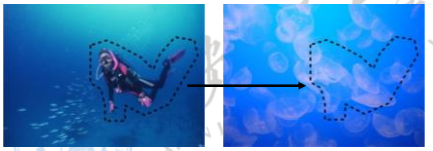
计算机视觉与模式识别

苏远歧 新型计算机研究所

Jianbo Shi, University of Pennsylvania

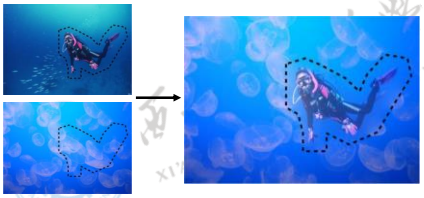
如何合成两幅图像(Blending)?



图像合成：图像修复.....

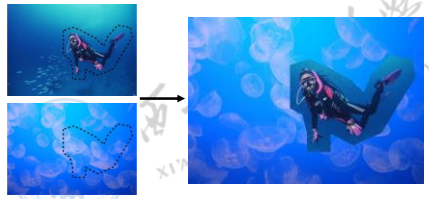
- 从一幅图像中扣出来一部分(如何自动化的扣，我们在图像分割这一章节有所涉及)
- 在另外一幅图像中恢复出来扣取的图像

合成 =拷贝+粘贴图像



图像合成是有关.....的艺术:
图像造假、将有关图像修复的艺术进行隐藏、使之看之自然
faking images, hiding evidence of image surgery, making it looks natural

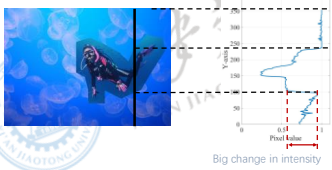
直接进行拷贝+粘贴



直接的方式：看起来似乎不太妙!
-- 我们在拷贝的区域人为地产生一个不自然的边界。

挑战: 颜色和亮度值的不匹配

垂直线上的蓝色分量值

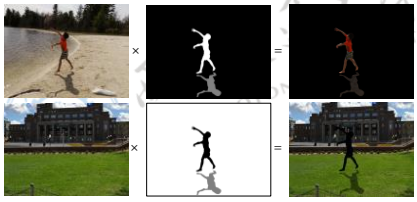


亮度上的差异产生了一个新的图像边界：
- 我们应该如何取消除这个边界?

