KnifeCut: 通过切割线修复细小部分分割

林铮* 南开大学计算机学院
段正鹏* 南开大学，计算机学院
张钊 商汤科技
郭春乐† 南开大学，计算机学院
程明明 南开大学，计算机学院

摘要

对当前的图像分割技术来说，处理拥有细小结构的物体仍然具有挑战性。这些技术通常在分割出物体的主体部分时表现良好，但是在输出对细小部分的分割却不尽人意。在实际应用中，不可避免地需要对这些输出进行后处理。然而，无论使用专业的编辑应用（例如 PhotoShop），还是借助当前的交互式图像分割方法（例如点击、涂鸦和多边形），修复这些输出都是费时费力的。为了改善预分割中不尽人意的细小结构，我们提出了一种有效的交互模式。在这种模式中，用户只需在错误标记的细小部分处划一条线，就像用刀切割一样。这种低压力、直观的操作不需要用户特意瞄准，并且在使用鼠标、触摸板和移动设备时非常友好。此外，由于线段穿过了包含细小部分像素的前景和背景区域，因此它能够提供对比度先验。基于这个交互思想，我们提出了 KnifeCut。此方法为用户提供了两个结果，其中一个结果只关注目标的细小部分，另一个为与目标部分具有相似特征的所有细小部分提供细化。据我们所知，KnifeCut 是第一种有针对性地解决交互式细小结构优化的方法。大量的实验和可视化结果进一步证明了此方法的友好性、便捷性和有效性。本文的项目页面详见网址[1]。

CCS CONCEPTS

• 以人为中心的计算 → 人机交互; **交互设计过程和方法**; **• 计算方法 → 人工智能**.

KEYWORDS

交互式分割; **细小物体**; **用户交互**

ACM Reference Format:

林铮, 段正鹏, 张钊, 郭春乐, and 程明明. 2022. KnifeCut: 通过切割线修复细小部分分割. In *Proceedings of the 30th ACM International Conference on Multimedia (MM '22), October 10–14, 2022, Lisboa, Portugal.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3503161.3547803

1 引言

在现实世界中，许多物体都具有细小的结构，例如栏杆、网和树枝。当前的图像分割方法在处理这些细小部分时往往失败。在大多数情况下，这些方法输出的分割在主体部分处表现地很好，但在细小部分处输出的结果不尽人意。因此，当需要高精度结果时，人工指导的后处理将不可避免地被引入。人们通常采用专业的图像处理工具（如 PhotoShop）或流行的交互式分割技术 [5, 24, 28, 38] 来优化细小的部分。

然而，目前的交互方式，例如笔刷、单击和多边形，都不能高效地解决细小的部分。如图 Fig. 1的顶行所示，用多边形标记边界或用笔刷绘制整个腿部可能既耗时又费力。基于单击的方法在一定程度上降低了注释的复杂性，但是将正点、负点分别放在腿上、腿旁边也是一个负担。在本文中，为了有效优化细小的结构，我们引入了一种快速、低压的交互模式，用户只需通过错误标记的像素绘制一条相交线，就像用刀切割一样。如图 Fig. 1所示，在错误标记的腿上进行切割的动作是直观并且快速的。图 Fig. 2展示了更加具体的样例。无论是在未分割的伞和球拍上，还是在过度分割的爪和网上，本文提出的切割动作都可以在瞬

*段正鹏是本文的共同第一作者
†郭春乐是本文的通讯作者
[1]项目页面为http://mmcheng.net/knifecut/

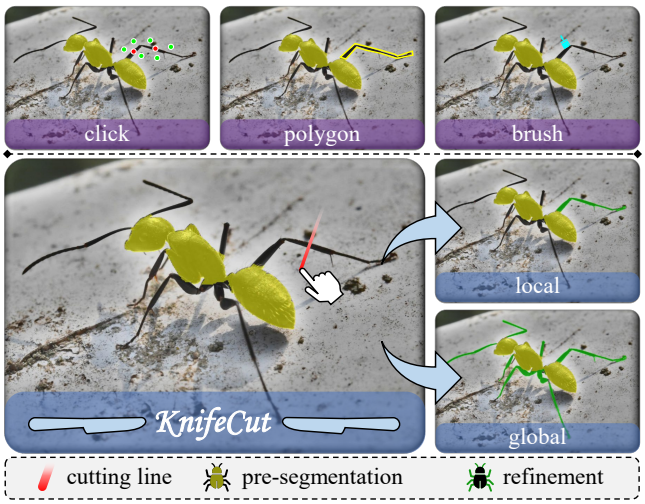

图 1: KnifeCut 的展示以及与其他细化交互的比较。顶部：处理细小部分时，其他细化交互（例如基于单击、多边形和笔刷）效率低下。底部：使用 KnifeCut，用户可以在细小部分上划出一条线，进行精细化处理，就像用小刀切割一样。KnifeCut 为用户提供了两个结果：一个是目标的细小部分，另一个是所有相关的细小部分。

间完成；而对于之前那些需要仔细瞄准前景和边界的交互而言，处理这些情况是难以想象的。此外，由于切割线上既有前景内容，也有背景内容，并且上面一定具有细小部分的像素，因此它提供了对比度先验。

基于提出的交互式思想，我们进一步设计了一个名为 KnifeCut 的框架。如图 Fig. 1所示，KnifeCut 旨在获得包含目标蚂蚁腿的局部细化和包含所有腿的全局细化；同时还保持了一个良好分割的主体，没有发生明显的变化。具体而言，我们的 KnifeCut 首先预测与切割线相邻的细小部分区域，然后通过相似性代理利用其对应的特征来激活所有相关的细小部分。利用切割线先验和激活的相似图，KnifeCut 估计局部和全局细小部分范围。它利用这些估计图进一步预测两个分支中的局部和全局细小部分优化，同时避免对主体的影响。此外，用户可以选择两个细化结果中的任何一个，其中局部分支仅关注目标细小部分，全局分支细化所有相关细小部分。

本文的贡献可归纳如下：

- 我们提出了一种用于细小部分分割优化的有效交互模式，该模式只需要在分割较差的区域（如切割）中绘制一条线。
- 基于交互思想，我们提出了 KnifeCut，它根据交互从局部和全局角度提供细化。
- 据我们所知，KnifeCut 是第一种设计用于交互式细小部分优化的方法。大量的实验进一步证明了它的便捷性和有效性。

2 相关的工作

2.1 交互式公共物体分割

大多数交互式分割方法将用户交互（如点和涂鸦）视为前景和背景的约束。在早期阶段，传统方法将约束与图像的低层特征相结合来预测背景/前景概率。Rother 等人 [37] 使用高斯混合模型来构建 GraphCut [5] 模型，并且通过最小切割/最大流算法 [4] 来解决这个问题。Li 等人 [21] 通过将图切割与预计算过分割相结合来实现即时反馈。Grady 等人 [11] 将每个像素分配给最有可能到达的用户定义标签，通过在 [18] 中引入重新启动模拟来改进。然而，仅考虑低级特征而忽略高级信息使得这些方法在复杂环境中无效。

