CS131 Lecture09：图像大小调整和分割

by：斯坦福大学计算机科学系

github: https://github.com/zhaoxiongjun/CS131_notes_zh-CN （包含中英文版课件及相关课程视频）

1 介绍

用于显示图像和视频的设备具有不同的大小和形状。因此，最佳的图像/视频查看配置因设备和屏幕大小而异。因此，图像尺寸调整在计算机视觉中具有重要意义。直观的想法是重新缩放或裁剪原始图像以适应新的设备，但这通常会导致伪影，甚至丢失图像中的重要内容。本课程讨论了在保留重要内容和限制工件的同时调整图像大小的技术。

1.1 问题描述

输入一个大小为 n*m 的图像，并返回一个大小为 n’*m’ 的图像，要使得它很好地代表原始图像。应满足：

1：新图像应符合设备几何约束。
2：新图像应该保留重要的内容和结构。
3：新图像应该具有有限的工件。

1.2 重要性的度量值

1．函数，S：p->[0,1]用于确定图像中哪些部分很重要；然后，可以使用不同的运算符来调整图像的大小。一种解决方案是使用最佳裁剪窗口提取最重要的内容，但这可能导致重要内容丢失。

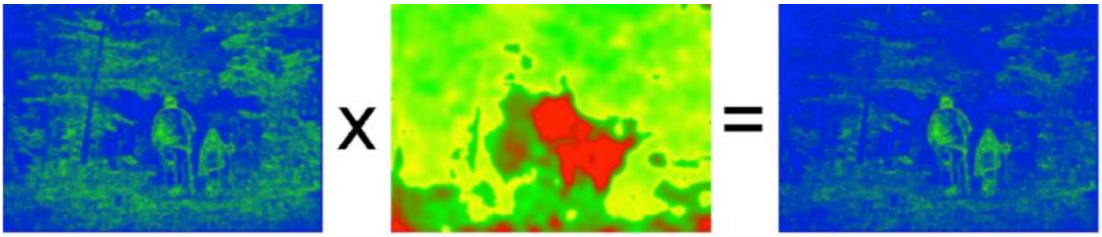


图 1：函数法度量重要性

2. 还有一些更复杂的技术可以测量具有更高重要内容的图像区域；它们包括但不限于注意力模型、眼睛跟踪（注视研究）和面部探测器。

图 2：更复杂的方法测量重要性，比如眼睛跟踪

2 Seam Carving

2.1 基本思想

人类的视觉对图像的边缘更敏感。因此，一个简单但有效的解决方案是从更平滑的区域中删除内容并保留包含边缘的信息更丰富的图像区域；这使用基于梯度的能量函数实现的，定义如下：

$$E(I) = |\frac{\partial}{\partial x}I| + |\frac{\partial}{\partial y}I|$$

因此，不重要的内容，对应像素的能量函数的值较小。

2.2 像素移除

去除不重要的像素有不同的方法，每种方法都会导致不同的视觉效果。下图展示了这类方法的三个示例；前两个（即最佳最小能量像素和行删除）被观察到会对图像质量产生负面影响。另一方面，最后一个（即能量最小的列删除）的效果明显更好，但它仍然会在新图像中造成大量的伪影。下一节将介绍另一种解决方案。

1. 删除能量函数值小的像素。
2. 删除能量函数值最小的行。
3. 删除能量函数最小的列。

图 3:不同的像素去除方法对图像质量的影响

2.3 A seam

1. Seam（线）被定义为像素从上到下（或从左到右）的连接路径。对于从上到下的像素，我们将从每行中精确地选取一个像素。数学定义是:

$$s^x = \{s_i^x\}_{i=1}^n = \{x(i), i\}_{i=1}^n, s.t. \forall i, |x(i) - x(i-1)| \leq 1$$
$$s^y = \{s_j^y\}_{j=1}^m = \{j, y(j)\}_{j=1}^m, s.t. \forall j, |y(j) - y(j-1)| \leq 1$$

2. 最佳接缝是基于像素梯度最小化能量函数。

$$s^* = argmin_s E(s), \quad \text{where} \quad E(I) = |\frac{\partial}{\partial x} I| + |\frac{\partial}{\partial y} I|$$

