背景：

瓷砖表面的色差是影响瓷砖外观质量的一个重要指标，需要一种有效的方法来自动分拣瓷砖中存在的色差。分析瓷砖的色差之前，要对图像实施定位，其目的是有效提取图像特征，通常会采用边缘检测方法。

算子：

1. 算子是一种数学运算符，它可以对一个或多个函数或变量进行操作，得到一个新的函数或变量。算子的作用是用来描述函数或变量之间的关系，或者用来提取函数或变量的某些特征。
2. 在图像处理中，算子通常是一种卷积核，它可以对图像的每个像素点进行卷积运算，得到一个新的图像。算子的作用是用来对图像进行滤波、增强、边缘检测、特征提取等操作。

算子的作用包括：

1. 边缘检测：算子可以通过对图像进行卷积操作来检测图像中的边缘，即图像中灰度值发生显著变化的地方，从而在图像中找到物体的轮廓。
2. 特征提取：算子可以通过对图像进行卷积操作来提取出图像中的特定特征，如纹理、角点、线条等，从而用于图像识别、目标检测等任务。
3. 图像增强：算子可以通过对图像进行滤波操作来增强或抑制图像中的某些频率成分，从而改变图像的亮度、对比度、锐化等特性。

边缘检测算法：

边缘检测所采用的技术主要有Roberts 算法、Prewitt 算 法、Sobel 算 法、Canny 算 法 以 及 Laplacian 算法。对于提取瓷砖灰度图像的边缘，效果最好的是 Prewitt 算法（论文提到）

**作用：**

图像边缘检测算子（如Prewitt算子和Sobel算子）的作用是在数字图像中寻找边缘和轮廓的位置。

1. 边缘在图像中通常表示着颜色、亮度或纹理等方面的突变或变化，是图像中不同区域之间的分界线。边缘检测是图像处理中的一项基本任务，广泛应用于计算机视觉、图像分析、图像识别等领域。
2. 边缘检测算子通过对图像中的像素值进行差分或卷积等操作，计算像素值的梯度或变化，从而找到图像中的边缘位置。这些算子通常使用局部领域内的像素值信息，通过与卷积核的卷积操作，得到一个新的像素值，表示该位置处的边缘强度和方向。

边缘检测原理：

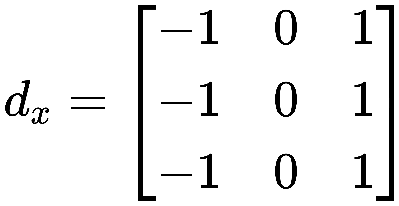
1. 图像边缘检测算子（如Prewitt算子和Sobel算子）的作用是在数字图像中寻找边缘和轮廓的位置。
2. 边缘在图像中通常表示着颜色、亮度或纹理等方面的突变或变化，是图像中不同区域之间的分界线。边缘检测是图像处理中的一项基本任务，广泛应用于计算机视觉、图像分析、图像识别等领域。
3. 边缘检测算子通过对图像中的像素值进行差分或卷积等操作，计算像素值的梯度或变化，从而找到图像中的边缘位置。这些算子通常使用局部领域内的像素值信息，通过与卷积核的卷积操作，得到一个新的像素值，表示该位置处的边缘强度和方向。

Prewitt算子和Sobel算子介绍：

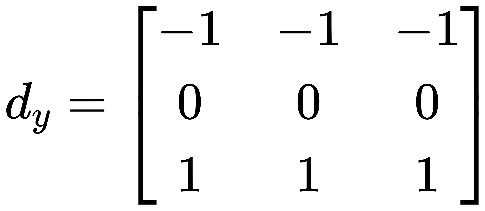
Prewitt算子和Sobel算子通常作用于灰度图像中的像素值，通过在图像的水平和垂直方向上计算梯度近似值，从而检测图像中的水平和垂直边缘。这些算子可以帮助提取图像中的边缘信息，用于图像的分割、特征提取、目标检测、图像增强等应用中。

Prewitt算子：   
Prewitt算子是一种离散差分算子，用于计算图像中的一阶导数，从而检测图像中的边缘。Prewitt算子分别在水平和垂直方向上使用两个3x3的卷积核进行卷积操作，得到水平和垂直方向上的梯度近似值。

