扩展或调整设计空间的方法，将使这些方法更适用于实际的原型制作工作流。

9 结论

成本是原型制作的核心：它决定了要构建什么、团队迭代的频率，以及哪些方向切实可行。我们调整了成本感知优化方法，以支持这些决策。该方法基于两个理念：将变更划分为结构化的成本类型（微调、交换和创建），并维护一份原型记录，追踪哪些组件已经构建完成。两者共同构成了一种反映真实原型制作工作量的成本模型。通过将该模型纳入采集函数，我们的方法根据单位成本期望改进选择配置，从而鼓励重复使用并避免重建。该方法可融入现有的贝叶斯优化框架，无需更改代理模型。仿真和一项用户研究表明，该方法在不降低性能的情况下减少了成本。更广泛地说，我们的工作将人机交互中的优化重新构想为一个资源分配问题：不仅要找到性能出色的实验设计，还要在实际预算约束下决定接下来值得构建什么。

致谢

本研究由欧洲研究委员会（101141916）和芬兰研究委员会（328400、345604、341763 和 357578）资助。代码可在此免费获取：https://github.com/aalto-ui/ CABOP。

参考文献

[1] Aline Abler、Juan Zárate、Thomas Langerak、Velko Vechev 和 Otmar Hilliges。2021。Hedgehog：基于中央电磁致动器的手持式球形针阵列。载于*2021 IEEE*世界触觉会议（*WHC*）。133–138。doi:10.1109/WHC49131.2021.9517197[2] David Ackley。2012。用于遗传爬山算法的连接主义机器。第28卷。Springer science & business media。[3] Jason Alexander、Anne Roudaut、Jürgen Steimle、Kasper Hornbæk、Miguel Bruns Alonso、Sean Follmer 和 Timothy Merritt。2018。形状可变界面研究的重大挑战。载于*2018*年*CHI*人机交互系统中的人为因素会议论文集。1–14。[4] Felix Berkenkamp、Andreas Krause 和 Angela P Schoellig。2023。带安全约束的贝叶斯优化：机器人学中的安全自动参数调优。机器学习 112，10（2023），3713–3747。[5] Eric Brochu、Tyson Brochu 和 Nando De Freitas。2010。用于程序化动画设计的贝叶斯交互式优化方法。载于*2010*年*ACM SIGGRAPH/Eurographics*动画计算研讨会论文集。103–112。[6] Bradley Camburn 和 Kristin Wood。2018。创客与DIY制造原则：以低成本实现设计原型。设计研究 58（2018），63–88。[7] Liwei Chan、Yi-Chi Liao、George B Mo、John J Dudley、Chun-Lien Cheng、Per Ola Kristensson 和 Antti Oulasvirta。2022。研究人在回路优化在设计交互技术时的正面与负面特质。载于*2022*年*CHI*人机交互系统中的人为因素会议论文集。1–14。[8] Edward James Christie、Daniel D Jensen、Richard T Buckley、Devin A Menefee、Kyle Kenneth Ziegler、Kristin L Wood 和 Richard H Crawford。2012。原型制作策略：文献综述与关键变量识别。载于*2012*年*ASEE*年度会议暨展览会。25–1091。[9] Nicola Dell 和 Neha Kumar。2016。面向发展的HCI的内外。载于*2016*年*CHI*人机交互系统中的人为因素会议论文集。2220–2232。[10] Bach Do 和 Ruda Zhang。2023。工程设计中的多保真贝叶斯优化。*arXiv*预印本 *arXiv:2311.13050*（2023）。[11] John J Dudley、Jason T Jacques 和 Per Ola Kristensson。2019。利用贝叶斯优化众包界面特征设计。载于*2019*年*CHI*人机交互系统中的人为因素会议论文集。1–12。[12] Anna Feit、Mathieu Nancel、Maximilian John、Andreas Karrenbauer、Daryl Weir 和 Antti Oulasvirta。2021。AZERTY ameliore：国家规模的计算设计。*ACM*通讯 （2021）。https://cacm.acm.org/magazines/2021/2/ 250082-azerty-amlior/fulltext[13] Dominik Fisch、Dominik Fisch、Martin Jänicke、Edgar Kalkowski 和 Bernhard Sick。2012。动态变化环境中的知识获取技术。*ACM*自主与自适应系统汇刊（*TAAS*）7，1 2012 1 25( ), – 。[14] Peter I Frazier。2018。贝叶斯优化。载于当代问题优化与建模的最新进展。Informs，255–278。[15] Saul Greenberg。2007。工具包与界面创造力。多媒体工具与应用 32，2（2007），139–159。[16] Xiaochi Gu、Yifei Zhang、Weize Sun、Yuanzhe Bian、Dao Zhou 和 Per Ola Kristensson。2016。Dexmo：一种用于虚拟现实中的动作捕捉和力反馈的低成本轻型机械外骨骼。载于*2016*年*CHI*人机交互系统中的人为因素会议论文集。1991–1995。[17] Matthew Hoffman、Bobak Shahriari 和 Nando de Freitas。2014。基于模型的多臂老虎机优化中的相关性与预算约束及其在自动机器学习中的应用。载于人工智能与统计学。365–374。