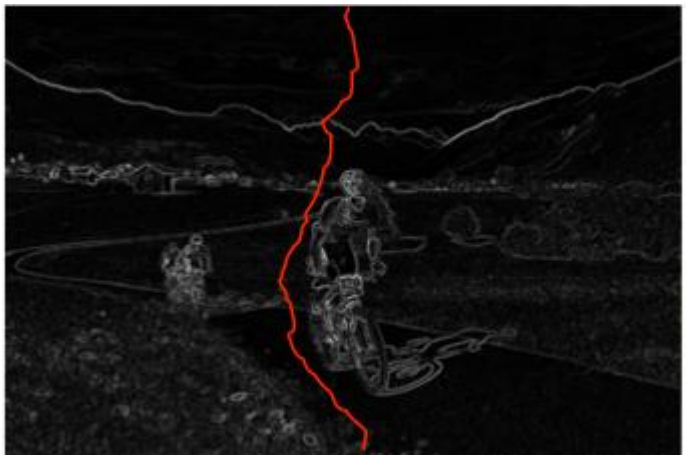


图 4：红线表示能量最小的最佳接缝位置

3. 递归关系可以用来寻找最优的接缝。如果 m（i, j）定义为穿过像素（i, j）接缝的最小能量成本，则递归关系为:

