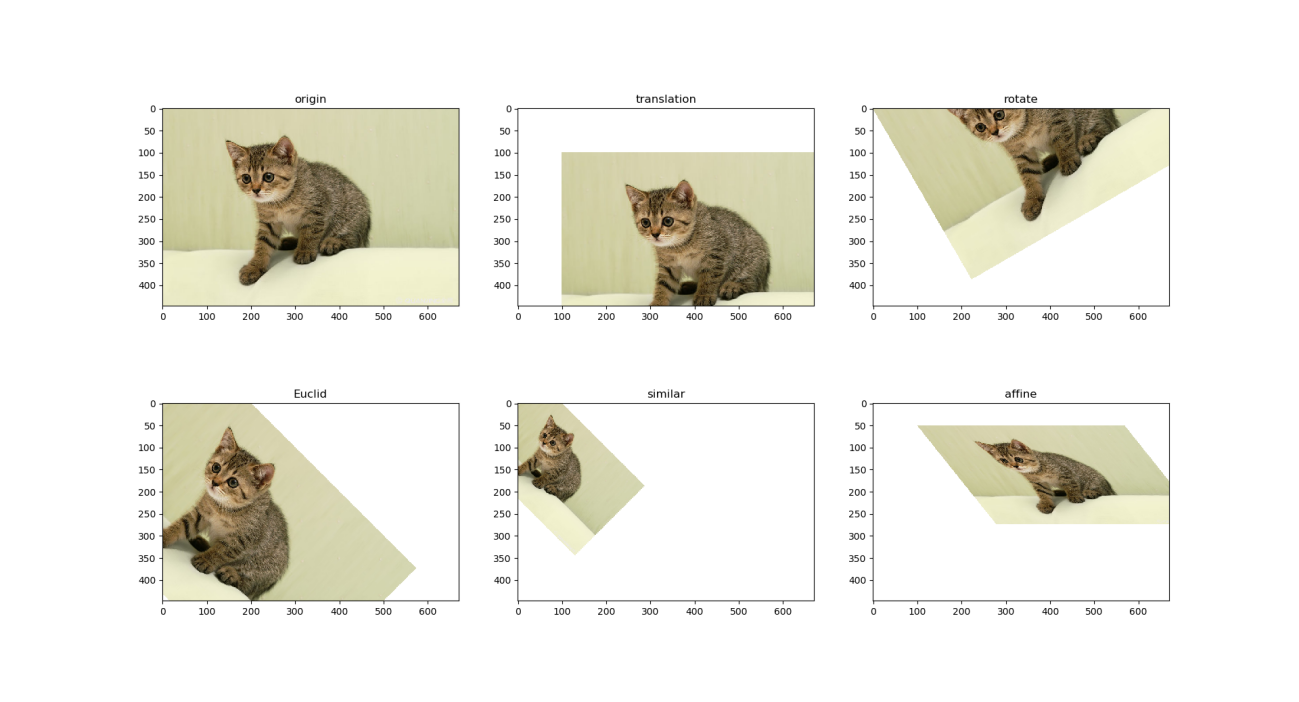
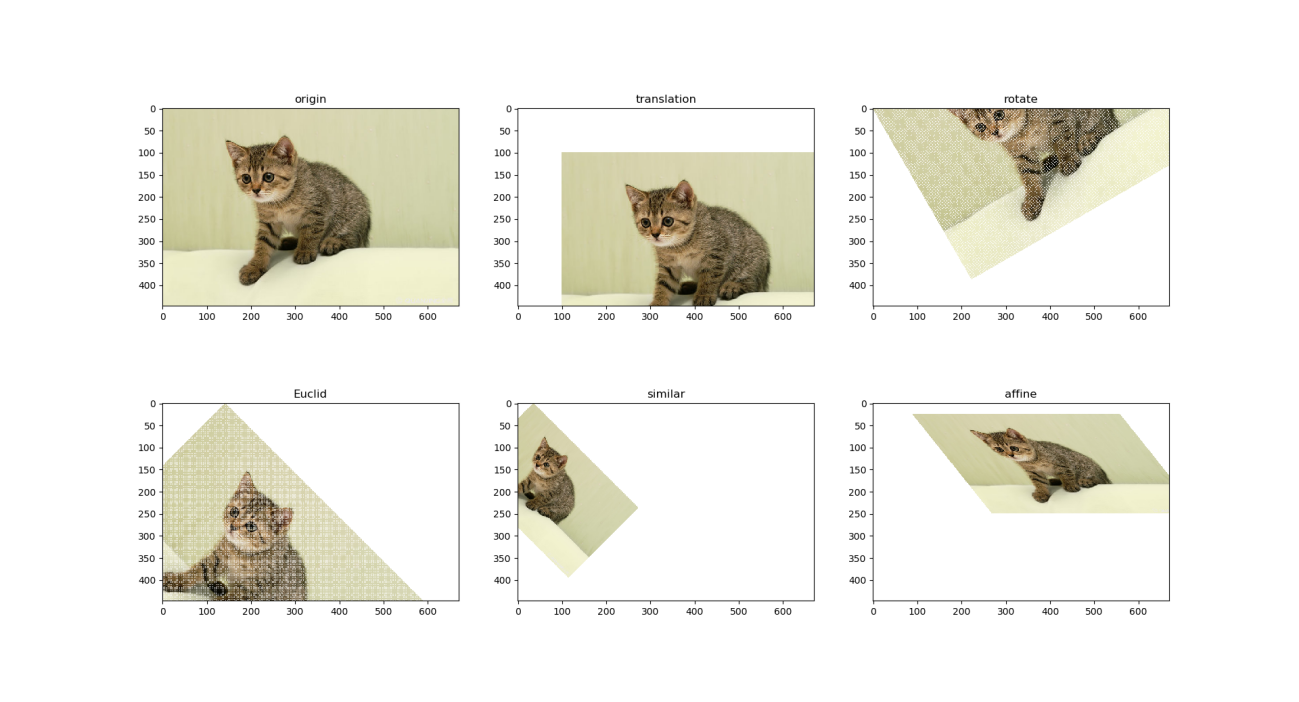
计算机视觉第二次实验 实验报告