[18] Meeri Kim、Erin Bradley、Samantha Chen-Hall、Rachel Hurst 和 Jennifer Mankoff。2016。临床与创客视角下，利用快速原型技术设计辅助技术。载于第 18 届 ACM SIGACCESS 计算机与无障碍国际会议论文集。251–256。[19] Lars Erik Holmquist。2023。比特很便宜，原子很昂贵：批判人机交互转向实体性的趋势。载于 2023 年 CHI 人机交互系统中的人为因素会议扩展摘要。1–8。[20] Pascal Jansen、Mark Colley、Svenja Krauß、Daniel Hirschle 和 Enrico Rukzio。2025。OptiCarVis：利用贝叶斯优化改进自动驾驶车辆功能可视化，以提升用户体验。载于 2025 年 CHI 人机交互系统中的人为因素会议论文集（CHI '25）。计算机协会，纽约，美国，论文 85，23 页。doi:10.1145/3706598.3713514[21] Kirthevasan Kandasamy、Gautam Dasarathy、Jeff Schneider 和 Barnabás Póczos。2017。利用连续近似进行多保真贝叶斯优化。载于国际机器学习会议。PMLR，1799–1808。[22] Mohammad M Khajah、Brett D Roads、Robert V Lindsey、Yun-En Liu 和 Michael C Mozer。2016。利用贝叶斯优化设计引人入胜的游戏。载于 2016 年 CHI 人机交互系统中的人为因素会议论文集。5571–5582。[23] Myunghee Kim、Ye Ding、Philippe Malcolm、Jozefien Speeckaert、Christoper J Siviy、Conor J Walsh 和 Scott Kuindersma。2017。人在回路中的可穿戴设备参数贝叶斯优化。PLOS ONE 12，9（2017），e0184054。[24] Scott R Klemmer 和 James A Landay。2009。支持整合物理与数字交互的工具包。人机交互 24，3（2009），315–366。[25] Yuki Koyama 和 Masataka Goto。2022。作为助手的贝叶斯优化：利用贝叶斯优化异步生成设计建议。载于第 35 届 ACM 用户界面软件与技术年度研讨会论文集。1–14。[26] Yuki Koyama、Issei Sato 和 Masataka Goto。2020。用于交互式视觉设计优化的序列图库。ACM 图形学汇刊（TOG）39，4（2020），88–1。[27] Yuki Koyama、Issei Sato、Daisuke Sakamoto 和 Takeo Igarashi。2017。通过群体进行高效视觉设计优化的序列线搜索。ACM 图形学汇刊（TOG）36，4（2017），1–11。[28] Manuel Laguna 和 Rafael Marti。2005。对用于多模态函数全局优化的高级散射搜索设计进行实验测试。《全球优化杂志》33，2（2005），235–255。[29] Raymond Lam、Karen Willcox 和 David H Wolpert。2016。有限预算下的贝叶斯优化：一种近似动态规划方法。载于《神经信息处理系统进展》。883–891。[30] Thomas Langerak、Juan José Zárate、David Lindlbauer、Christian Holz 和 Otmar Hilliges。2020。Omni：用于动态触觉反馈的被动磁性工具体积式感知与驱动。载于第 33 届 ACM 用户界面软件与技术年度研讨会论文集。594–606。[31] Thomas Langerak、Juan José Zárate、Velko Vechev、David Lindlbauer、Daniele Panozzo 和 Otmar Hilliges。2020。电磁触觉引导系统的最优控制。载于第 33 届 ACM 用户界面软件与技术年度研讨会论文集。951–965。[32] Eric Hans Lee、Valerio Perrone、Cedric Archambeau 和 Matthias Seeger。2020。成本感知贝叶斯优化。arXiv 预印本 arXiv:2003.10870（2020）。[33] Yingzhen Li 和 Yao Yao。2019。带方向约束搜索的贝叶斯优化。arXiv 预印本 arXiv:1906.09459（2019）。[34] Yi-Chi Liao。2023。《人在回路中的设计优化》。博士论文。阿尔托大学。[35] Yi-Chi Liao、João Belo、Hee-Seung Moon、Jürgen Steimle 和 Anna Maria Feit。2026。通过从用户模型中学习的先验实现高效的人在回路优化。载于 2026 年 CHI 人机交互系统中的人为因素会议论文集（西班牙巴塞罗那）（CHI '26）。ACM，西班牙巴塞罗那，24 页。doi:10. 1145/3772318.3791976[36] Yi-Chi Liao、John J. Dudley、George B. Mo、Chun-Lien Cheng、Liwei Chan、Antti Oulasvirta 和 Per Ola Kristensson。2023。多目标贝叶斯优化与交互设计。《IEEE 普适计算》22，1（2023），29–38。doi:10.1109/MPRV.2022.3230597[37] Silvia Lindtner、Garnet D Hertz 和 Paul Dourish。2014。人机交互创新的新兴场所：创客空间、硬件初创企业与孵化器。载于 SIGCHI 人机交互系统中的人为因素会议论文集。439–448。[38] Roman Marchant 和 Fabio Ramos。2012。面向智能环境监测的贝叶斯优化。载于 2012 年 IEEE/RSJ 智能机器人与系统国际会议。IEEE，2242–2249。[39] Umberto Noè 和 Dirk Husmeier。2018。关于一种新型基于改进的贝叶斯优化采集函数。arXiv 预印本 arXiv:1808.06918（2018）。[40] Fabian Pedregosa、Gaël Varoquaux、Alexandre Gramfort、Vincent Michel、Bertrand Thirion、Olivier Grisel、Mathieu Blondel、Peter Prettenhofer、Ron Weiss、Vincent Dubourg 等。2011。Scikit-learn：Python 机器学习工具。《机器学习研究杂志》12（2011），2825–2830。[41] Victor Picheny、Tobias Wagner 和 David Ginsbourger。2013。含噪优化中基于克里金法的填充准则基准研究。《结构与多学科优化》48卷3期（2013），607–626。[42] Michael Riesener、Eric Rebentisch、Christian Dölle、Sebastian Schloesser、Maximilian Kuhn、Jan Radermacher 和 Günther Schuh。2019。复杂技术系统敏捷开发中面向依赖关系的原型制作模型。《Procedia CIRP》84（2019），1023–1028。[43] Alexandre A Schoepfer、Jan Weinreich、Ruben Laplaza、Jerome Waser 和 Clemence Corminboeuf。2024。成本引导的贝叶斯反应优化。《Digital Discovery》3卷11期（2024），2289–2297。[44] Lawrence F Shampine。1982。Rosenbrock 方法的实现。《ACM 数学软件汇刊》（TOMS）8卷2期（1982），93–113。[45] Jasper Snoek、Hugo Larochelle 和 Ryan P Adams。2012。机器学习算法的实用贝叶斯优化。《神经信息处理系统进展》25（2012）。[46] Jialin Song、Yuxin Chen 和 Yisong Yue。2019。基于高斯过程的多保真度贝叶斯优化通用框架。载于《第22届国际人工智能与统计学会议》。PMLR，3158–3167。[47] Gerald Staib、Andreas Dörrhöfer 和 Markus Rosenthal。2013。组件与系统：模块化构造——设计、结构与新技术。Walter de Gruyter。[48] Ye Tao、Guanyun Wang、Caowei Zhang、Nannan Lu、Xiaolian Zhang、Cheng Yao 和 Fangtian Ying。2017。Weavemesh：一种用于柔性组装创建的三维模型的低保真、低成本原型制作方法。载于《2017年 CHI 人机交互系统中的人为因素会议论文集》。509–518。[49] Yuki Tatsukawa、I-Chao Shen、Mustafa Doga Dogan、Anran Qi、Yuki Koyama、Ariel Shamir 和 Takeo Igarashi。2025。FontCraft：使用交互式贝叶斯优化进行多模态字体设计。载于《ACM 人机交互系统中的人为因素会议论文集》（CHI）。[50] Edward Tiong、Olivia Seow、Bradley Camburn、Kenneth Teo、Arlindo Silva、Kristin L Wood、Daniel D Jensen 和 Maria C Yang。2019。设计原型制作的经济性与维度：价值、时间、成本与保真度。《机械设计杂志》141卷3期（2019）。[51] Pauli Virtanen、Ralf Gommers、Travis E Oliphant、Matt Haberland、Tyler Reddy、David Cournapeau、Evgeni Burovski、Pearu Peterson、Warren Weckesser、Jonathan Bright 等。2020。SciPy 1.0：Python 科学计算的基础算法。《Nature Methods》17卷3期（2020），261–272。[52] Robert A Virzi。1989。从低保真原型中能学到什么？载于《人因学会年会论文集》第33卷。SAGE 出版社，SAGE CA：洛杉矶，加利福尼亚州，224–228。[53] Mark Weiser。1998。校园普适计算的未来。《Commun. ACM》41卷1期（1998），41–42。[54] David Winfield、Dongheng Li、Jason Babcock 和 Derrick J Parkhurst。2005。面向人机交互应用的稳健、低成本眼动追踪开放硬件与开放软件工具包。《爱荷华州立大学人机交互技术报告 ISU-HCI》（2005）。[55] Erwin Wu、Ye Yuan、Hui-Shyong Yeo、Aaron Quigley、Hideki Koike 和 Kris M Kitani。2020。Back-Hand-Pose：通过手背形变网络，利用腕戴式相机进行三维手部姿态估计。载于《第33届 ACM 用户界面软件与技术研讨会论文集》。1147–1160。[56] David C Wynn 和 Claudia M Eckert。2017。关于设计与开发中迭代的观点。《工程设计研究》28卷2期（2017），153–184。[57] Xbox Design Lab。2025。适用于 Xbox 控制器的 Adaptive Thumbstick Toppers。https://xbox.design/en-us/accessories/adaptive-thumbstick-toppers-for-xbox-controllers。[访问日期：10-09-2025]。[58] Kenta Yamamoto、Yuki Koyama 和 Yoichi Ochiai。2022。摄影师在环贝叶斯优化的摄影布光设计。载于《第35届 ACM 用户界面软件与技术研讨会论文集》。1–11。[59] Lizhi Yang、Zhongyu Li、Jun Zeng 和 Koushil Sreenath。2022。贝叶斯优化与混合零动态相结合：双足运动控制的安全参数学习。载于《2022年国际机器人与自动化会议》（ICRA）。IEEE，10456–10462。[60] Juan Jose Zarate、Thomas Langerak、Bernhard Thomaszewski 和 Otmar Hilliges。2020。借助球形电磁铁实现无接触非平面触觉。载于《2020年 IEEE 触觉研讨会》（HAPTICS）。IEEE，698–704。

