Jobs in September

9 月份工作汇报

TanLin

October 10,2016

Contents

Classification Of Interpolation

Interpolation:

- Nearest Neighbor Interpolation
- Billnear Interpolation
- Bicubic Interpolation

.

Code

Nearest Neighbor Interpolation 具体的代码如下：
Billnear Interpolation 具体的代码如下：
Bicubic Interpolation 具体的代码如下：

Running Results

Classification Of Arithmetic Operations

Arithmetic Operations:

- Image Average $\overline{g(x,y)} = \frac{1}{K}\sum_{i=1}^{K} g_i(x,y)$
- Image Subtraction $g(x,y) = f(x,y) - h(x,y)$
- Image Multiplication $g(x,y) = f(x,y) \times h(x,y)$
- Image Division $h(x,y) = g(x,y) \div f(x,y)$

Image Average's Introduction

- 对噪声图像进行相加求平均主要是为了降噪
- 具体操作是：设 $f(x, y)$ 是无噪图像,$\eta(x, y)$ 是噪声,$g(x, y)$ 是噪声干扰图像. 则

$$g(x, y) = f(x, y) + \eta(x, y)$$

- 对 K 副不同的噪声图像进行进行求平均得到, 则

$$\overline{g(x, y)} = \frac{1}{K} \sum_{i=1}^{K} g_i(x, y)$$

Image Average's Code

因为是对 K 副不同噪声图像进行求平均，第一步是添加噪声得到 K 副噪声图像

Add Gaussian Noise 具体的代码如下：

Image Average's Code

Image Average 具体的代码如下：

Image Average's Running Results

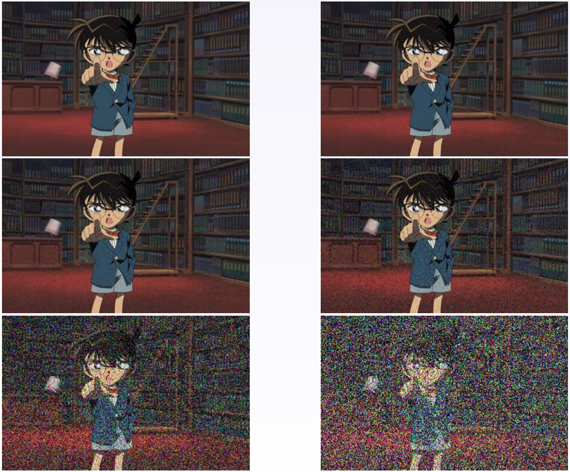


Image Average's Running Results

Image Subtraction’s Introduction

- 图像的相减经常用于增强图像之间的差.

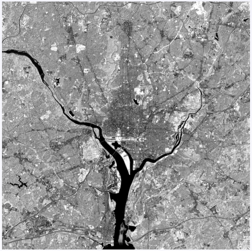
$$g(x, y) = f(x, y) - h(x, y)$$

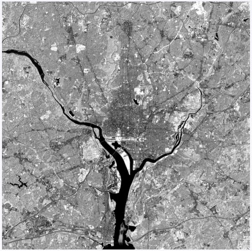
- 因为需要先将图像的每一个最低阶比特置为 0, 所以首先工作是得到每一个像素的最低有效比特为 0 的图像。
- 在 256 级灰度图像中, 每一个像素的灰度是有 8 比特组成的. 也就是一副 8 比特的图像是由 8 个比特平面组成, 平面 1 包含图像中所有像素的最低阶比特. 所以最低阶比特置为 0, 就是讲像素值 0~1 变为 0.

