数图hw3 coding

2017年10月18日　　13:30

> imshow(img,[]) 中，中括号表示扩展显示数据的范围。
> 因为imshow这个函数默认只能显示0-255的值，所以如果图片数据类型不是0-255的话，显示上会出问题，要么加中括号改变范围，要么自己改数据类型。
>
> 简单地说如果一个图片是uint8，则可以直接显示；而一个图片是double，则要改为uint8后显示，或加中括号。

例 4.15　由小空间模板得到频率域滤波器。

在这个例子中，我们从一个空间模板开始，并说明如何产生相应的频率域滤波器。然后，我们比较用频率域滤波和空间技术得到的滤波结果。当希望比较给定的空间模板相对于一个或多个频率域全滤波器候选者的性能时，或者深入理解模板的性能时，这类分析是很有用的。为简单计，我们使用来自图 3.41 (e) 中的 3×3 Sobel 垂直边缘检测器。图 4.38(a)显示了一幅 600 × 600 像素的图像 $f(x, y)$，我们要对它进行滤波，图4.38 (b) 显示了该图像的谱。

图4.39 (a) 显示了这个 Sobel 模板 $h(x, y)$ (透视图在下面解释)。因为输入图像的大小是 600×600 像素，而滤波器的大小为 3×3，为避免缠绕错误，我们根据式 (4.6-29) 和式 (4.6-30) 将 f 和 h 填充为 602×602 像素大小，Sobel 模板是奇对称的，它被嵌入到一个偶数尺寸的 0 阵列中 (见例 4.10)。为保持这种对称性，我们把 $h(x, y)$的中心放在 602×602 这个填充后的阵列的中心处。对滤波器生成来说，这是一个很重要的方面。如果我们在形成 $h_p(x, y)$的过程中，关于填充后的阵列保持奇对称，则由表 4.1 中的性质 9 可知 $H(u, v)$将完全是虚函数。如我们在本例末尾显示的那样，这将得到用 $h(x, y)$对图像进行空间滤波相同的结果。如果不保持这种对称性，则结果将不再相同。

用以产生 $H(u, v)$的过程是：(1) 用$(-1)^{x+y}$乘以 $h_p(x, y)$以便频率域滤波器“中心化”；(2) 计算 (1) 中结果的正 DFT；(3) 考虑寄生的实部 (我们知道 $H(u, v)$必须是纯虚函数)，将得到的 DFT 的实部置 0；(4) 用$(-1)^{u+v}$乘以该结果。最后一步反过来用$(-1)^{u+v}$乘以 $H(u, v)$，它隐含着 $h(x, y)$被移到 $h_p(x, y)$的中心。图4.39 (a) 显示了 $H(u, v)$的透视图，而图4.39 (b) 以图像的方式显示了 $H(u, v)$。如期望的那样，该函数是奇函数，因而其关于中心反对称。在 4.7.3 节略述的步骤中，函数 $H(u, v)$被用做任何其他的频率域滤波器。

图 4.39 (c) 是在 4.7.3 节略述的步骤中使用刚才得到的滤波器对图 4.38 (a) 中的图像滤波的结果。正如我们对推导出的滤波器所期望那样，边缘被增强了，所有的恒值灰度区域被减小为 0 (为了显示，灰度色调进行了标定)。图4.39 (d) 显示了在 3.6.4 节给出的步骤中使用 $h(x, y)$直接在空间域对同一图像滤波的结果，结果相同。

288 ~ 291

因为在这里我们处理的是离散量，傅里叶变换的计算使用 DFT 算法执行。如果我们选择使用两个函数变换的乘积的 IDFT 来计算空间卷积，则必须考虑在 4.6.3 节中讨论过的周期性问题。我们给出一个关于该问题的一维的例子，然后将结论扩展到两个变量。图 4.28 左边一列执行两个函数 f 和 h 的卷积，它使用式(3.4-2)的一维等式，因为两个函数具有相同的尺寸，该卷积写为