A 应用示例

示例 *1*：*Dexmo*——用于虚拟现实中动作捕捉和力反馈的机械外骨骼。

目的。 Dexmo 是一种轻量级机械外骨骼，专为虚拟现实中的手指级运动追踪和被动力反馈而设计 [16]。它采用伺服驱动关节锁定来模拟刚性接触，无需使用大型致动器或系绳。

组件。 Dexmo 可分解为三个主要组件：(1) 手指连杆机构，(2) 力反馈单元（FFU），以及 (3) 控制器和通信模块。每个组件都有可调参数。连杆机构可调整杆长和关节位置。FFU 包含伺服类型和锁定几何结构。控制器则允许更改采样率、滤波策略和分辨率。

成本结构。 重复使用很简单。之前打印的连杆机构部件通常可以交换到新配置中 (交换)，而滤波等软件更改通常只需微调 (微调)。更换或重新设计 FFU 需要高精度操作和伺服校准，因此属于成本高昂的 创建 操作。这种结构使优化器能够综合考虑硬件和软件组件的成本。如何获取成本?

优化目标。 系统可以针对用户感知的真实感或定量追踪精度进行优化。我们的成本模型允许系统探索新力反馈单元带来的性能提升是否值得其制造成本，同时鼓励重复使用无需改动的组件。

示例 *2*：*Omni* — 一种电磁触觉反馈设备。

用途。 Omni 是一种体积式触觉器件，通过触控笔和定制的电磁致动器 [30, 60] 将输入与输出相结合。嵌入触控笔中的永磁体与球形电磁铁相互作用，同时，霍尔传感器阵列追踪触控笔的位置。

组件。 该器件可分解为三个组件：触控笔、电磁铁和传感器阵列。电磁铁的参数包括磁芯尺寸、绕组匝数和线径，通常需要进行有限元仿真。通过改变磁体长度和半径，可以调整触控笔。传感器阵列的数量、布局和滤波设置也各不相同。

成本结构。 电磁铁更改成本高（创建），涉及建模、绕制和高昂的材料成本。触控笔和传感器配置通常可以重复使用（交换），而一些软件层面的改动（例如滤波器长度）则较为简单（微调）。这些组件级区别使优化器能够识别成本高昂的改动，并尽可能优先重复使用。

优化目标。 Omni 可以通过用户对触觉保真度的评分，或通过测量力和追踪准确度来进行优化。成本模型可以通过尽量减少不必要的硬件变更并优先安排影响重大的调优步骤，帮助管理工作量。

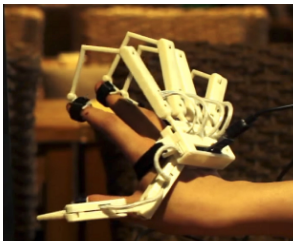
示例 *3*：*Back-Hand-Pose*——使用机器学习进行手部姿态估计的腕戴式摄像头。

用途。 Back-Hand-Pose 是一个实时系统，通过佩戴在手腕上的 RGB摄像头拍摄手背，实现 3D 手部姿态估计 [55]。它采用基于长短期记忆网络的双流网络（Dorsal-Net），根据皮肤形变推断手部关节位置，无需依赖外部摄像头或标记点。