尽管还有一些工作 [17, 41, 42] 进一步提升传统的方法，基于深度学习的方法最近已成为该领域引入高级特征的主流。这类方法充分利用高层次特征来使用约束。Xu 等人 [43] 提出了第一个基于深度学习的方法和多个随机点采样策略，这些已经成为这个领域的标准协议。一些工作聚焦于网络结构的设计。Zhao 等人 [47] 使用交互式涂鸦训练区域和边界域的模型，并进行凸推理以获得最终结果。Hu 等人 [13] 提出了一种双流融合网络来解耦图像和交互。对于指导图谱，Majumder 等人 [31] 利用用户点击生成内容感知的指导图谱。Lin 等人 [27] 强调第一次点击的重要作用，并且将它视作特殊的指导。为了促使每个相互作用的像素被正确分割，Jang 等人 [15] 提出了 BRS，f-BRS [38] 对此方法进行了提升。Kontogianni 等人 [19] 通过从用户校正中学习，对模型参数进行连续调整。Lin 等人 [29] 在注释过程中进一步使用了更多信息。由于用

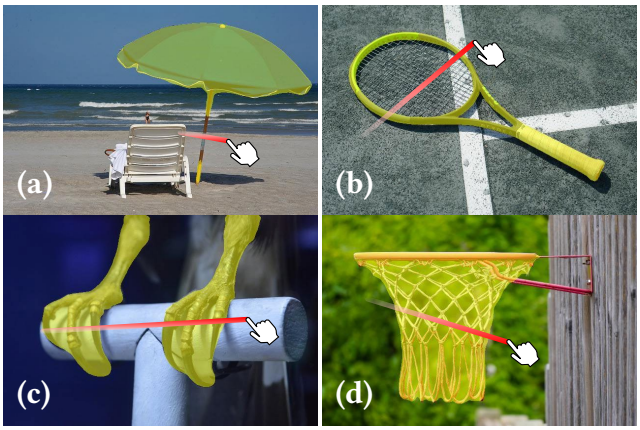

图 2: 本文提出的 KnifeCut 处理的不同细小状况。(a) 沙滩伞的把手漏掉了一部分；(b) 网球拍的网尚未被分割；(c) 爪子没有被很好地分割；(d) 网被分割，包括间隙中的背景。

户交互不可避免地具有模糊性，因此已经开展了一些研究 [22, 25] 来解决这个问题。为了充分利用用户交互并更好地结合本地和全局特征，Liew 等人 [23] 提出了 RIS-Net 来捕获每个点击对周围的信息以进一步细化。Chen 等人 [8] 引入了非局部方法，以将信息从点击点正确传播到目标区域。Lin 等人 [26] 引入了焦点视图来利用每次点击的周围信息。

虽然大多数研究是通过对背景和前景的注释进行的，但是这项任务探索了多种交互模式。Mortensen 等人 [35] 提出智能剪刀，通过鼠标简单的手势动作来提取物体。Castrejon 等人 [6] 以图像裁剪为输入，以循环方式生成轮廓对象的多边形顶点，Polygon-RNN++ [1] 对此进行了改进。使用相同的输入法，Ling 等人 [30] 使用图卷积网络同时预测所有顶点。Jain 等人 [14] and Le 等人 [20] 采用边界点击作为交互。还有一些工作结合不同的交互模式以获得更好的效果，例如边界框和点击 [3, 45]。极值点已被证明对常见物体 [32]、具有细小结构的物体 [24] 和完整图像 [2] 有效。

2.2 交互式细小物体分割

细小物体的交互式分割已经研究了很长时间。Vicente 等人 [39] 在 graph cut [5] 之前施加了额外的连接性来处理这个任务。Jegelka 等人 [16] 在分割具有细小物体的图像时为均匀边界引入了一个折扣。Mansilla 等人 [33, 33] 提出了 COIFT，它结合了 OIFT 中的连通性约束来改进细小物体的分割。Dong 等人 [10] 设计了一种新的带有标签的亚马尔可夫随机游走算法来解决这个问题。Liew 等人 [24] 提出了使用极值点作为交互的 TOS-Net，以及一个名为 ThinObject-5K 的大规模数据集用于此任务。但是，该方法在实际应用中无法处理细小物体的优化任务。为了解决这个问题，基于我们的交互思想，我们提出了 KnifeCut，它在预分割掩模上对细小部分进行优化，对主体部分的影响很小。

3 提出的交互

3.1 交互的引入

为了更好地解释以下部分，我们在这里定义了一些符号。G 表示图像 I 的分割标签。通过使用与 [24] 相同的策略，我们从 G 中提取细小部分的分割标签 $\mathbf{G}_{\text{thin}}$。而非细小部分的分割标签 $\mathbf{G}_{\text{non-thin}}$ 可以通过 $\mathbf{G}_{\text{non-thin}} = \mathbf{G} - \mathbf{G}_{\text{thin}}$ 获取。

我们将通过分割方法获取的预分割表示为 $\mathbf{P}'$。以 IoU 分数为衡量标准，$\mathbf{P}'$ 通常比 G 取得更好的性能。然而，这些方法往往无法处理具有细小部分的对象，这是由于细小部分的像素太少，这很难通过 IoU 分数反映。如 Fig. 2所示，$\mathbf{P}'$ 主要面临以下两种细小结构的状况：1) **未分割**：细小部分的细节，甚至整个区域都丢失了，例如 Fig. 2 (a-b) 中的沙滩伞和网球拍。2) **过度分割**：细小部分（包括间隙中的背景）保持得太密而不能分离，例如 Fig. 2 (c-d) 中的爪子和网。

为了克服这一问题，我们提供了一种专门为细小部分设计的高效细化工具。与 $\mathbf{G}_{\text{non-thin}}$ 相比，$\mathbf{P}'$ 已经表现得很好了；因此我们只关注主体分割微小变化情况下的细小区域优化。考虑到上述复杂情况，并受到人类行为的启发，我们提出了一种有效的交互模式以供改进。正如人们倾向于用刀切割细小物体一样，用户只需要在细小部分的错误标记区域绘制一条相交线。Fig. 2还展示了我们的方法处理的实际样例。无论是未

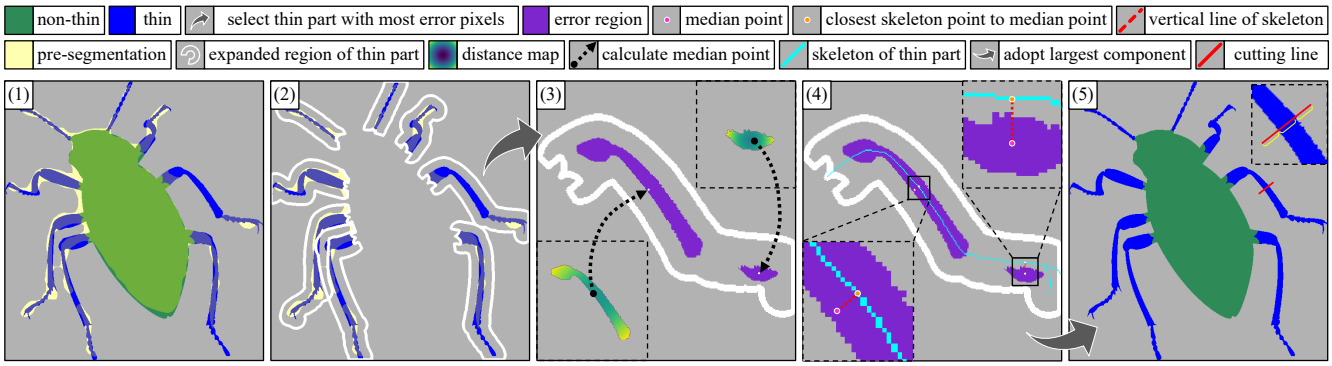

图 3: 测试阶段我们的模拟算法的可视化步骤。(1) 提出了预分割、非细小区域和细小区域。(2) 细小部分的扩展区域由白色边界线表示。对每个扩展区域中的误差像素进行计数后，选择误差像素最多的细小部分。(3) 计算距离图谱，其中每个像素值对应于到其他点的距离和，然后在误差区域中找到中间点。(4) 首先提取细小部分的骨架，然后使用离中间点最近的骨架点作为锚点。基于锚点，绘制骨架的垂直线。(5) 采用误差区域的最大分量制作切割线。两端将在细小部分上被扩展相同的长度。最终效果如图所示。