水平方向的Prewitt算子如下：



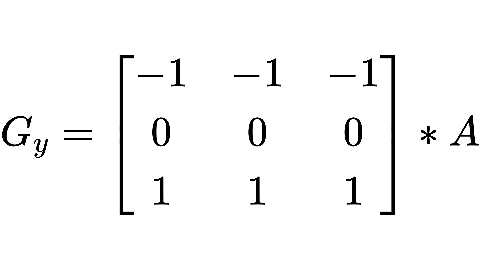
垂直方向的Prewitt算子如下：

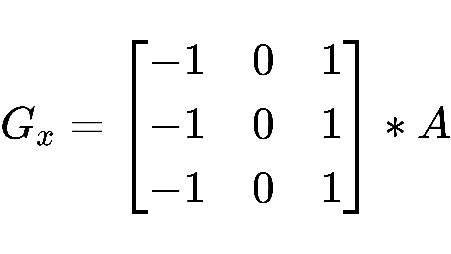


Prewitt算子通过计算图像中的梯度变化，可以在图像中检测到边缘的位置和方向。

Prewitt算法的理论依据是：  
如果像素点的灰度新值不小于阈值，则这些像素点都属于边缘点。也就是说，选用合适的阈值T，如果P(i,j) ≥ T，则(i,j)就是边缘点，P(i,j)就是边缘图像。

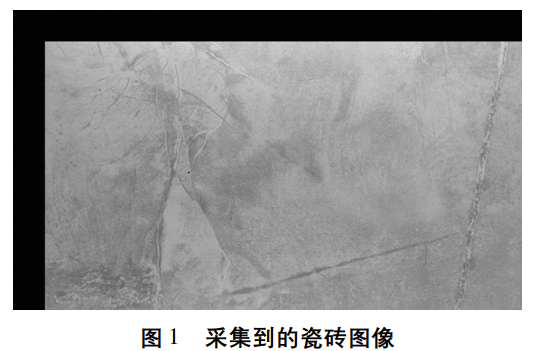
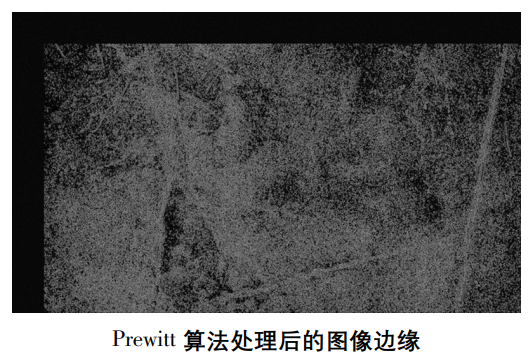
[如果我们定义A(x,y)为源图像，Gx和Gy为两个图像，分别包含水平和垂直导数近似，那么它们可以计算为](https://en.wikipedia.org/wiki/Prewitt_operator)：





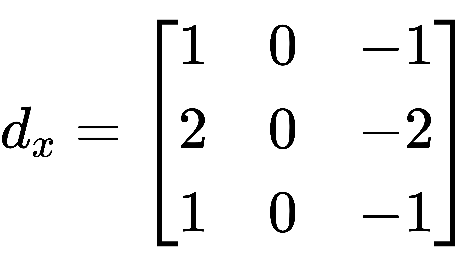
那么

或者

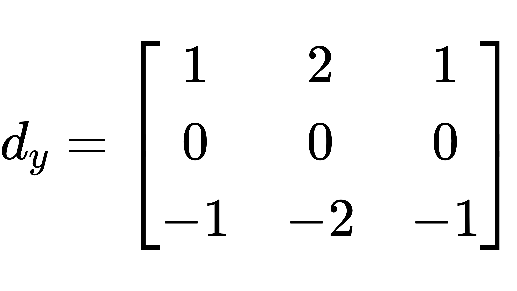
 

Sobel算子：   
Sobel算子也是一种常用的离散差分算子，用于计算图像中的一阶导数，用于边缘检测。Sobel算子也使用了两个3x3的卷积核，在水平和垂直方向上进行卷积操作，得到水平和垂直方向上的梯度近似值。过计算图像中的梯度变化，可以检测图像中的边缘

