SIFT特征提取实验

实验目的

试编程实现SIFT特征点检测算法，并在下图上进行测试。

实验环境

- OpenCV2
- MatPlotLib

实验原理

高斯差分与金字塔

在SIFT（Scale-Invariant Feature Transform）算法中，为了实现尺度不变性，首先要构建图像的尺度不变性尺度空间。给定一张图像$I(x,y)$，其尺度空间可以表示为

$$L(x,y,\sigma) = K(x,y,\sigma) * I(x,y)$$

其中，$K(x,y,\sigma)$是尺度为σ的卷积核。

研究发现，高斯拉普拉斯尺度空间具有尺度不变性，而高斯差分尺度空间是其良好的近似，且计算简单，高斯差分核定义为：

$$DoG(x,y,\sigma) = G(x,y,k\sigma) - G(x,y,\sigma)$$

其中，$G(x,y,\sigma)$是尺度为σ的高斯核，k是一个常数（通常取$k=\sqrt{2}$）。先求出高斯核的尺度空间$L(x,y,\sigma)$，高斯差分尺度空间$D(x,y,\sigma)$可以表示为：

$$D(x,y,\sigma) = L(x,y,k\sigma) - L(x,y,\sigma)$$

通过对图像进行多层次的模糊处理并计算其高斯差分图，可以构建出尺度空间金字塔，每层金字塔中的图像包含相应尺度的特征信息。

关键点定位

关键点是通过检测高斯差分尺度空间的局部极值点得到的。

26邻域

在定位关键点时，每个像素需要与其26邻域中的所有像素进行比较，26邻域包括当前尺度层的8个邻域像素，以及上一层和下一层的各9个邻域像素。如果一个像素点在其26邻域中同时具有最大或最小的DoG值，则认为它是一个潜在的关键点。

极值点

为了进一步精确定位关键点的位置，SIFT使用泰勒级数展开来近似关键点在尺度空间中的位置。将DoG函数在当前点(x,y,σ)附近进行二阶泰勒展开得到：

$$D(x) \approx D + \frac{\partial D}{\partial x}x + \frac{1}{2}x^T\frac{\partial^2 D}{\partial x^2}x$$

其中，$x=(x,y,\sigma)^T$是位置的偏移向量，$\frac{\partial D}{\partial x}$是DoG在当前点的梯度向量，$\frac{\partial^2 D}{\partial x^2}$是Hessian矩阵。通过拟合并求解上述公式的极值，可以得到关键点的亚像素位置。

不稳定点剔除

在实际应用中，许多检测到的极值点是由噪声或边缘引起的，而非稳定的特征点。

低对比度点

在 SIFT 算法中，为了剔除对比度较低的点，采用了一个简单的阈值判断方法。计算每个潜在关键点的 DoG 值 $D(x)$ 并与设定的阈值 T_c（通常$T_c=0.03$）进行比较。如果 $|D(x)|<T_c$，则认为该关键点为低对比度点，不具有显著的特征信息，可以直接将其舍弃。

边缘点

为了剔除边缘点，SIFT使用一个主曲率比值的判别方法。首先，通过Hessian矩阵的特征值来确定主曲率方向，定义Hessian矩阵为：

$$H = \begin{bmatrix} D_{xx} & D_{xy} \\ D_{xy} & D_{yy} \end{bmatrix}$$

设Hessian矩阵的两个特征值为λ_1和λ_2，则主曲率比可以表示为：

$$\frac{\lambda_1}{\lambda_2}$$

通常设定一个阈值，如果主曲率比值过大，则认为该点是由边缘效应引起的，可以剔除掉。

在OpenCV中，可以通过`cv2.SIFT_create()`来创建一个SIFT对象，主要参数有：

- `nOctaveLayers`：每组金字塔中的层数，通常取值为3。
- `contrastThreshold`：对比度阈值，用于过滤低对比度的关键点。
- `edgeThreshold`：边缘阈值，用于剔除边缘的响应。
- `sigma`：高斯模糊的标准差，用于控制尺度空间的初始模糊度。