$$M(i,j) = E(i,j) + min(M(i-1,j-1), M(i-1,j), M(i-1,j+1))$$

在原算法中采用 O（snm），S=3 的动态规划，可以有效地解决这一问题。给定能量函数值为：

5	8	12	3
4	2	3	9
7	3	4	2
5	5	7	8

图 5：用于 seam carving 算法的能量函数示例

递归关系如下：

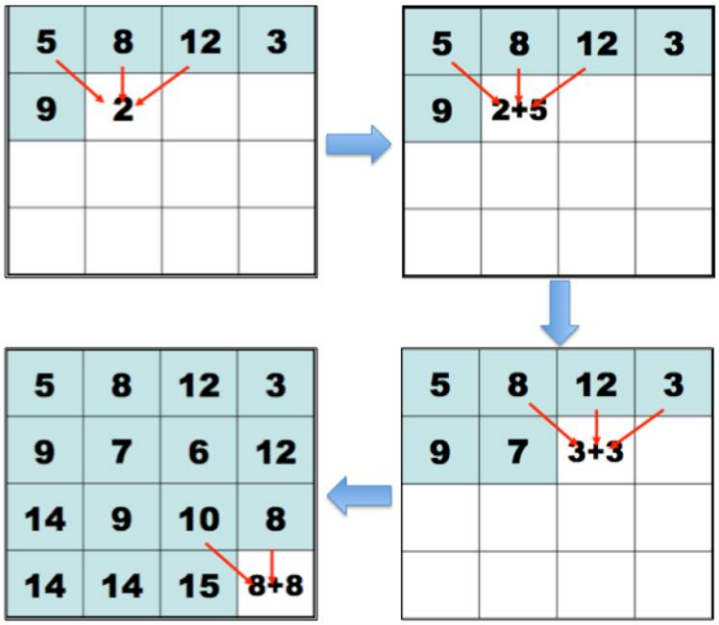


图 6:使用关系递归计算缝成本的过程

4. 为了寻找最佳接缝，引入了回溯法。它从底部具有最低能量值的像素开始，然后向上移动到图像的顶部（即第一个图像行）。

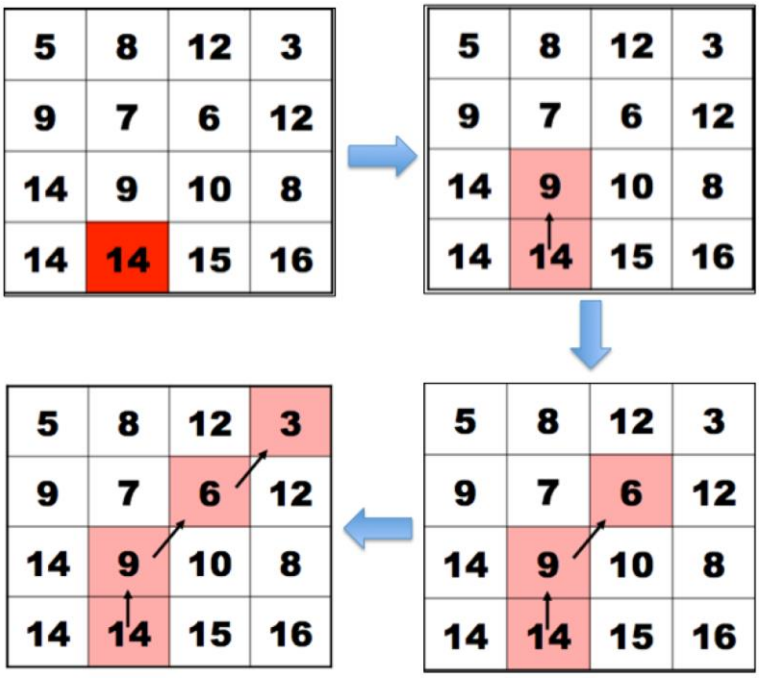


图 7:使用回溯找到最优接缝

2.4 Seam carving 算法

此算法复杂度为 O（（n －n’）mn）。在每轮循环中，E、s 和 im 的更新都需要 O（mn）。对于垂直调整大小，可以对图像进行转置，以便使用相同的算法。

Algorithm 1 Seam-Carving

```
im ← original image of size m × n
n' ← desired image size n'

Do (n-n') times:
    E ← Compute energy map on im
    s ← Find optimal seam in E
    im ← Remove s from im

return im
```

由于算法去除了低能量像素，图像的平均能量会增加。所述的算法可用于修改纵横比，实现物体去除，以及图像尺寸调整。如果翻转图像，则过程相同。调整大小时，需要删除水平和垂直接缝。通过动态编程可以解决两个方向的接缝添加和移除顺序。具体来说，递推关系为：

$$T(r,c) = min(T(r-1,c) + E(s^x(I_{n-r-1\times m-c})), T(r,c-1) + E(s^y(I_{n-r\times m-c-1}))$$

更多信息请参阅论文：https://dl.acm.org/citation.cfm?doid=1276377.1276390
Or 博客：https://blog.csdn.net/hjimce/article/details/44916869

3 先进的 Seam Carving

3.1 图像展开

可以使用类似的方法来增加图像的大小。通过扩展图像中最不重要的区域（如接缝所示），可以在不影响主要内容的情况下增加图像尺寸。一种简单的方法是迭代地查找和复制最低能量层。但是，这为我们提供了如下所示的结果：

图 9:复制的最小能量接缝图像扩张并不是一个好策略

在图像的右侧（图 9），一个接缝被重复复制。这是程序反复检索相同（最小能量）的接缝。一个更有效的实现是一次找到前 k 个接缝，并复制每个接缝：

图 10:使用前 k 个低能量接缝扩张战略得到更有效的图像

如上所述，第二个图像扩展策略显著提高了调整大小的图像的质量。但是，请注意，此方法只能将图像放大 2 倍（因为没有足够的接缝进行复制）。对于非常大的放大，您可以迭代放大 1.4-1.5x。

3.2 多尺寸图像表示

虽然我们已经看到图像大小调整是非常有效的，但它仍然需要大量的计算。在实践中，图像与接缝的表示一起存储，以放弃接缝的重新计算，并简化设备上的图像大小调整。这些接缝表示与图像的维度相同，但它们没有像素强度，而是根据能量值（从最小能量到最高能量）对接缝排序的路径进行编号。为了在预处理步骤中计算这些接缝，必须在每次接缝移除后重新计算图像能量（这是由于成本函数的变化）。有关这种表示的示例，请参见以下内容：

