

数字图像处理大作业报告

1. 数据集

1.1 自然场景

1.1.1 高空摄影

数据集为**城市区域分布分类**, (比赛数据, 现已不开放下载), 包括9类城市空间类型, 40万张图片。高空摄影, 图像比较模糊。



1.1.2 正常场景

数据集为**iccv09Data**, 715张图像, 地址: https://github.com/mmmmmmiracle/digital_image_processing/blob/master/data/iccv09Data.tar.gz。自然场景下的人和物。



1.1.3 水下摄影

数据集为**水下目标检测**, 5543张图像。地址: <https://www.kesci.com/u/7a6126>。深海拍摄, 图像对比度低, 雾化现象严重。



1.1.4 质地纹理

数据为**纹理质地图片数据集**, 476张图像, 地址: <https://www.kesci.com/home/dataset/5e903545e7ec38002d015fbb/files>。是在奥地利萨尔茨堡附近拍摄采集的476种彩色纹理图像的大集合。



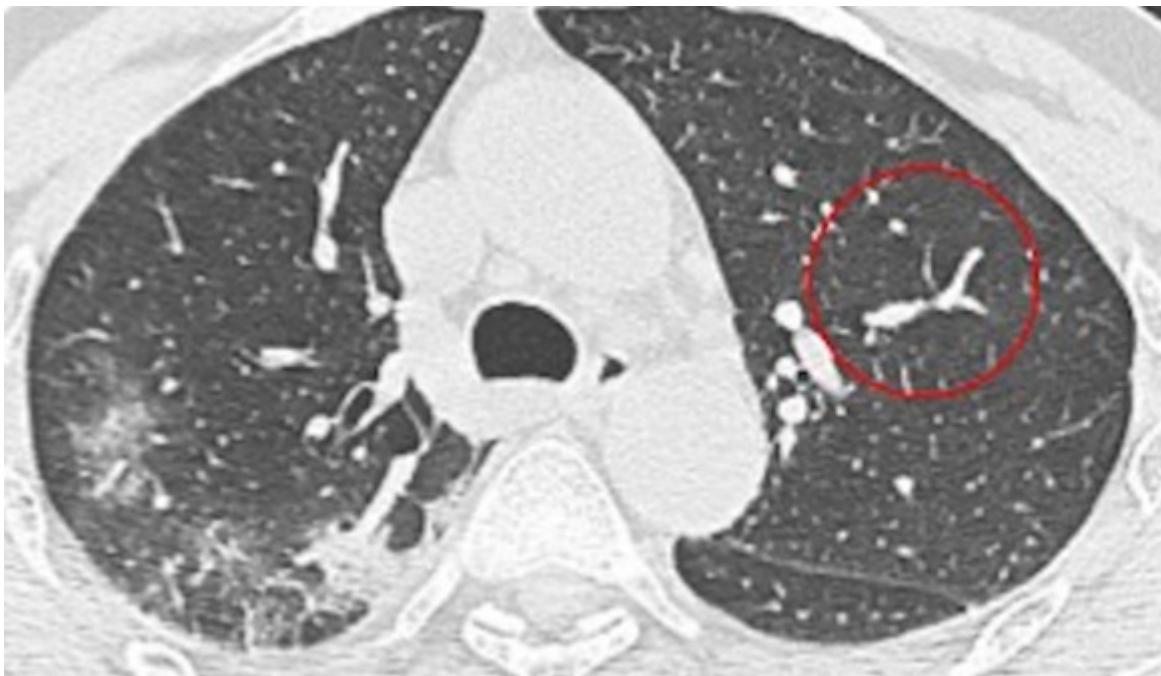
1.1.5 植物病理

数据集为**植物病理学2020-FGVC7**, 1820张图像, 地址: <https://www.kesci.com/home/dataset/5e870dc995b029002ca84507/files>。植物病理特征, 图像阴影比较多。



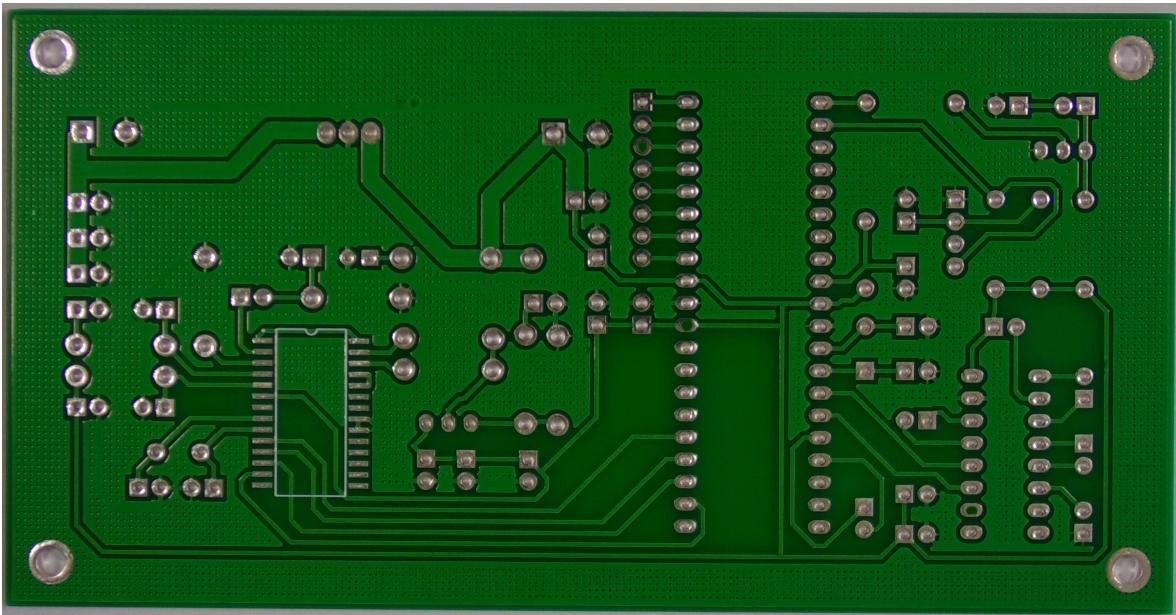
1.2 医学图像

数据集为**COVID-CT**, 349张图像, 地址: <https://github.com/mmmmmmiracle/COVID-CT/tree/master/Images-processed> 。新冠肺炎患者的肺部影像图。



1.3 工业图像

数据集为**PCB-defect-detection**, 12428张图像, 包含六种PCB板的缺陷。地址: https://www.dropbox.com/s/32kolsaa45z2mpj/PCB_DATASET.zip?dl=0 。



1.4 多机数据

用两种手机拍摄的图像，拍摄机器：oppo手机(1080 * 1440)，一加手机(1080, 2160)。

1.5 MNIST

手写数字数据集，共七万张图像。

2. 图像增强

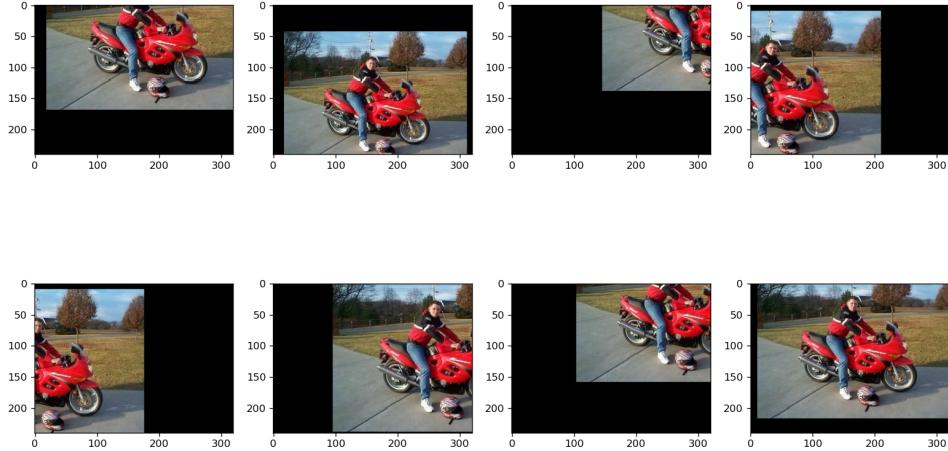
[实验代码](#)

