中国矿业大学

China University of Mining and Technology

《OpenSource GIS 开发》

课程作业

学　　院：环境与测绘学院

班　　级：地理信息科学 17-1 班

姓　　名：张清昱

学　　号：07172336

2020 年 12 月

OpenSource 综合设计

一、系统介绍

很多时候我们可能只需要一些简单的功能来处理遥感影像，却需要打开类似 ENVI 等这样体量庞大的软件，即浪费了时间，又浪费了电脑内存，通过 Python 实现的简单几个常用的小软件实现对遥感影像的打开、关闭、清空、结果保存、展示信息、查看直方图、波段变换，实现轻量、简易的目的来处理简单的遥感影像。

二、总体设计

本系统开发环境为 MacOS Big Sur 操作系统，在 PyCharm CE 中使用编程语言 Python 开发，使用了 Pillow 包、GDAL 包、tkinter 包、turtle 包、time 包进行辅助。使用全局变量 FILEWAY 存储当前窗口的文件路径信息，initWin 存储默认设置的窗口大小，以及 w_box, h_box 设置打开遥感影像后的窗口大小，default_dir 存储默认存储路径，代码前半段为函数，在后面皆数被引用，数据全部存储在运行路径本地。

三、关键代码与步骤

本次选择 Python 作为编程语言，理由之一是其在图形 GUI 开发上有轻量、简易的优势，其二是由于前期实验过程中就已经使用且配置好了 Python 环境，这对于本次开发将节省大量的时间，最后因为则是 Python 作为解释性编程语言，只需新建一个后缀名为.py 的文件就可编写程序，为交作业与写作业都提供了便利。

选好了编程语言，接下来考虑使用哪个框架进行编写，Python 一般常用的有 Qt、Tkinter、wxPython 等，其中以 Qt 为代表的一类功能十分强大，几十种组件可以选择，可自定义程度也十分之高，但对于本次作业其实是杀鸡用牛刀了，把眼光投向 Python 自带的 Tkinter 库，其实对于本次这些简易的功能，我们使用 Tkinter 进行简单的定制即可。

主力影像处理工具当然选择 GDAL，而考虑到还有画图功能，本次也引入了 Python 自带的图形库 turtle 进行直方图的绘制，Python 的方便之处就在于安装成功库的情况下，在文件头部直接进行语句引入即可。

创建主窗口 root，使用 tkinter 的 tk()方法就可以实现，然后创建菜单栏并编辑菜单栏的下拉框，添加功能按钮，一定要在主窗口的基础上进行编辑控件，不然就无法提供接下来将要写的功能接口：

```
# 创建菜单栏
menubar = tk.Menu(root)
filemenu = tk.Menu(menubar, tearoff=0)
menubar.add_cascade(label='文件', menu=filemenu)

filemenu.add_command(label='清空', command=_clear)
filemenu.add_command(label='打开', command=_open)
filemenu.add_command(label='保存', command=_save)
filemenu.add_separator()  # 分隔线
filemenu.add_command(label='退出', command=root.quit)

# 继续编辑菜单栏
editmenu = tk.Menu(menubar, tearoff=0)
menubar.add_cascade(label='操作', menu=editmenu)

editmenu.add_command(label='信息', command=_showInfo)
editmenu.add_command(label='变换波段', command=_swapBand)
editmenu.add_command(label='显示直方图', command=_gram)

root.config(menu=menubar)
```

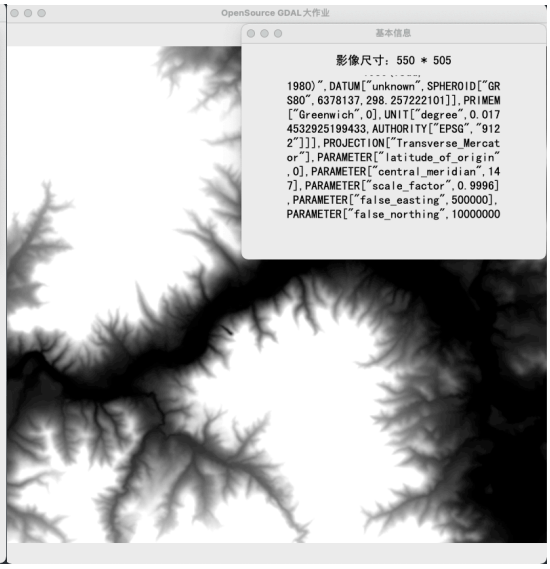
为了引导用户更好的使用该软件，可以在一打开的主页面添加文字引导“请选择文件->打开，打开遥感影像”，只需添加 label 控件，并且定义 label 内的文字填充，设定好初始窗口的大小，即可专注于逻辑层的编写上了。