人工智能92 陈睿阳 2173214280

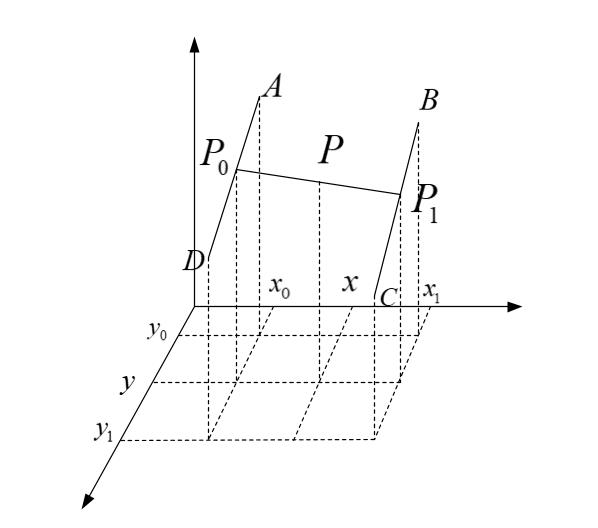
1. 图像的参数化几何变换原理
   1. 图像的几何变换，从某种意义上来讲是将原图中的一个点映射到新的图像中的一个点，即确立一个函数关系：. 对于计算机而言，要处理给定的几种变换，就是要做线性变换（这个很容易理解，因为x’和y’均可以写成x与y的线性组合），自然而然就想到了矩阵.
      1. 下面简单推导一下：
      2. 下面不加证明地给出从平移变换到仿射变换这五种变换的变换矩阵（其实就是把方程组写成矩阵的形式）：
         1. 平移变换
         2. 旋转变换
         3. 欧式变换（平移加旋转）
         4. 相似变换
         5. 仿射变换
      3. 下面给出几何变换的结果图