2.1 平移

利用如下矩阵可以计算出平移前后坐标的映射关系。

$$\begin{bmatrix} x_1 \\ y_1 \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & d_x \\ 0 & 1 & d_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_0 \\ y_0 \\ 1 \end{bmatrix}$$

其中 `dx`, `dy` 为坐标轴两个方向的平移距离。



2.2 Resize

- 最近邻插值：

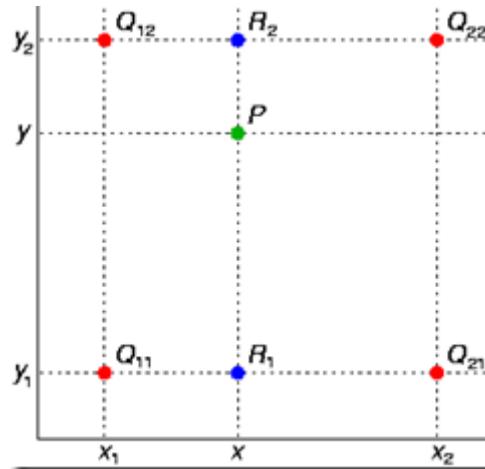
$$x_0 = x_1 * \frac{width_0}{width_1}$$

$$y_0 = y_1 * \frac{height_0}{height_1}$$

首先得到目标图与原图像的坐标映射关系，然后在原图中找到一点 p ，使得改点到 (x_0, y_0) 的欧式距离最短。将该点的值赋给目标图 (x_1, y_1) 的像素值。

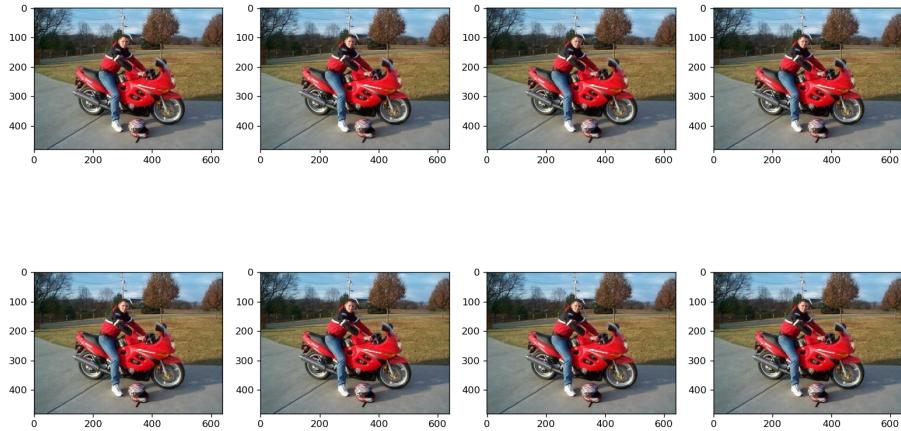
- 双线性插值：

假设目标图某一点 N 的坐标映射到原图像的 P 处：



则 N 点的像素值可以由如下公式计算得到：

$$f(N) = \frac{(x_2 - x)(y_2 - y)}{(x_2 - x_1)(y_2 - y_1)} f(Q_{11}) + \frac{(x - x_1)(y_2 - y)}{(x_2 - x_1)(y_2 - y_1)} f(Q_{21}) + \frac{(x_2 - x)(y - y_1)}{(x_2 - x_1)(y_2 - y_1)} f(Q_{12}) + \frac{(x - x_1)(y_2 - y)}{(x_2 - x_1)(y_2 - y_1)} f(Q_{22})$$



2.3 旋转

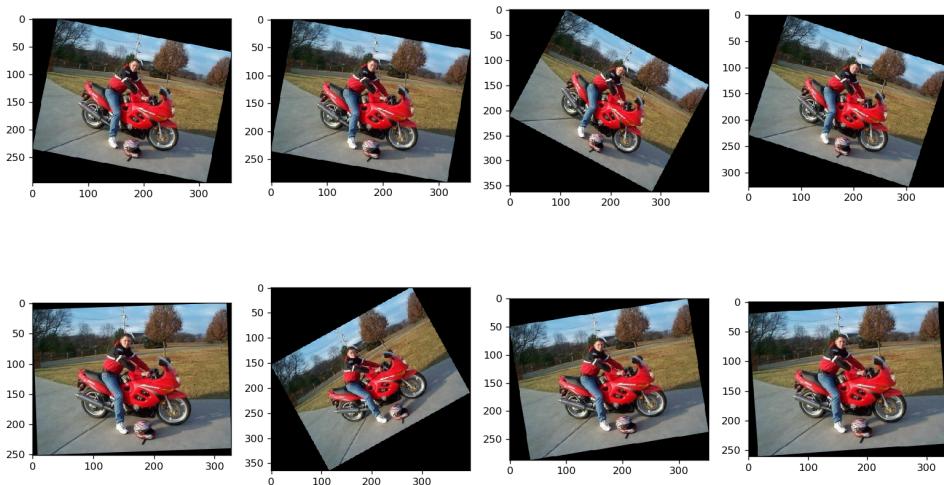
1. 首先计算得到旋转后的图像的大小:

```
[m, n, o] = size(img);
new_m = ceil(abs(m*cosd(degree)) + abs(n*sind(degree)));
new_n = ceil(abs(n*cosd(degree)) + abs(m*sind(degree)));
```

2. 然后通过如下公式计算得到原图与目标图的坐标映射关系:

$$[x_0, y_0, 1] = [x_1, y_1, 1] \begin{bmatrix} 1 & 0 & 0 \\ 0 & -1 & 0 \\ -0.5 * W_1 & 0.5 * H_1 & 1 \end{bmatrix} \begin{bmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & -1 & 0 \\ -0.5 * W_0 & 0.5 * H_0 & 1 \end{bmatrix}$$

其中 W_1, H_1 为目标图的宽和高, W_0, H_0 为原图像的宽和高, θ 为旋转角度

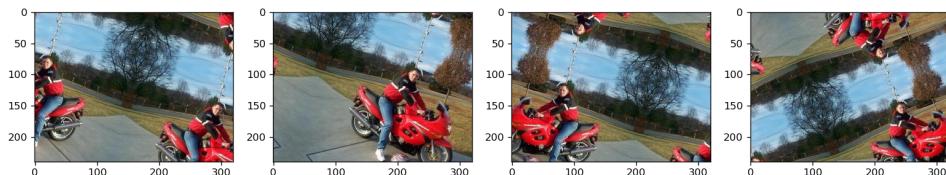
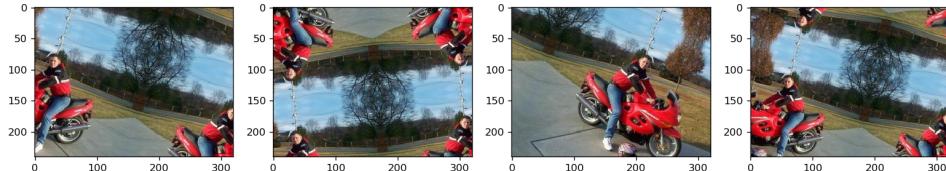


2.4 仿射变换

相当于对于图像做了一个平移、旋转、缩放、剪切、对称, 计算矩阵如下:

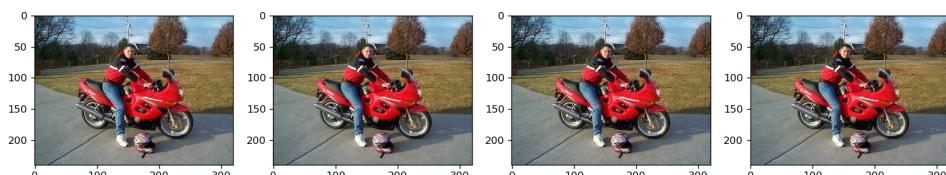
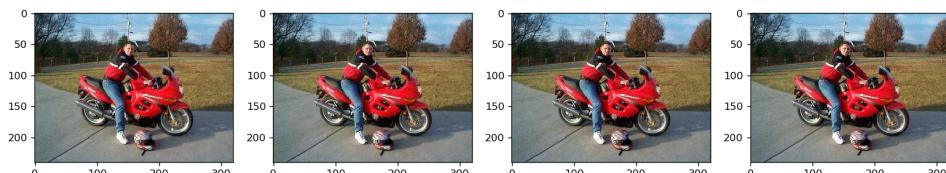
$$\begin{bmatrix} x_0 \\ y_0 \\ 1 \end{bmatrix} = \begin{bmatrix} a_1 & a_2 & d_x \\ a_3 & a_4 & d_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ y_1 \\ 1 \end{bmatrix}$$