分割的还是过度分割的，交互都可以在不需要思考和努力的情况下完成。

除了方便和高效之外，切割线还包含其他交互方法无法提供的信息。因为它既有背景内容，也有前景内容，所以切割线包含对比的先验。Sec. 5.2中的进一步实验将证明我们的方法在处理细小部分时相较于基于点击的方法的优越性。

3.2 交互的模拟

众所周知，即使面对同一张图片，不同的用户往往也会做出不同的交互。为了让 KnifeCut 更好地适应这一点，我们需要各种交互来进行训练。由于从真实用户收集这些交互太昂贵且不切实际，因此我们设计了一种模拟算法来模拟用户并自动生成对。此外，为了避免训练过程中的过拟合，模拟算法具有一定的随机性。但是在测试的时候，我们需要保证每次测试的公平性，迎合用户的操作习惯。因此，我们在假设用户大致垂直于骨架切割最大误差区域的假设下，对模拟算法进行了一点改动。仿真算法的细节如下：

步骤 1： 在图像中的所有细小部分中，用户倾向于首先细化预分割最差的部分，因此我们需要评估其当前质量。为了简化，我们假设 $\mathbf{G}_{\text{thin}}$ 上的每个组件可以被视为一个细小的部分，用 $\mathbf{T}$ 表示。遵循 [24] 中的做法，我们首先设置了一个可以用如下公式表示的阈值 τ：

$$\tau = 10 \times \frac{\max(H_{\text{box}}, W_{\text{box}})}{300}, \tag{1}$$

其中，H_{box} 和 W_{box} 分别指的是包围物体的边界框的高度和宽度。

我们使用 $d(\mathbf{p}_1, \mathbf{p}_2)$ 表示 $\mathbf{p}_1$ 和 $\mathbf{p}_2$ 两点之间的欧式距离。给定一个标签 $\mathbf{G} \in \{0,1\}^{H \times W}$，其中，$H$ 和 W 分别指的是高度和宽度，我们根据如下公式表示出函数 $\phi(\mathbf{p}, \mathbf{G})$ 来计算距离变换：

$$\phi(\mathbf{p}, \mathbf{G}) = \min_{\forall \{\mathbf{q} | \mathbf{G}_\mathbf{q} = 1\}} d(\mathbf{p}, \mathbf{q}), \tag{2}$$

其中，$\mathbf{G}_\mathbf{q}$ 指的是像素位置 $\mathbf{q}$ 处的 $\mathbf{G}$ 值。

因此，每个细小部分的扩展区域 $\mathbf{E}$（也被用作评价掩膜）可以用如下公式表示：

$$\mathbf{E}^i = \{\mathbf{p} : (\phi(\mathbf{p}, \mathbf{T}^i) \le \tau) \wedge ((\mathbf{G}_{\text{non-thin}})_\mathbf{p} = 0)\}, \tag{3}$$

其中，$\mathbf{T}^i$ 和 $\mathbf{E}^i$ 分别指的是第 i 个细小的部分和对应的扩展区域。采用了一个白色的边界线表明扩展区域 $\mathbf{E}$，如 Fig. 3(2) 所示。

步骤 2： 我们计算 $\mathbf{T}^i$ 中错误像素的数目 $\mathbf{N}^i$：

$$\mathbf{N}^i = \sum (|\mathbf{P}' - \mathbf{G}| \times \mathbf{E}^i), \tag{4}$$

其中，× 指的是逐元素乘法。

对于训练过程，我们将为交互随机地挑选细小的部分，其中随机概率与 $\mathbf{N}^t$ 成正比。但是，在测试阶段，将为后续步骤选择误差像素最多的细小部分。Fig. 3 (2-3) 展示了这个过程的一个样例。

步骤 3：对于选定的细小部分，我们希望直线穿过误差区域。因此，我们首先计算误差区域，该区域在 Fig. 3中以紫色表示。对于训练阶段，我们将在错误区域中随机采样一个点 $\mathbf{m}$。但是在测试阶段期间，我们将选择错误区域的最大分量，并将其表示为 $\mathbf{C}$。点 $\mathbf{m}$ 将位于 $\mathbf{C}$ 的中点，该中点与其他点的距离和最小。这个过程可以用如下公式表示：

$$\mathbf{m} = \underset{\forall\{\mathbf{p}|\mathbf{C_p}=1\}}{\arg\min} \sum_{\mathbf{C_q}=1} d(\mathbf{p},\mathbf{q}). \tag{5}$$

Fig. 3 (3) 展示了距离图谱的两个示例，其中每个像素值对应于到其他点的距离和。

步骤 4：为了使线垂直于细小的部分，我们首先提取所选细小部分的骨架，并在上面选择一个锚点 $\mathbf{a}$。我们使用 [46] 提出的算法提取骨架，并将其记为 $\mathbf{K}$。无论在训练阶段还是测试阶段，都将从 $\mathbf{K}$ 中选择与点 $\mathbf{m}$ 的距离最短的锚点 $\mathbf{a}$，这个过程可以用如下公式表示：

$$\mathbf{a} = \underset{\forall\{\mathbf{p}|\mathbf{K_p}=1\}}{\arg\min}\, d(\mathbf{p},\mathbf{m}). \tag{6}$$

确定锚点后，我们需要选择线的角度 θ。在模型训练，为了避免训练过程中的过拟合，角度 θ 在 $[0, 180°)$ 内随机设置，线从 x 轴正向处顺时针开始旋转。在测试阶段，我们根据锚点 $\mathbf{a}$ 制作一条骨架的垂直线，它倾向于通过点 $\mathbf{m}$。Fig. 3 (4) 展示了垂直线的两个样例。

步骤 5：确定了锚点 $\mathbf{a}$ 和垂直于骨架的角度，我们就可以画出切割线了。我们首先在细小部分的边界上定位线的两端。为了迎合用户的操作习惯，我们给这条线的两端增加了一定的扩展 l。最初的 l 与细小部分上的线长度相同。在训练阶段，扩展 l 分别在两端随机缩放。然而，在测试阶段，两端将被初始的 l 等长扩展。Fig. 3 (5) 显示了模拟线的最终可视化结果。