由于遥感影像多种多样大小不一，所以需要设置一个固定的大小，将图像拉伸或缩放至该大小，以便程序的统一性，我们使用 Pillow 库进行图片的编辑，封装函数 resize，输入原大小与想要变成的大小（窗口大小），即可返回一个大小变换过后的 PIL 图像对象，后期只要转成 Tkimage 对象就可以添加到视图窗口里了：

```
def resize(w, h, wbox, hbox, pil_image):
    f1 = 1.0 * wbox / w
    f2 = 1.0 * hbox / h
    factor = min([f1, f2])
    width = int(w * factor)
    height = int(h * factor)
    return pil_image.resize((width, height), Image.ANTIALIAS)
```

效果如下：

2592 * 2693 大小的影像



550 * 505 大小的影像

考虑到在对图像进行处理后需要清空窗口内的原有图像，此块程序需要多次调用，所以对此块也进行函数封装，此时便需要调用上述的 resize 函数，先将原图像在窗口清空，再导入新的图像并放置，并且更新为了指明当前窗口上影像路径的全局变量 FILEWAY：

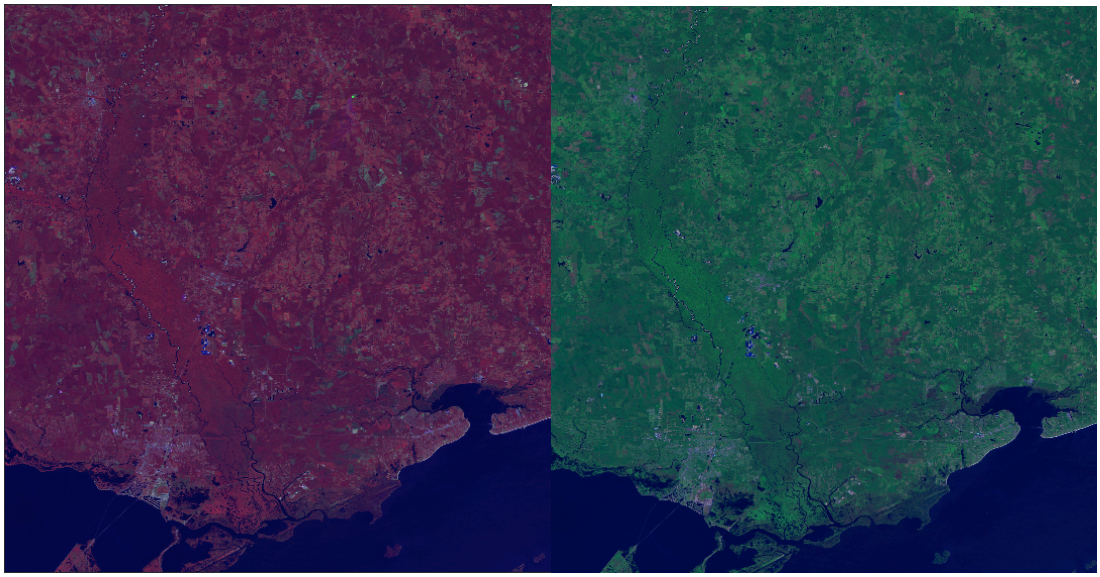
```
def refreshImage(fway):
    global labelImage
    labelImage.place_forget()
    pil_image = Image.open(fway)
    # 获取图像的原始大小
    w, h = pil_image.size
    # 缩放图像让它保持比例，同时限制在一个矩形框范围内
    pil_image_resized = resize(w, h, w_box, h_box, pil_image)
    # 把PIL图像对象转变为Tkinter的PhotoImage对象
    tk_image = ImageTk.PhotoImage(pil_image_resized)

    root.geometry('{}x{}+600+150'.format(w_box, h_box))
    labelImage = tk.Label(root, image=tk_image, width=w_box, height=h_box)
    labelImage.image = tk_image
    labelImage.place(relx=0, rely=0)
    global FILEWAY
    FILEWAY = fway
```