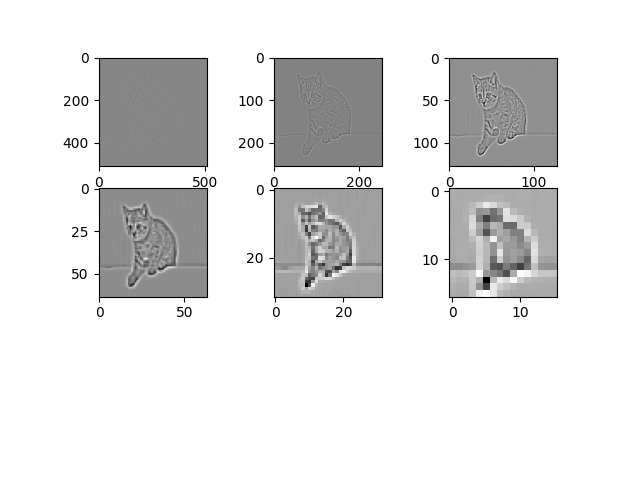
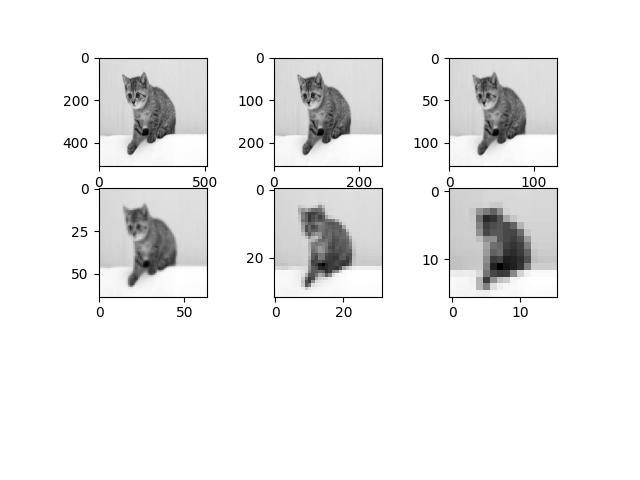
* + - 1. 上图从左到右、从上到小是对如左上所示的小猫，依次进行平移、旋转、欧式变换、相似变换以及仿射变换的结果，具体参数详见附录的geo\_inverse.py文件.
  1. 图像的前向变换以及逆向变换的从本质上来讲，是映射与反映射的关系. 前向变换是将原图中的一个点映射到新的图像中的一个点，而逆向变换是寻找新的图像某一点是由原图中哪一点映射而来的. 听起来可能觉得它们差不多，但是它们却有本质的区别，下面用一个例子说明：



* + 1. 上图是将前述几何变换采用前向变换的结果，内插方法为最近邻插值. 显而易见的是在旋转以及欧式变换的结果中，均出现了白色的花纹. 白色花纹代表在其中的点没有找到原图中对应的点. 那为什么在后向变换中没有这个问题，前向变换中就出现了呢？其实很简单，举个例子就很容易解释清楚了：
       1. 假设原图中存在两点(2,3),(4,5)，这两点经过旋转变换映射到目标图像的(2.3,3.1),(1.7,2.6)两点，由最近邻可知，这两点实际上会映射到目标图像的(2,3)处. 但注意旋转变换是不改变点的密度的，所以如果原图中的多个点映射到新的图像中的同一个点，那必然会导致像素点密度减少，也就是信息的丢失，白色花纹便代表着这种信息丢失（像素点的值没有被更新，还是255）.
       2. 后向变换为什么不会出现这种情况呢？因为后向变换是给定目标图像中的某点，找在原图中的最近邻，不管怎么样目标图像中的点都会被遍历到，不会出现前向变换中只遍历部分点的情况.
    2. 用数学的思想来看可以参考单射、满射等概念. 在这个例子中前向映射是满射，但不是单射；后向映射是满射也是单射，故是双射，双射保证了图像信息的完整性.
  1. 图像的下抽样原理与内插方法原理
     1. 图像的下抽样原理简单来说就是滤波加抽样，对一幅图像首先要进行高斯滤波使其平滑，平滑过后采取去偶数行、列的策略进行下取样，这样可以保证取样的结果与真实缩放的结果更加相似.
     2. 内插方法是对于非整数坐标像素点的处理，最近邻内插是选取距离坐标点最近的整数点，而双线性插值是根据两个维度的线性函数决定取值，其原理图如下所示：