4 提出的网络

4.1 特征提取器

随着基于深度学习的方法的普及，通常采用经典的神经网络来提取图像的特征。由于大多数网络是通过从高分辨率到低分辨率的方式提取特征，因此在下采样过程中可能会丢失细小部分的信息，从而导致细小结构的分割性能不佳。因此，为了在整个架构中保持高分辨率特征，我们将 HRNet [40] 作为我们的特征提取器。我们将该网络输出的特征用作 KnifeCut 中使用的特征，并将其记为 $\mathbf{F}$。值得注意的是，此特征的尺寸为原始图像分辨率的 $\frac{1}{4}$。

此外，为了与其他交互式分割方法保持一致，我们还采用 ResNet [12] 作为特征提取器。但是，由于它是通过从高到低的过程来提取特征的，前几层输出的高分辨率特征包含有限的高级表示。因此，为了保持与 HRNet 相同分辨率的特征，我们构建了 DeepLab v3+ [7] 的类似架构。ResNet 输出的特征将被送入空洞空间金字塔池化（Atrous Spatial Pyramid Pooling，ASPP）模块和解码器模块，同时其分辨率将恢复为 $\frac{1}{4}$。由 HRNet 和 ResNet 提取的特征的效果详见 Sec. 5.3。

4.2 相似性模块

基于观察到物体的细小部分（例如蚂蚁的腿）通常具有相似的特征，我们设计了一个简单但有效的模块来利用相似信息。

我们首先将线掩膜 $\mathbf{L}$ 和特征 $\mathbf{F}$ 连接在一起，并将它们与批归一化层和 ReLU 激活函数一起输入到七个 3×3 卷积层。将输出乘以 $\mathbf{L}$，我们便可以得到与直线相邻的细部分的预测，并将其记为 $\mathbf{P}_{\text{line}}$。基于特征 $\mathbf{F}$ 和预测 $\mathbf{P}_{\text{line}}$，我们可以使用如下公式计算出细小部分的代理向量 $\boldsymbol{s}$：

$$\boldsymbol{s} = \frac{\tilde{\sum}(\mathbf{F}\times\mathbf{P}_{\text{line}})}{\sum \mathbf{P}_{\text{line}}}, \tag{7}$$

其中，$\tilde{\sum}$ 指的是逐个元素相加的操作，$\boldsymbol{s}$ 的维度与 $\mathbf{F}$ 的网络层具有相同的数目。

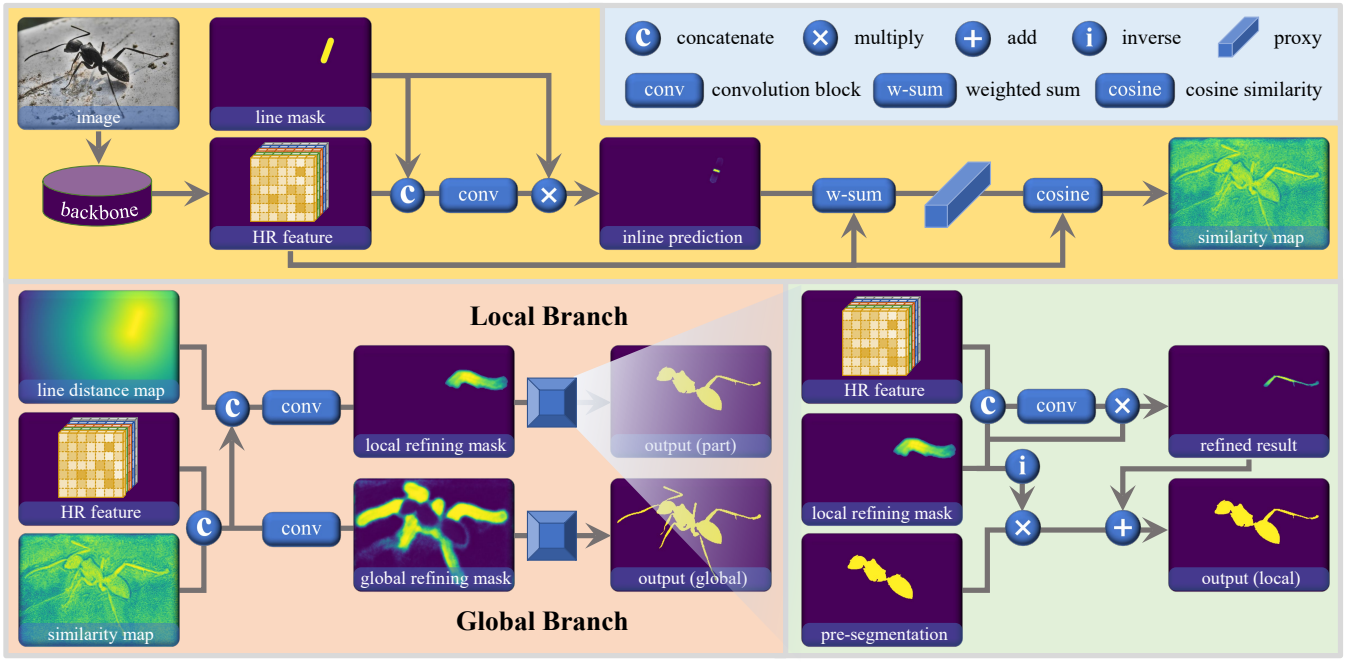

图 4: KnifeCut 的网络结构。顶部是细小相似性模块，它通过切割线上的相似性代理激活所有相关的细小部分。底部是具有两个分支的分割细化模块。它首先估计局部和全局细化掩膜，然后在两个掩膜的帮助下获得细化结果。结合细化结果和预分割掩膜，网络输出局部和全局结果。请在 Sec. 4查阅详细信息。

然后，我们通过余弦相似度获得相似性图谱 $\mathbf{S}$：

$$\mathbf{S}_\mathbf{p} = \frac{\boldsymbol{s} \cdot \mathbf{F}_\mathbf{p}}{\|\boldsymbol{s}\|\|\mathbf{F}_\mathbf{p}\|}, \tag{8}$$

其中，$\|\boldsymbol{s}\|$ 和 $\|\mathbf{F}_\mathbf{p}\|$ 分别指的是 $\boldsymbol{s}$ 和 $\mathbf{F}_\mathbf{p}$ 的规范化形式. $\mathbf{F}_\mathbf{p}$ 指的是像素位置 $\mathbf{p}$ 处的特征向量，此向量和 $\boldsymbol{s}$ 具有相同的维度。可视化的 $\mathbf{S}$ 详见 Fig. 4和 Fig. 5。

4.3 分割细化模块

考虑到用户的不同意图，我们为分割细化模块配备了两个分支。一个分支是掌握用户提供的交互，只对目标细小部分进行优化，而另一个分支则利用细小部分共享的相似性，一次性优化所有细小部分。

我们首先通过 $\mathbf{D}_\mathbf{p} = \phi(\mathbf{p}, \mathbf{L})$ 得到距离图 $\mathbf{D}$。为了利用交互信息，局部分支以特征 $\mathbf{F}$、相似度图 $\mathbf{S}$ 和距离图 $\mathbf{D}$ 为输入；而全局分支的输入不包括 $\mathbf{D}$。

使用类似的网络结构，拼接而成的输入被送入到卷积块中。因此，我们可以分别获得局部和全局分支的精炼掩膜 $\mathbf{M}$。然后我们将 $\mathbf{M}$ 与 $\mathbf{F}$ 拼接起来，并输入到卷积块中，从而获得两个分支的预测结果 $\mathbf{R}$。

