

**“数字图像处理基础”实验报告**

|  |  |  |  |
| --- | --- | --- | --- |
| 题目 | 实验二：图像文件读写 | | |
| 姓名 | 钟军凯 | 学号 | 22211374 |
| 指导教师 | 彭亚辉 | | |
| 日期 | 2025年 3月5日 | | |

**电子信息工程学院**

目 录

[1. 实验目的分析 3](#_Toc191473896)

[2. 实验方案设计 3](#_Toc191473897)

[3. 实验结果 4](#_Toc191473898)

[4. 分析与讨论 1](#_Toc191473899)1

[5. 问题发现与探究 1](#_Toc191473900)2

[6. 实验总结 1](#_Toc191473901)5

[指导老师评价 2](#_Toc191473902)0

1. 实验目的分析

本次实验的主要目标是帮助学生掌握常见图像格式（如JPEG、PNG等）的读取和保存方法，同时学习如何设计和实现自定义图像格式的编码与解码。实验中，学生将通过文件对话框选择并打开指定的图像文件，直观地操作图像数据。此外，实验会让学生使用特定的模块，让学生熟悉相应模块的使用，同时提升学生的代码编写和调试能力。

1. 实验方案设计
   1. **实验内容1**

首先使用ttk.Button创建一个按键，这个按键的command属性定义到open\_file函数，也就是说，按下这个按键的时候，将会执行open\_file函数。

open\_file函数的大致内容为：使用tkinter的filedialog中的askopenfilename()函数来打开文件选择窗口，该函数只能选择一个文件，最终返回文件的路径。获取到图片文件的路径之后，使用PIL中的Image.open()函数打开图片，得到一个Image对象，再用PIL的ImageTk.PhotoImage()函数将Image对象转换为tkinter能够解析的PhotoImage对象，最后使用ttk.Label来显示图片即可。

* 1. **实验内容2**

实验内容1中得到的Image对象，使用Image.mode获取图片类型，从而知道图片有哪些颜色通道，然后用Image.size获取图片的宽和高。

* 1. **实验内容3**

对于read\_raw(file\_name)，使用open()函数，以“rb”的方式打开raw文件，接着使用struct.unpack()函数得到raw文件的宽和高，具体为：raw\_width = struct.unpack(“i”, raw\_file.read(4))[0]和raw\_height = struct.unpack(“i”, raw\_file.read(4))[0]。接着再次使用struct.unpack()获取图片的像素数据，用“B”的格式unpack。然后用close()函数关闭文件。将获取到的图片像素数据转换为二位数组，使用numpy.array(raw\_data).reshape(raw\_height, raw\_width)。最后返回该二位数组。

对于write\_raw(file\_name, array)，使用array.shape获取width和height，再使用array.reshape()重构成一维数组，使用struct.pack()将width、height和转化得到的一维数组以raw文件的格式打包，得到data。基于file\_name，得到一个新的文件名，使用open(),以“wb”的方式打开这个新的文件，使用write()写入data，最终close()，完成写入。

* 1. **实验内容4**

在实验内容1中的open\_file()中，将askopenfilename()改为askopenfilenames()，这时可以打开多个文件，文件路径以元组的形式返回。使用for循环对文件路径遍历，对于每一个raw文件使用read\_raw(file\_name)函数。

* 1. **实验内容5**

将实验内容4中，由read\_raw(file\_name)函数得到的数组放入write\_raw(file\_name, array)即可。

* 1. **拓展实验**

使用ttk.Button.bind()函数，将按键与鼠标行为绑定，同时不同的鼠标行为对应不同的回调函数，鼠标左键对应img\_backward()，鼠标右键对应img\_forward()。定义一个索引变量file\_index，用于文件路径元组中的定位，鼠标进行操作后，无论进入img\_backward()，还是进入img\_forward()，内容都是改变file\_index，使其定位不同的文件路径，从而显示不同的图片。