仿射变换保持了二维图像的“平直性”和“平行性”，直线之间的相对位置关系保持不变，平行线经仿射变换后依然为平行线，且直线上点的位置顺序不会发生改变。



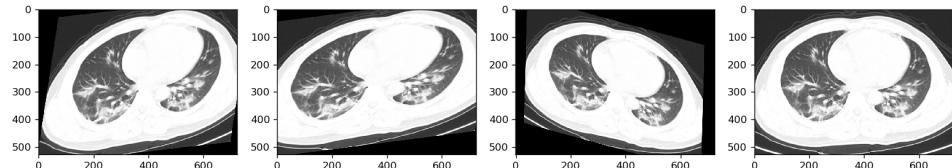
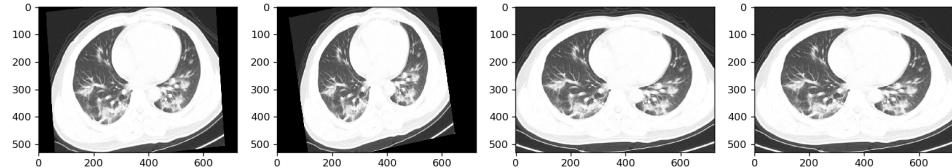
2.5 高斯噪声

随机在图像上加入高斯噪声。



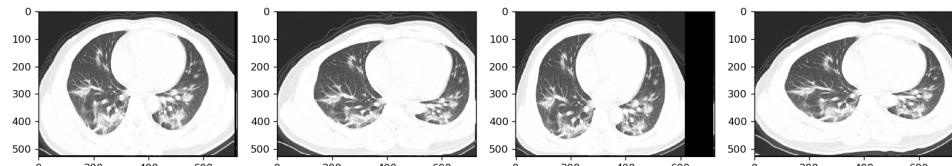
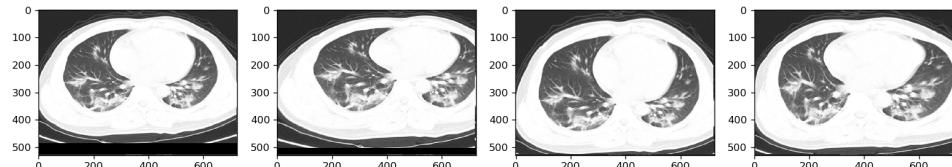
2.6 弹性形变

- 首先需要对图像中的每个像素点 (x, y) 产生两个 $-1 \sim 1$ 之间的随机数， Δx 和 Δy ，分别表示该像素点的 x 方向和 y 方向的移动距离；
- 生成一个以 0 为均值，以 σ 为标准差的高斯核 k_{nn} ，并用前面的随机数与之做卷积，并将结果作用于原图像。



2.7 网格畸变

将图像分成一定数量的小网格图像，对每个小网格进行畸变处理。



3. 图像配准

[实验代码](#)

3.1 配准方法

3.1.1 sift

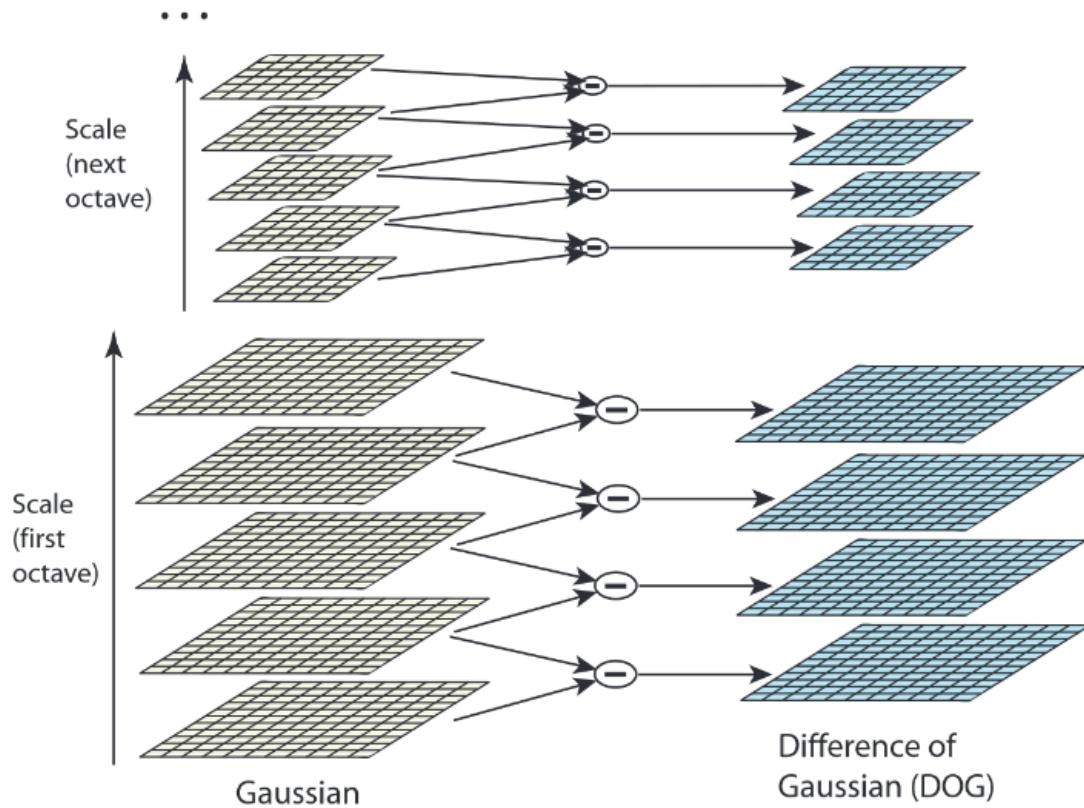
建立高斯差分金字塔

$$L(x, y, \sigma) = G(x, y, \sigma) * I(x, y) \quad (1)$$

$$G(x, y, \sigma) = \frac{1}{2\pi\sigma^2} e^{-(x^2+y^2)/2\sigma^2} \quad (2)$$

$$\begin{aligned} D(x, y, \sigma) &= (G(x, y, k\sigma) - G(x, y, \sigma)) * I(x, y) \\ &= L(x, y, k\sigma) - L(x, y, \sigma) \end{aligned} \quad (3)$$

- 对同一组内的大小相同的图片，利用不同的高斯核模糊图像，以此来模拟物体在实际中远近不同对人眼的影响。
- 然后对同一组内的相邻高斯核的图像相减。如下图所示：



- 最后进行 `downsampling`，得到小一些的图像，重复上两步。
- 最后可以得到多组 `DOG`，以此来近似空间位置不变性。

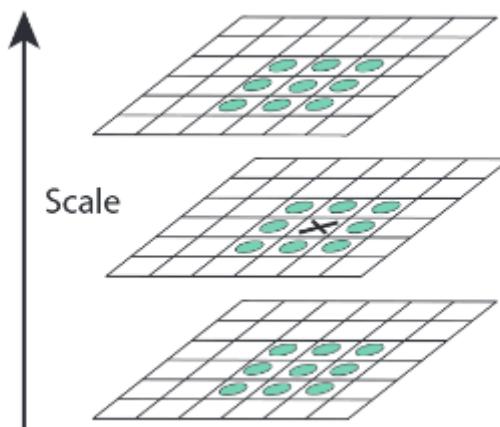
关键点位置确定（极值点）

- 对差分图像进行阈值化：

$$\text{abs(val)} > 0.5 * T / n \quad T = 0.04$$

其中 n 为 DOG 的组数

- 在高斯差分金字塔中找极值，如下图所示：



中心点需要同周围26个点比较（假设kernel size 为 $3*3$ ）。这样可以找到近似的极值点位置，但是可能同实际的极值点位置有所偏差，可以对极值点位置进行二阶泰勒展开，微调极值点的位置。