组件。 该系统由三个组件构成：佩戴在手腕上的摄像头、预处理流程（例如掩膜处理和运动历史）以及神经网络模型。参数包括摄像头视场、安装位置、图像变换和模型架构。

成本结构。 该系统的主要成本在于重新训练。摄像头安装位置或数据表示的变化可能导致先前的训练数据失效，进而需要投入大量成本进行数据收集和训练周期（创建）。预处理逻辑的变化可能使数据得以重复使用，并只需进行部分重新训练（交换）。调整学习率等超参数通常只需投入较少的人力，但可能会产生运行时间成本（微调）。这种结构凸显了软件原型设计与硬件原型制作一样，都能受益于成本敏感型视角。

优化目标。 系统可以针对姿态准确度、鲁棒性或泛化能力进行优化。成本感知优化器可以在重新训练成本与模型性能之间取得平衡，从而有选择地进行改进，而无需重新启动整个流程。



(a) Dexmo [16]



(b) Omni [30, 60]



(c) Back-Hand-Pose [55]

图 13：将我们的成本模型回顾性地应用于人机交互文献中的三个系统。每个系统都具有独特的组件结构和成本不对称性，展现了该模型在不同领域中的灵活性。

B 用户研究

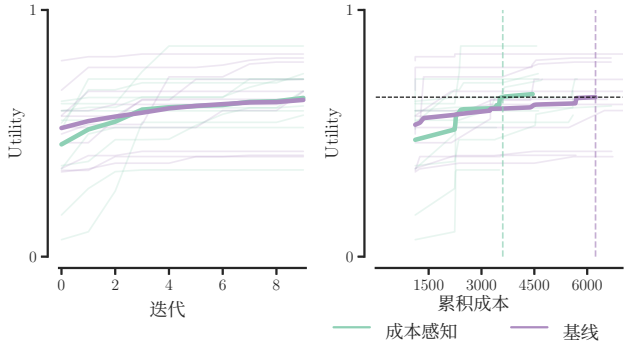


图 14：用户研究的结果，两张图均在背景中显示了各个用户。左：效用随迭代次数的变化。右：效用随累积成本的变化。我们的方法以基线 ~ 70% 的成本，达到相近的质量。

C 附加真实函数

在论文正文（第 6 节）中，我们将方法与人机交互领域常用的贝叶斯优化基线进行了比较 [34], 该基线通过期望改进来选择候选项，而不考虑成本。相比之下，我们的方法优化的是按成本归一化后的期望改进。为了进行系统评估，我们使用二维 Rosenbrock 函数作为真实效用曲面。

为评估我们的方法并消除真实函数的影响，我们报告了三个附加函数上的结果，除了 Rosenbrock 外，还包括：Ackley [2],Goldstein-Price [41],和 Levy [28]。这些函数表现出不同的特性，表明我们的方法能够推广到单个测试函数之外。

Ackley. 阿克利函数常用于测试优化方法，因为它在原点的全局极小值周围有许多局部极小值：

$$f(x_1,x_2) = -20\exp\left(-0.2\sqrt{0.5(x_1^2+x_2^2)}\right) - \exp\left(0.5(\cos(2\pi x_1) + \cos(2\pi x_2))\right) + 20 + e \tag{14}$$

在人机交互原型设计中，它反映了这样一类实验设计问题：许多次优解看起来都很有希望（局部最优解），但只有经过仔细探索，才能找到真正的全局解。

Goldstein–Price。Goldstein–Price函数具有复杂曲面，包含多个局部极小值，并在 $(0,\ -1)$ 处有单一全局极小值：

$$f(x_1,x_2) = \left[1+(x_1+x_2+1)^2(19-14x_1+3x_1^2-14x_2+6x_1x_2+3x_2^2)\right]\times \left[30+(2x_1-3x_2)^2(18-32x_1+12x_1^2+48x_2-36x_1x_2+27x_2^2)\right] \tag{15}$$

其多个吸引域模拟了这样的实验设计情境：多个可行原型在质量上相互竞争，因此需要采用能够区分全局更优解与仅仅够用的解的策略。

Levy。 莱维函数是一个多峰函数，其全局极小值位于 (1, 1)。其定义如下：

$$f(x_1,x_2) = \sin^2(\pi w_1) + (w_1-1)^2\left[1+10\sin^2(\pi w_1+1)\right] + (w_2-1)^2\left[1+\sin^2(2\pi w_2)\right], \quad w_i = 1+\frac{x_i-1}{4},\quad i\in\{1,2\} \tag{16}$$

对人机交互而言，它刻画了这样的场景：要取得进展，必须在高度不规则的设计空间中探索；微小的参数变化可能导致结果质量出现大幅波动，例如在不确定性下调节交互式硬件或软件参数。

有效性。对于每项实验，我们仅提供简要介绍，更多信息请参阅正文。评估的运行方式与正文报告完全一致，仅真实函数发生变化。我们重点给出除研究4之外所有研究的补充结果；研究4需要更高维的表述，因此在此省略。

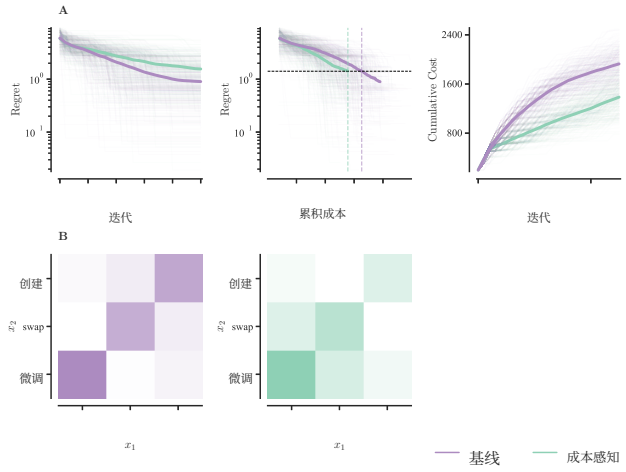
所有研究均依赖多个随机实例、多次重复，以及每次重复所用的随机种子。已知最优解支持客观评估，从而确保结果稳健、可解释且可复现。