核心功能：变化波段，用户可以在弹出的选框中选择新的三个波段由哪几个波段组成，即若设置为 2、1、3 即意为交换原影像的第一和第二波段，使得影像变化为真彩色或假彩色影像，真彩色：R,G,B 三波段的合成显示图。假彩色：任意非 R,G,B 波段的合成图。若红波段（R）、绿波段（G）、蓝波段（B）三幅图像分别赋予 R、G、B 三色，所生成的是“真彩色”或“天然”彩色合成图像，如 TM3、2、1（RGB）。若三幅其他任何波段图像赋予 R、G、B 三色，则得假彩色合成图像，如 TM1、2、3（RGB），TM3、5、4（RGB）等。若近红外波段（NIR）、红波段（R）、绿波段（G）三幅图像分别赋予 R、G、B 三色，则得标准假彩色合成图像，如 TM4、3、2（RGB），SPOT3、2、1（RGB）等。

我们通过 gdal_array 的 SaveArray 来实现，在保存时重新设置第一个参数 src_array，改变原有的[0, 1, 2]波段顺序，设置为任意，即可实现波段变化，并指定保存路径名称与保存格式（format 属性），继承原影像：

```
def _swapBand():
    def change(band1, band2, band3):
        arr = gdal_array.LoadFile(FILEWAY)
        gdal_array.SaveArray(arr[[band1, band2, band3], :], "swap.tif",
                             format="GTiff", prototype=FILEWAY)
        resWay = FILEWAY[:FILEWAY.rfind('/') + 1] + "swap.tif"
        time.sleep(2)
        refreshImage(resWay)
        rootBand.destroy()
```

以下为交换第一第二波段后的变化的真、假彩色影像：

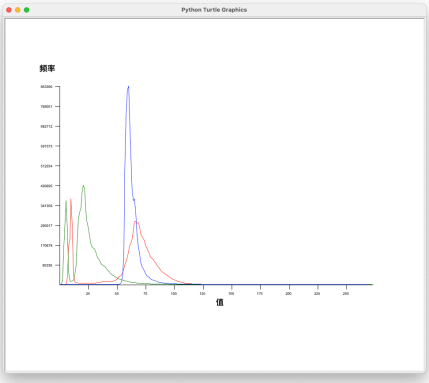


假彩色　　真彩色

图像直方图包含辐射信息，没有空间信息。直方图可以被看作离散概率分布，因为在图像子块中某一个亮度值柱条的相对高度（被像素总数归一化）表示在该图像中找到该图像中具有该特定亮度值像素的概率。直方图对比度的拉伸（一些被映射到同样新亮度值的柱条将被叠加，幅值大小不变），因为遥感图像亮度值低且对比度低，方法有线对比度增强，饱和线性对比度增强、自动对比度增强、对数和指数对比度增强、分段线性对比度修正。图像直方图由于其计算代价较小，且具有图像平移、旋转、缩放不变性等众多优点，广泛地应用于图像处理的各个领域，特别是灰度图像的阈值分割、基于颜色的图像检索以及图像分类。使用 turtle 进行直方图的绘制，实质是以列表数据形式读取影像数据并展示在二维坐标系中，此处仅展示参数计算部分，不展示绘画计算部分：

```python
def histogram(a):
    fa = a.flat
    n = gdal_array.numpy.searchsorted(gdal_array.numpy.sort(fa), list(range(0, 256)))
    n = gdal_array.numpy.concatenate([n, [len(fa)]])
    hist = n[1:] - n[:-1]
    return hist
```