- 调整极值点位置：

在检测到的极值点 $X_0(x_0, y_0, \sigma_0)^T$ 处做三元二阶泰勒展开

$$f\left(\begin{bmatrix} x \\ y \\ \sigma \end{bmatrix}\right) = f\left(\begin{bmatrix} x_0 \\ y_0 \\ \sigma_0 \end{bmatrix}\right) + \left[\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}, \frac{\partial f}{\partial \sigma}\right] \left(\begin{bmatrix} x \\ y \\ \sigma \end{bmatrix} - \begin{bmatrix} x_0 \\ y_0 \\ \sigma_0 \end{bmatrix}\right)$$

$$+ \frac{1}{2} \left(\begin{bmatrix} x \\ y \\ \sigma \end{bmatrix} - \begin{bmatrix} x_0 \\ y_0 \\ \sigma_0 \end{bmatrix}\right)^T \begin{bmatrix} \frac{\partial^2 f}{\partial x \partial x}, \frac{\partial^2 f}{\partial x \partial y}, \frac{\partial^2 f}{\partial x \partial \sigma} \\ \frac{\partial^2 f}{\partial x \partial y}, \frac{\partial^2 f}{\partial y \partial y}, \frac{\partial^2 f}{\partial y \partial \sigma} \\ \frac{\partial^2 f}{\partial x \partial \sigma}, \frac{\partial^2 f}{\partial y \partial \sigma}, \frac{\partial^2 f}{\partial \sigma \partial \sigma} \end{bmatrix} \left(\begin{bmatrix} x \\ y \\ \sigma \end{bmatrix} - \begin{bmatrix} x_0 \\ y_0 \\ \sigma_0 \end{bmatrix}\right)$$

矢量形式: $f(X) = f(X_0) + \frac{\partial f^T}{\partial X} \hat{X} + \frac{1}{2} \hat{X}^T \frac{\partial^2 f}{\partial X^2} \hat{X}$

求函数导数, 令导数为0, 解得:

$$\hat{X} = -\frac{\partial^2 f^{-1}}{\partial X^2} \frac{\partial f}{\partial X}$$

将解带入 $f(x)$, 得到极值点:

$$\begin{aligned} f(X) &= f(X_0) + \frac{\partial f^T}{\partial X} \hat{X} + \frac{1}{2} \left(-\frac{\partial^2 f^{-1}}{\partial X^2} \frac{\partial f}{\partial X} \right)^T \frac{\partial^2 f}{\partial X^2} \left(-\frac{\partial^2 f^{-1}}{\partial X^2} \frac{\partial f}{\partial X} \right) \\ &= f(X_0) + \frac{\partial f^T}{\partial X} \hat{X} + \frac{1}{2} \frac{\partial f^T}{\partial X} \frac{\partial^2 f^{-T}}{\partial X^2} \frac{\partial^2 f}{\partial X^2} \frac{\partial^2 f^{-1}}{\partial X^2} \frac{\partial f}{\partial X} \\ &= f(X_0) + \frac{\partial f^T}{\partial X} \hat{X} + \frac{1}{2} \frac{\partial f^T}{\partial X} \frac{\partial^2 f^{-1}}{\partial X^2} \frac{\partial f}{\partial X} \\ &= f(X_0) + \frac{\partial f^T}{\partial X} \hat{X} + \frac{1}{2} \frac{\partial f^T}{\partial X} (-\hat{X}) \\ &= f(X_0) + \frac{1}{2} \frac{\partial f^T}{\partial X} \hat{X} \end{aligned}$$

- 舍去低对比度的点:

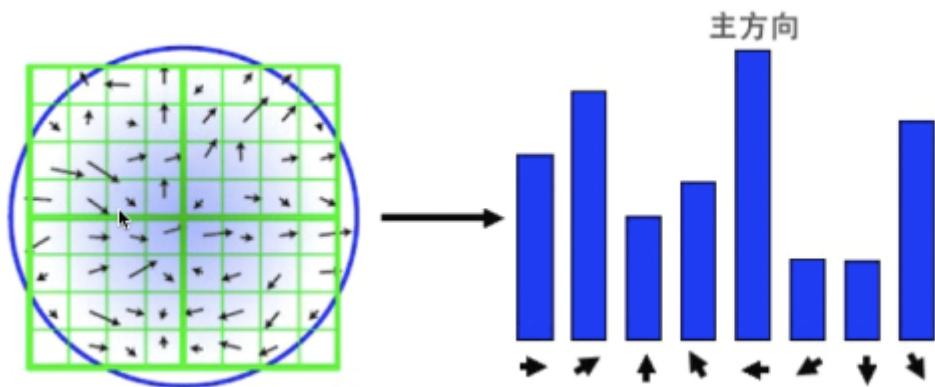
$$\text{若 } |f(X)| < \frac{T}{n} \quad , \text{ 则舍去点 } X$$

- 消除边缘效应(Hessian矩阵):

$$\begin{aligned} \mathbf{H}(x, y) &= \begin{bmatrix} D_{xx}(x, y) & D_{xy}(x, y) \\ D_{xy}(x, y) & D_{yy}(x, y) \end{bmatrix} \\ \text{Tr}(\mathbf{H}) &= D_x + D_y = \alpha + \beta \\ \text{Det}(\mathbf{H}) &= D_x D_{yy} - (D_y)^2 = \alpha \beta \\ \text{其中: } \alpha > \beta \quad \text{且} \quad \alpha = \gamma \beta \\ \text{若 } \text{Det}(\mathbf{H}) < 0 \quad \text{舍去点 } X \\ \frac{\text{Tr}(\mathbf{H})^2}{\text{Det}(\mathbf{H})} &= \frac{(\alpha + \beta)^2}{\alpha \beta} = \frac{(\gamma \beta + \beta)^2}{\gamma \beta^2} = \frac{(\gamma + 1)^2}{\gamma} \\ \text{若不满足 } \frac{\text{Tr}(\mathbf{H})}{\text{Det}(\mathbf{H})} &< \frac{(\gamma + 1)^2}{\gamma} \quad \text{舍去点 } X \text{ (建议取 10.0)} \end{aligned}$$

确定关键点主方向

统计以特征点为圆心, 以该特征点所在的高斯图像的尺度的1.5倍为半径的圆内的所有的像素的梯度方向及其梯度幅值。



构建关键点描述符

在关键点周围区域划分 4×4 个网格，对每个网格统计梯度方向，得到8个方向的数字描述。最终对每个点得到一个128维的向量描述。最后利用最近邻算法(KDTree)找寻最接近的两个描述符，即两张图片中最佳匹配的特征点。

手写代码实现

[CODE](#)

3.1.2 surf

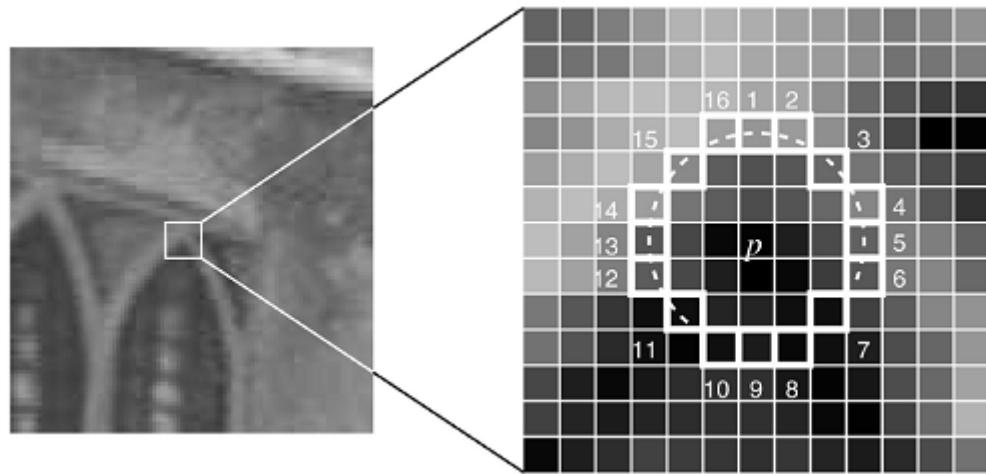
SURF是对SIFT实现上的加速，核心点在于采用积分图对计算加速。Sift采用的是DOG图像，而surf采用的是Hessian矩阵行列式近似值图像。利用箱式滤波器（box filter）简化二维高斯滤波，不需要再进行降采样，通过Haar小波特征设定特征点主方向，构建的特征描述子就是64维的(sift构建的是128维的)。