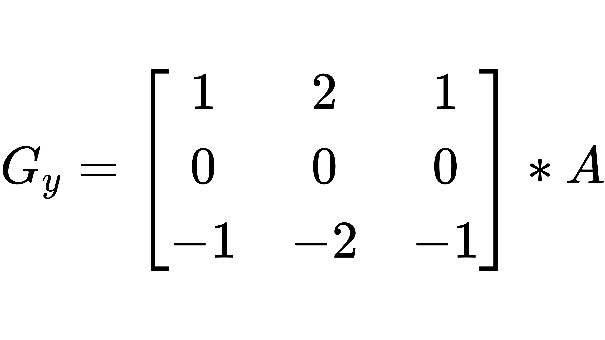
水平方向的Sobel算子如下：

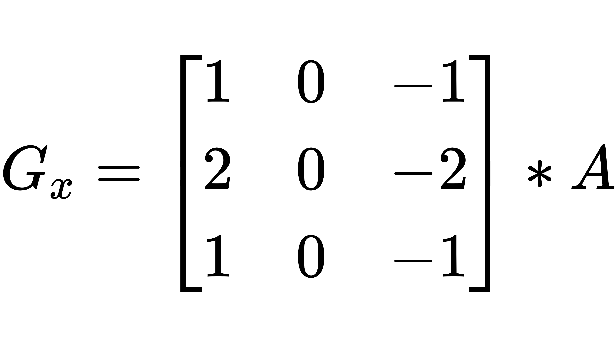


垂直方向的Sobel算子如下：

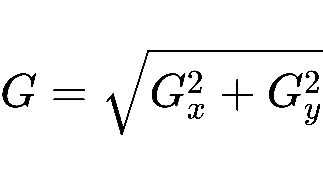


[如果我们定义A(x,y)为源图像，Gx和Gy为两个图像，分别包含水平和垂直导数近似，那么它们可以计算为](https://en.wikipedia.org/wiki/Prewitt_operator)：

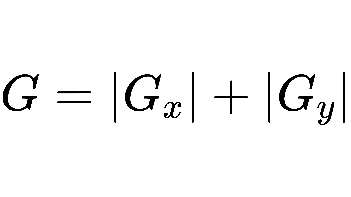




将两个分量结合为梯度：



或者表示为



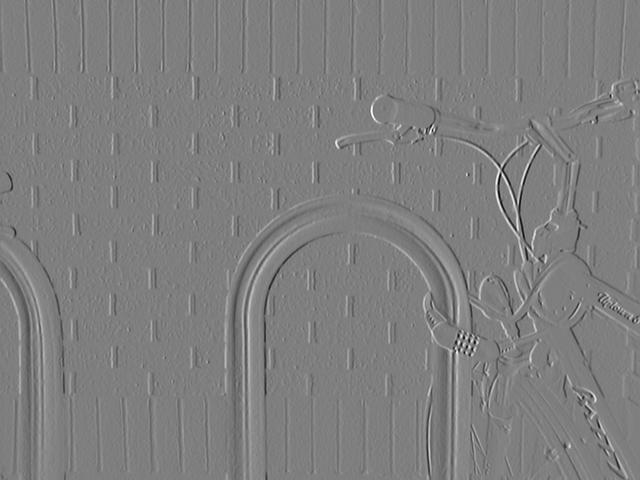
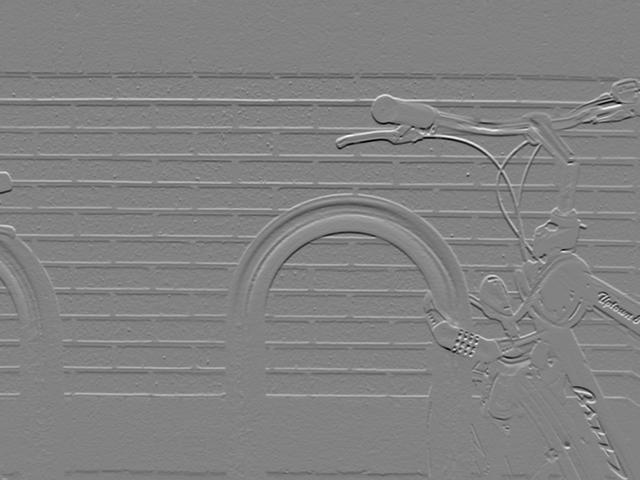
同样地，如果梯度G大于某一阀值则认为该点 (x,y)为边缘点。

以下是实际展示Sobel算子的效果：

原始图像vs处理后的图像

X轴方向处理vs Y轴方向处理

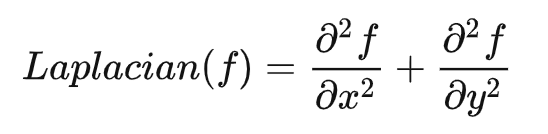
 

Prewitt算子和Sobel算子在边缘检测中广泛应用，具体使用哪种算子取决于应用场景和需求。这两种算子都可以用于检测图像中的边缘，但由于其卷积核的不同，它们在对边缘方向和强度的响应上可能会有所不同。