由于我们的目标是根据其他方法获得的预测 $\mathbf{P}'$ 来优化细小部分的结果，因此我们保留了 $\mathbf{P}'$ 的结果用于主体部分，仅更改细小结构的部分。

两个分支的最终预测结果 $\mathbf{P}$ 可以用如下公式表示：

$$\mathbf{P}_\text{branch} = \mathbf{M}_\text{branch} \times \mathbf{R}_\text{branch} + (1 - \mathbf{M}_\text{branch}) \times \mathbf{P}', \tag{9}$$

其中，`branch` 指的是 `local` 或者 `global`。

4.4 损失函数

对于二值分割任务，通常采用二值交叉熵（BCE），权重映射 $\mathbf{W}$ 作为损失函数，其可以表示为：

$$\ell(\mathbf{P}, \mathbf{G}, \mathbf{W}) = -\frac{1}{N}\sum \mathbf{W}\times(\mathbf{G}\times\log(\mathbf{P})+(1-\mathbf{G})\times\log(1-\mathbf{P})), \tag{10}$$

其中，$\mathbf{P}$ 指的是预测掩膜的概率，$\mathbf{G}$ 指的是由 1 和 0 组成的标签，N 是像素的总数目。

权重图谱用于强制网络专注于细小部分的分割质量。在细小相似性模块中，我们采用 $\ell(\mathbf{P}_{\text{line}}, \mathbf{G}_{\text{thin}}, \mathbf{L})$ 来监督在线预测。对于分割细化模块中的细化结果，权重图谱 $\tilde{\mathbf{W}}$ 可以使用如下公式计算：

$$\tilde{\mathbf{W}}_{\text{branch}} = 1 - \mathbf{G}_{\text{non-thin}} + \mathbf{M}_{\text{branch}}, \tag{11}$$

对于监督两个分支的最终输出的损失函数，我们采用传统的 BCE 损失。标签与每个分支的差异可以使用如下公式计算：

$$\tilde{\mathbf{G}}_{\text{branch}} = \begin{cases} \mathbf{P}' \times (1 - \mathbf{E}^*) + \mathbf{G} \times \mathbf{E}^*, & \text{branch} = \text{local} \\ \mathbf{P}' \times (1 - \mathbf{E}) + \mathbf{G} \times \mathbf{E}, & \text{branch} = \text{global} \end{cases}, \tag{12}$$

其中，$\mathbf{E}^*$ 指的是选中的细小部分的扩展区域，$\mathbf{E}$ 指的是所有的扩展区域。

因此，两个分支的损失可以用如下公式计算：

$$\mathcal{L}_{\text{branch}} = \ell(\mathbf{R}_{\text{branch}}, \mathbf{G}, \tilde{\mathbf{W}}_{\text{branch}}) + \ell(\mathbf{P}_{\text{branch}}, \tilde{\mathbf{G}}_{\text{branch}}, \mathbf{A}), \tag{13}$$

其中，$\mathbf{A}$ 指的是一个全 1 矩阵。

总损失 $\mathcal{L}_{\text{total}}$ 可以用如下公式表示：

$$\mathcal{L}_{\text{total}} = \alpha\mathcal{L}_{\text{line}} + \beta\mathcal{L}_{\text{local}} + \gamma\mathcal{L}_{\text{global}}, \tag{14}$$

其中，$\mathcal{L}_{\text{local}}$ 和 $\mathcal{L}_{\text{global}}$ 是通过给 `branch` 赋予 `local` 或者 `global` 并依据 Equ. (13) 计算得到的。

在我们的实验中，我们对切割线的分割施加了硬约束。因此，α、β 和 γ 分别被设置为 10、1 和 1.

5 实验

5.1 设置

数据集 我们采用以下数据集进行实验：

- **ThinObject-5K [24]：** 该数据集分别包含 4748、500 和 500，用于训练、验证和测试。该数据集中的所有图像都是通过在背景图像上合成前景对象而合成的。在本文中，我们在训练集上训练我们的模型，并在测试集上对其进行评估。
- **HRSOD [44]：** 该数据集最初是为显着目标检测任务提出的。正如之前的工作 [24] 所做的那样，我们使用相同的 287 张图像进行评估，其中包括 305 个带有细小部分的物体。
- **COIFT [33, 34]：** 遵循之前的工作 [24]，该数据集包含 280 张图像，它结合了 [33, 33] 中的 3 个鸟类和昆虫数据集。需要注意的是，此数据集中的图像的分辨率比其他两个数据集低得多。

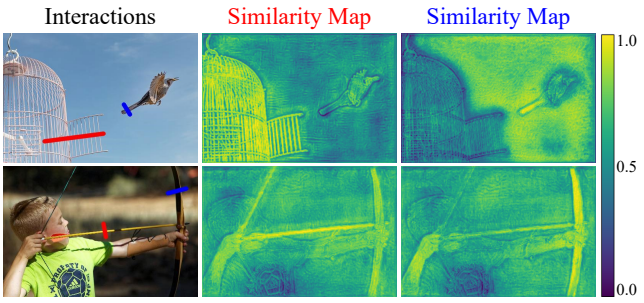

图 5: 根据切割线得到的不同的相似性图谱的展示。左边和右边的相似性图谱分别对应红线和蓝线。相似性图谱的激活区域与交互的细小部分高度相关。

评价指标 由于我们的方法是专门针对细小物体或部分提出的，并且属于细小区域的像素数量通常非常小，因此无法在交并比（Intersection over Union，IoU）指标上明显反映改进。遵循 [24]，我们在扩展的细小部分区域 E（详见 Sec. 3.2）上采用细小的 IoU 分数（IoU_{thin}），以更好地反映分割性能。我们首先通过 $\mathbf{P}_{\text{thin}} = \mathbf{P} \times \mathbf{E}$ 获得细小部分的预测。因此，IoU_{thin} 可以表示为：

$$\text{IoU}_{\text{thin}} = \frac{\sum(\mathbf{P}_{\text{thin}} \cap \mathbf{G}_{\text{thin}})}{\sum(\mathbf{P}_{\text{thin}} \cup \mathbf{G}_{\text{thin}})}. \tag{15}$$

此外，我们还采用边界度量 $\mathcal{F}$ [36] 来评估分割边缘的质量。我们通过形态学算子计算出在 $c(\mathbf{P}_{\text{thin}})$ 和 $c(\mathbf{G}_{\text{thin}})$ 的轮廓点之间的基于轮廓的准确率和召回率 P_c 和 R_c。$\mathcal{F}_{\text{thin}}$ 可以用如下公式表示：

$$\mathcal{F}_{\text{thin}} = \frac{2P_cR_c}{P_c + R_c}. \tag{16}$$

对于这两个指标，值越大，表明模型的性能越好。

实现细节 本文实验中使用的 ResNet [12] 和 HRNet [40] 分别是在 ImageNet [9] 上预训练过的 ResNet-50 和 HRNet-18。我们在 ThinObject-5K 的训练集上训练我们的 KnifeCut。训练过程持续 30 个轮次，批量尺寸设置为 5。对

表 1: KnifeCut 的核心消融研究。我们使用指标 IoU_{thin} 和 $\mathcal{F}_{thin}$ 来评价细小部分的分割。符号↑表示指标值越高，模型的性能越好。'A/B' 指的是采用 ResNet 和 HRNet 作为特征提取器。$G_{non\text{-}thin}$ 被采用作为预分割。'SM' 和 'DM' 分别指的是相似性模块和距离图谱。符号 → 表示取代操作。

