算法基础大作业

PB21010456 杨隽🄵

2024 年 6 月 20 日

1 问题描述

图像处理中的一些问题可转换为求解一个最优化问题，当目标优化函数满足某些性质时，可通过最大流最小割算法来求解该问题。基于学习的最大流算法，实现以下两个图像处理作业：

1. 实现文章 “Graphcut Textures: Image and Video Synthesis Using Graph Cuts”，基于 graphcut 的图像纹理合成.

2. 实现” GrabCut -Interactive Foreground Extraction using Iterated Graph Cuts”，基于 graphct 的图像分割.

2 实验算法

2.1 纹理合成

2.1.1 简介

在第一篇文章的方法中，来自样本图像或视频的子块被转换并复制到输出，然后沿着最佳接缝缝合在一起，以生成新的（更大的）输出。与其他技术相比，块的大小不是先验地选择的，而是使用图切技术来确定输入和输出纹理之间任何给定偏移量的最佳 patch 区域。与动态规划不同，这里图割技术适用于任何维度的接缝优化。除了常规的图像合成外，还专门在 2D 和 3D 中探索它来执行视频纹理合成。文章提出了近似偏移搜索技术，可以很好地与提出的补丁大小优化相结合。文章展示了合成规则、随机和自然图像和视频的结果。还演示了如何使用这种方法交互式地合并不同的图像以生成新的场景。

2.1.2 具体内容

算法主要分为两步。第一步，初始化放置部分原始图像中的局部图像放置到输出图像上。第二步，通过 graphcut 算法来寻找局部图像和原输出图像中已有像素之间的像素缝隙，并确认输出图像中真正变为局部图像块的像素，和保持不变的部分。

始化的目的是在已有的输出图像中插入一个新的图像局部块，这两个图像之间会有一些重叠区域。我们的目标是尽量使接缝不显眼。为此，可以让接缝左侧主要由左侧图像 A 提供像素，而接缝右侧主要由右侧图像 B 提供像素。假设我们的输出是一个图，可以给它添加两个端点 A 和 B，并将每个像素视为一个节点。我们考虑从这个图中移除一些边，以使被移除的边的总代价最小，这

实际上是一个经典的最大流最小割问题。对于重叠区域中的相邻像素 s 和 t，我们定义如下的连接代价：

$$M(s,t,A,B) = \|A(s) - B(s)\|_2 + \|A(t) - B(t)\|_2$$

那么所求问题约化为

$$\Sigma_{s,t} M(s,t,A,B)$$

用 Fordfolkson 算法很容易实现。

对于多个图的情形，假设在输出纹理中已经放置了几个 patch，并且我们希望在多个 patch 已经相遇的区域放置一个新补丁。在旧 patch 之间的边界上有可能出现可见的接缝，我们可以使用我们在铺设这些 patch 时解决的图切问题的弧线成本来测量这一点。此时为了得到令人满意的合成结果，应该保存上一轮接缝的分割代价，并且将新计算的分割代价加入到老的分割代价中形成总体代价，根据总体代价来确认新的分割位置。

因此此时的算法如下：首先在分割线中添加 SeamNode，然后把缝节点与新的 PatchB 相连，并且分配相应的代价。相应的代价根据下面的三个规则来确认：（如图 1 所示）

1. 规则 1：如果 M(1,4,A1,B) 经过图割运算后任然是缝合线，那么说明原有的缝合线合理，缝合线两边至少有一个像素被替换为新 Patch 的像素。

2. 规则 2：如果 M(1,4,A1,B) 经过图割运算后任然不属于 min-cut，那么说明原有的接缝会消失，这意味着缝合线两边的像素都会维持老像素或者被替换成新像素。

3. 规则 3：如果与某个缝节点以及相邻像素间的连接被切割，那么需要将切割代价与原有代价，形成新的切割代价。

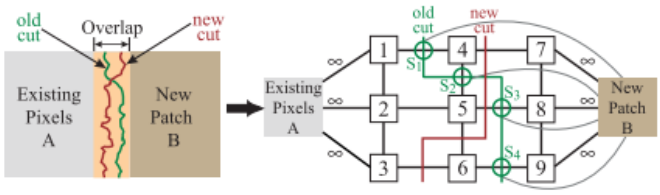

Figure 3: (Left) Finding the best new cut (red) with an old seam (green) already present. (Right) Graph formulation with old seams present. Nodes s_1 to s_4 and their arcs to **B** encode the cost of the old seam.