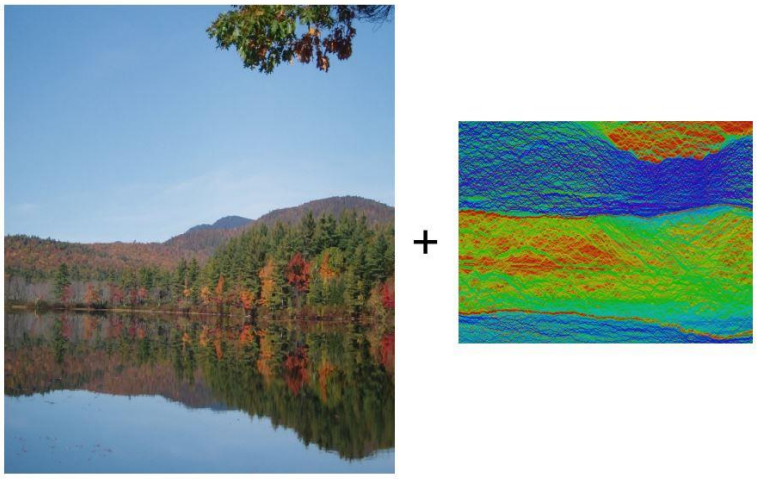


图 11:预计算接缝的例子用于快速图像调整设备上

给定这种表示方式，一个试图从原始图像中删除 k 个接缝的程序可以删除与接缝图像（在右侧）中标记为 1 到 k 的那些对应的像素。

3.3 对象移除

通过用户指定图像的哪个区域以提供高能量或低能量，我们可以使用 seam

carving 来专门保存或删除某些对象。算法专门选择接缝，以便它们通过给定的对象（下面的绿色部分）。

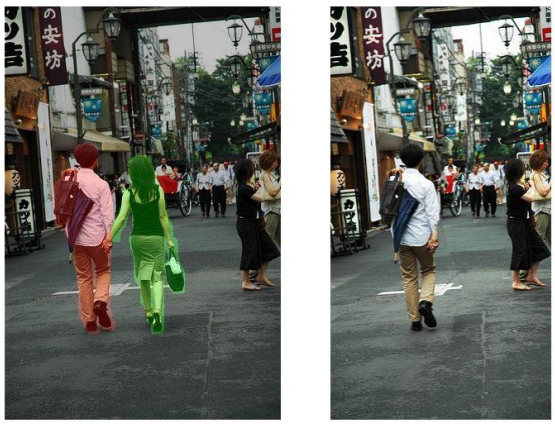


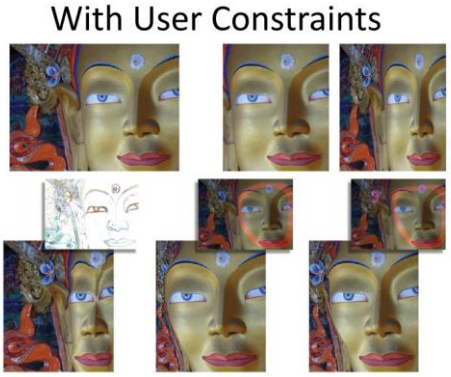
图 12：通过为图像的一部分指定低能量值，seam carving 算法可用于对象移除。

3.4 限制

Seam Carving 是一种有效的图像尺寸调整方法。但是，也有局限性：1）主要局限性是当图像的大小发生剧烈变化时，对有效性的下限和上限；2）在可能是低能量的对象的上下文中无法识别重要特征。让我们来看看下面的图片。

图 13：seam Carving 的局限性

有些平坦平滑的图像区域（即低梯度）对图像很重要，但它们被移除；例如，这包括女性的脸颊和前额。虽然这些区域的能量很低，但它们是人类感知的重要特征，应予以保留。为了解决这些限制，可以修改能量函数以考虑附加信息。例如，人脸检测器可用于识别重要内容（即人脸），或者用户可应用其他约束。



3.5 正向能量

当我们进行 seam carving 时，我们去除了能量最低的像素，并保留了能量最高的像素。因此，平均图像能量增加，可能导致伪影和锯齿状边缘。避免这个问题的一种方法是将焦点放在去除那些将最小能量插入图像的接缝上。这种方法被称为正向能量；我们的原始累计成本矩阵是通过添加来自相应新邻居的正向能量进行修改的，如下图所示。最初引入的方法称为“反向”能量。

$$M(i,j) = P(i,j) + \min \begin{cases} M(i-1,j-1) + C_L(i,j) \\ M(i-1,j) + C_U(i,j), \\ M(i-1,j+1) + C_R(i,j) \end{cases}$$

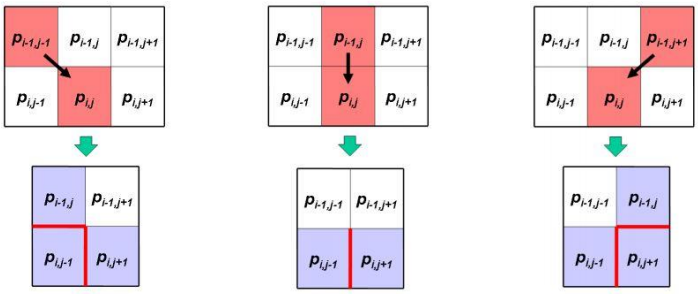


图 15：正向能量计算

下图比较了图像大小调整前后计算的性能。传统的“反向”能量方法导致扳手手柄和植物茎部出现锯齿状边缘。另一方面，“向前”能量方法将增加的能量最小化，并保持边缘的平滑。

