Contents SIFT

1 导入图像

In [2]:

```
from PIL import Image
import re
import numpy as np
from matplotlib import pyplot as plt

path = 'E:\chip design competition\IMAGE\\5.jpg'
img = Image.open(path, "r")

img_L = img.convert('L')
pixels = img_L.load()
height,width = img.height, img.width
all_pixels = []
for row in range(height):
  for col in range(width):
    cpixel = pixels[col, row]
    all_pixels.append(cpixel)

img = np.array(all_pixels).reshape((height,width))
plt.axis('off')
plt.imshow(img, cmap='gray')
print("图像尺寸：", img.shape)
```

图像尺寸： (512, 512)

2 高斯模糊

2.1 生成高斯算子

In [3]:

```
import math
import numpy as np

#生成高斯模板的函数
#输入：sigmma为参数，size为模板的边长
#输出：np.array类型矩阵
def create_gaussian_kernal(sigmma, size):
    Gaussian_kernal = [[float(0) for i in range(size)] for
    sum_my = float(0)
    for row in range(size):
        for col in range(size):
            Gaussian_kernal[row][col] = math.exp(-((row-(si
            sum_my = sum_my + Gaussian_kernal[row][col]
    for row in range(size):
        for col in range(size):
            Gaussian_kernal[row][col] = Gaussian_kernal[row]
    return np.array(Gaussian_kernal, dtype="float32")

#是否进行测试?
test_enable = 0
if test_enable:
    size, sigmma = 7, 0.84089642
    print("高斯模糊模板：", size, "*", size, ",sigmma: ", si
    print(create_gaussian_kernal(sigmma, size))
else:
    print("complete")
```

```
complete
```

3 生成高斯金字塔

In [4]:

```
from PIL import Image
import re
import numpy as np
from matplotlib import pyplot as plt
import cv2

# 是否需要显示效果
show_enable = 1

src = cv2.imread(path,cv2.IMREAD_GRAYSCALE)
sigmma0 = 1.6
S = 4
size = 15
img_gaussian_list = [img.copy() for i in range(S)]
if show_enable:
    plt.figure(dpi=150)
    plt.axis('off')
for i in range(4):
    sigmma = sigmma0 * 2**(i/(S-1))
    gaussian_kernal = create_gaussian_kernal(sigmma, size)
    dst = cv2.filter2D(src, ddepth = -1, kernel = gaussian_
    img_gaussian_list[i] = np.array(dst)
    if show_enable:
        plt.subplot(1,4,i+1)
        plt.axis('off')
        plt.imshow(img_gaussian_list[i], cmap = 'gray')
    else:
        print("complete")
```

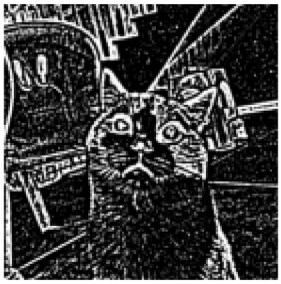
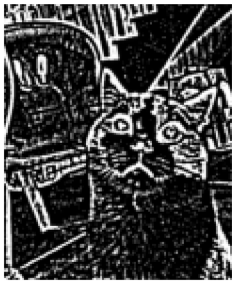


4 生成差分金字塔

Contents SIFT

In [5]:

```
#是否显示结果?
show_enable = 1
img_gaussian_deta_list = [img.copy() for i in range(S)]
if show_enable:
    plt.figure(dpi = 100)
    plt.axis('off')
for i in range(3):
    img_gaussian_deta_list[i] = img_gaussian_list[i] - img_g
    if show_enable:
        plt.subplot(1,3,i+1)
        plt.axis('off')
        plt.imshow(img_gaussian_deta_list[i], cmap='gray')
    else:
        print(i, ": complete, size:", img_gaussian_deta_list

```

5 寻找关键点

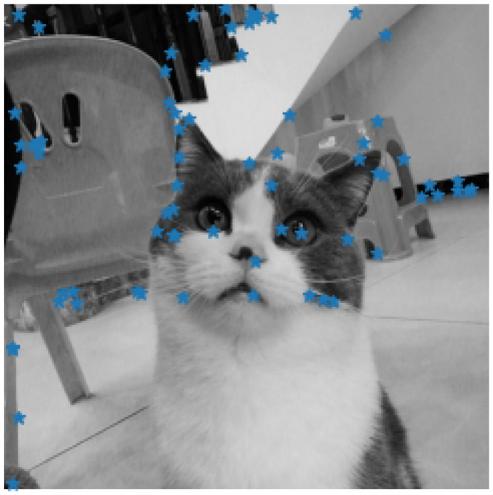
In [6]:

```
from matplotlib import pyplot as plt

feature_row = []
feature_col = []

for row in np.arange(1,510):
    for col in np.arange(1,510):
        center = img_gaussian_deta_list[1][row][col]
        flag = 1
        for z in [0,1,2]:
            for x in [-1,0,1]:
                for y in [-1,0,1]:
                    if z==1 and x==0 and y==0:
                        continue
                    elif center >= img_gaussian_deta_list[z
                        flag = 0
        if flag==1:
            feature_row.append(row)
            feature_col.append(col)
```

In [7]:

```
plt.figure(dpi = 80)
plt.imshow(img, cmap='gray')
plt.plot(np.array(feature_col), np.array(feature_row), '*')
plt.axis('off')
print("关键点数量：", len(feature_col))
```