Alpha blending

Alpha channel encodes the transparency of the object

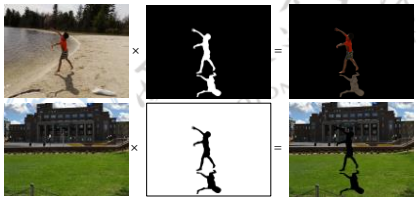
Alpha blending

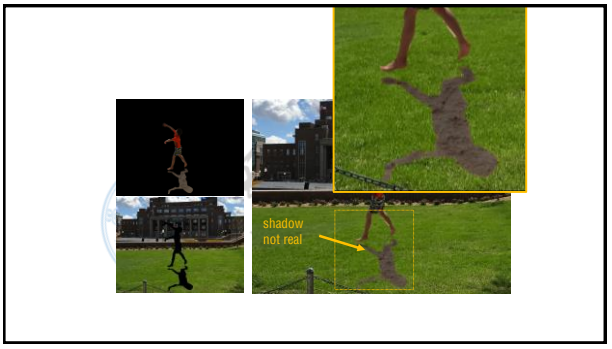


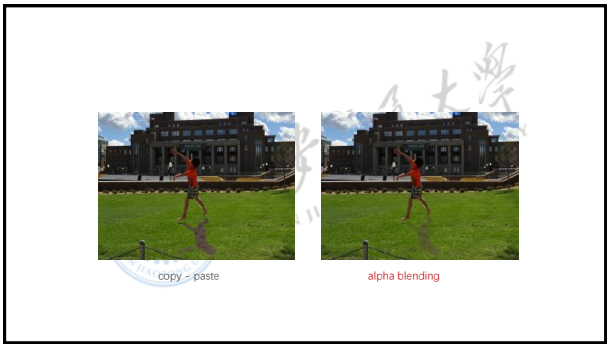
Alpha blending

Alpha blending

我们直接的拷贝和粘贴是一种特殊的Alpha合成——基于二值Alpha图像的

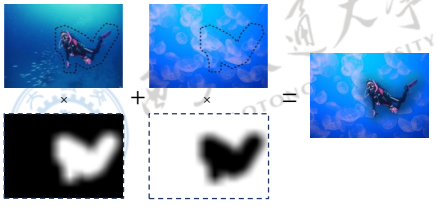


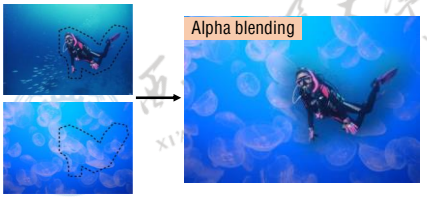




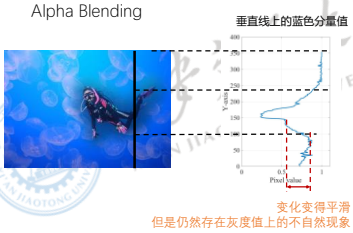
Alpha blending 可以处理透明的问题

Alpha blending的调节



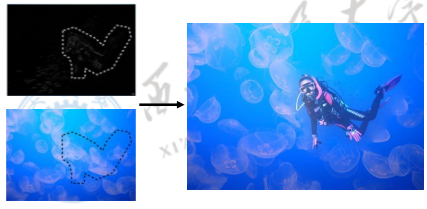


Alpha Blending



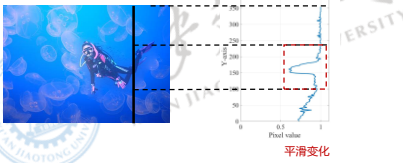
变化变得平滑
但是仍然存在灰度值上的不自然现象

解决方法: 仅拷贝梯度值+重构



梯度合成:

不再有灰度值上的突变



图像合成

copy - paste　　alpha blending　　Gradient blending

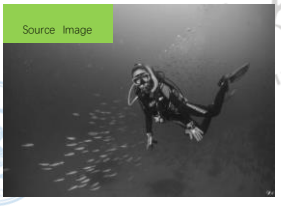
梯度合成的示例

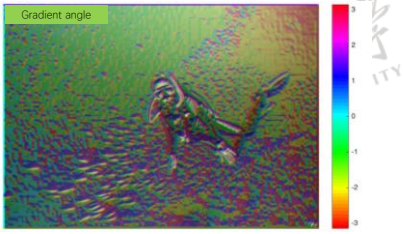
Source (figure)　　Target (background)　　Result

帽子的颜色和背景色发生了融合

-- 成功的图像修复... 也产生了一些有趣的副作用

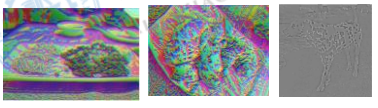
为什么要通过
梯度重构图像

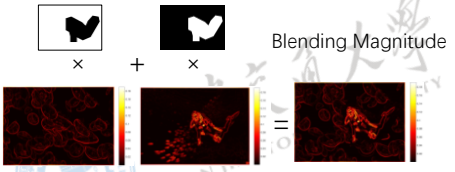




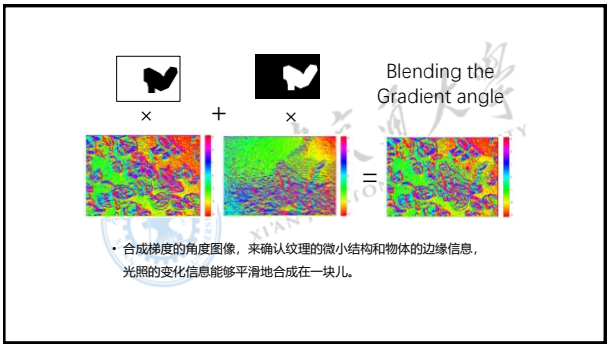
梯度域

- 梯度记录了有关形状和阴影的几乎全部信息
- 记录了纹理的微小结构信息
- 记录了光照的微小变化信息
- 在金字塔合成中，我们将图像分解成2阶的拉普拉斯图像，记录了形状的信息。