#	Candidate	ThinObject5K		HRSOD		COIFT	
		IoU_{thin}↑	$\mathcal{F}_{thin}$↑	IoU_{thin}↑	$\mathcal{F}_{thin}$↑	IoU_{thin}↑	$\mathcal{F}_{thin}$↑
	Pre-segmentation ($G_{non\text{-}thin}$)	0.000	0.000	0.000	0.000	0.000	0.000
local	KnifeCut	0.637/0.661	0.776/0.785	0.400/0.413	0.674/0.659	0.448/0.493	0.667/0.713
	KnifeCut (w/o SM)	0.604/0.655	0.738/0.778	0.371/0.403	0.629/0.659	0.416/0.482	0.640/0.702
	KnifeCut (w/o SM & DM)	0.554/0.618	0.692/0.749	0.164/0.206	0.283/0.339	0.279/0.403	0.447/0.602
	KnifeCut (Line → Click)	0.584/0.615	0.692/0.716	0.361/0.379	0.575/0.578	0.386/0.439	0.524/0.578
global	KnifeCut	0.798/0.821	0.926/0.932	0.497/0.504	0.809/0.792	0.671/0.688	0.903/0.922
	KnifeCut (w/o SM)	0.780/0.803	0.910/0.915	0.285/0.297	0.497/0.488	0.600/0.628	0.838/0.850
	KnifeCut (Line → Click)	0.793/0.822	0.916/0.931	0.489/0.498	0.788/0.779	0.667/0.690	0.894/0.918

于每个轮次，我们采用初始学习率为 7×10^{-3}、γ 为 0.9 的指数学习率衰减策略。我们采用了 momentum 值为 0.9、权重衰减为 5×10^{-4} 的随机梯度下降（Stochastic Gradient Descent，SGD）作为参数优化器。

训练阶段，我们采用随机翻转来增强数据。为了帮助相似性模块识别细小部分间的差异，我们还采用了一种随机粘贴策略，详见 Sec. 5.2。对于测试，Sec. 3.2章节引入了模拟算法。值得注意的是，我们的评估中包括后处理以及根据阈值对精炼掩膜进行二值化。

速度分析 我们在单个 NVIDIA Titan XP GPU 上测试 KnifeCut 的推理时间。采用 HRNet、ResNet-based Deeplab v3+ 作为特征提取器时，处理一张分辨率为 1024×1024 的图片分别需要花费 0.057 秒和 0.281 秒。这两个速度都足以满足实时推理的需要。

5.2 消融实验

我们开展消融实验来证明局部和全局细化所采用的每个模块的有效性。实验结果如表 Tab. 1所示。我们使用 $G_{non\text{-}thin}$ 作为我们预分割的掩膜，并选择 IoU_{thin} 和 $\mathcal{F}_{thin}$ 作为评估指标。这些模块将逐渐从 KnifeCut 中移除。此外，我们用位于细小部分处的一个点击替换将切割线来展示我们交互模式的优越性。基于 HRNet 和 ResNet 的两个实验都将在 ThinObject-5K、HRSOD 和 COIFT 三个数据集上进行。

相似性模块 我们首先移除局部和全局分割模块的相似性模块，实验结果如表 Tab. 1所示。对于局部分支，以 ResNet 为特征提取器的 KnifeCut 的 IoU_{thin} 和 $\mathcal{F}_{thin}$ 指标在三个数据集上大约减少了 3%。但是，对于以 HRNet 为特征提取器的 KnifeCut，相似度图谱带来了微弱的有限提升。这可能是因为 HRNet 保持了高分辨率的特征，而 ResNet 在下采样过程中丢失了一些细小部分的信息，所以细小部分的激活产生了更大的影响。因此，这一实验结果表明相似度图谱有助于网络更好地识别细小部分，特别是对于那些具有低分辨率特征的网络。

全局分割细化的情况则是更加糟糕。一方面，较差的性能证明了相似度图在全局细化中的重要作用。在图像上激活的所有相似的细小部分，网络可以准确地估计细小部分的范围并专注于对其进行细化，这符合我们的预期。另一方面，模型性能不同程度的恶化可以归因于数据集分布的不同。HRSOD 由环境复杂的

真实图像组成，因此增加了分割难度；而 ThinObject-5K test 的数据分布与我们的训练集相同，并且仅包含具有明显边界的合成图像。因此，对 HRSOD 的改进更好地说明了相似度图在实际应用中的必要性。

相似度图谱的另一个不能被统计反映的优点是可以区分属于不同物体的细小部分。无论其特征如何，激活所有细小部分会使网络混淆要细化的位置，并可能将它们全部分割，这在大多数情况下会违反用户的意图。但是，受限于训练集，很少有不同的细物体同时出现。为了使网络更好地处理这种情况，我们在训练过程中采用了随机粘贴策略。将随机选择两张合适的图像粘贴在一起并输入到网络中，而只分割一个细小物体。实验结果如 Fig. 5 所示。可以看到，在切割笼子的时候，只有笼子会被激活，鸟尾巴也是如此。第二张图片也表现不错，弓和箭根据切割线分别激活。需要注意的是，由于相似度图谱独立于预分割的图谱，我们没有在 Fig. 5 中显示它。

线性距离图谱 对于局部分割细化模块，我们进一步去除了距离图谱。如 Tab. ??tab: 消融} 所示，模型的性能在三个数据集不同程度地出现下降。在不同数据集上下降不同的原因与相似性模块相同。因此，这一实验也证明了距离图谱在局部细化中的主导作用。在其指导下，网络知道细小部分的局部范围，并专注于标记区域的细化。

切割线 作为 KnifeCut 的一项重要贡献，我们还开展了交互模式的消融研究。由于我们将点击视为线条的退化，因此我们在 KnifeCut 中用一次点击替换切割线。遵循基于点击的交互方法的惯例 [43]，我们将点击放置在最大细错误区域的中心。实验结果如 Tab. 1 所示。可以看出，模型在 $\mathrm{IoU}_{\mathrm{thin}}$ 和 $\mathcal{F}_{\mathrm{thin}}$ 两个指标上的性能都在一定程度上变差了。与点击相比，切割线更容易穿过多个细小的部分，而不仅仅是一个。由于切割线穿过背景和前景，因此它还可以提供对比的先验。此外，绘制切割线对用户来说方便直观，对大多数设备，例如鼠标、触摸板和移动设备，也都很友好；而对于点击来说，仔细瞄准并点击每个细小的部分既费时又费力。

5.3 对比

性能评价 如 Tab. 2 所示，我们在三个数据集，即 ThinObject-5K、HRSOD 和 COIFT 上对比了 KnifeCut 和其他方法的性能。由于我们的方法是专门为细化而设计的，所以我们只使用关于细小部分的评估指标。这些采用的指标已在 Sec. 5.1 中引入。至于常见物体的交互式分割方法，我们选择 f-BRS [38] 和 FCA Net [27]，因为它们的代码维护良好。为了保持这些方法的公平性，我们通过两次点击采用它们的结果作为我们 KnifeCut 的预分割。由于切割线可以被视为两次点击（两个端点），我们通过 4 次点击将我们的结果与他们的结果进行比较。值得注意的是，我们使用 ThinObject-5K 训练集对这些模型进行了微调。可以看到，我们的方法对细小部分的改进对于局部和全局细化都是相当大的。这不仅证明了我们的方法作为细化工具的可行性，而且表明了切割线处理细小部分分割的适用性和有效性。