C.1 研究1：无限预算下的性能

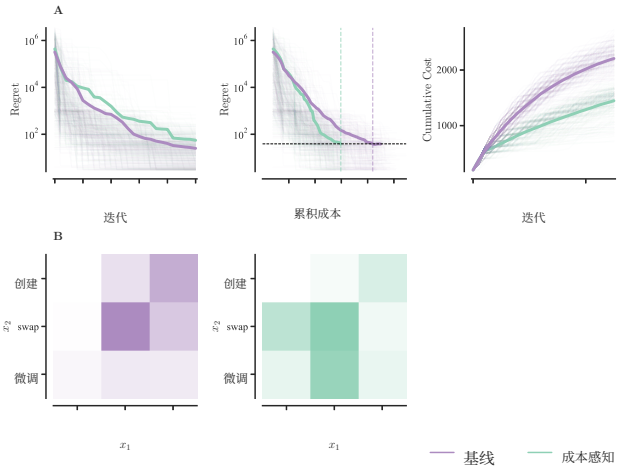
本研究检验成本感知方法在未施加显式预算时，是否能以更低的累积成本取得相近结果。我们对每个真实函数进行了25次迭代优化，每种方法进行250次独立试验。两种方法均从三个随机 Sobol 样本开始。在每一步，我们记录遗憾以及动作成本类型（微调，交换，创建）。虽然迭代次数固定，但累积成本并不固定，从而使我们能够比较样本效率（步数）与成本效率（所用资源）。由于分布不满足正态性，统计分析采用 Mann–Whitney U 检验。

Ackley。 结果如图15a所示。在遗憾值最低时，成本感知方法所需的成本（M = 1010.23, SD = 302.72 M = 1448.25 SD = 361.34 ）显著低于基线（, ），Mann–Whitney U = 10504.00, p < .001。然而，成本感知方法的最终遗憾值（M = 1.56, SD = 1.05）高于基线（M = 0.90, SD = 0.68），U = 43808.00, p < .001。在低成本预算下，即所有运行中总成本最低的情况下（预算 = 781），成本感知方法的遗憾值（M = 3.21, SD = 1.24）低于基线（M = 3.98, SD = 1.36），U = 20582.00, p < .001。使用完整样本时，我们的方法最终成本低于基线（成本：M = 1386.23，SD = 248.49；标准：M = 1927.28，SD = 227.48），双样本t检验证实了这一差异（$t(498)$ = −25.39, p < .001 ）。两种方法均未达到全局最优值。这些结果表明，尽管成本感知方法的最终遗憾值略高，但始终以显著更低的成本实现这一点。

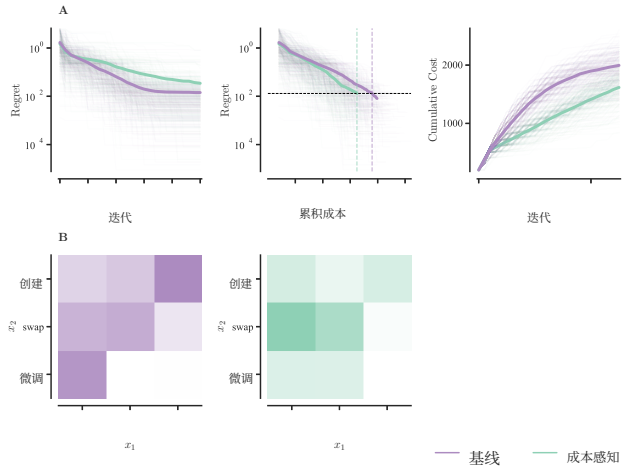
Goldstein–Price. 结果见图15b。在遗憾值最低的点，成本感知方法所需的成本显著低于基线（998.80 SD = 312.89 M = 1590.56 SD = 514.59 ，），U = 10731.50 p < .001。成本感知方法的最终遗憾值再次高于基线（M = 55.60, SD = 116.68）和（M = 25.32, SD = 33.97 U = 39169.00 p < .001 ），，。在匹配的成本水平下，



(a) Ackley



(b) Goldstein–Price



(c) Levy

图16：研究1：在固定迭代次数下比较成本感知贝叶斯优化与基线贝叶斯优化。x_1 和 x_2 是 x 中的两个参数。A - 左：遗憾值随迭代次数的变化。A - 中：遗憾值与累积成本的关系；成本感知方法以更低的成本实现了相近的遗憾值。A - 右：各次迭代的累积成本；成本感知方法始终需要更低的成本。B：采样编辑类型的分布（微调、交换、创建）；成本感知采样的分布更加均衡。

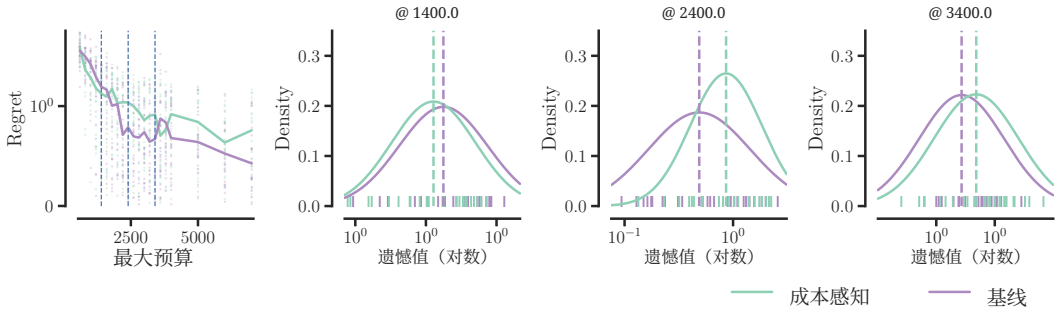
在所有运行中最终成本最低（预算 = 709）时，成本感知方法取得的遗憾值显著低于基线（$M = 8973.39$, $SD = 51980.67$）（$M = 15437.55$, $SD = 60678.71$），$U = 22441.00$, $p < .001$。使用完整样本时，我们的方法最终成本低于基线（成本：$M = 1447.29$，$SD = 206.94$；标准：M 2207 46，= 281 08 .S D.），曼–惠特尼检验支持这一差异（$U = 1781.00$, $p < .001$）。因此，尽管最终遗憾值较高，成本感知方法仍能在成本水平相同或更低的情况下持续降低遗憾值。

Levy。结果见图 15c。在遗憾值最低的点，成本感知方法所需成本显著低于基线（M = 1164.48, SD = 318.99 $M = 1614.22$ $SD = 370.13$ $U = 9672.50$ ），（，），，$p <$ .001 $M = 0.04$ $SD = 0.09$ 。成本感知方法的最终遗憾值高于基线（，）（$M = 0.01$, $SD = 0.06$），$U = 39687.00$, $p < .001$。在成本水平相近时（预算 = 835），差异仍然显著，$U = 26461.00$, $p = .003$。使用所有样本时，我们的方法最终成本仍低于基线（成本：M = 1616.56，SD = 233.88；标准：M = 1995.60，SD = 240.01），曼–惠特尼检验支持了这一差异（U = 7403.00, $p < .001$ ）。这些发现表明，尽管最终遗憾值较高，成本感知方法始终能以较低成本实现相近的结果。