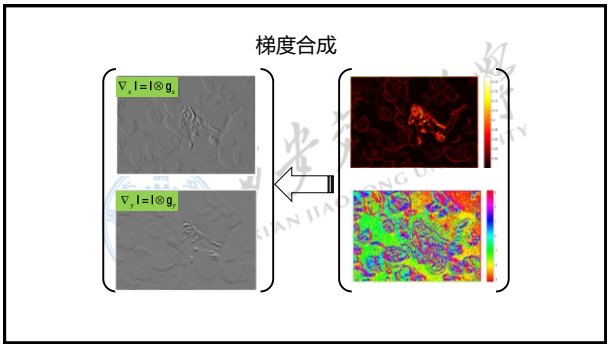


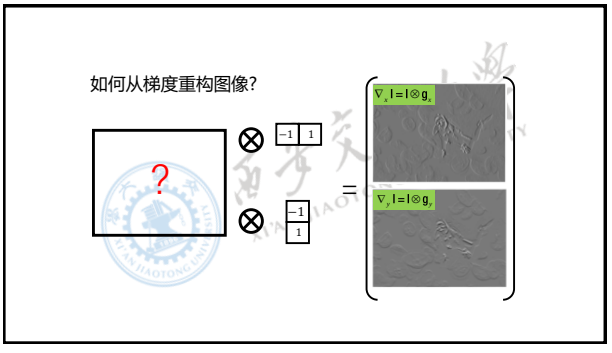


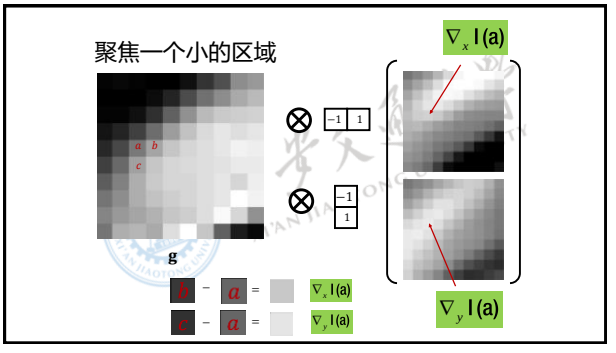
- 我们首先合成梯度的幅度，来构建一个无缝（Seamless）的梯度幅度图像

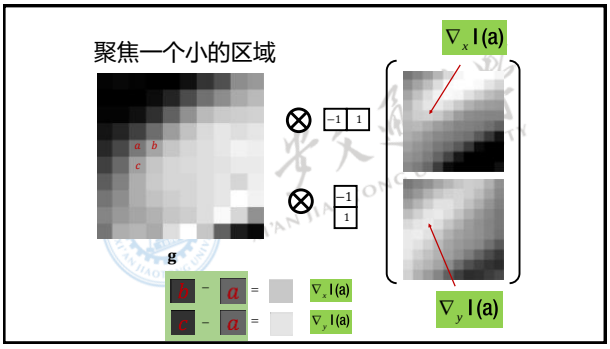


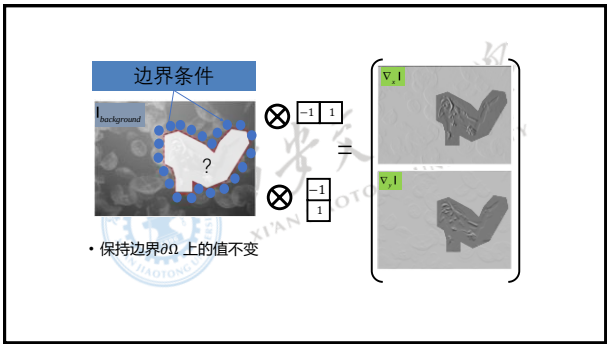
- 合成梯度的角度图像，来确认纹理的微小结构和物体的边缘信息，光照的变化信息能够平滑地合成在一块儿。