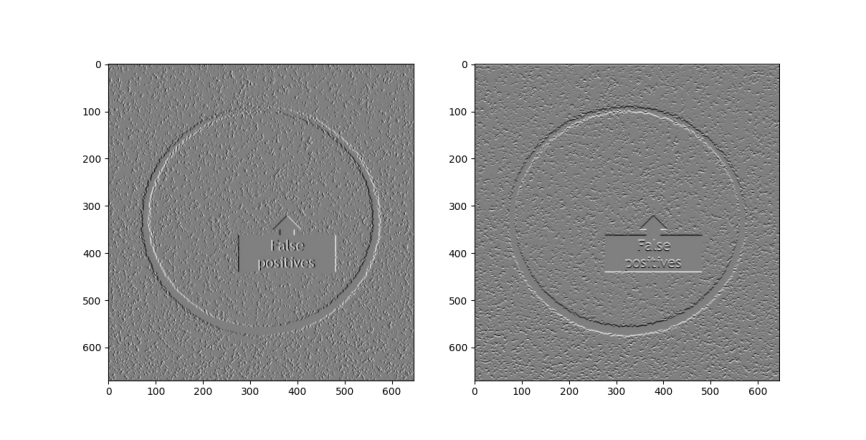
* + 1. 高斯金字塔与拉普拉斯金字塔的图片如下所示：



1. 特征检测
   1. 高斯一阶微分
      1. 如果要提取图像的边缘，一个最直观的想法是要提取图像颜色变化较大的区域，因为边缘往往对应着颜色的分界面，分界面两侧的像素差值可以用梯度来体现.
      2. 图像的高斯一阶微分即图像每一个像素的梯度以及幅角，梯度和幅角的又由在x方向以及y方向的偏导数决定，其关系如下：

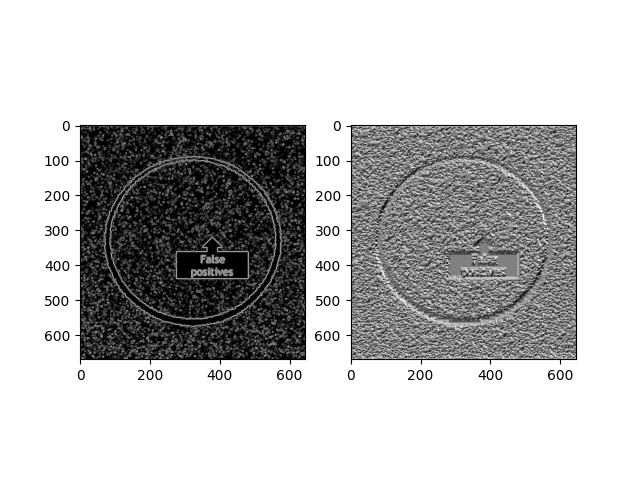
x方向与y方向偏导数的求法不唯一，这里采用Sobel算子求取，即对原图分别用x方向与y方向的Sobel算子进行卷积.