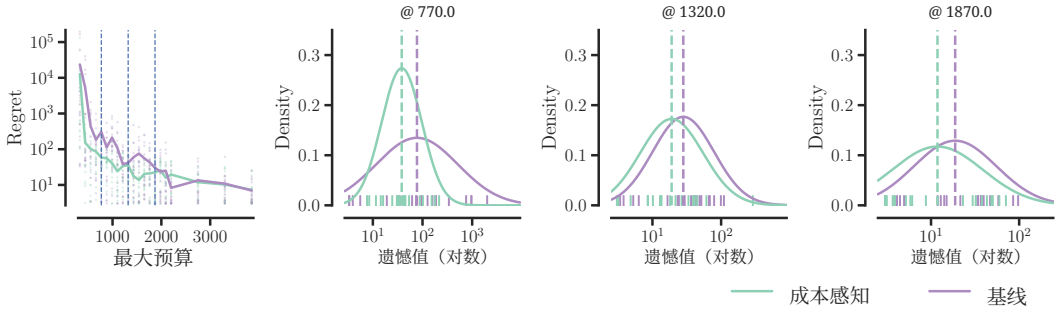
C.2 研究2：预算约束下的性能

本研究评估在总可用预算设有上限时，成本感知方法能否取得更好的结果。我们模拟了固定成本预算下的优化运行，各函数的预算范围分别确定。每种方法均使用全部预算，并在预算耗尽后终止。对于每个预算水平，我们记录最终遗憾值。每个水平下，每种方法均运行了 75 次优化。与研究1相同，由于结果分布不服从正态分布，我们使用曼–惠特尼 U 检验评估统计显著性。

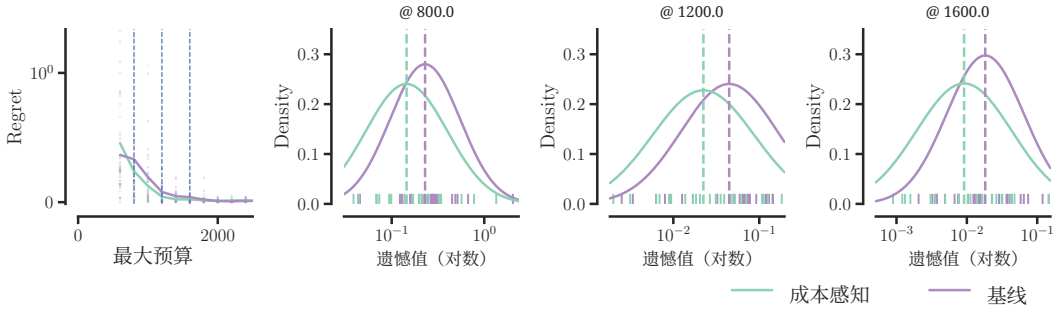
Ackley。图17a显示，在预算为1400时，成本感知方法取得的遗憾值略低于基线，但该



(a) Ackley



(b) Goldstein–Price



(c) Levy

图17：研究2：不同成本预算下的最终遗憾值和遗憾值分布。每一行展示一个基准函数。左侧：不同成本预算范围内的最终遗憾值，以及各预算下的分布。右侧：选定成本预算下的遗憾值分布。在低预算和中等预算范围内，成本感知方法往往优于基线，但差异在三个函数之间有所不同。

差异并不显著（$p = .065$）。在更高预算（2400和3400）下，基线的表现显著更好（$p = .001$ 和 $p = .027$）。因此，只有最低预算对成本感知方法略有利。这种有限的影响可能反映了Ackley景观的特点：均匀分布的局部极小值和平滑的过渡降低了对成本分配的灵敏度，因此两种方法都收敛到相近的解。这一点也体现在左图所示的噪声中，即使预算更高，也不能保证任一方法找到更好的解。

Goldstein–Price。结果如图 17b 所示。在预算为 770 时，成本感知方法的遗憾值显著低于基线（$M = 58.58$, $SD = 58.20$），而基线为（$M = 304.76$, $SD = 496.81$），$U = 1296.00$, $p = .008$。在 1320 时，成本感知方法再次优于基线（$M = 36.01$, $SD = 60.76$ 对比 $M = 41.51$, $SD = 30.20$ $U = 1233.00$ $p = .003$ 1870, ），，。在成本感知方法的遗憾值仍低于基线（$M = 22.20$, $SD = 22.01$ 对比 $M = 30.09$, $SD = 26.25$），但这一差异不具有统计显著性（$U = 1485.00$, $p = .099$）。这些结果表明，在较低和中等范围的预算下，成本感知方法具有持续优势，但在更高的预算分配下，差距会缩小。

Levy. 结果见图 17c。在预算为 800 时，成本感知方法的遗憾值显著低于基线（$M = 0.24, SD = 0.29$）

（$M = 0.33, SD = 0.38$），$U = 1944.00$, $p = .001$ 1200, 。在该预算下，这一差异依然存在，成本感知方法的遗憾值仍低于基线（$M = 0.04, SD = 0.05$），（$M = 0.08, SD = 0.07$），$U = 1881.00$, $p < .001$。在 1600 时，两种方法的差距继续扩大，成本感知方法（$M = 0.02, SD = 0.03$）优于基线（$M = 0.04$, $SD = 0.05$ $U = 2259.00$ $p = .038$ ），，。这些结果表明，成本感知方法在不同预算下均具有稳定且统计上可靠的优势，尤其是在较低和中等预算范围内，提升更为显著。

C.3 研究3：对成本不对称性的灵敏度

本研究评估成本感知方法是否会适应此类成本不对称性，而基线方法则对所有评估一视同仁。我们分别将硬件和软件**创建**成本设为 {100, 400, 1600, 6400}，由此得到16种成本条件。每种条件下，我们进行了50次优化试验，每次迭代25轮。我们记录了按组件和操作类型（**微调**、**交换**、**创建**）划分的样本数量。我们采用广义线性混合效应模型，评估这些方法是否会根据成本不对称性调整采样行为。

Ackley。成本感知方法会根据成本不对称性调整采样行为（图18a）。随着硬件成本增加，该方法显著减少了**创建**操作，并将工作量转向成本较低的**交换**和**微调**操作（$p < .001$ 用于交互效应）。基线方法没有表现出类似的适应性，硬件和软件操作的采样率几乎保持不变。事后对比证实，基线方法系统性地对硬件**创建**操作过度采样，对软件编辑采样不足（$p < .0001$）。相比之下，成本感知方法会根据硬件成本的变化，在不同操作类型之间重新分配工作量，表明它能够动态适应不对称性，而基线方法则保持静态。