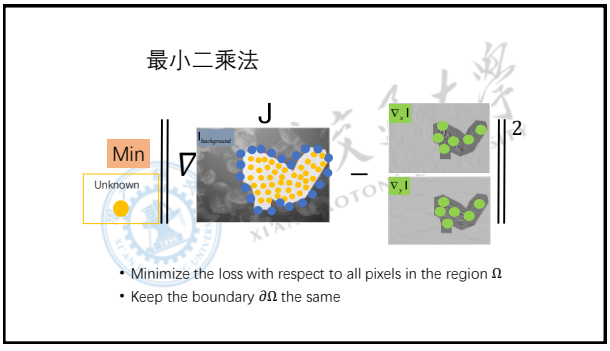


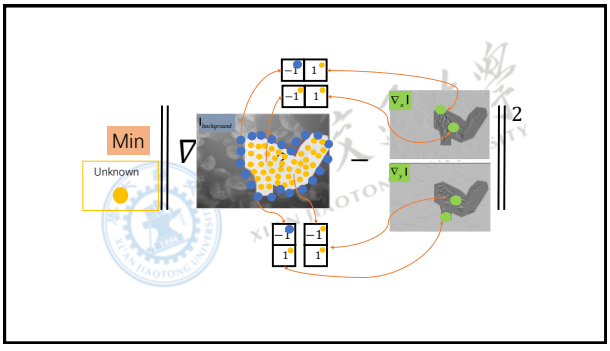




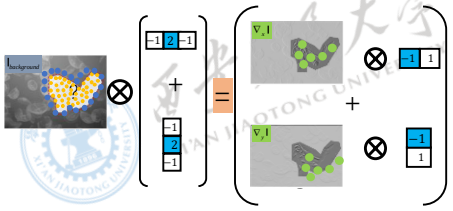

聚焦一个小的区域
$\nabla_x I(a)$
$\nabla_y I(a)$
g


边界条件
$I_{background}$
?
$\nabla_x I$
$\nabla_y I$
• 保持边界$\partial\Omega$ 上的值不变

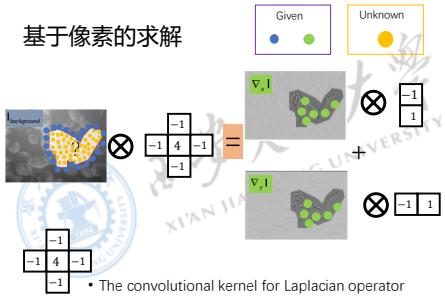

最小二乘法
J
Min
Unknown
$I_{background}$
$\nabla_x I$
$\nabla_y I$
2
• Minimize the loss with respect to all pixels in the region Ω
• Keep the boundary ∂Ω the same


Min
Unknown
$I_{background}$
$\nabla_x I$
$\nabla_y I$
2

最小二乘法



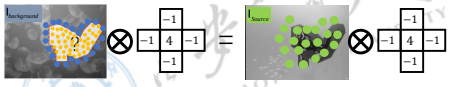
基于像素的求解



- The convolutional kernel for Laplacian operator

特例

- 当 $\nabla_x I$ $\nabla_y I$ 是由一幅原图像 I_{Source} 计算的(without editing)



- 保持边界$\partial\Omega$ 上的值不变
- RGB三个通道分开处理
- 每个待求像素的拉普拉斯算子对应着一个方程式

二维函数的拉普拉斯算子

$$\Delta I = \frac{\partial^2 I}{\partial x^2} + \frac{\partial^2 I}{\partial y^2}$$

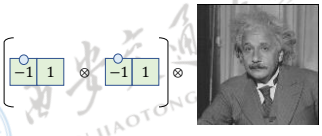
最简单的拉普拉斯算子

最简单的拉普拉斯算子

$\frac{\partial^2 I}{\partial x^2} = \frac{\partial}{\partial x} \otimes \frac{\partial}{\partial x} \otimes I \quad \Rightarrow$