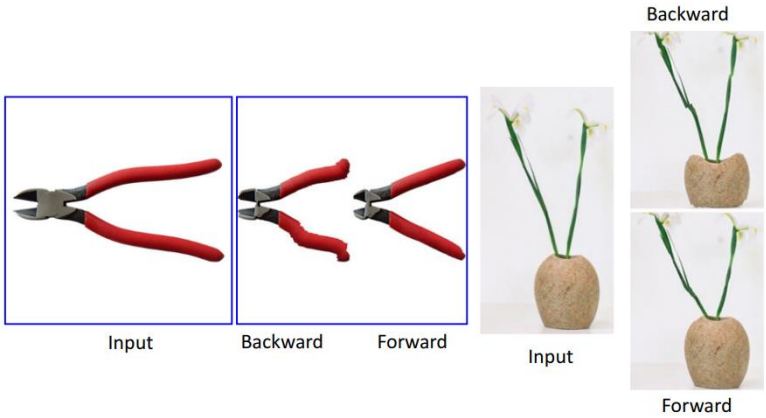


图 16:向前能源 content-aware 图像缩放方法

3.6 Seam-Carving 应用在视频

虽然对 seam carving 的强大功能进行了探索，但如何将其应用于视频仍然是个问题。该算法在视频大小调整方面面临挑战。视频很难调整大小。我们面临两个主要问题。

首先，让我们考虑一个一分钟窗口的电影录制 30 帧。在这段时间内，我们有 1800 帧。如果我们的 seam carving 算法需要一整分钟来处理一个图像，我们需要 30 个小时来完全处理视频。

第二个问题是时间一致性。可以考虑采用直观、朴素的 seam carving 方法独立处理每一个帧。然而，这并不一定保留有关连续帧关系的重要内容。由于人眼对运动特别敏感，如果不考虑图像中的上下文，可能会产生一个外观不佳的视频，并且在帧与帧之间存在明显的失真。跨帧的更改没有一致性。一种更有效的方法是将视频视为三维空间，其中每个垂直平面都是视频中的图像。最低能量的二维接缝可以计算整个视频的长度。这会产生更好的结果，但它仍然面临着限制。

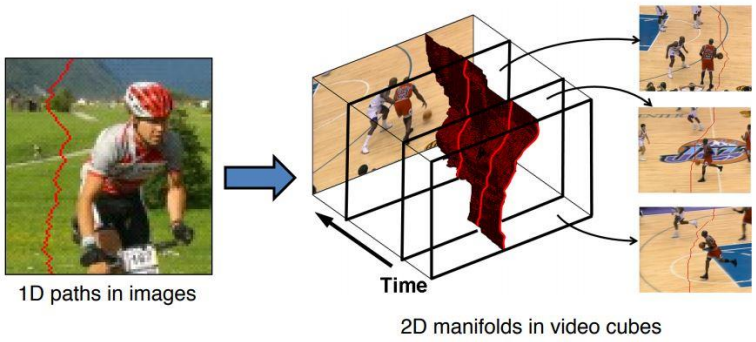


图 17:改进 seam carving 视频

4 分割

在计算机视觉中，我们经常感兴趣的是识别一组像素。我们称之为图像分割问题。人类凭直觉进行图像分割。例如，两个人在看同一个错觉时可能会看到不同的东西，这完全取决于他们的大脑如何分割图像。在下面的图片中，你可能看到斑马，或者你可能看到狮子。

