程设第五次作业

20377383 樊思涵

调用库

```python
import matplotlib.pyplot as plt
from PIL import Image
from PIL import ImageFilter
import os
import glob
```

实现基类 Filter

```python
class Filter:
    """
    基类Filter
    """
    def __init__(self,image,parameters):
        """
        image:待处理的图片实例
        parameters:滤波器参数
        """
        self.image = image
        self.parameters = parameters

    def filter(self):
        """
        在基类中不进行实现
        实现细节交给子类
        """
        pass
```

实现 Filter 的四个子类

```python
class Edge(Filter):
    """
    边缘提取子类Edge
    """
    def __init__(self,image,parameters):
        super(Edge,self).__init__(image,parameters)

    def filter(self,img):
        img = img.filter(ImageFilter.FIND_EDGES)
        return img

```

```python
class Blur(Filter):
    """
    模糊子类Blur
    """
    def __init__(self,image,parameters):
        super(Blur,self).__init__(image,parameters)

    def filter(self,img):
        img = img.filter(ImageFilter.BLUR)
        return img

```

```python
class Sharpen(Filter):
    """
    锐化子类Sharpen
    """
    def __init__(self,image,parameters):
        super(Sharpen,self).__init__(image,parameters)

    def filter(self,img):
        img = img.filter(ImageFilter.SHARPEN)
        return img

```

```python
class Resize(Filter):
    """
    大小调整子类Resize
    """
    def __init__(self,image,parameters):
        super(Resize,self).__init__(image,parameters)

    def filter(self,img):
        img = img.resize((self.parameters[0],self.parameters[1]))
        return img

```

实现图片处理类及其具体的几个方法

初始参数

```
    def __init__(self,formation,path,Image_list,Image_process):
        """
        formation:图片格式
        path:图片目录
        Image_list:储存图片实例（初始传参传入空列表）
        Image_process: 储存处理过的图片（初始传参传入空列表）
        """
        self.formation = formation
        self.path= path
        self.Image_list = Image_list
        self.Image_process = Image_process

```

加载图片（作为内部方法在图片处理过程中调用）

```
    def __load_images(self):
        """
        内部方法
        加载指定目录下所有格式为formation的图片
        """
        self.Image_list = glob.glob(os.path.join(self.path,'*'+self.formation))

```

批量处理图片内部方法

```
    def __batch_ps(self,Filter):
        """
        处理图片的内部方法
        """
        for i in range(len(self.Image_process)):
            img = Filter.filter(self.Image_process[i])  #调用对应Filter类的filter方法
            self.Image_process[i] = img

```

批量处理图片的对外公开方法

```
    def batch_ps(self,*args):
        """
        批量处理图片的对外公开方法
        args为不定长的tuple形如(operation,parameters)
        """
        ImageShop.__load_images(self)                   #加载图片路径
        for i in self.Image_list:
            self.Image_process.append(Image.open(i))
        for image in self.Image_process:
            for i in range(len(args[0])):
                if args[0][i][0] == 'Edge':
                    e = Edge(image,args[0][i][1])
                    ImageShop.__batch_ps(self,e)
                elif args[0][i][0] == 'Sharpen':
                    s = Sharpen(image,args[0][i][1])
                    ImageShop.__batch_ps(self,s)
                elif args[0][i][0] == 'Blur':
                    b = Blur(image,args[0][i][1])
                    ImageShop.__batch_ps(self,b)
                elif args[0][i][0] == 'Resize':
                    r = Resize(image,args[0][i][1])
                    ImageShop.__batch_ps(self,r)

```

图片的展示

```
124     def display(self,row = 3,column= 3,maximum = 27):
125         """
126         利用subplot函数批量显示处理后图片
127         row:每行图片数
128         column:每列图片数
129         maximum:处理图片最大数量
130         在默认情况最多输出3页
131         """
132         if len(self.Image_process) > maximum:
133             self.Image_process = self.Image_process[:maximum]
134         plt.ion()       #为了使循环能够正常进行
135         for page in range(0,len(self.Image_process),row * column):      #控制每一页生产的图片数量
136             for i in range(row * column):                       #控制每张子图展示图片数量
137                 if page + i <len(self.Image_process):
138                     img = self.Image_process[page + i]
139                     plt.subplot(row,column,i + 1)
140                     plt.imshow(img)
141                 else:
142                     continue
143             plt.show()
144             plt.pause(10)    #等待10秒后关闭当前页
145             plt.close('all')
146
```