* + 1. x方向与y方向的偏导的结果分别如下所示：

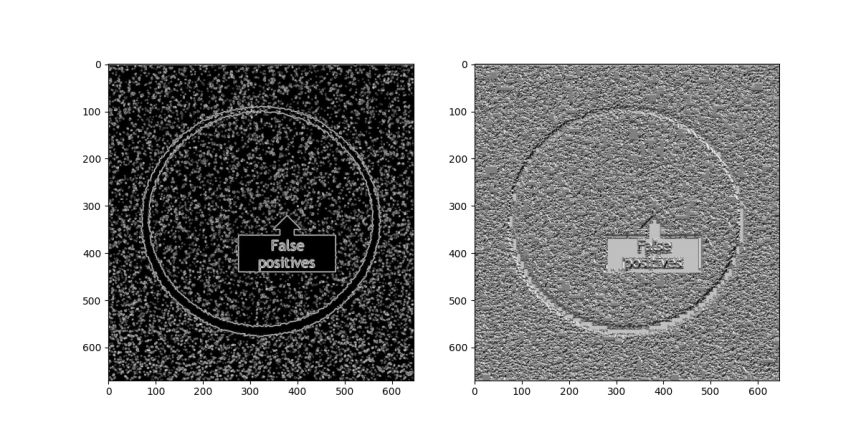


易见x方向的偏导更关注从左到右的变化，而y方向的偏导更关注从上到下的变化

* + 1. 梯度图如下所示：

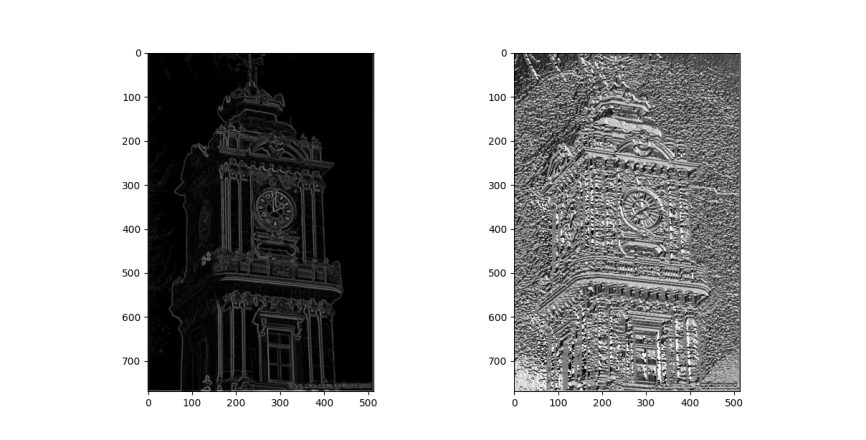


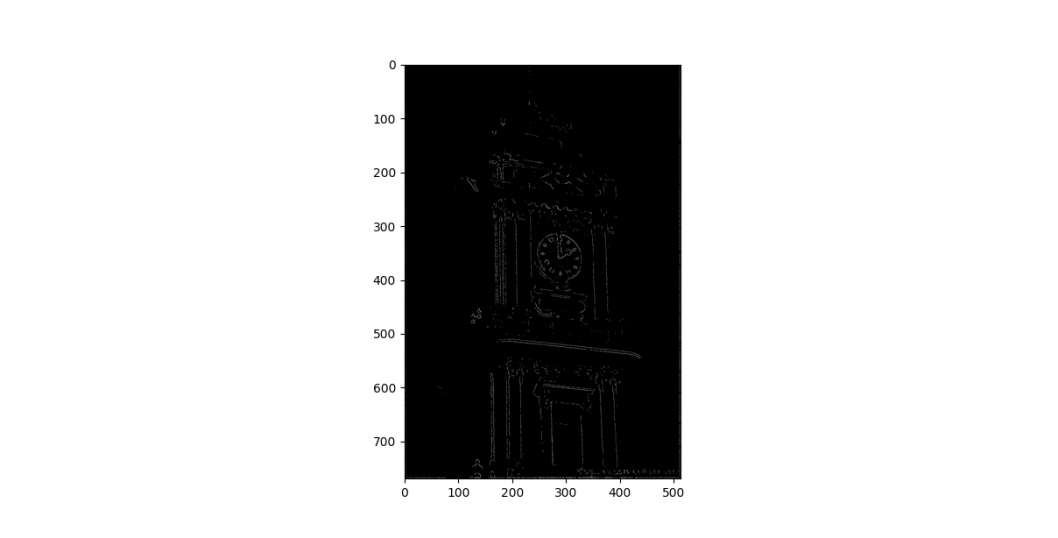
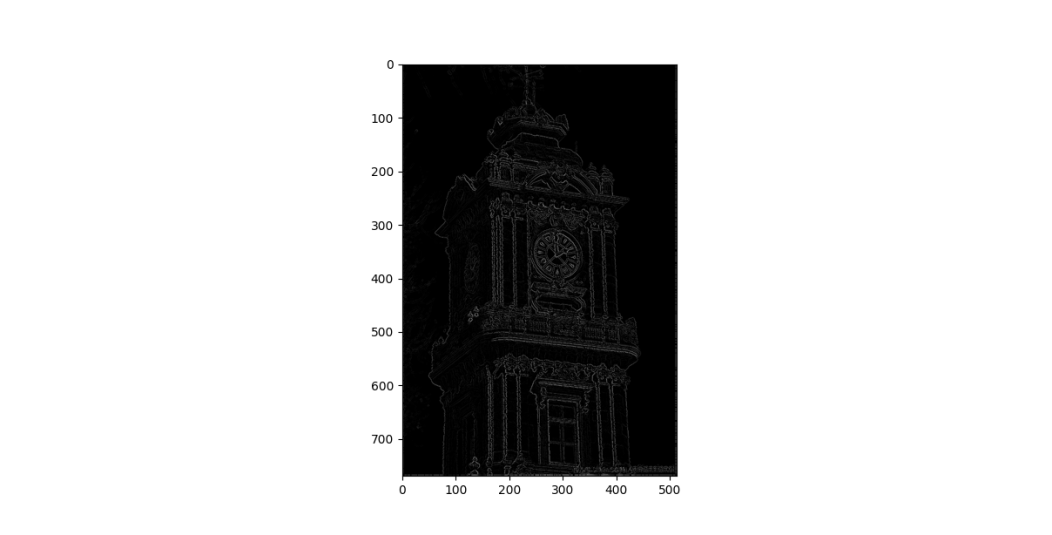