对于交互式细小物体分割，我们还将 KnifeCut 与 TOS-Net [24] 进行了比较。同样为了控制相同的点击次数，我们使用边界框（对称角位置的两次外部点击）作为交互模式重新训练模型，该模型具有与 TOS-Net 相同的网络结构。模型得到的结果将作为预分割，这样总的交互次数仍然可以控制为 4。可以看到，经过 KnifeCut 细化后，对细小部分的分割得到了改进，并且超过了 TOS-Net。

定性结果 Fig. 6展示了 KnifeCut 适合处理的几种情况。当细小部分未被分割时，KnifeCut 可以细化预分割，从而仅通过一条穿过未分割细小部分的切割线生成出色的预测。例如，标志的左极丢失，用户需要画一条线穿过它。作为局部结果，左极点将在掩膜上完成；而正确的一个也将同时包括在全局结果中。至于更复杂的情况，球拍缺少网的分割，这对于现有的细化工具是无法想象的。幸运的是，即使在这种困难的情况下，KnifeCut 也会采用同样的方式来解决，而无需增加任何时间和考虑。在另一种情况下，细小部分经常太靠近，导致过度分割。我们以鸟的翅膀为例来说明情况。切割过分割的部分，然后在局部分支中细化目标细小部分，同时两翼用于全局结果。梳子是更加复杂的情况，但 KnifeCut 效果良好。值得注意的是，

表 2: 在三个评价数据集上同其他方法比较 IoU_{thin} 和 $\mathcal{F}_{thin}$ 指标。符号 † 和 § 分别指的是局部分支和全局分支的输出。'A/B' 指的是采用 ResNet 和 HRNet 作为特征提取器的结果。

Method	Interaction	ThinObject5K		HRSOD		COIFT	
		$IoU_{thin}\uparrow$	$\mathcal{F}_{thin}\uparrow$	$IoU_{thin}\uparrow$	$\mathcal{F}_{thin}\uparrow$	$IoU_{thin}\uparrow$	$\mathcal{F}_{thin}\uparrow$
FCA-Net [27]	4 guidance clicks	0.645	0.776	0.372	0.618	0.498	0.726
KnifeCut†	2 guidance clicks + 1 line	0.758/0.770	0.887/0.888	0.488/0.501	0.814/0.800	0.601/0.623	0.869/0.888
KnifeCut§	2 guidance clicks + 1 line	0.811/0.826	0.927/0.927	0.519/0.532	0.850/0.843	0.678/0.694	0.932/0.939
f-BRS [38]	4 guidance clicks	0.784	0.872	0.455	0.713	0.631	0.855
KnifeCut†	2 guidance clicks + 1 line	0.812/0.823	0.914/0.916	0.502/0.512	0.823/0.805	0.666/0.679	0.914/0.921
KnifeCut§	2 guidance clicks + 1 line	0.828/0.840	0.929/0.931	0.523/0.535	0.850/0.847	0.692/0.704	0.939/0.943
TOS-Net [24]	4 extreme clicks	0.865	0.938	0.651	0.916	0.764	0.962
KnifeCut†	1 bounding box + 1 line	0.874/0.875	0.944/0.945	0.666/0.668	0.916/0.917	0.772/0.775	0.965/0.966
KnifeCut§	1 bounding box + 1 line	0.873/0.877	0.946/0.946	0.664/0.670	0.917/0.917	0.786/0.793	0.968/0.969

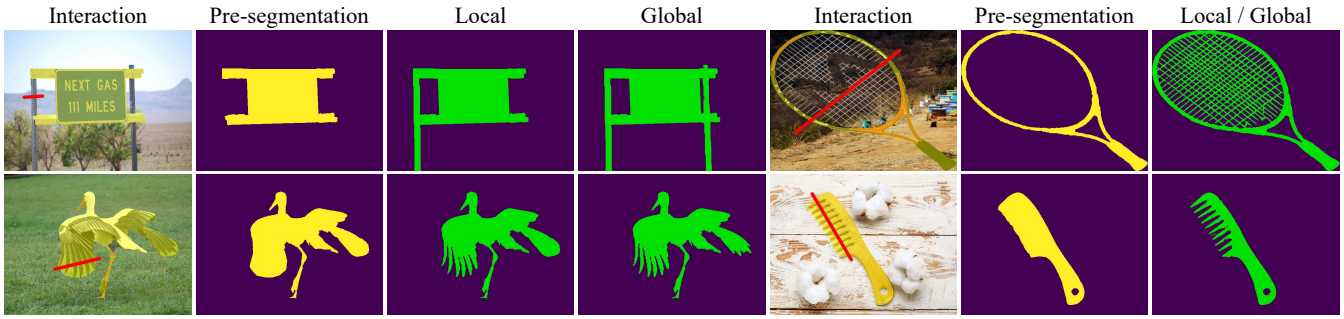

图 6: KnifeCut 的定性结果。预分割和切割线如上图所示。局部和全局细化结果均有展示。'局部/全局' 表示结果几乎完全相同，并且展示了局部结果。

因为对于网和梳子，局部和全局的结果几乎相同，我们只显示局部的。

6 结论

在本文中，为了分割令人烦恼的细小物体，我们提出了一种新颖的交互模式，只需要用户画一条穿过错误标记的细小部分的线。与目前的交互方式相比，用户的负担得到了有效的减轻，并且易于通过鼠标、触摸板和移动设备进行控制。为了充分探索和展示该模式的优越性，我们提出了一种后处理模型 KnifeCut，以进一步细化其他分割方法获得的预分割，特别是对于未分割或过度分割的细小部分。在三个数据集上的实验均证明了 KnifeCut 作为标注工具的优越效果。

ACKNOWLEDGMENTS

本工作由国家重点研发计划项目(NO.2018AAA0100400)、国家自然科学基金（NO.61922046）、中央高校基本科研业务费专项资金、南开大学以及中国博士后科学基金（NO.2021M701780）资助。

REFERENCES

[1] David Acuna, Huan Ling, Amlan Kar, and Sanja Fidler. 2018. Efficient interactive annotation of segmentation datasets with polygon-rnn++. In *CVPR*.

[2] Eirikur Agustsson, Jasper RR Uijlings, and Vittorio Ferrari. 2019. Interactive full image segmentation by considering all regions jointly. In *CVPR*.

[3] Rodrigo Benenson, Stefan Popov, and Vittorio Ferrari. 2019. Large-scale interactive object segmentation with human annotators. In *CVPR*.

[4] Yuri Boykov and Vladimir Kolmogorov. 2004. An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision. *IEEE TPAMI* (2004).

[5] Yuri Y Boykov and M-P Jolly. 2001. Interactive graph cuts for optimal boundary & region segmentation of objects in ND images. In *ICCV*.

[6] Lluis Castrejon, Kaustav Kundu, Raquel Urtasun, and Sanja Fidler. 2017. Annotating object instances with a polygon-rnn. In *CVPR*.