SIFT特征与SURF特征的比较：

- **构建图像金字塔:** SIFT特征利用不同尺寸的图像与高斯差分滤波器卷积；SURF特征利用原图片与不同尺寸的方框滤波器卷积。
- **特征描述子:** SIFT特征有 $4 \times 4 \times 8 = 128$ 维描述子，SURF特征有 $4 \times 4 \times 4 = 64$ 维描述子
- **特征点检测方法:** SIFT特征先进行非极大抑制，再去除低对比度的点，再通过Hessian矩阵去除边缘响应过大的点；SURF特征先利用Hessian矩阵确定候选点，然后进行非极大抑制
- **特征点主方向:** SIFT特征在正方形区域内统计梯度幅值的直方图，直方图最大值对应主方向，可以有多个主方向；SURF特征在圆形区域内计算各个扇形范围内x、y方向的haar小波响应，模最大的扇形方向作为主方向

3.1.3 fast + brief

fast特征提取：



如上图所示，一个以像素p为中心，半径为3的圆上，有16个像素点（p1、p2、...、p16），FAST算法的过程如下：

Algorithm FAST9-16 特征检测

```

1: 定义一个阈值 t，计算 p1、p9 与中心 p 的像素差
2: if 它们绝对值都小于阈值 then
3:   p 点不是特征点
4: else
5:   计算 p1、p9、p5、p13 与中心 p 的像素差
6:   if 它们的绝对值有至少 3 个小于阈值 then
7:     p 点不是特征点
8:   else
9:     计算 p1 到 p16 这 16 个点与中心 p 的像素差
10:    if 它们的绝对值有至少 9 个小于阈值 then
11:      p 点不是特征点
12:    else
13:      p 是特征点，周围 16 个点中最大的连续 10 个点的绝对值差的最小值，且满足  $x >$  阈值的值
        即为得分，在以特征点 p 为中心的一个邻域（如  $3 \times 3$  或  $5 \times 5$ ）选取得分最高的点。
14:    end if
15:  end if
16: end if

```

接下来，使用ID3算法训练一个决策树，将特征点圆周上的16个像素输入决策树中，以此来筛选出最优的FAST特征点。接着，非极大值抑制去除局部较密集特征点。使用非极大值抑制算法去除临近位置多个特征点的问题。为每一个特征点计算出其响应大小。计算方式是特征点P和其周围16个特征点偏差的绝对值和。在比较临近的特征点中，保留响应值较大的特征点，删除其余的特征点。

建立金字塔，来实现特征点的多尺度不变性。设置一个比例因子 `scaleFactor`（opencv默认为1.2）和金字塔的层数 `nlevels`（opencv默认为8）。将原图像按比例因子缩小成 `nlevels` 幅图像。缩放后的图像为：`I' = I/scaleFactor (k=1, 2, ..., nlevels)`。`nlevels` 幅不同比例的图像提取特征点总和作为这幅图像的 FAST 特征点。

brief特征描述：

BRIEF算法计算出来的是一个二进制串的特征描述符。它是在一个特征点的邻域内，选择n对像素点 `pi、qi (i=1, 2, ..., n)`。然后比较每个点对的灰度值的大小。如果 `I(pi) > I(qi)`，则生成二进制串中的1，否则为0。所有的点对都进行比较，则生成长度为n的二进制串。一般n取128、256或512，opencv默认为256。另外，为了增加特征描述符的抗噪性，算法首先需要对图像进行高斯平滑处理。

在旋转不是非常厉害的图像里，用BRIEF生成的描述子的匹配质量非常高。但在旋转大于30°后，BRIEF的匹配率快速降到0左右。steered BRIEF对BRIEF旋转不变性做了改进，rBRIEF改进了特征点描述子的相关性。

- steered BRIEF：在使用BRIEF特征描述时，要将图像转换到相应的尺度图像上，然后在尺度图像上的特征点处取SxS邻域，然后选择点对并旋转，得到二进制串描述符。
- rBRIEF：使用steeredBRIEF方法得到的特征描述子具有旋转不变性，但是却在描述符的可区分性不如原始的BRIEF算法。这个性质对特征匹配的好坏影响非常大。描述子是特征点性质的描述。描述子表达了特征点不同于其他特征点的区别。描述子要尽量的表达特征点的独特性。如果不同特征点的描述子的可区分性比较差，匹配时不容易找到对应的匹配点，引起误匹配。

3.1.4 orb

ORB算法分为两部分，分别是特征点提取和特征点描述。特征提取是由FAST (Features from Accelerated Segment Test) 算法发展来的，特征点描述是根据BRIEF (Binary Robust Independent Elementary Features) 特征描述算法改进的。

ORB算法提出使用矩 (moment) 法来确定FAST特征点的方向。也就是说通过矩来计算特征点以r为半径范围内的质心，特征点坐标到质心形成一个向量作为该特征点的方向。矩定义如下：

$$m_{pq} = \sum x^p y^q I(x, y)$$

其中， $I(x, y)$ 为图像灰度表达式。该矩的质心为 $(m_{10}/m_{00}, m_{01}/m_{00})$

假设角点坐标为O，则向量的角度即为该特征点的方向。计算公式

$$\theta = \arctan(m_{10}/m_{01})$$

为了解决描述子的可区分性和相关性的问题，ORB使用统计学习的方法来重新选择点对集合。首先建立300k个特征点测试集。对于测试集中的每个点，考虑其31x31邻域。不同于原始BRIEF算法的地方是，在对图像进行高斯平滑之后，使用邻域中的某个点的5x5邻域灰度平均值来代替某个点对的值，进而比较点对的大小。这样特征值更加具备抗噪性。另外使用积分图像加快求取5x5邻域灰度平均值的速度。

在31x31的邻域内共有 $(31-5+1) \times (31-5+1) = 729$ 个这样的子窗口，取点对的方法共有M=265356种，要在这些M种方法中选取256种取法，选择的原则是这256种取法之间的相关性最小，选取步骤如下：

- 在300k特征点的每个31x31邻域内按M种方法取点对，比较点对大小，形成一个 $300k \times M$ 的二进制矩阵Q。矩阵的每一列代表300k个点按某种取法得到的二进制数。
- 对Q矩阵的每一列求取平均值，按照平均值到0.5的距离大小重新对Q矩阵的列向量排序，形成矩阵T。
- 将T的第一列向量放到R中。
- 取T的下一列向量和R中的所有列向量计算相关性，如果相关系数小于设定的阈值，则将T中的该列向量移至R中。
- 按照上一步的方式不断进行操作，直到R中的向量数量为256。

3.2 实验结果

实验环境：

- windows10 + python3.7.4 + opencv-contrib-python 3.4.2.17 + albumentations 0.4.5
- 处理器：i7-6700HQ 2.6HZ，四核八线程
- 内存：16GB

3.2.1 耗时对比

实验数据为 `mnist` 数据集，并将其重新调整大小 `256*256`。

- `detectors and descriptors : SIFT, SURF, ORB, STAR + BRIEF`
- `matching methods: BRUTE-FORCE, FLANN(Fast Library for Approximate Nearest Neighbors)`