图 18：关于视觉错觉图像分割的问题

图像分割背后的动机之一是将图像分割成连贯的对象。以下是这类细分的两个示例：

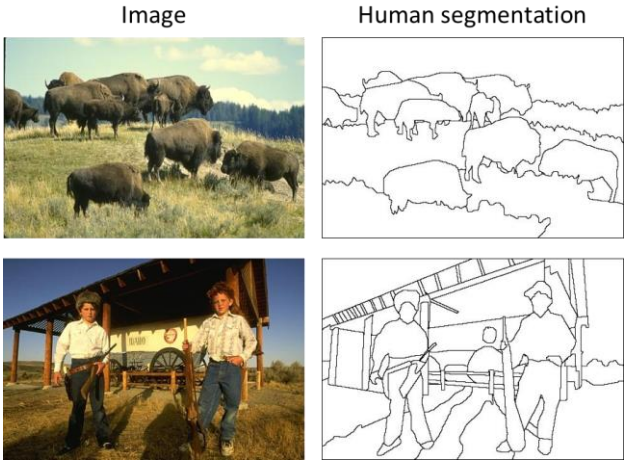


图 19:人类图像分割的例子

我们还可能希望根据附近像素的相似性将图像分割成许多组。我们称这些组为“超级像素”。超级像素允许我们将许多单个像素视为一个簇，从而实现更快的计算。下面是一个由超像素分割的图像示例：

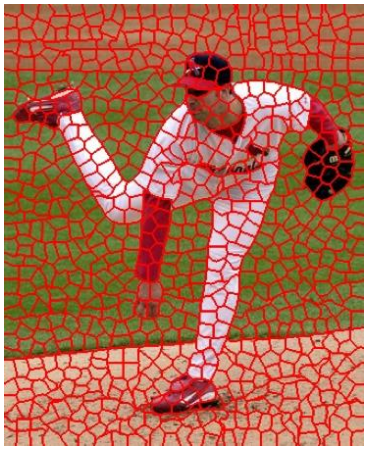


图 20：超像素允许通过聚类像素更快地进行计算

超像素分割和其他形式的分割有助于支持特征。我们可以将像素组视为一个特征，从中获取图像的信息，图像分割也有利于一些常见的照片效果，如背景去除。如果我们能够正确地分割一个图像，我们将只能保留我们想要的组并删除其他组。虽然分割是非常有用的并且有许多实际应用，但是没有一种方法可以分割图像，我们必须比较不同的分割算法来找到我们的最佳解决方案。如果图像的组数很少或过多，则容易出现分割不足和过度分割的情况。然而，即使是适当分割的照片也可能有多个不同的可能分组。

为了解决如何分割图像的问题，我们将分割视为聚类。通过聚类，我们可以将相似的数据点组合在一起，并用一个奇异值来表示它们。这再次有助于我们操纵图像或从中提取特征的能力。但是，我们必须决定几个重要问题：

1. 如何确定两个像素、像素块或图像是否相似？
2. 我们如何根据成对相似性计算一个整体分组？

对于这些问题，聚类算法有不同的答案；它们将在下一节中进行深入讨论。

一般来说，两大类聚类算法是自上而下和自下而上的。如果像素和像素块位于同一视觉实体上，则自顶向下的聚类方法将它们分组在一起。而自下而上的方法将局部相干的像素分组在一起。

我们也可以使用某些人类可识别的视觉模式进行聚类算法。一些示例模式包括将相似的对象分组在一起或使用对称来帮助分割。在某些情况下，我们也可以看到“共同命运”。共同命运意味着一组物体似乎在一起移动，因此它们共享相同的“命运”。下面是骆驼的例子，我们可以根据它们的共同命运来分组。

图 21:共同命运提供了视觉线索的分割问题

我们也可以用这种光学错觉来说明共同命运。这种错觉被称为 Muller-lyer 错觉，它欺骗我们认为底线部分比顶线部分长（如下图），即使它们实际上是相同的长度（忽略四条“小尾巴”）。

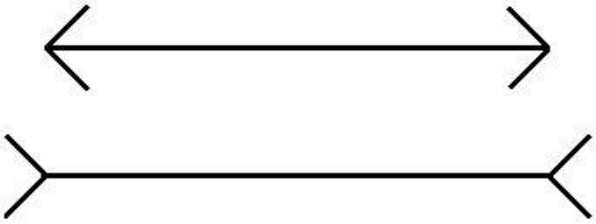


图 22:共同命运导致一种光学错觉

另一种我们可以将对象分组的方法是通过近邻法。将对象与它们看起来距离近的对象分组。例如，在这张图片中，我们可以将前景中的三个人分组在一起。

图 23:距离可以帮助图像分割