保存图片

```
147     def save(self,filepath):
148         """
149         保存图片到指定路径
150         """
151         for num in range(len(self.Image_process)):
152             img = self.Image_process[num]
153             img.save(filepath+'\{}'.format(num) + self.formation)
154
```

实现测试类

```
155 class TestImageShop:
156     """
157     测试类
158     """
159     def __init__(self,formation,path,Image_list,Image_process):
160         self.Test = ImageShop(formation,path,Image_list,Image_process)
161
162     def batch(self,*args):
163         self.Test.batch_ps(args)
164
165     def save(self,filepath):
166         self.Test.save(filepath)
167
168     def display(self):
169         self.Test.display()
170
```

main 函数

```python
def main():
    """
    main函数
    """
    #给定的参数
    parameters = [640,480]
    path = r'C:\Users\LF\Desktop\animals'   #图片集路径
    formation = '.png'
    Image_list,Image_process = [],[]
    operation = ['Edge','Sharpen','Blur','Resize']
    filepath = r'C:\Users\LF\Desktop\week6'
    #使用测试类测试
    test = TestImageShop(formation,path,Image_list,Image_process)
    #test.batch((operation[0],0),(operation[1],0),(operation[2],0),(operation[3],parameters))
    test.batch((operation[0],0))
    test.save(filepath)
    test.display()

if __name__ == '__main__':
    main()
```