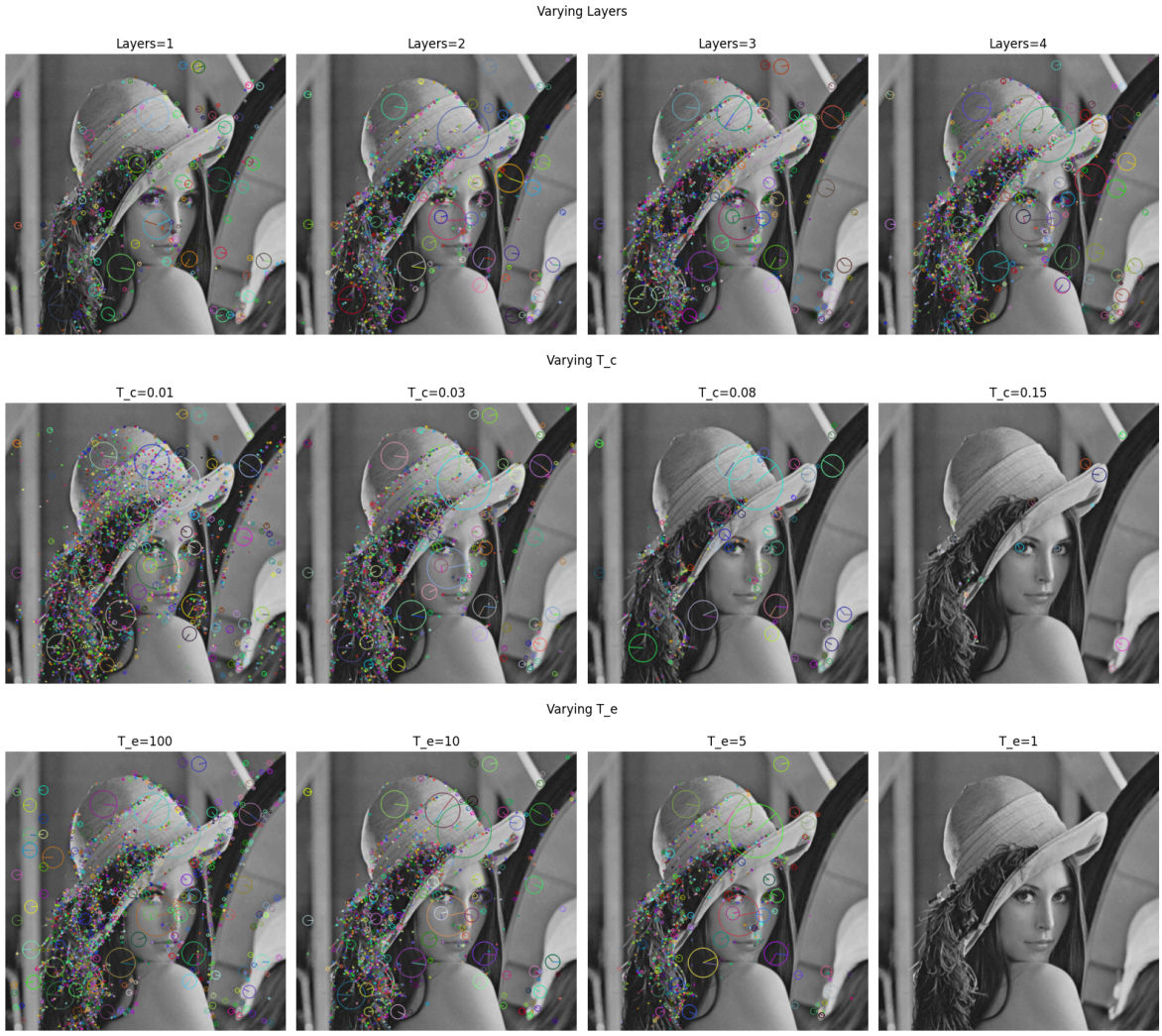
在SIFT对象创建后，可以通过调用`detectAndCompute`方法来对图像进行关键点检测和描述符计算。该方法接收一个输入图像，并返回检测到的关键点和对应的描述符。