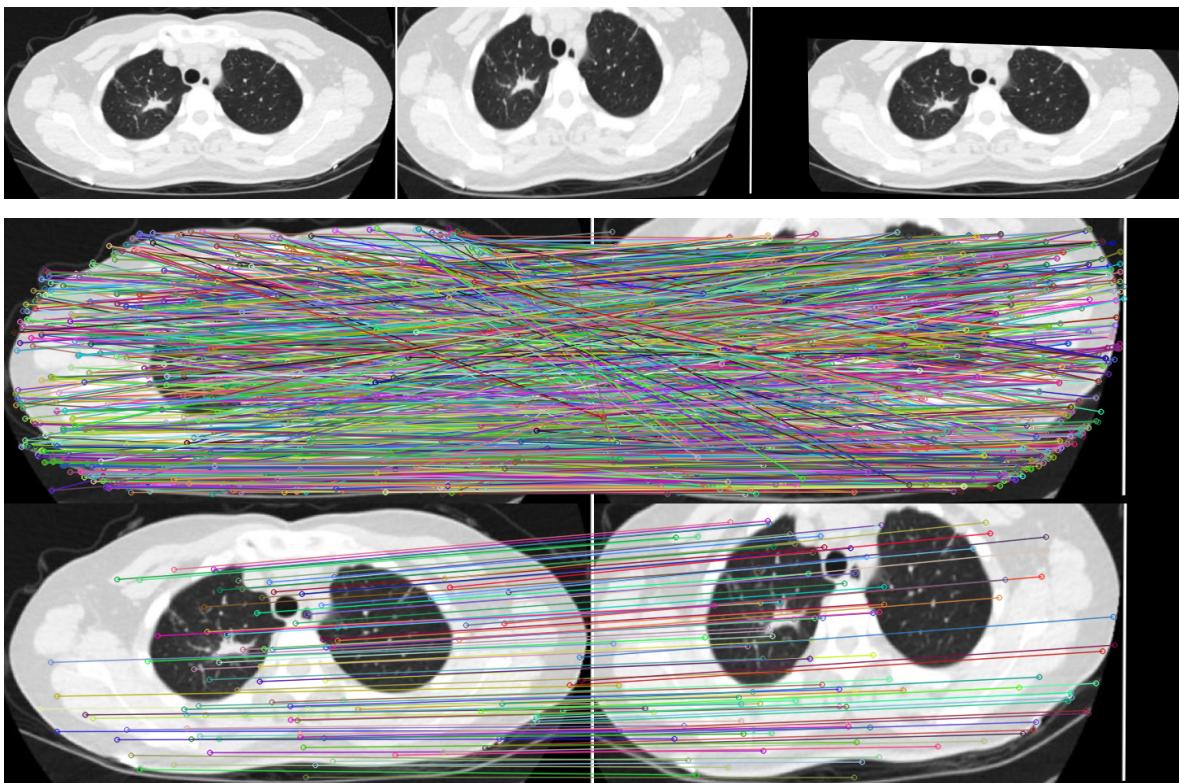
	BRUTE-FORCE	FLANN
SIFT	606.758 s	678.236
SURF	515.923 s	576.093
ORB	292.548 s	-
FAST + BRIEF	258.984 s	-

FLANN 只能适用于 SIFT , SURF

3.2.2 配准结果

医学图像

数据增强方式：弹性形变(Elastic Transform)

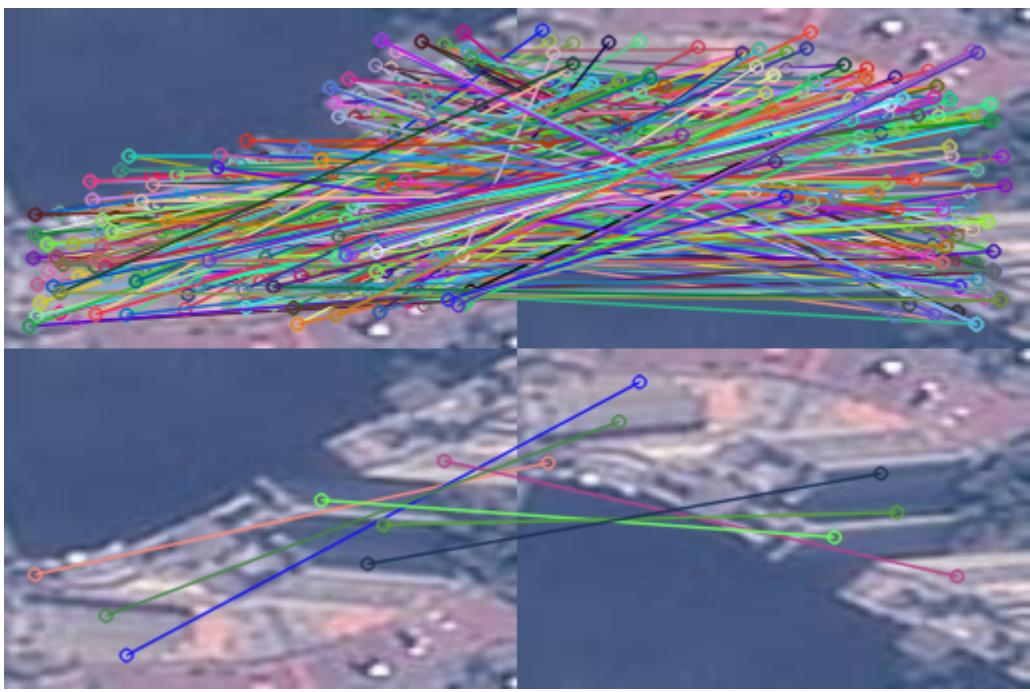


自然场景

高空摄影(雾)

数据增强方式：垂直翻转





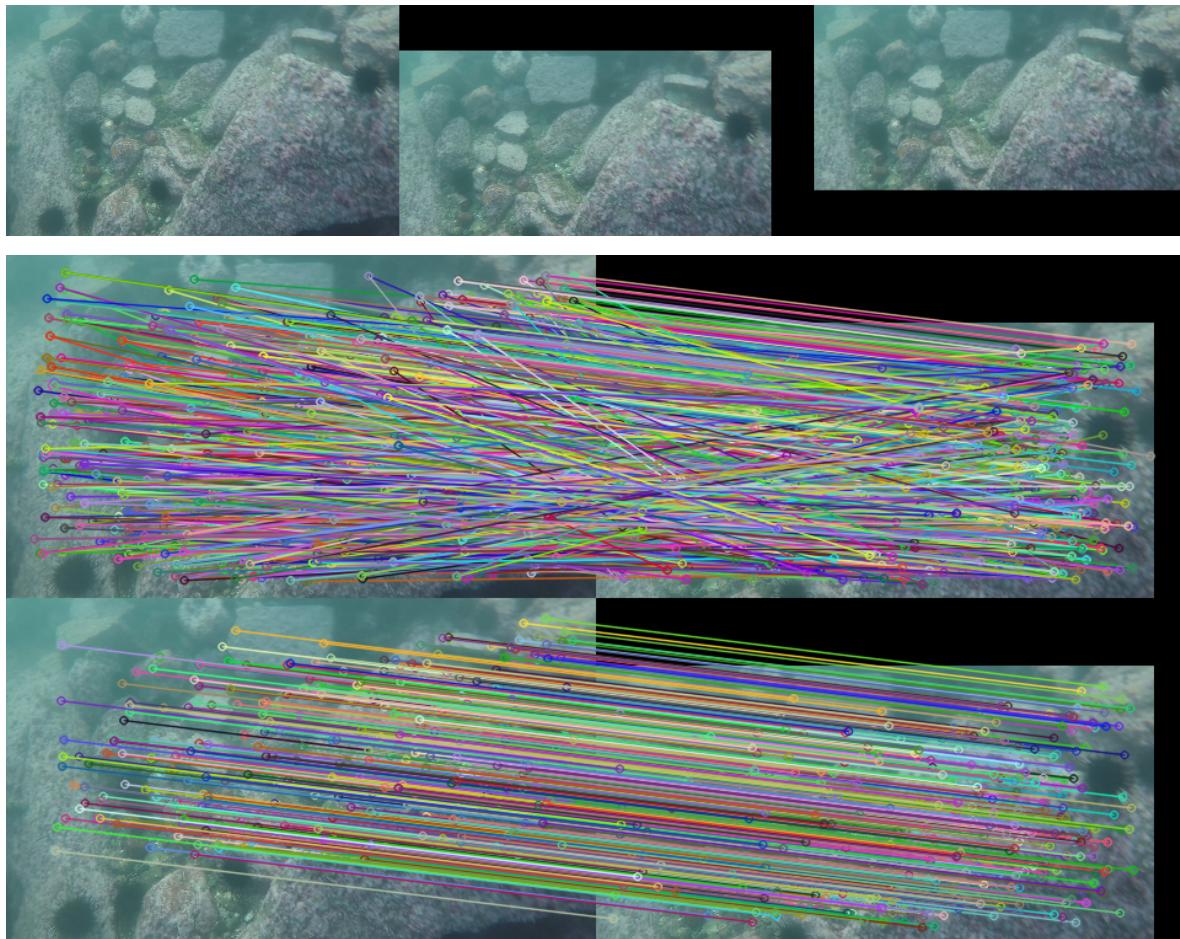
地面摄影(正常场景)

数据增强方式：水平翻转



水下摄影(雾)