测试结果展示
在网络上下载格式为 png 的图片若干
理由：png 格式图片轮廓清晰，便于展示


eb3cb00e21f4013ed1584d05fb1d4e9fe777ead218ac104497f5c97faeebb5bb_640


eb3db20b28f3093ed1584d05fb1d4e9fe777ead218ac104497f5c97faee8b1b8_640


eb3db20b28fc043ed1584d05fb1d4e9fe777ead218ac104497f5c97faee8b1b8_640


eb3db20b28fc053ed1584d05fb1d4e9fe777ead218ac104497f5c97faee8b1b8_640


eb31b2092bfd043ed1584d05fb1d4e9fe777ead218ac104497f5c97faeebb5bb_640


eb32b70f21f2013ed1584d05fb1d4e9fe777ead218ac104497f5c97faeebb5bb_640


eb32b70f21f2033ed1584d05fb1d4e9fe777ead218ac104497f5c97faee8b1b8_640


eb32b6072cfc073ed1584d05fb1d4e9fe777ead218ac104497f5c97faee8b1b8_640

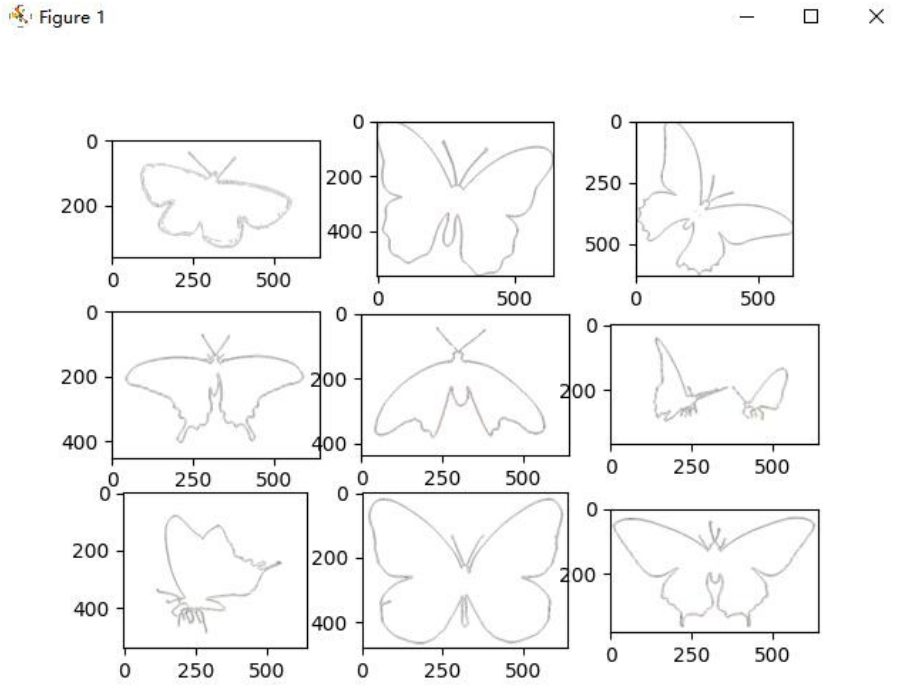

eb32b6072cfd003ed1584d05fb1d4e9fe777ead218ac104497f5c97faeebb5bb_640

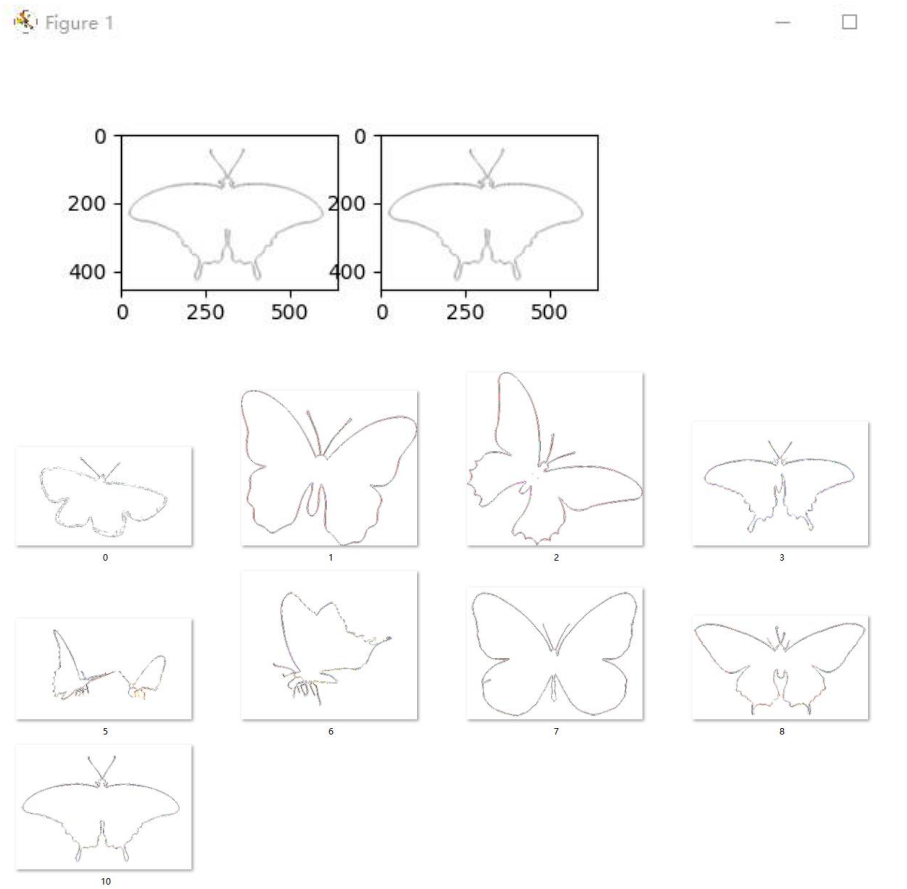

eb32b6092afd033ed1584d05fb1d4e9fe777ead218ac104497f5c97faeebb5bb_640


eb32b6092afd043ed1584d05fb1d4e9fe777ead218ac104497f5c97faeebb5bb_640

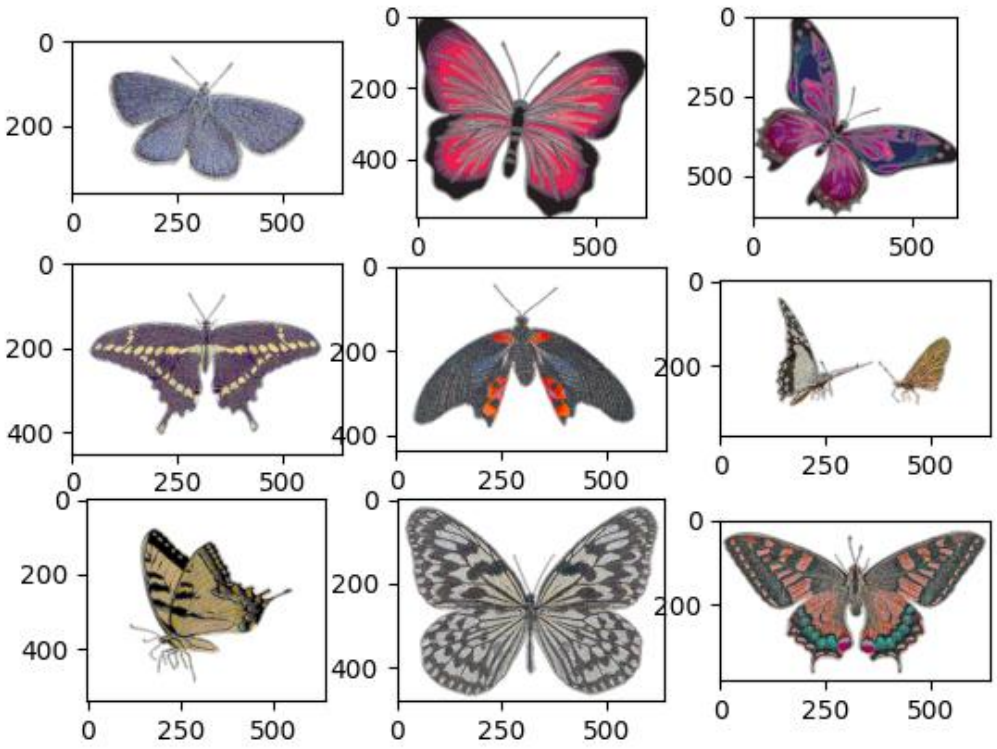
先做单一操作，由于共 11 张图故分两页展示，再在资源管理器中展示