最终效果图如下：



附录：

```python
# -*- coding:utf-8 -*-

import os
from PIL import Image, ImageTk
import tkinter as tk
from tkinter import filedialog
from tkinter import ttk
from tkinter import messagebox
from osgeo import gdal
from osgeo import gdal_array
import turtle
import time

root = tk.Tk()
FILEWAY = ""
labelImage = tk.Label()
initWin = '500x100+750+400'
w_box, h_box = 800, 800
default_dir = "/Users/apple/Documents/GDAL"


def resize(w, h, wbox, hbox, pil_image):
    f1 = 1.0 * wbox / w
    f2 = 1.0 * hbox / h
    factor = min([f1, f2])
    width = int(w * factor)
    height = int(h * factor)
    return pil_image.resize((width, height), Image.ANTIALIAS)


def refreshImage(fway):
    global labelImage
    labelImage.place_forget()
    pil_image = Image.open(fway)
    # 获取图像的原始大小
    w, h = pil_image.size
    # 缩放图像让它保持比例，同时限制在一个矩形框范围内
    pil_image_resized = resize(w, h, w_box, h_box, pil_image)
    # 把 PIL 图像对象转变为 Tkinter 的 PhotoImage 对象
    tk_image = ImageTk.PhotoImage(pil_image_resized)

    root.geometry('{}x{}+600+150'.format(w_box, h_box))
```

```
    labelImage = tk.Label(root, image=tk_image, width=w_box, height=h_box)
    labelImage.image = tk_image
    labelImage.place(relx=0, rely=0)
    global FILEWAY
    FILEWAY = fway


def _clear():
    global labelImage
    labelImage.place_forget()
    root.geometry(initWin)
    global FILEWAY
    FILEWAY = ""


def _open():
    global FILEWAY
    FILEWAY = filedialog.askopenfilename(title="选择遥感影像
", initialdir=(os.path.expanduser(default_dir)))
    if FILEWAY:
        refreshImage(FILEWAY)


def _showInfo():
    if FILEWAY:
        dataset = gdal.Open(FILEWAY)
        im_width = dataset.RasterXSize
        im_height = dataset.RasterYSize
        im_proj = dataset.GetProjection()
        rootInfo = tk.Toplevel()
        rootInfo.geometry('450x320+750+300')
        rootInfo.title('基本信息')
        labelInfo_size = tk.Label(rootInfo, text="影像尺寸:
{} * {}".format(im_width, im_height), font=("黑体", 18), height=2)
        labelInfo_size.pack()
        labelInfo_proj = tk.Label(rootInfo, text="投影信息:
" + im_proj, font=("黑体", 18), height=10, wraplength=320)
        labelInfo_proj.pack()
    else:
        messagebox.showerror("提示", "未打开任何图像！")


def _swapBand():
    def change(band1, band2, band3):
```

```
84.         arr = gdal_array.LoadFile(FILEWAY)
85.         gdal_array.SaveArray(arr[[band1, band2, band3], :], "swap.tif", form
    at="GTiff", prototype=FILEWAY)
86.         resWay = FILEWAY[:FILEWAY.rfind('/') + 1] + "swap.tif"
87.         time.sleep(2)
88.         refreshImage(resWay)
89.         rootBand.destroy()
90.
91.     if FILEWAY:
92.         rootBand = tk.Toplevel()
93.         rootBand.geometry('150x100+750+300')
94.         rootBand.title('波段选择')
95.         tk.Label(rootBand, text="波段 1:  ").grid(row=0)
96.         tk.Label(rootBand, text="波段 2:  ").grid(row=1)
97.         tk.Label(rootBand, text="波段 3:  ").grid(row=2)
98.
99.         number1 = tk.StringVar()
100.          band1Chosen = ttk.Combobox(rootBand, width=6, textvariable=number1,
    state='readonly')
101.          band1Chosen['values'] = (1, 2, 3)
102.          band1Chosen.grid(row=0, column=1)
103.          band1Chosen.current(0)
104.
105.          number2 = tk.StringVar()
106.          band2Chosen = ttk.Combobox(rootBand, width=6, textvariable=number2,
    state='readonly')
107.          band2Chosen['values'] = (1, 2, 3)
108.          band2Chosen.grid(row=1, column=1)
109.          band2Chosen.current(1)
110.
111.          number3 = tk.StringVar()
112.          band3Chosen = ttk.Combobox(rootBand, width=6, textvariable=number3,
    state='readonly')
113.          band3Chosen['values'] = (1, 2, 3)
114.          band3Chosen.grid(row=2, column=1)
115.          band3Chosen.current(2)
116.
117.          confirm = tk.Button(rootBand, text="确认
    ", width=8, command=lambda: change(int(band1Chosen.get()) - 1,
118.
         int(band2Chosen.get()) - 1,
119.
         int(band3Chosen.get()) - 1))
120.          confirm.grid(row=3, columnspan=2)
```

```
    else:
        messagebox.showerror("提示", "未打开任何图像！")


def _gram():
    def histogram(a):
        fa = a.flat
        n = gdal_array.numpy.searchsorted(gdal_array.numpy.sort(fa), list(range(0, 256)))
        n = gdal_array.numpy.concatenate([n, [len(fa)]])
        hist = n[1:] - n[:-1]
        return hist

    def draw_histogram(hist, scale=True):
        turtle.color("black")
        axes = ((-355, -200), (355, -200), (-355, -200), (-355, 250))
        turtle.up()
        for p in axes:
            turtle.goto(p)
            turtle.down()
        turtle.up()
        turtle.goto(0, -250)
        turtle.write("值", font=("SimHei, ", 18, "bold"))
        turtle.up()
        turtle.goto(-400, 280)
        turtle.write("频率", font=("SimHei, ", 18, "bold"))
        x = -355
        y = -200
        turtle.up()
        for i in range(1, 11):
            x = x + 65
            turtle.goto(x, y)
            turtle.down()
            turtle.goto(x, y - 10)
            turtle.up()
            turtle.goto(x, y - 25)
            turtle.write("{}".format((i * 25)), align="center")
        x = -355
        y = -200
        turtle.up()
        pixels = sum(hist[0])
        if scale:
            maxValue = 0
            for h in hist:
```

```
                hmax = h.max()
                if hmax > maxValue:
                    maxValue = hmax
            pixels = maxValue
        label = int(pixels / 10)
        for i in range(1, 11):
            y = y + 45
            turtle.goto(x, y)
            turtle.down()
            turtle.goto(x - 10, y)
            turtle.up()
            turtle.goto(x - 15, y - 6)
            turtle.write("{}".format((i * label)), align="right")
        x_ratio = 709.0 / 256
        y_ratio = 450.0 / pixels
        colors = ["red", "green", "blue"]
        for j in range(len(hist)):
            h = hist[j]
            x = -354
            y = -199
            turtle.up()
            turtle.goto(x, y)
            turtle.down()
            turtle.color(colors[j])
            for i in range(256):
                x = i * x_ratio
                y = h[i] * y_ratio
                x = x - (709 / 2)
                y = y + -199
                turtle.goto((x, y))

if FILEWAY:
    turtle.tracer(False)
    histograms = []
    arr = gdal_array.LoadFile(FILEWAY)
    for b in arr:
        histograms.append(histogram(b))
    draw_histogram(histograms)
    turtle.pen(shown=False)
    turtle.done()
else:
    messagebox.showerror("提示", "未打开任何图像！")