我们将在 4.11 节中讨论计算 DFT 的有效方法。

$$f(x)\bigstar h(x)=\sum_{m=0}^{399}f(x)h(x-m)$$

这个等式与式(4.4-10)相同，但是，对位移 x 的要求是它必须足够大，以便使反转的 h 完全滑过 f。换句话说，该过程包括：(1)关于原点求 h 的镜像(即使 h 旋转 180°)［见图4.28(c)］，(2)以数量 x 平移镜像函数［见图4.28(d)］，(3)对每一个平移的 x 值，计算前面公式右边的全部乘积之和。依据图4.28，这意味着对每个 x 值用图4.28(d)中的函数乘以图4.28(a)的函数。位移 x 的范围是要求 h 完全滑过 f 的所有值。图 4.28(c)显示了这两个函数的卷积。注意，卷积是位移变量 x 的函数，并且，在这个例子中，要求 h 完全滑过 f 的 x 的范围是从 0 到 799。

如果我们使用 DFT 和卷积定理得到与图4.28左列相同的结果，我们必须考虑 DFT 表达式中固有的周期。这与图 4.28(f)和图 4.28(g)中对两个周期函数进行卷积是等价的。卷积过程与我们刚才讨论的相同，但是，现在这两个函数是周期函数。按前一段介绍的方法处理这两个函数将得到图4.28(j)所示的结果，很明显，它是不正确的。因为，我们卷积两个周期函数，所以卷积本身也是周期的。在图4.28 中，周期的靠近使它们互相干扰而导致所谓的缠绕错误。根据卷积定理，如果我们计算两个 400 点的函数 f 和 h 的 DFT，两个函数相乘，然后，计算反 DFT，那么我们将得到如图4.28(j)所示卷积的 400 点的错误的一段。

幸运的是，解决缠绕错误问题很简单。考虑分别有 A 个样本和 B 个样本的两个函数 $f(x)$和 $h(x)$，可以证明(Brigham[1988])，如果我们把 0 添加到这两个函数中，使它们具有相同的长度，用 P 来表示，然后，可以这样选择以避免缠绕：

$$P\geqslant A+B-1 \tag{4.6-26}$$

271 ~ 273

在我们的例子中，每一个函数有 400 个点，我们应该使用的最小值是 $P = 799$，这意味着我们需要在每个函数的结尾处添加 399 个 0。这种处理称为 0 填充。作为练习，读者可以自己证明：如果图4.28(f)和图4.28(g)中的函数的周期已通过至少填充 399 个 0 的方法加长了，其结果就是一个周期卷积，其中每个周期都与图4.28(e)中的正确结果相等。对 DFT 使用卷积定理将得到与图4.28(e)相同的 799 点的空间函数。其结论是：若要使第 3 章介绍的“直接”卷积公式法与 DFT 方法产生的卷积结果相同，后者必须对函数补足周期再计算它们的变换。

零可添加到函数的开始处，或分别添加到函数的开始处和结尾处。但将零添加到函数的结尾处更为简单。

形象地表示二维下的例子较为困难，但是，我们可以得出关于缠绕错误及对函数补 0 的相同的结论。令 $f(x, y)$和 $h(x, y)$分别是大小为 $A\times B$ 和 $C\times D$ 像素的图像阵列。在循环卷积中的缠绕错误可以通过对这两个函数进行零填充来避免，方法如下：

273 ~ 274

$$f_p(x,y)=\begin{cases} f(x,y), & 0\leqslant x\leqslant A-1 \text{ 和 } 0\leqslant y\leqslant B-1 \\ 0, & A\leqslant x\leqslant P \text{ 或 } B\leqslant y\leqslant Q \end{cases} \tag{4.6-27}$$

和

$$h_p(x,y)\begin{cases} h(x,y), & 0\leqslant x\leqslant C-1 \text{ 和 } 0\leqslant y\leqslant D-1 \\ 0, & C\leqslant x\leqslant P \text{ 或 } D\leqslant y\leqslant Q \end{cases} \tag{4.6-28}$$