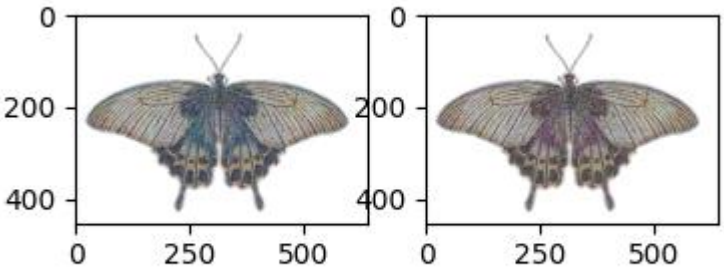
1.边缘提取

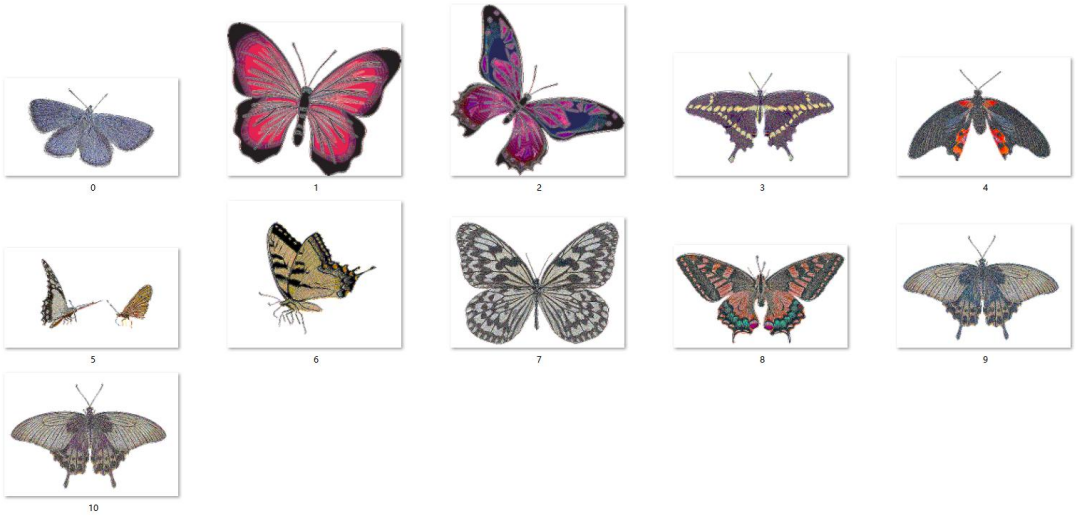




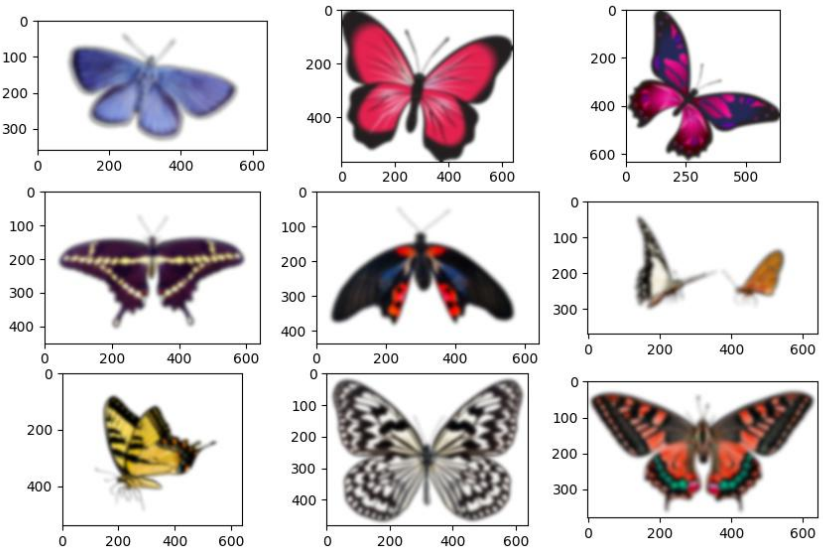
2.模糊

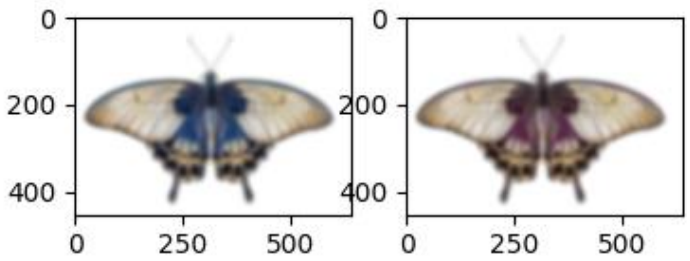

0
200
0
250
500
0
200
400
0
500
0
250
500
0
200
0
250
500
0
200
400
0
250
500
0
200
0
250
500
0
200
400
0
250
500
0
200
400
0
250
500
0
200
0
250
500


0
200
400
0
250
500
0
200
0
250
500

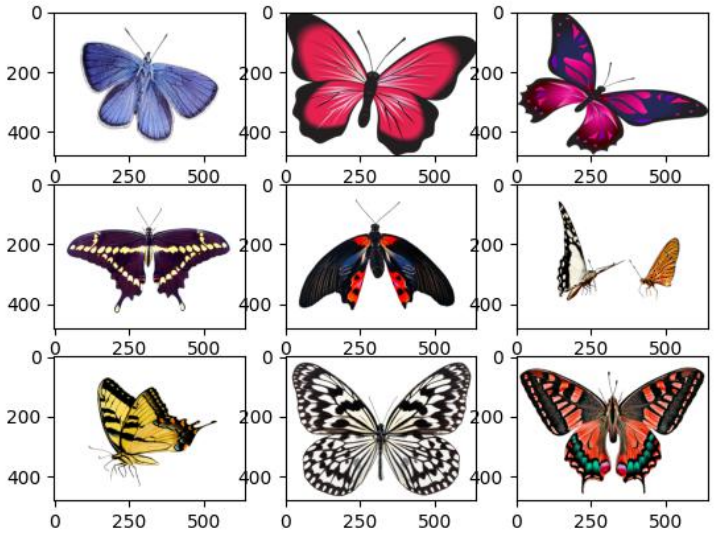

0
1
2
3
4
5
6
7
8
9
10

3.锐化





4.大小调整



5.多种操作（锐化+模糊）