关键点数量： 72



6 生成描述向量

6.1 计算一阶差分矩阵Dx，Dy

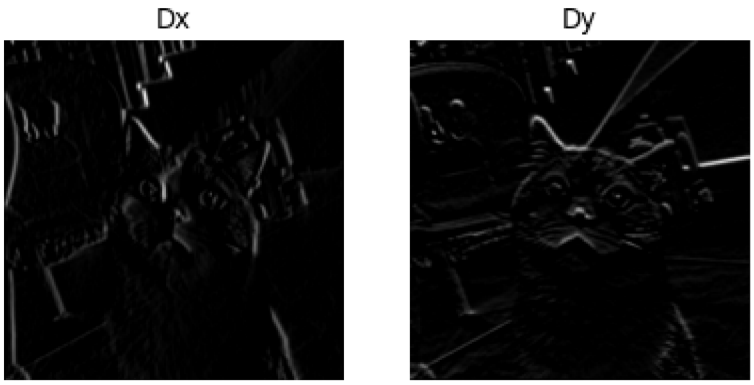
In [8]:

```
import numpy as np
from matplotlib import pyplot as plt
import math
import warnings

warnings.filterwarnings("ignore")

Dx, Dy = np.zeros((height, width)), np.zeros((height, width)
img_temp = img_gaussian_list[1]
Dx_kernel, Dy_kernel = np.array([-1,0,1]).reshape((1,3)), np
Dx = (cv2.filter2D(img_temp, ddepth = -1, kernel = Dx_kerne
Dy = (cv2.filter2D(img_temp, ddepth = -1, kernel = Dy_kerne

print("complete")
```

complete

In [9]:

```
#是否显示Dx,Dy矩阵的效果
show_enable = 1
if show_enable == 1:
    plt.figure(dpi = 80)
    plt.subplot(1,2,1)
    plt.title("Dx")
    plt.imshow(Dx, cmap='gray')
    plt.axis('off')

    plt.subplot(1,2,2)
    plt.title("Dy")
    plt.imshow(Dy, cmap='gray')
    plt.axis('off')
```



6.2 计算幅值矩阵和角度矩阵

Contents SIFT

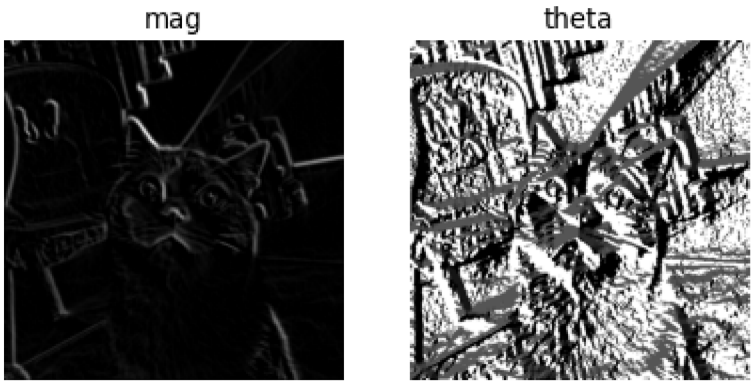
In [16]:

```
import numpy as np
from matplotlib import pyplot as plt
import math
import warnings

warnings.filterwarnings("ignore")

mag, theta = np.zeros((height, width)), np.zeros((height, wi
for row in range(height):
    for col in range(width):
        dx = Dx[row][col]
        dy = Dy[row][col]
        #计算梯度
        mag[row][col] = abs(dx) + abs(dy)
        #计算象限
        if dx>0 and dy>=0:
            XiangXian = 1
        elif dx<=0 and dy>0:
            XiangXian = 2
        elif dx<0 and dy<=0:
            XiangXian = 3
        else:
            XiangXian = 4
        #计算绝对值的角度
        dx, dy = abs(dx), abs(dy)
        if dx>dy*5:
            temp=0
        elif dx*2>dy*3:
            temp=1
        elif dx*3>dy*2:
            temp=2
        elif dx*5>dy:
            temp=3
        else:
            temp=4
        #根据象限计算角度
        if XiangXian == 1:
            temp=temp
        elif XiangXian == 2:
            temp = 8-temp
        elif XiangXian == 3:
            temp = 8+temp
        elif XiangXian == 4:
            temp = 16-temp
        if temp==16:
            temp=0
        theta[row][col] = temp

print("complete")
```

```
complete
```

In [18]:

```
#是否显示mag,theta矩阵的效果
show_enable = 1
if show_enable == 1:
    plt.figure(dpi = 80)
    plt.subplot(1,2,1)
    plt.title("mag")
    plt.imshow(mag, cmap='gray')
    plt.axis('off')

    plt.subplot(1,2,2)
    plt.title("theta")
    plt.imshow(theta, cmap='gray')
    plt.axis('off')
```



6.3 子区域描述向量生成函数

In [13]:

```
def sub_region_descripitor(center_row, center_col):
    sigmma = sigmma0*1.5
    size = size
    kernel_temp = create_gaussian_kernal(sigmma, size)
    resu = [0 for i in range(8)]
    for row_deta in range(5):
        for col_deta in range(5):
            direction = theta

```

In [ ]:

```

```

In [ ]:

```

```

In [ ]:

```

```

In [ ]:

```
1
```

In [ ]:

```
1
```

In [ ]:

```
1
```

In [ ]:

```
1
```

7 基于cv2的sift关键点查找

In [ ]:

```
# 是否使能?
enable = 0

if enable == 1:
    image1 = cv2.imread("E:\chip design competition\IMAGE\gi

    sift = cv2.SIFT_create()
    kp1, des1 = sift.detectAndCompute(image1, None)
    kp_image1 = cv2.drawKeypoints(image1, kp1, None)

    plt.figure()
    plt.imshow(kp_image1)
    plt.savefig('kp_image1.png', dpi = 300)
```

In [ ]:

```
1
```