Image Subtraction's Code

低阶为 0 具体的代码如下:
Image Subtraction 具体的代码如下:

Image Subtraction's Running Results





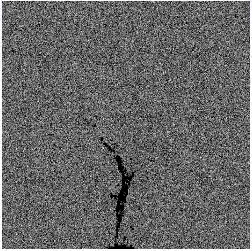


Image Multiplication's Introduction

- 设是是由完美图像与阴影图像的乘积来建模的图像. 即

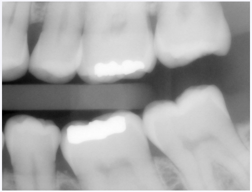
  $$g(x,y) = f(x,y) \times h(x,y)$$

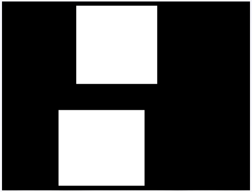
- 图像的乘积 (相除) 常用于阴影矫正.
  比如, 有一副牙齿的片, 你只需看看它的某一部分, 这是需要用到了图像乘法, 将图像与模板图像相乘, 它的模板图像的 ROI 区域为 1, 其他为 0, 也就是白色对应 1, 黑色对于 0.

Image Multiplicationn's Code

Image Multiplication 具体的代码如下：

Image Multiplication's Running Results





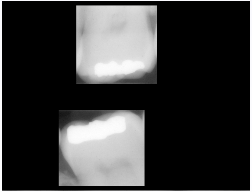


Image Division's Introduction

- 图像相除是将一个图像亮度调高,

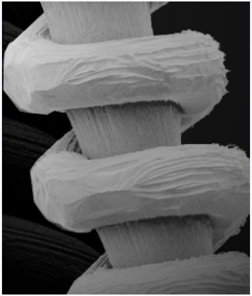
  $$h(x, y) = g(x, y) \div f(x, y)$$

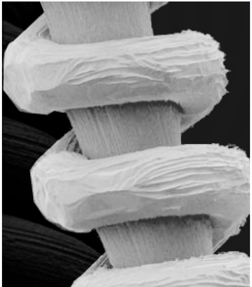
  因为模板的像素值是属于 0~1, 用指定图像除以模板, 那么得到的图像像素值肯定变高. 但是得到的像素值可能 >=255, 那么, 就会导致冗余.

Image Division's Code

Image Division 具体的代码如下：

Image Division's Running Results





Set Operation's Introduction

Set Operations:

- Union Operation
- Intersection Operation
- Complement Operation

Complement Operation

- The complement is defined as the pairwise differences between a constant and the intensity of every pixel in an image.
- set $A=\{(x,y,z)|(x,y,z)\in \mathrm{A}\}$, $\mathrm{A}^c=\{(x,y,K-z)|(x,y,z)\in \mathrm{A}\}$.
  其中 $K=2^k-1$, 我们这里是用的 8 比特的图, 所以 $K=255$.
  此时,A^c 与 A 是大小相同的图像.

Union Operation

- The union operations for gray-scale values usually are defined as the max of corresponding pixels pairs.
- A 与 B 的并集可以表示为:

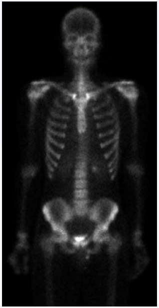
$$A \cup B = \{max(a, b) | a \in A, b \in B\}$$

- B 是与 A 大小相同的图像,m 为 A 中的元素的平均灰度值,B 的 z 为 $3m$.
- 对 A 与 B 求并集, 得到的图像与 A,B 大小是一样的, 如果 A 的值超过 $3m$, 则用 A 表示, 其余的为 $3m$, 即中间灰度值.

Set And Logical Operation's Code

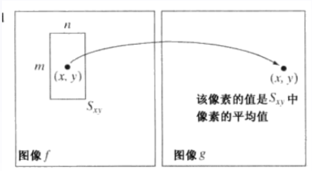
Set And Logical Operations 具体的代码如下：

Set And Logical Operations's Running Results



Neighborhood Operation's Introduction

- 邻域操作就是计算一个大小为 $m \times n$, 中心在 (x, y) 的矩形邻域中求像素的平均值, 并且映射到输出图像的相应坐标位置.
  设 $g(x, y)$ 是输出图像,$f(r, c)$ 是原图像,
  $$g(x, y) = \frac{1}{mn} \cdot \sum f(r, c)$$