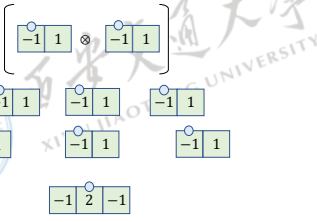
最简单的拉普拉斯算子

$\frac{\partial^2 I}{\partial x^2} = \frac{\partial}{\partial x} \otimes \frac{\partial}{\partial x} \otimes I \quad \Rightarrow$



最简单的拉普拉斯算子

$\frac{\partial^2 I}{\partial x^2} = \frac{\partial}{\partial x} \otimes \frac{\partial}{\partial x} \otimes I \quad \Rightarrow$



最简单的拉普拉斯算子

$$\frac{\partial^2 I}{\partial y^2} = \frac{\partial}{\partial y} \otimes \frac{\partial}{\partial y} \otimes I \quad \Rightarrow$$

$$\begin{bmatrix} -1 \\ 1 \end{bmatrix} \otimes \begin{bmatrix} -1 \\ 1 \end{bmatrix}$$

$$\begin{bmatrix} -1 & -1 \\ 1 & 1 \end{bmatrix} \quad \begin{bmatrix} -1 & -1 \\ 1 & 1 \end{bmatrix} \quad \begin{bmatrix} -1 & -1 \\ 1 & 1 \end{bmatrix}$$

$$\begin{bmatrix} -1 \\ 2 \\ -1 \end{bmatrix}$$

二维函数的拉普拉斯算子

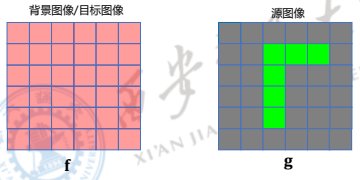
$$\Delta I = \frac{\partial^2 I}{\partial x^2} + \frac{\partial^2 I}{\partial y^2}$$

$$\begin{bmatrix} & -1 & \\ -1 & 4 & -1 \\ & -1 & \end{bmatrix}$$

- 各向同性的最小阶线性算子!
- 最简单的拉普拉斯算子易受噪声影响

如何求解这个逆向的拉普拉斯方程?

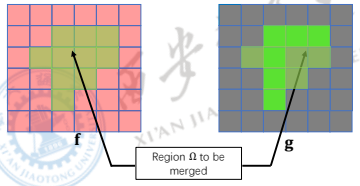
一个示例



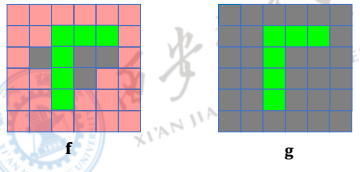
出于显示考虑, 这里在背景图像 f 上叠加了一个掩模

f图像的RGB值为： (255,0,0), 图像 g 的背景值为 (0,0,0)