* + 1. 若要考虑不同方差对梯度图的影响，可以采用不同的高斯核对图像进行滤波，下图是将上图高斯核的sigma从1改到0.01的结果：



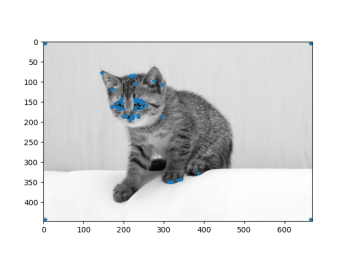
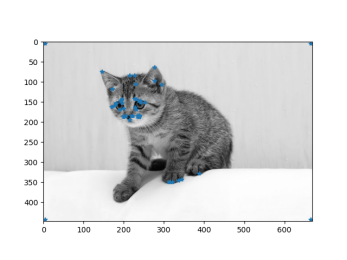
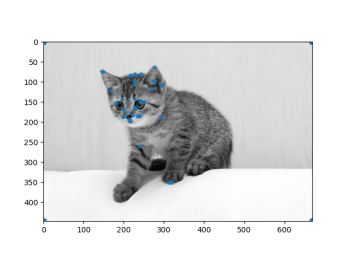
对比可以发现，高斯核方差对图像梯度的影响主要体现在梯度的精度，较大方差的高斯核会使图像更加模糊，梯度的区分度也就越小.