图 1: 多个图的缝合

对于 Patch 的选择，主要有几下三种方法：

1. 随机放置: 在这种方法中，新块 (整个输入图像) 被转换到随机偏移位置。图切算法从这个 patch 中选择一块放置到输出图像中，然后我们重复这个过程。这是最快的合成方法，并且对随机纹理有很好的效果。

2. 整块匹配: 为了解释输入和输出之间的部分重叠，我们用重叠区域的面积将平方差和 (SSD) 代价归一化。

3. 子块匹配: 这是所有 patch 选择技术中最通用的。它也是随机纹理以及涉及纹理运动 (如水波和烟雾) 的视频序列的最佳方法。这些序列中的运动在空间上是非结构化的，图像的不同区域表

现出不同的运动; 然而，运动本身是结构化的，因为它是局部连贯的。在子块匹配中，我们首先在输出纹理中选择一个小的子块 (通常比输入纹理小得多)。在我们的实现中，这个输出子块与我们想要放置块的错误区域相同或略大。现在我们在输入纹理中寻找与输出子块匹配的子块。同样地，我们寻找输入的翻译，使得与输出子补丁重叠的部分很好地匹配——只有那些允许输入与输出子补丁完全重叠的翻译才会被考虑。

本文提出的图切技术的主要优势之一是它允许直接扩展到视频合成。将视频序列视为体素的 3D 集合，其中一个轴是时间。视频中的 patch 是视频的整个 3D 时空块，可以放置在 3D(时空) 体的任何位置。因此，视频纹理合成同样需要图像纹理合成的两个步骤，即 patch 放置和接缝查找。与 2D 纹理相似，视频的 patch 选择方法也必须根据视频的类型来选择。一些视频序列只显示时间平稳性，而另一些则显示空间和时间的平稳性。对于那些只显示时间平稳性的，在所有三个维度 (空间和时间) 搜索 patch 翻译是不必要的。我们可以将我们的搜索限制在时间上的 patch 偏移量，也就是说，我们只寻找 patch 的时间平移。然而，对于空间和时间静止的视频，我们会在所有三个维度上进行搜索。我们现在描述一些我们的视频合成结果。我们首先展示了一些时间静止纹理的例子，在这些纹理中，我们发现了视频过渡的时空接缝。这些结果改进了视频纹理，并优于动态纹理。

2.1.3 总结

总的来说，本文通过提供以下优点，显著提高了纹理合成的技术水平:(a) 对将创建接缝的区域的形状没有限制，(b) 考虑旧的接缝成本，(c) 易于推广到视频接缝表面的创建，(d) 添加了约束的简单方法。

2.2 图像切割

2.2.1 简介

高效、交互式的静态图像前景/背景分割问题在图像编辑中具有重要的现实意义。经典的图像分割工具使用纹理 (颜色) 信息，例如：Magic Wand、或边缘 (对比度) 信息如 Intelligent Scissors。最近，一种基于图切割的优化方法成功地结合了这两种信息。本文从三个方面对 graph-cut 方法进行扩展。第一：我们开发了一个更强大的迭代优化版本；第二：迭代算法的强大功能大大简化了给定质量结果所需的用户交互；第三：提出了一种稳健的 border matting 算法，可以同时估计物体边缘的 alpha-matte 和前景像素的颜色。我们证明，对于中等难度的例子，提出的方法优于竞争工具。

2.2.2 具体内容

图像是由 RGB 颜色空间中的像素 z_n 组成的。由于构建足够的颜色空间直方图是不切实际的，我们遵循已经用于软分割的实践的有 K 个高斯分量的全协方差 GMM，一个用于背景，一个用于目标区域。对于 N 个图像的像素，取向量 $\boldsymbol{k} = (k_1, \cdots, k_n, \cdots, k_N)$，其中 $k_n \in 1, \cdots, K$。，由此我们可以知道整个图像的 Gibbs 能量为

$$E(\alpha, \boldsymbol{k}, \theta, z) = U(\alpha, \boldsymbol{k}, \theta, z) + V(\alpha, z)$$

其中

$$U(\alpha, \boldsymbol{k}, \theta, z) = \Sigma_n D(\alpha_n, k_n, \theta, z_n)$$

$$D(\alpha_n, k_n, \theta, z_n) = -\log \pi(\alpha_n, k_n) + \frac{1}{2}\log det\Sigma(\alpha_n, k_n) + \frac{1}{2}(z_n - \mu(\alpha_n, k_n))^T \Sigma(\alpha_n, k_n)^{-1}(z_n - \mu(\alpha_n, k_n))$$

$$\theta = \{\pi(\alpha, k), \mu(\alpha, k), \Sigma(\alpha, k), \alpha = 0, 1, k = 1, \cdots, K\}$$

U 表示区域项。计算 Gibbs 能量时，只要确认 GMM 的三个参数 $\pi(\alpha, k), \mu(\alpha, k), \Sigma(\alpha, k)$, 就可以直接根据像素的颜色值来确认它属于目标和背景的 GMM, 从而知道它分别位于上述两个区域的概率。

我们又知道

$$V(\alpha, z) = \gamma\Sigma_{(m,n)\in C}[\alpha_n \neq \alpha_m] \exp((-\beta)\|z_n - z_m\|^2)$$