Laplacian算子

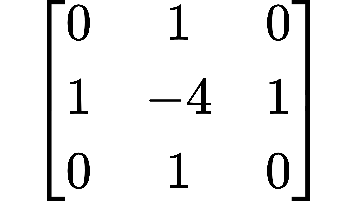
Laplacian算子具有各方向同性的特点，能够对任意方向的边缘进行提取，具有无方向性的优点，因此使用Laplacian算子提取边缘不需要分别检测X方向的边缘和Y方向的边缘，只需要一次边缘检测即可。Laplacian算子是一种二阶导数算子，对噪声比较敏感，因此常需要配合高斯滤波一起使用。

计算公式如下：



在二维图像上表示成：

Laplacian算子可以表示为模板形式：



[Laplacian算子是一种二阶微分算子，它可以用来测量一个函数在某个点的曲率，也就是变化的程度。](https://blog.csdn.net/qq_49478668/article/details/123808815)可以[突出图像中强度发生快速变化的区域，因此常用在边缘检测任务当中。它可以直接对图像进行卷积运算，得到一个新的图像，其中边缘处的像素值较大，而平坦区域的像素值接近于零](https://blog.csdn.net/weixin_40647819/article/details/91463946)。

Laplacian算子能够检测出图像中的边缘，但也有一些缺点：

1. Laplacian算子没有考虑边缘的方向信息，因此无法区分水平、垂直或其他方向的边缘。
2. Laplacian算子对噪声非常敏感，因为它是一个二阶导数算子，会放大高频分量。因此，在使用Laplacian算子之前，通常需要先对图像进行平滑处理，以降低噪声的影响
3. Laplacian算子得到的边缘是双边的，也就是说，在边缘两侧都有响应值。这会导致边缘不够细致和清晰。

LoG算子的提出：

[为了解决这些问题，一种常用的方法是在进行Laplacian运算之前先对图像进行高斯平滑滤波处理，这样可以减少噪声，并且使得边缘更加平滑。这种方法称为高斯拉普拉斯（Laplacian of Gaussian, LoG）算子](https://blog.csdn.net/qq_49478668/article/details/123808815)。

LoG算子的结构： LoG算子的整体结构包括两个步骤：高斯滤波和拉普拉斯算子。首先，图像经过高斯滤波器，滤波后的图像变得模糊，噪声也被消除。然后，对滤波后的图像应用拉普拉斯算子，计算图像在各个方向上的二阶导数，从而得到图像中的边缘信息。

具体而言，LoG算子的整体结构可以表示为以下公式：

高斯函数：

LoG算子的表达式：

原图像与高斯核函数卷积后再做laplace运算：

DoG算子

DoG算子是LoG算子的一种近似方法，与LoG算子类似，DoG算子也是基于高斯滤波的边缘检测算法，但采用了两个不同尺度的高斯滤波器的差异，从而近似地计算了LoG算子。

DoG算子是高斯函数的差分，具体到图像中，就是将图像在不同参数下的高斯滤波结果相减，得到差分图。

DoG算子的作用：

由于使用了两个不同尺度的高斯滤波器，DoG算子对不同尺度的边缘有更好的响应，因此在实际应用中，DoG算子常用于多尺度边缘检测和图像。

DoG算子的表达式如下：

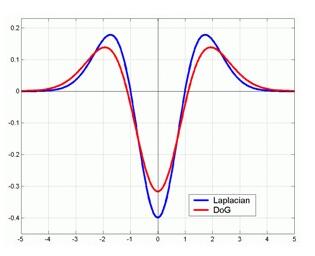
其中高斯核函数：

可以得到：

根据一阶导数的定义得到：

最后得到：

总的来说，就是用两个不同标准的高斯核去对图像进行平滑操作，结果相减就能近似得到LoG算子处理后的结果，由于高斯差分的计算更加简单，因此可用DoG算子近似替代LoG算子。



reference

<https://senitco.github.io/2017/06/20/image-feature-LoG-DoG/>

<https://zhuanlan.zhihu.com/p/92143464>

[Laplacian算子-Log算子-Dog算子边缘检测原理合集及实现\_dog算子的优缺点\_小武~~的博客-CSDN博客](https://blog.csdn.net/weixin_40647819/article/details/91463946)

[【从零学习OpenCV 4】Laplacian算子 - 知乎 (zhihu.com)](https://zhuanlan.zhihu.com/p/101162962)

《OpenCV算法精解》---张平