```

```
def _save():
    if FILEWAY:
        image = Image.open(FILEWAY)
        saveWay = filedialog.askdirectory(title="选择保存路径
", initialdir=(os.path.expanduser(default_dir)))
        image.save(saveWay + '/result.tif')
    else:
        messagebox.showerror("提示", "未打开任何图像！")


root.title('OpenSource GDAL大作业')
root.geometry(initWin)
labelNotice = tk.Label(root, text="请选择文件->打开，打开遥感影像", font=("黑
体", 30), height=3).pack()

# 创建菜单栏
menubar = tk.Menu(root)
filemenu = tk.Menu(menubar, tearoff=0)
menubar.add_cascade(label='文件', menu=filemenu)

filemenu.add_command(label='清空', command=_clear)
filemenu.add_command(label='打开', command=_open)
filemenu.add_command(label='保存', command=_save)
filemenu.add_separator()  # 分隔线
filemenu.add_command(label='退出', command=root.quit)

# 继续编辑菜单栏
editmenu = tk.Menu(menubar, tearoff=0)
menubar.add_cascade(label='操作', menu=editmenu)

editmenu.add_command(label='信息', command=_showInfo)
editmenu.add_command(label='变换波段', command=_swapBand)
editmenu.add_command(label='显示直方图', command=_gram)

root.config(menu=menubar)

root.mainloop()
```