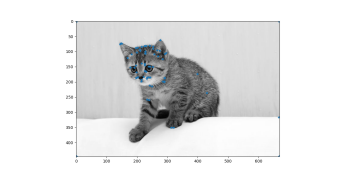
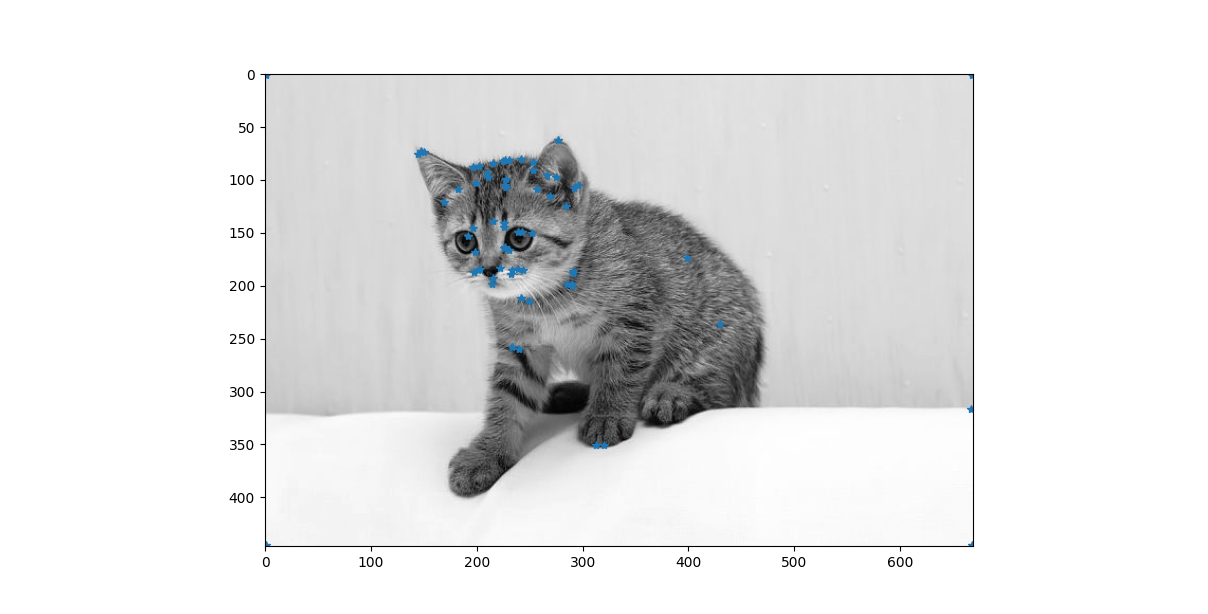
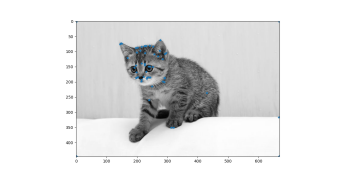
* 1. Canny边缘检测
     1. Canny边缘检测的过程如下：
        1. 对图像进行高斯滤波，使其平滑；
        2. 求得图像的梯度，包括幅值以及幅角；
        3. 对求得的梯度进行非极大抑制，这一步的目的是为了突出边缘；
        4. 采用边缘链接，利用阈值进行筛选.
     2. 边缘检测的结果如下所示，图像依次为梯度图，非极大抑制的结果以及边缘链接的结果：