数据增强方式：平移



自然纹理(纹理特征)

数据增强方式：仿射变换

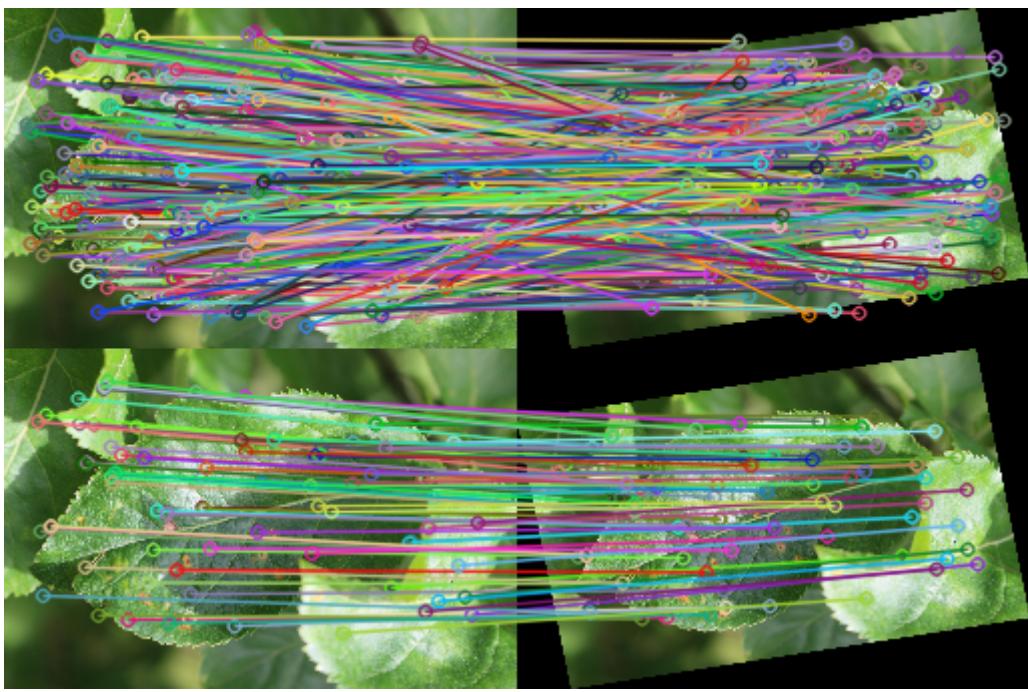




植物病理(阴影)