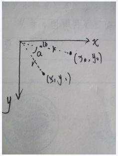
Neighborhood Operation's Code

Neighborhood Operations 具体的代码如下:

Neighborhood Operation's Running Results

Rotation's Introduction

- 旋转的原理:



- 以图像的左上角是其坐标原点，X 轴沿着水平方向向右，Y 轴沿着竖直方向向下。而在旋转的过程一般使用旋转中心为坐标原点的笛卡尔坐标系，所以图像旋转的第一步就是坐标系的变换，原坐标是(x_0, y_0)，旋转之后坐标是(x_1, y_1)，一开始角度是 b，旋转角度是 a，如图所示，可以很容易推导出旋转坐标方程为:

$$\begin{cases} x_1 = x_0\cos(a) - y_0\sin(a) \\ y_1 = x_0\sin(a) + y_0\cos(a) \end{cases}$$

Rotation's Introduction

- 设原图像的四个点分别是(0,0),(src.cols,0),(0,src.rows),(src.cols,src.rows), 那么旋转之后的坐标可以由上面的公式求出来.
- 将这四个坐标取出旋转之后, 新的四个坐标. 设 top 为旋转后最高点的纵坐标，down 为旋转后最低点的纵坐标，left 为旋转后最左边点的横坐标，right 为旋转后最右边点的横坐标, 此时的原点是(left,top).
  旋转后的宽和高为 newWidth,newHeight，则可得到下面的关系:

$$\begin{cases} newHeight = down - top \\ newWidth = right - left \end{cases}$$

- 那么在新的坐标系下, 原图像坐标是以(j+left,i+top)的变化映射到到新的图像的, 其中 (j,i) 为新的图像的坐标.

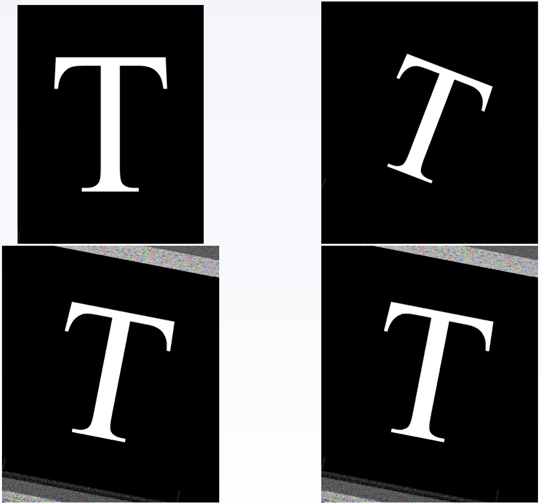
Rotation's Code

Nearest Neighbor Rotation 具体的代码如下:

Bilinear Rotation 具体的代码如下:

Bicubic Rotation 具体的代码如下:

Rotation’s Running Results



Geometrically Distorted Image's Introduction

- 图像配准前, 需要一个参考图像和一个几何畸变图像. 几何畸变就是图像的垂直和水平方向都发生了变化.
- 图像的几何畸变就是运用的 Shear(vertical) 和 Shear(horizontal),

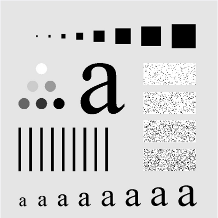
$$\begin{cases} x = i + s * j \\ y = s * i + j \end{cases}$$

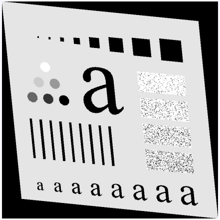
- (i,j) 是原图像的像素点,(x,y) 是偏移之后图像的像素点, 其中 s 是偏移角度.

Geometrically Distorted Image's Code

Image Registration 具体的代码如下：

Geometrically Distorted Image's Running Results





Processing And Plans

Processing $\bigstar$ *Plans*