Goldstein–Price。成本感知方法对成本不对称性的适应性很强（图18b）。随着硬件成本增加，该方法减少了**创建** 采样，并转而采取更多**交换** 和 **微调** 操作（ $p <$ .001 交

互效应）。相比之下，基线几乎没有调整，仍在各类操作间分配相近的工作量。事后对比证实，基线对硬件**创建** 采样过多，而对微调操作采样不足（$p < .0001$）；成本感知方法则根据成本，在不同模态和操作类型间重新分配采样。这些结果表明，只有成本感知方法会对不对称性作出动态响应，而基线仍然僵化。

Levy。成本感知方法再次适应了成本不对称性（图 18c）。硬件成本较高时，该方法减少了对**创建**操作的依赖，并增加了对**交换**和**微调**（$p < .001$ 适用于多组对比）的使用。基线几乎没有调整，硬件采样和软件采样的比例基本保持不变。事后检验证实，基线始终过度采样**创建**操作，而成本感知方法则更灵活地重新分配工作量（例如，**创建**与**交换**和**微调**之间的对比均 $p < .001$）。这些结果进一步表明，只有成本感知方法会根据成本差异调整采样策略，而基线则保持不变。

C.4 研究5：重新调整成本权重

本研究评估成本感知方法能否随着复杂度增加而有效扩展，并在周期中途成本发生变化时重新分配工作量。我们在两种条件下进行了评估，每种条件包含 24 次迭代和三组随机初始化。在固定条件下，成本权重保持不变。在动态条件下，硬件**创建**成本在第 10 次迭代时增加十倍（例如，工具不可用），随后在第 17 次迭代时降至十分之一（例如，发现更快的制造方法）。我们比较三个成本阶段中的**创建**操作频率：第 4–10 次、第 11–17 次和第 18–24 次迭代。每种条件均重复进行 50 次优化试验。我们采用广义线性混合效应模型评估条件、阶段和操作类型之间的交互作用，重点关注**创建**采样。

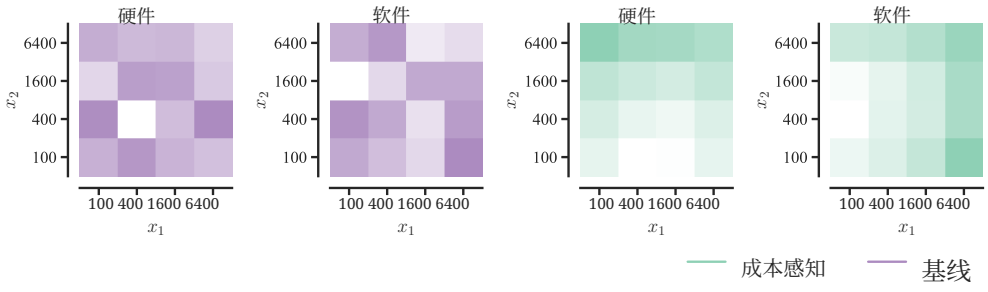
Ackley。成本感知方法对成本变化的适应性很强。在动态条件下，**创建** 采样在成本增加后大幅下降（$p < .0001$），并在成本再次降低时显著上升（$p < .0001$）。相比之下，恒定条件下的基线在各阶段均没有明显变化（所有 $p > .47$）。在不同条件之间比较，动态运行在高成本阶段采样的**创建** 操作更少（$p < .0001$），成本降低后则更多（ .0001 $p = .76$ ）；初始阶段没有差异（）。尽管效用曲面相同，仍出现了这些效应，这表明成本感知方法响应的是成本结构，而非结果。这些结果表明，成本感知方法会根据变化中的约束重新分配采样力度，而基线不会。

Goldstein–Price。成本感知方法再次适应了成本变化。在动态条件下，**创建** 采样在成本增加后略有下降（$p = .16$），并在成本下降时大幅上升（$p < .0001$）。恒定条件下的基线在各阶段均未出现显著变化（均为 $p > .59$）。在不同条件之间，初始阶段没有差异（$p = .88$），但动态运行在高成本阶段采样的**创建**动作更少（$p = .008$），而在成本下降后采样更多（$p < .0001$）。这些发现证实，即使在不同的真实值下，成本感知方法也会依照不断变化的成本约束重新分配采样力度，而基线则保持静态。

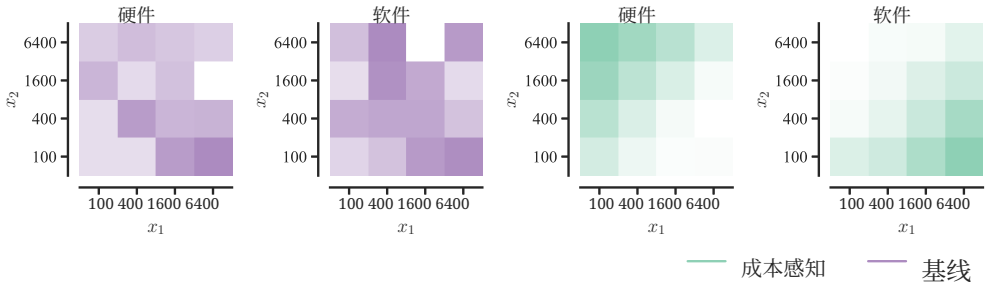
Levy。成本感知方法再次对不断变化的成本表现出强适应。在动态条件下，**创建** 采样在成本增加后显著下降（$p < .0001$），并在成本降低时急剧上升（$p < .0001$）。恒定条件下的基线在各阶段均无变化（均为 $p \geq .74$）。在不同条件之间，初始阶段没有差异（$p = .22$），但动态运行在高成本阶段采样的**创建** 动作更少（$p < .0001$），而在成本再次下降时采样更多（$p < .0001$）。这些发现进一步表明，成本感知方法会根据动态成本变化重新分配采样力度，而基线则保持静态。