[7] Liang-Chieh Chen, Yukun Zhu, George Papandreou, Florian Schroff, and Hartwig Adam. 2018. Encoder-decoder with atrous separable convolution for semantic image segmentation. In *ECCV*.

[8] Xi Chen, Zhiyan Zhao, Feiwu Yu, Yilei Zhang, and Manni Duan. 2021. Conditional Diffusion for Interactive Segmentation. In *ICCV*.

[9] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. 2009. Imagenet: A large-scale hierarchical image database. In *CVPR*.

[10] Xingping Dong, Jianbing Shen, Ling Shao, and Luc Van Gool. 2015. Sub-Markov random walk for image segmentation. *IEEE TIP* (2015).

[11] Leo Grady. 2006. Random walks for image segmentation. *IEEE TPAMI* (2006).

[12] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *CVPR*.

[13] Yang Hu, Andrea Soltoggio, Russell Lock, and Steve Carter. 2019. A fully convolutional two-stream fusion network for interactive image segmentation. *NN* (2019).

[14] Suyog Dutt Jain and Kristen Grauman. 2019. Click carving: Interactive object segmentation in images and videos with point clicks. *IJCV* (2019).

[15] Won-Dong Jang and Chang-Su Kim. 2019. Interactive Image Segmentation via Backpropagating Refinement Scheme. In *CVPR*.

[16] Stefanie Jegelka and Jeff Bilmes. 2010. *Cooperative cuts for image segmentation.* Technical Report. University of Washington.

[17] Meng Jian and Cheolkon Jung. 2016. Interactive image segmentation using adaptive constraint propagation. *IEEE TIP* (2016).

[18] Tae Hoon Kim, Kyoung Mu Lee, and Sang Uk Lee. 2008. Generative image segmentation using random walks with restart. In *ECCV*.

[19] Theodora Kontogianni, Michael Gygli, Jasper Uijlings, and Vittorio Ferrari. 2020. Continuous adaptation for interactive object segmentation by learning from corrections. In *ECCV*.

[20] Hoang Le, Long Mai, Brian Price, Scott Cohen, Hailin Jin, and Feng Liu. 2018. Interactive boundary prediction for object selection. In *ECCV*.

[21] Yin Li, Jian Sun, Chi-Keung Tang, and Heung-Yeung Shum. 2004. Lazy snapping. *ACM TOG* (2004).

[22] Zhuwen Li, Qifeng Chen, and Vladlen Koltun. 2018. Interactive image segmentation with latent diversity. In *CVPR*.

[23] JunHao Liew, Yunchao Wei, Wei Xiong, Sim-Heng Ong, and Jiashi Feng. 2017. Regional interactive image segmentation networks. In *ICCV*.

[24] Jun Hao Liew, Scott Cohen, Brian Price, Long Mai, and Jiashi Feng. 2021. Deep Interactive Thin Object Selection. In *WACV*.

[25] Jun Hao Liew, Scott Cohen, Brian Price, Long Mai, Sim-Heng Ong, and Jiashi Feng. 2019. Multiseg: Semantically meaningful, scale-diverse segmentations from minimal user input. In *ICCV*.

[26] Zheng Lin, Zheng-Peng Duan, Zhao Zhang, Chun-Le Guo, and Ming-Ming Cheng. 2022. FocusCut: Diving Into a Focus View in Interactive Segmentation. In *CVPR*.

[27] Zheng Lin, Zhao Zhang, Lin-Zhuo Chen, Ming-Ming Cheng, and Shao-Ping Lu. 2020. Interactive image segmentation with first click attention. In *CVPR*.

[28] Zheng Lin, Zhao Zhang, Ling-Hao Han, and Shao-Ping Lu. 2022. Multi-Mode Interactive Image Segmentation. In *ACM MM*.

[29] Zheng Lin, Zhao Zhang, Zi-Yue Zhu, Deng-Ping Fan, and Xia-Lei Liu. 2022. Sequential Interactive Image Segmentation. *CVMJ* (2022).

[30] Huan Ling, Jun Gao, Amlan Kar, Wenzheng Chen, and Sanja Fidler. 2019. Fast interactive object annotation with curve-gcn. In *CVPR*.

[31] Soumajit Majumder and Angela Yao. 2019. Content-Aware Multi-Level Guidance for Interactive Instance Segmentation. In *CVPR*.

[32] Kevis-Kokitsi Maninis, Sergi Caelles, Jordi Pont-Tuset, and Luc Van Gool. 2018. Deep extreme cut: From extreme points to object segmentation. In *CVPR*.

[33] Lucy AC Mansilla and Paulo AV Miranda. 2016. Oriented image foresting transform segmentation: Connectivity constraints with adjustable width. In *SIGGRAPH*.

[34] Lucy AC Mansilla, Paulo AV Miranda, and Fábio AM Cappabianco. 2016. Oriented image foresting transform segmentation with connectivity constraints. In *ICIP*.

[35] Eric N Mortensen and William A Barrett. 1995. Intelligent scissors for image composition. In *SIGGRAPH*.

[36] Federico Perazzi, Jordi Pont-Tuset, Brian McWilliams, Luc Van Gool, Markus Gross, and Alexander Sorkine-Hornung. 2016. A benchmark dataset and evaluation methodology for video object segmentation. In *CVPR*.

[37] Carsten Rother, Vladimir Kolmogorov, and Andrew Blake. 2004. Grabcut: Interactive foreground extraction using iterated graph cuts. *ACM TOG* (2004).

[38] Konstantin Sofiiuk, Ilia Petrov, Olga Barinova, and Anton Konushin. 2020. f-BRS: Rethinking backpropagating refinement for interactive segmentation. In *CVPR*.

[39] Sara Vicente, Vladimir Kolmogorov, and Carsten Rother. 2008. Graph cut based image segmentation with connectivity priors. In *CVPR*.

[40] Jingdong Wang, Ke Sun, Tianheng Cheng, Borui Jiang, Chaorui Deng, Yang Zhao, Dong Liu, Yadong Mu, Mingkui Tan, Xinggang Wang, et al. 2020. Deep high-resolution representation learning for visual recognition. *IEEE TPAMI* (2020).

[41] Tao Wang, Zexuan Ji, Quansen Sun, Qiang Chen, Qi Ge, and Jian Yang. 2018. Diffusive likelihood for interactive image segmentation. *PR* (2018).

[42] Tao Wang, Jian Yang, Zexuan Ji, and Quansen Sun. 2018. Probabilistic diffusion for interactive image segmentation. *IEEE TIP* (2018).

[43] Ning Xu, Brian Price, Scott Cohen, Jimei Yang, and Thomas S Huang. 2016. Deep interactive object selection. In *CVPR*.

[44] Yi Zeng, Pingping Zhang, Jianming Zhang, Zhe Lin, and Huchuan Lu. 2019. Towards High-Resolution Salient Object Detection. In *ICCV*.

[45] Shiyin Zhang, Jun Hao Liew, Yunchao Wei, Shikui Wei, and Yao Zhao. 2020. Interactive object segmentation with inside-outside guidance. In *CVPR*.

[46] Tongjie Y Zhang and Ching Y. Suen. 1984. A fast parallel algorithm for thinning digital patterns. *Commun. ACM* (1984).

[47] Yibiao Zhao, Song-Chun Zhu, and Siwei Luo. 2010. Co3 for ultra-fast and accurate interactive segmentation. In *ACM MM*.