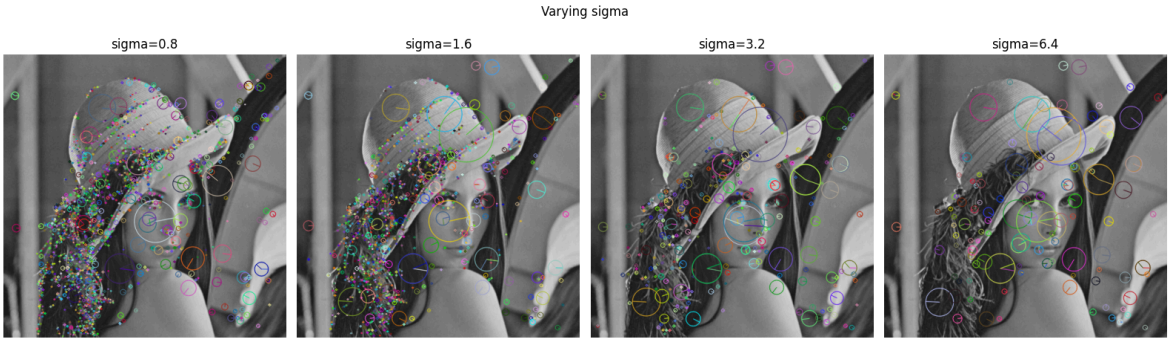
实验过程

分别改变四个参数——金字塔层数、对比度阈值、边缘阈值与高斯核方差，对比检测结果。

In [26]:
```python
import cv2
import matplotlib.pyplot as plt

# 读取图像
image = cv2.imread('lena.png')
image = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)

# 基础参数设置
base_params = {
    'Layers': 3,
    'T_c': 0.03,
    'T_e': 10,
    'sigma': 1.6
}

# 参数列表，每次只改变一个参数
experiments = {
    'Layers': [1, 2, 3, 4],
    'T_c': [0.01, 0.03, 0.08, 0.15],
    'T_e': [100, 10, 5, 1],
    'sigma': [0.8, 1.6, 3.2, 6.4]
}

for param_name, values in experiments.items():
    plt.figure(figsize=(16, 5))
    for i, value in enumerate(values, 1):
        # 当前实验参数
        test_params = base_params.copy()
```

```
        test_params[param_name] = value

        # 创建SIFT对象
        sift = cv2.SIFT_create(
            nOctaveLayers=test_params['Layers'],
            contrastThreshold=test_params['T_c'],
            edgeThreshold=test_params['T_e'],
            sigma=test_params['sigma']
        )

        # 检测关键点和计算描述符
        keypoints, descriptors = sift.detectAndCompute(image, None)

        # 绘制关键点图像
        img_keypoints_rich = cv2.drawKeypoints(image, keypoints, None, flags=cv2

        # 显示结果
        plt.subplot(1, 4, i)
        plt.imshow(img_keypoints_rich, cmap='gray')
        plt.title(f"{param_name}={value}")
        plt.axis('off')

    plt.suptitle(f"Varying {param_name}")
    plt.tight_layout()
    plt.show()
```





可以看到：

- 层数越高，检测到的关键点越多，因为更多的层可以得到更大的尺度空间，有更多可能的极值点；
- 较低的对比度阈值通常会增加关键点数量，引入噪声；
- 较高的边缘阈值可以过滤掉更多边缘特征，在图块中的特征点的比例明显提高了；
- 更高的方差会得到更少的特征点，因为卷积核尺度变大，图像更平滑，极值点的数目就减少了。