C.5 研究6：成本估算不准确的影响

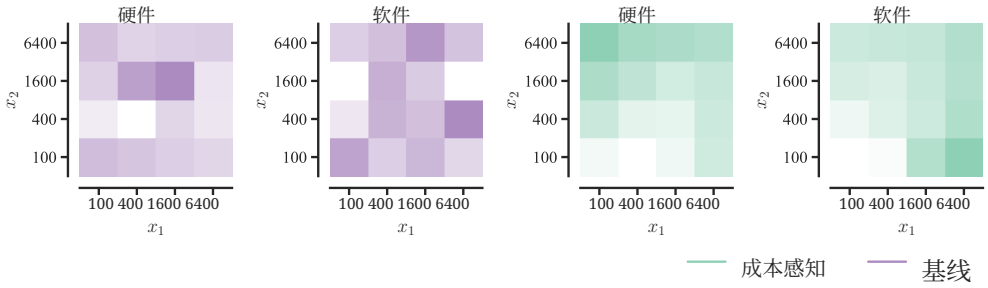
全部。对于其他函数，我们通过将真实成本按 $\alpha \in [10^{-1}, 10^{1}]$ 缩放，并在整个优化过程中保持认知成本不变，重复进行成本误估实验。对于所有函数，定性表现均与主要结果一致。提高某个算子的认知成本会降低其采样频率，



(a) Ackley



(b) Goldstein–Price



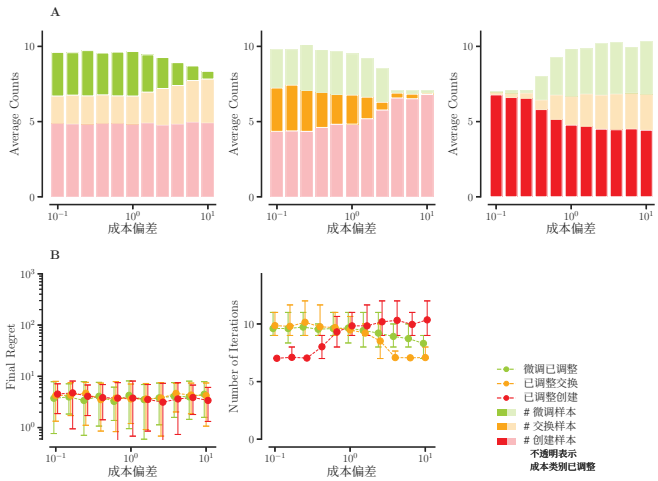
(c) Levy

图 18：研究3：按方法、组件和成本不对称性划分的样本数量。热图展示了不同采集成本下硬件和软件编辑操作的频率。成本感知方法会调整采样，以避免高成本操作；而基线方法则均匀分配工作量。为便于查看，每张热图均单独进行了归一化。

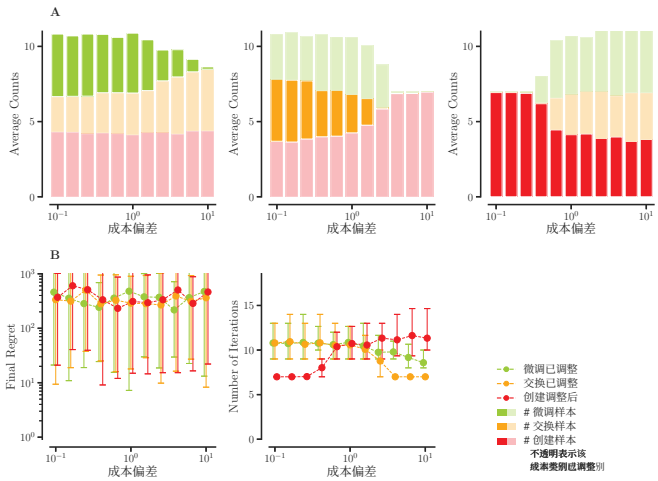
而降低其认知成本则会增加其使用量。后续影响遵循相同规律：交换的成本误估会造成最显著的变化，低估创建的成本会迅速耗尽预算，而微调的成本误估影响则相对较小。尽管采样模式发生了这些变化，整体性能仍保持稳定，且该方法能够以可预测的方式进行适应。与之前一样，保持成本关系的比例一致比精确数值更重要。

C.6 总结

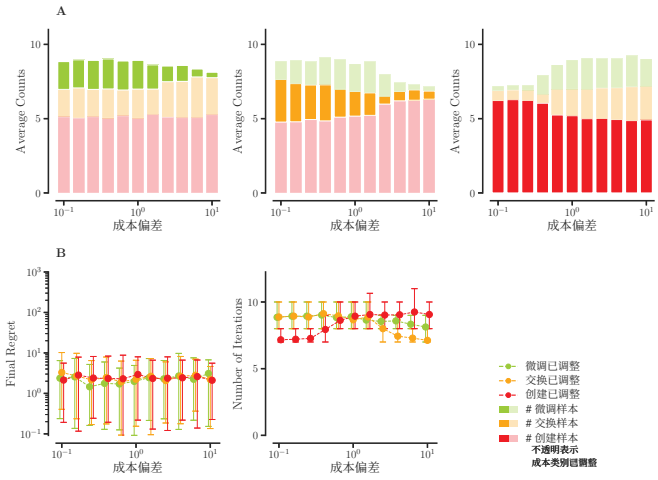
表3汇总了各项实验和真实值的结果。在所有基准测试中，成本感知方法始终展现出较高的成本效率（实验1）、对不对称性的有效适应能力（实验3），以及在周期中途对变化的成本作出响应的能力（实验5）。在预算约束下（实验2），其优势则较为复杂：在 Goldstein–Price 和 Levy 函数上，其表现与基线相当或更优；在 Ackley 上未显示出可靠的优势；在 Rosenbrock 上则取得了持续改进。总体而言，研究结果证实，成本感知优化能够推广到多种不同函数，但其在严格预算限制下的优势取决于问题特征。



(a) Ackley



(b) Goldstein–Price



(c) Levy

图 20：研究6：不准确的成本估计对优化行为的影响。认知成本是真实成本按一个系数缩放后的版本，该系数 α 范围从 10^{-1} 到 10^{1}。图中展示了这些错误估计如何影响 微调、交换和 创建 过程中的最终遗憾、迭代次数和采样模式。误差条表示 95% 置信区间。

实验	Rosenbrock	Ackley	Goldstein-Price	Levy
成本效率	✓	✓	✓	✓
预算约束	✓	X	✓	✓
成本不对称性	✓	✓	✓	✓
周期中途改变成本	✓	✓	✓	✓
对成本估算的敏感性	✓	✓	✓	✓

表3：成本感知方法在四个实验（行）和四种不同真实值（列）上的性能。每个单元格表示在给定真实值下是否展现了该能力。Rosenbrock 评估的结果可在论文正文中找到。