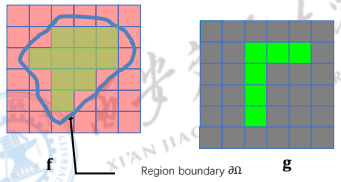
Step1: 定义要合并的区域



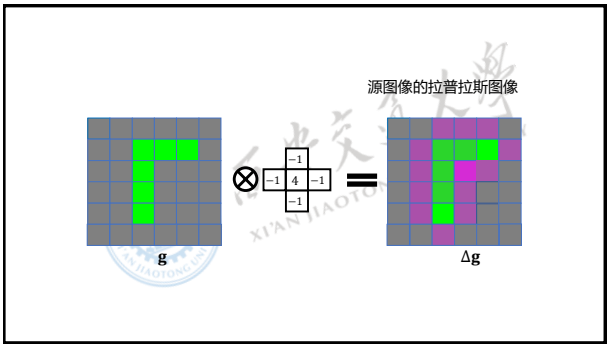
直接拷贝——粘贴

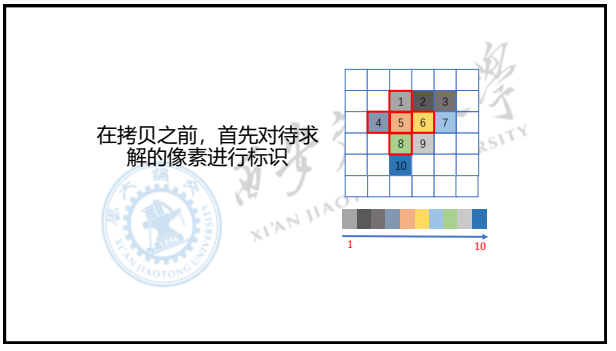


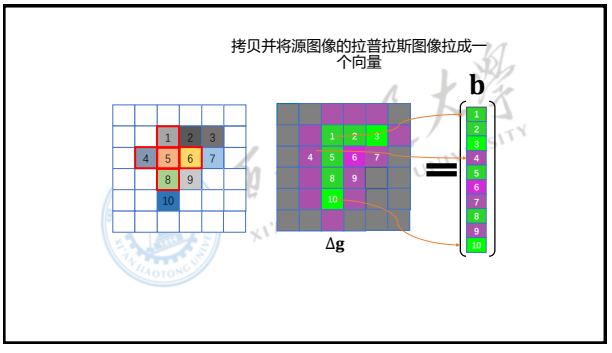
保持背景图像 f 在边界上的值不变

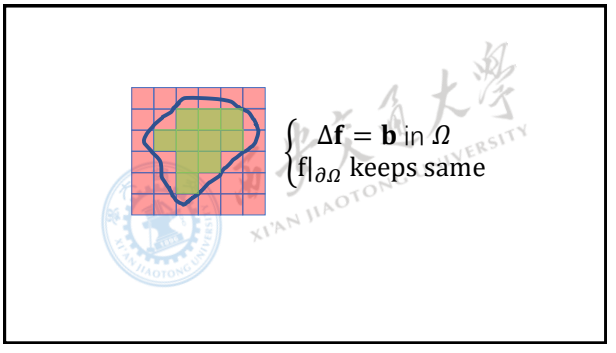


Keep f the same on the boundary ∂Ω


源图像的拉普拉斯图像
-1
-1
4
-1
-1
g
Δg


在拷贝之前，首先对待求解的像素进行标识
1
2
3
4
5
6
7
8
9
10
1
10


拷贝并将源图像的拉普拉斯图像拉成一个向量
b
Δg


$\Delta \mathbf{f} = \mathbf{b}$ in Ω
$\mathbf{f}|_{\partial\Omega}$ keeps same

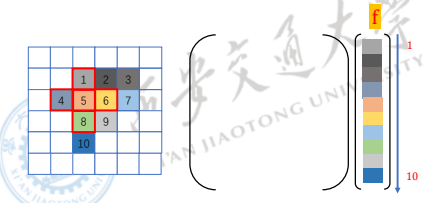
$$\Delta \mathbf{f} = \mathbf{b}$$

Δ 是一个线性的算子…

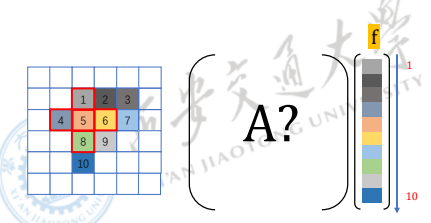
$$\mathbf{A}\mathbf{f} = \mathbf{b}$$

…We can use a matrix **A** to encode it!

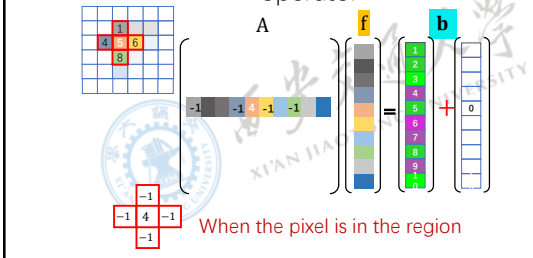
Indexing the unknowns



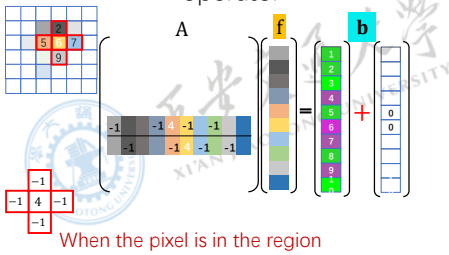
Laplacian as a matrix?