其中，

$$P\geqslant A+C-1 \tag{4.6-29}$$

和

$$Q\geqslant B+D-1 \tag{4.6-30}$$

填充后的图像大小为 $P \times Q$。如果两个阵列大小相同，都是 $M \times N$，则要求

$$P \geqslant 2M - 1 \tag{4.6-31}$$

和

$$Q \geqslant 2N - 1 \tag{4.6-32}$$

在 4.7.2 节，我们将给出一个例子来说明图像上出现缠绕错误所带来的影响。一般来说，DFT 算法对偶数尺寸的阵列执行较快，因此最好选择 P 和 Q 为满足上面方程的最小偶整数。如果两个阵列大小相同，则意味着 P 和 Q 选择为该阵列大小的两倍。

在图 4.28(a) 和图 4.28(b) 中，两个函数的值在取样间隔的尾端方便地变为 0。如果这两个函数中有一个函数或两个函数的值在取样区间的尾端不是 0，当把 0 添加到函数上以消除缠绕错误时，就将创建一个不连续的函数。这类似于用一个“盒子”与一个函数相乘，在频率域它意味着原始变换与一个 sinc 函数的卷积(见例 4.1)，依次地，这将造成一个由 sinc 函数的高频分量导致的所谓频率泄漏。泄漏会在图像上产生块效应。虽然漏泄从来没有被完全消除过，但可以用另外一个函数乘以取样函数来明显地加以减少，这个函数在取样数据的两端平滑地过渡到接近于零，从而减弱“盒子”的（高频分量）急剧过渡。当在图像重建(如高清晰度图形)中希望保真度高的时候，这种称为开窗或切趾的方法是重要的考虑。如果您需要开窗，较好的方法是使用二维高斯函数(见 4.8.3 节)。该函数的的优点是其傅里叶变换也是高斯的，这样会产生较低的泄漏。

一个简单的切趾函数是位于取样数据中心的三角形，它会在取样数据的两端过渡到 0。这称为布特沃斯窗。其他一些常见的窗口是汉明窗和汉宁窗。我们甚至可以使用一个高斯函数。在 5.11.5 节中，我们将会回到加窗问题。

4.7.3 频率域滤波步骤小结

前两节的内容可以总结如下：

1. 给定一幅大小为 $M \times N$ 的输入图像 $f(x, y)$，从式(4.6-31)和式(4.6-32)得到填充参数 P 和 Q。典型地，我们选择 $P = 2M$ 和 $Q = 2N$。
2. 对 $f(x, y)$添加必要数量的 0，形成大小为 $P \times Q$ 的填充后的图像 $f_p(x, y)$。

283 ~ 285

3. 用 $(-1)^{x+y}$ 乘以 $f_p(x, y)$移到其变换的中心。
4. 计算来自步骤 3 的图像的 DFT，得到 $F(u, v)$。
5. 生成一个实的、对称的滤波函数 $H(u, v)$，其大小为 $P \times Q$，中心在 $(P/2, Q/2)$ 处[①]。用阵列相乘形成乘积 $G(u, v) = H(u, v)F(u, v)$；即 $G(i, k) = H(i, k)F(i, k)$。
6. 得到处理后的图像：

$$g_p(x, y) = \left\{\text{real}\left[\Im^{-1}\left[G(u, v)\right]\right]\right\}(-1)^{x+y}$$

如前所述，关于中心对称有助于形象地描述滤波过程并生成滤波函数本身，但它不是基本的需求。

其中，为忽略由于计算不准确导致的寄生复分量，选择了实部，下标 p 指出我们处理的是填充后的阵列。

7. 通过从 $g_p(x, y)$的左上象限提取 $M \times N$ 区域，得到最终处理结果 $g(x, y)$。

图 4.36 说明了上述步骤。图中的符号解释了每一个图像源。如果它被放大，图 4.36(c)将会显示图像中插入的黑点，因为，为了显示，负的灰度被裁剪为 0 了。注意图 4.36(h)中图像暗边的特点，它是使用零填充处理后由低通滤波造成的。