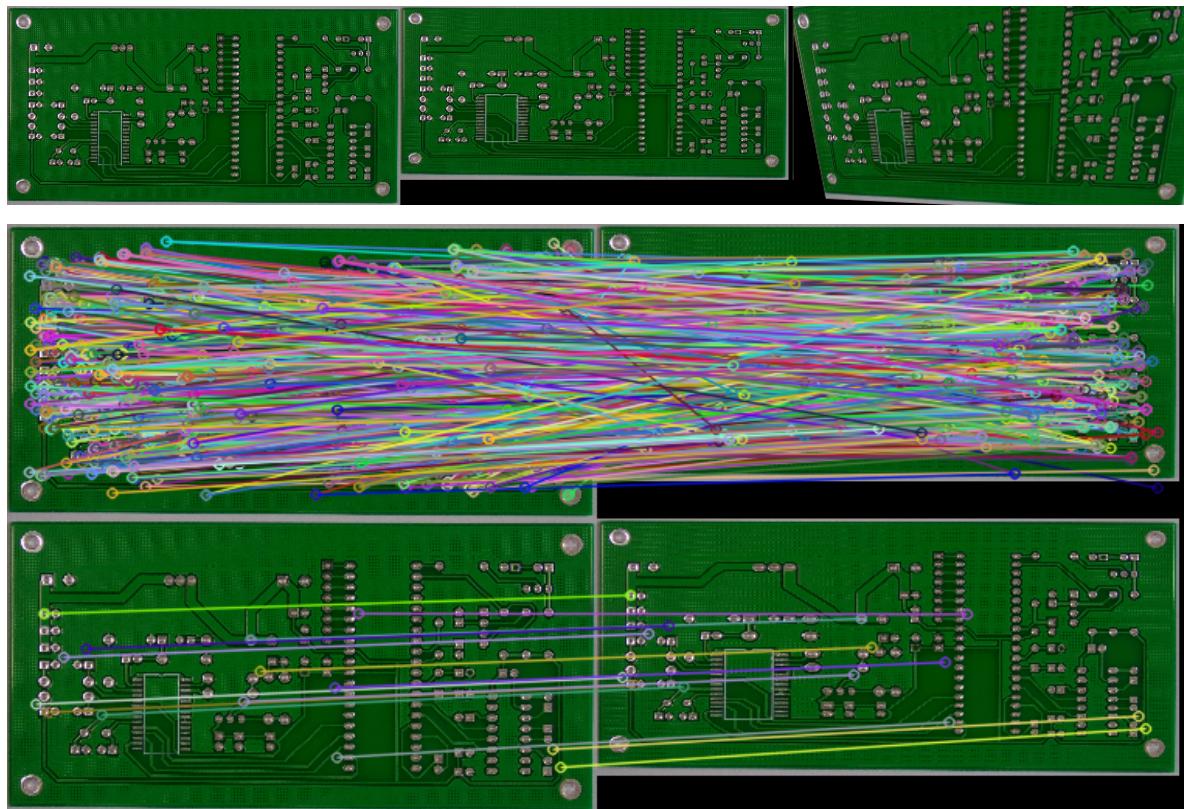
数据增强方式：旋转





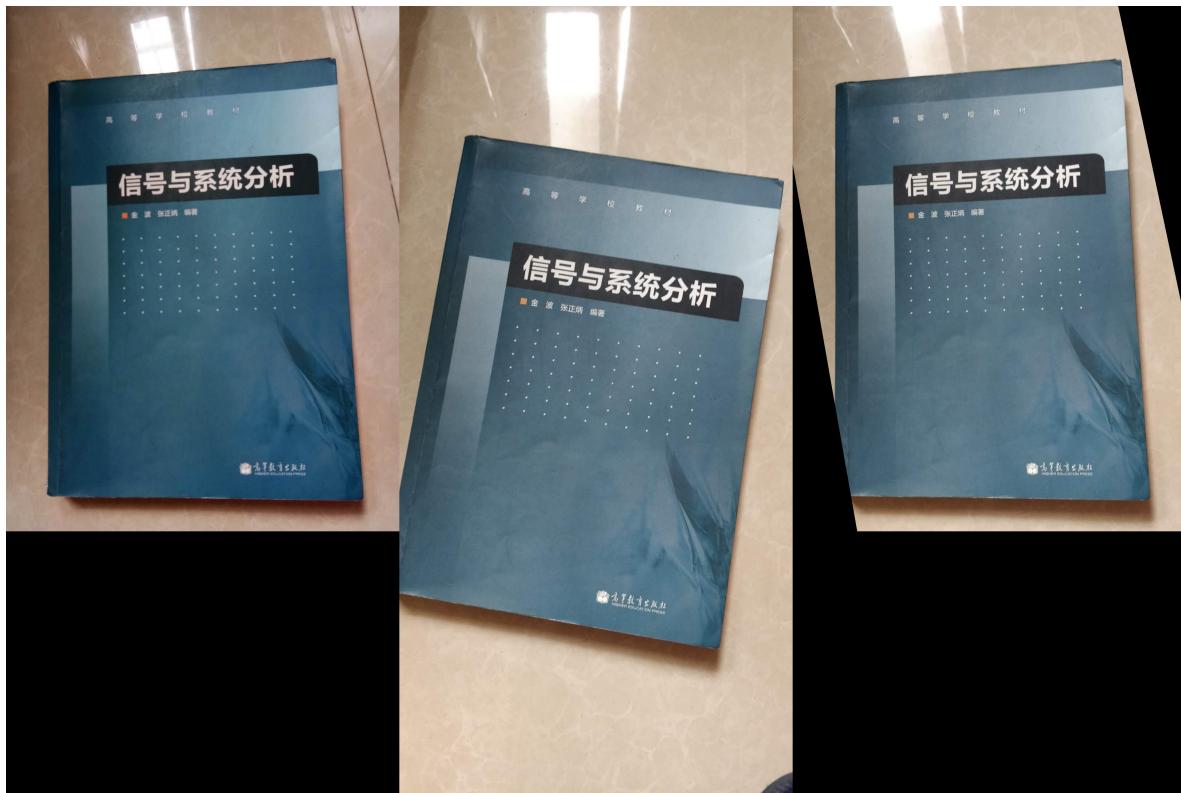
工业图像

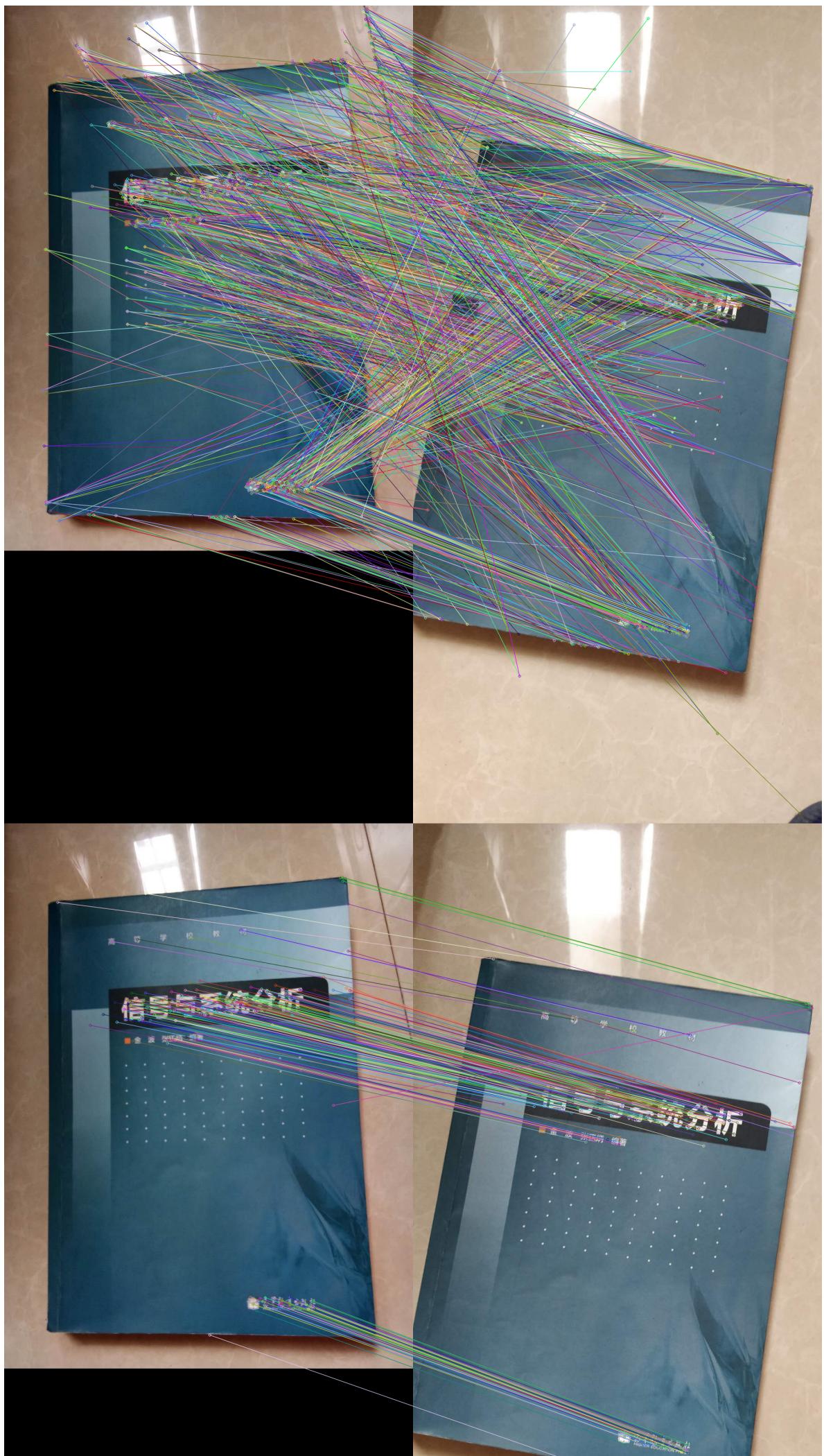
数据增强方式: 网格畸变(GridDistortion)



多机数据

拍摄机器: oppo手机(1080 * 1440), 一加手机(1080, 2160)

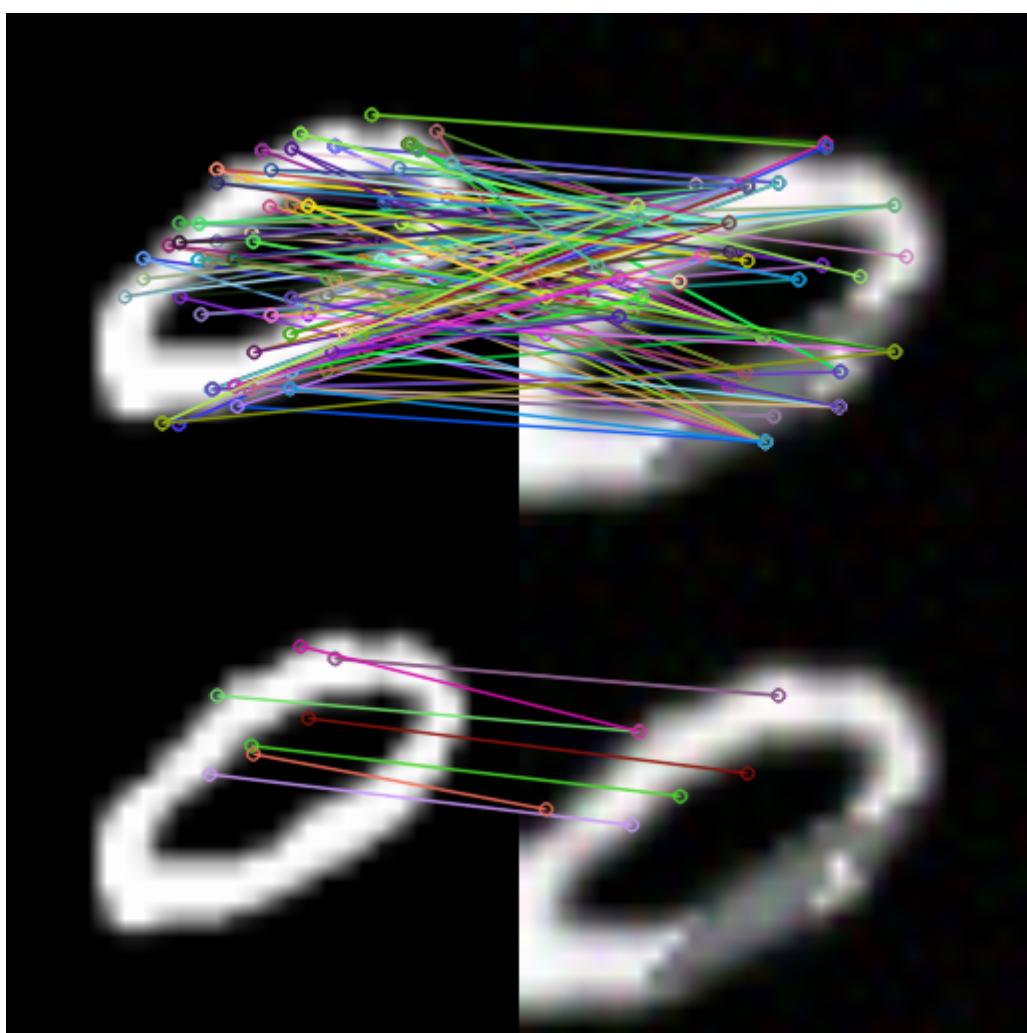






mnist

数据增强方式：平移 + 缩放 + 旋转 + 雾化 + 阴影 + 高斯噪声



References

OPENCV DOC

[Feature Detection and Description](#)

[Feature Matching](#)

[Feature Matching + Homography to find Objects](#)

BLOGS

[SIFT图像匹配技术详细指南](#)

[Python进行SIFT图像对准](#)

[特征点检测： Harris, SIFT, SURF, ORB](#)

[图像配准算法大总结](#)

PAPER

[SIFT](#)

[SURF](#)