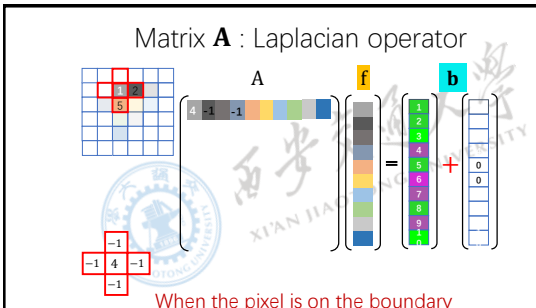
Matrix **A** encoding the Laplacian operator



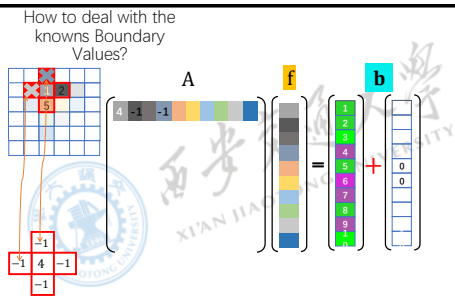
Matrix **A** encoding the Laplacian operator

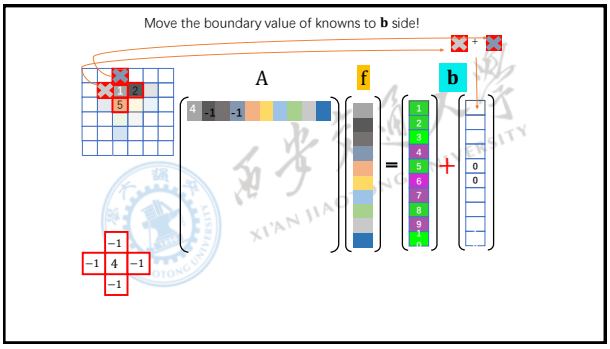


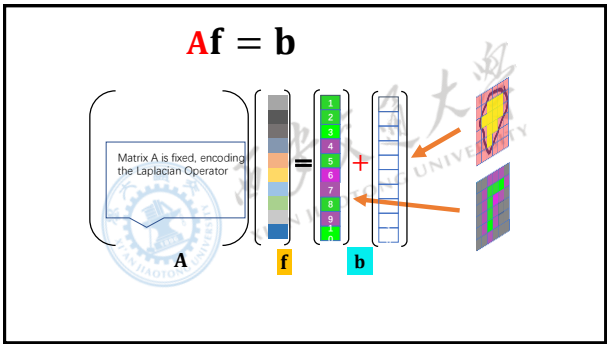
Matrix **A** : Laplacian operator

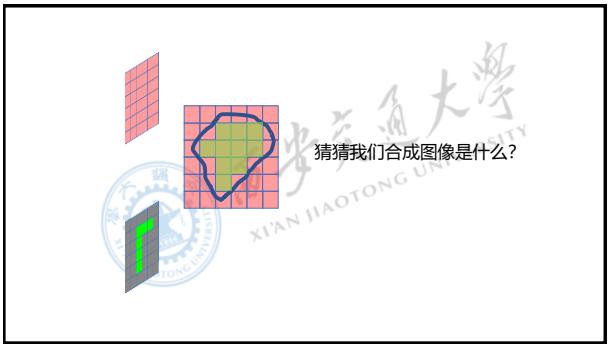


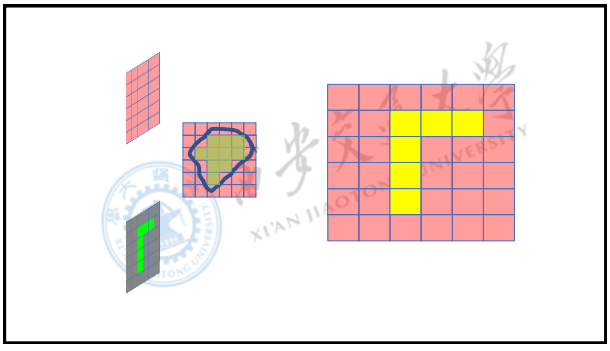
How to deal with the knowns Boundary Values?