训练发现取 $\gamma = 50$ 是一个比较优秀的数值，从而 V 我们也可以很便捷的求出。于是就可以根据上面的公式算出图像的能量，就可以开始进行分割了。

GrabCut 中新的能量最小化方案以迭代方式工作，取代了这种一次性算法。这种方法的优点是可以自动细化不透明度 α，因为新标记的像素来自初始 trimap 的 T_u 区域，用于细化颜色 GMM 参数 θ。

初始化：用户先直接框选一个目标得到初始的 T，这时认为方框外的像素都是背景 T_B，方框内的元素 T_U 为可能是目标的像素。对 T_B 元素初始化 $\alpha_n = 0$，对 T_u 元素初始化 $\alpha_n = 1$。

再进行如下三步迭代：

1. 对每个像素分配高斯分量 $k_n = \arg\max_{k_n} D_n(\alpha_n, k_n, \theta, z_n)$
2. 从数据中学习得到 GMM 的参数 $\theta = \arg\max_\theta U(\alpha, k, \theta, z)$
3. 求解最大流最小割问题 $\min_{\alpha_n : n\in T_u} \min_k E(\alpha, k, \theta, z)$

重复上述步骤直到算法收敛即可。综上，我们相当于把图片中的每个像素视为图中的一个节点，然后添加 source 节点和 sink 节点，所有前景连接到 source 节点，背景连接到 sink 节点，连接到源结点或者结束节点的权重根据 GMM 的概率分布来定义，像素之间的权重根据相似相似度来定义，越不相似权重越低。这样的最大流最小割算法可以给出一个近似的前景背景分割。

与单次图割方法相比，GrabCut 中迭代最小化的附加能力如何显著减少完成分割任务所需的用户交互量。这在两个方面都很明显。首先，在初始化和优化之后，降低所需的用户编辑程度。其次，初始交互可以更简单，例如通过允许用户进行不完整的标记。

2.2.3 总结

综上所述，本文提出并演示了一种新的前景提取算法，该算法可以在用户不太费力的情况下，为中等难度的图像获得高质量的前景 alpha 图像。该系统结合了硬分割的迭代图切优化与边界抠图，以处理模糊和混合像素的对象边界。

3 实验结果

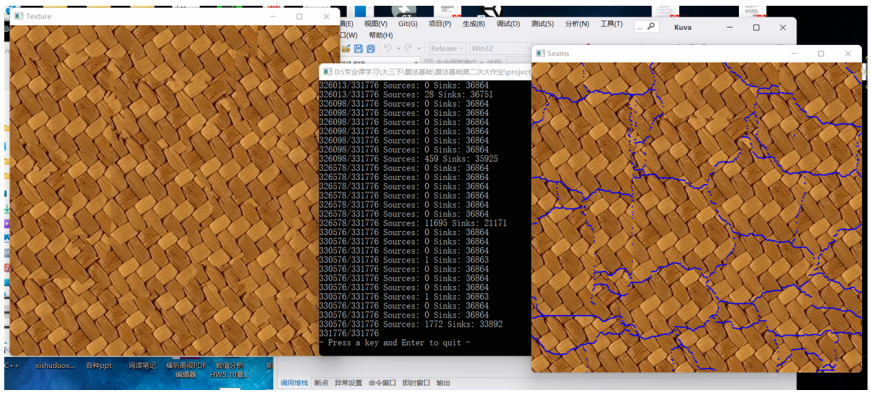

图 2: 纹理合成

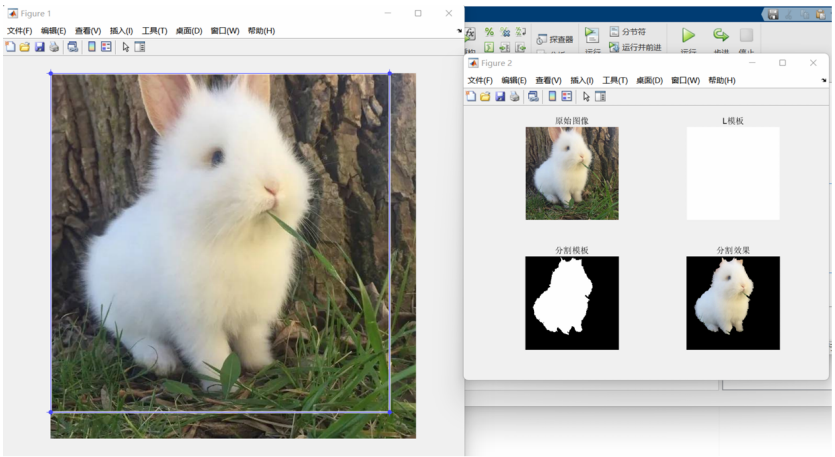

图 3: 图像切割