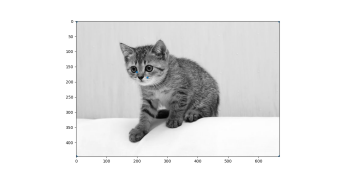
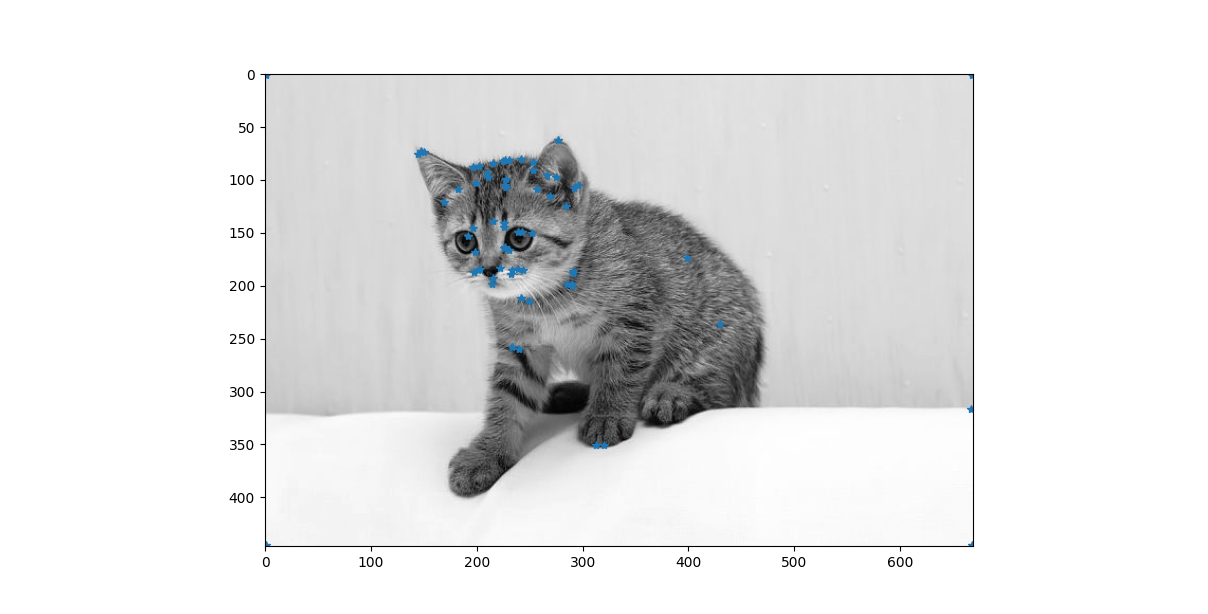
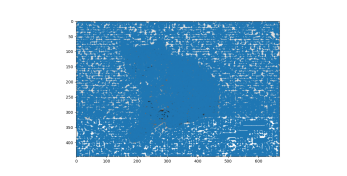


* + 1. 显然经过若干步骤后可以得到相当准确的边缘，对物体的描摹更加精确.
  1. Harris角点检测原理：如果存在任意方向上，都有着较大灰度变化，那么我们可以认为该窗口中存在角点。
     1. 角点检测过程如下：
        1. 计算原图像的x，y两个方向上的梯度；
        2. 将x\*x，y\*y，x\*y方向上的梯度做出乘积；
        3. 对2中三个梯度值进行高斯加权，计算窗口对应的矩阵M；
        4. 计算(x,y)位置上的Harris响应值R；
        5. 进行非极大值抑制。
     2. 实验结果如下：

  
上图从左到右依次为均匀分布的窗口大小从5\*5到7\*7再到9\*9的结果. 当窗口增大时，检测到的特征点就会增加.



上图从左到右依次为高斯窗口（sigma=1）大小从5\*5到7\*7再到9\*9的结果，除了窗口增大检测到的特征点增加之外，sigma=1的高斯窗口相比均匀分布的窗口能检测到更多的特征点.



上图依次为高斯窗口（size=7\*7）sigma从到0.5到1再到2的结果，sigma越大，意味着更平滑的滤波效果，相比而言检测到的特征点就会越少.

注：打印图片不清晰，清晰的图片见电子版

* + 1. 结论：当角点检测中窗口越大时，就有更多的角点被检测出来，但是当窗口过大时，窗口内所包含的内容太多了，无法识别其中的角点；当窗口变小时，图像中也会有一些角点没有被检测出来，因此只有当窗口选择和图像尺度比较匹配时，角点检测效果较为明显。
    2. 角点检测的不变形：具有旋转不变性，因为椭圆转过一定角度后的形状不变，即特征值不变，因此不同角度照片的角点检测出来的角点数量是相近的，提取的还是相同事物的边缘特征。
    3. 角点检测的等变性：由于角点检测精度为像素级的，当图像尺度不发生改变时，角点检测出的角点数量是不变的，但是当进行图像缩放后，检测出的角点数量会随尺度变化和图像内容变化而发生改变。
    4. 角点检测的定位精度：由于角点位于两个边缘的交点处，可以做到精确定位（像素级），甚至能够达到亚像素的精度。