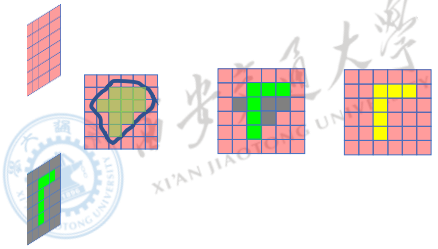



Move the boundary value of knowns to **b** side!
A
f
b
=
+
−1
−1 4 −1
−1

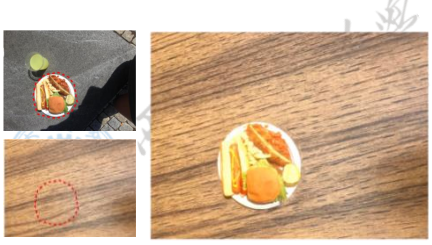

Af = b
Matrix A is fixed, encoding the Laplacian Operator
A
=
+
f
b


猜猜我们合成图像是什么?

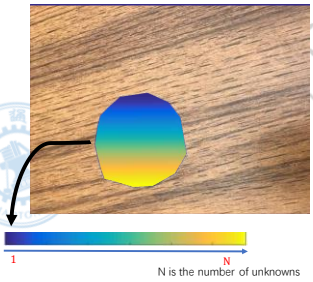




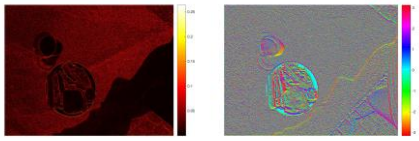
我们来看看真实图像对应的例子



Step1.标记需要求解的像素

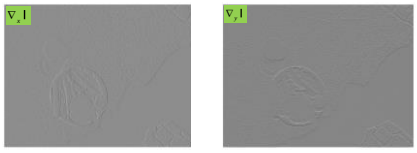


Step2. 从源图像中获取梯度图像



Gradient magnitude　　Gradient angle

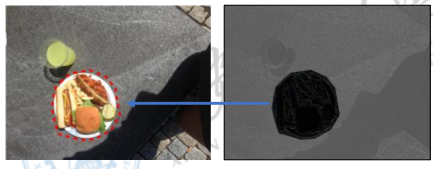
Step2. 从源图像中获取梯度图像



Step2. 计算源图像的拉普拉斯图像

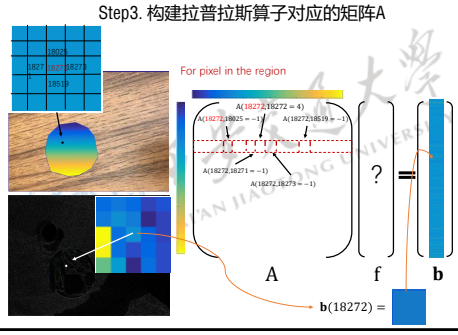
Source image　　Laplacian

Step2. 从源图像中拷贝梯度

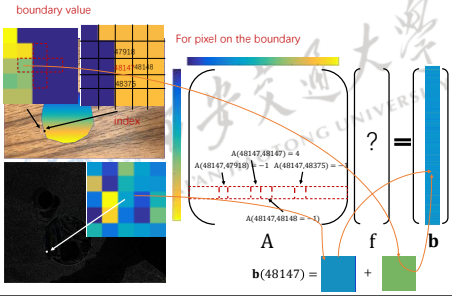


Step2. 扣出源图像中的拉普拉斯图像

Step3. 构建拉普拉斯算子对应的矩阵A



Step3. 构建拉普拉斯算子对应的矩阵A



Step4. 求解线性方程组，并将像素值拷贝回目标图像中

Where is waldo

Where is waldo

Where is waldo

Where is waldo