1. 实验结果

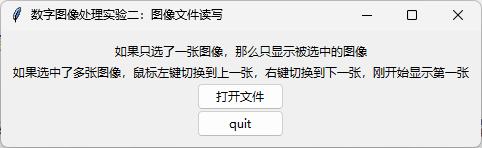


图 1：初始界面

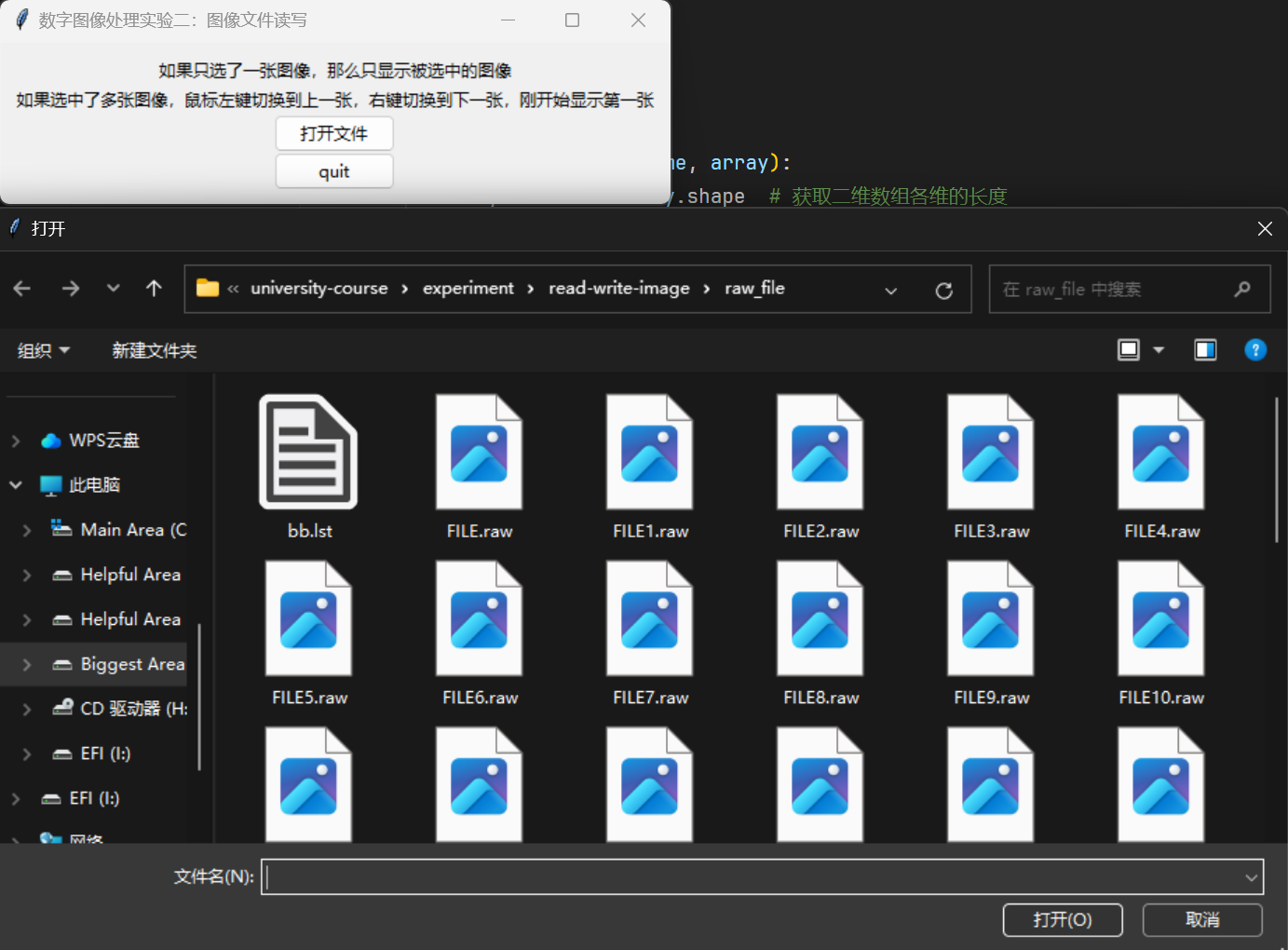


图 2：文件选择界面的弹出



图 3：jpg图片格式的显示



图 4：png图片格式的显示



图 5：bmp图片格式的显示



图 6：raw文件图像显示



图 7：写入新raw文件后新raw文件显示



图 8：raw文件数据写入重命名文件



图 9：鼠标切换图片初始界面



图 10：鼠标左键切换图片效果

1. 分析与讨论

本实验实现了如下的效果：

如果只选择了一张图片，则切换图片的按键会被隐藏；如果选择了多张图片，则切换图片的按键将会出现。随着对切换图片按键的点击，图片将会被循环显示，左键显示上一张，右键显示下一张，图片顺序由图片的文件名排序而得。

无论选择的是常见的图片文件，还是raw文件，都可以展现出上一段所描述的效果。如果选择了raw文件，还会有将raw文件的数据写入一个新的raw文件的操作，新的raw文件的文件名为原raw文件的文件名后加上“\_new”。

1. 问题发现与探究
   1. **问题一**

【**问题描述**】

ttk.Button中command属性的误解，之前创建quit按键的时候，command设置为了tkinter.Tk.destroy，这个时候，我认为这个destroy是一个变量，或者一个标志。

【**问题探究**】

通过对ttk模块相关文档的阅读，我发现command属性需要的是一个函数。也就是说，command是用来定位某一操作的，当Button被按下，使用command来定位的函数所定义的操作将会被执行。

* 1. **问题二**

【**问题描述**】

file\_data = struct.unpack("B", file.read())报错：struct.error: unpack requires a buffer of 1 bytes。

【**问题解决**】

使用struct.unpack的时候，需要指定长度，所以进行修改：file\_data = struct.unpack(f"{file\_width \* file\_height}B", file.read())。如果只是单纯一个“B”，那么只会解包一个数，但是实际上有很多数据，所以需要加上数据的个数。

* 1. **问题三**

【**问题描述**】

首先，先给出一份在本实验的开始阶段写的代码：

|  |
| --- |
| import tkinter as tk  from tkinter import ttk  from tkinter import filedialog as fd  from PIL import Image  from PIL import ImageTk  import numpy as np  import struct  ''' 创建图窗 '''  # 创建基本的图窗  root = tk.Tk()  root.title("数字图像处理实验二：图像文件读写") # 设置图窗标题  frame = ttk.Frame(root, padding = 10)  frame.grid()  ''' 显示图片 '''  def show\_image (image\_path):  image = Image.open(image\_path) # 通过图片路径打开图片  image\_tk = ImageTk.PhotoImage(image) # 将图像转化为tkinter可用的PhotoImage对象  image\_label = ttk.Label(frame, image = image\_tk)  image\_label.grid(row = 1, column = 0)  ''' 打开文件 '''  def open\_file ():  file = fd.askopenfilename()  if file:  print("读取文件成功，文件路径为：" + file)  else:  print("读取文件失败，请确保文件存在或者文件未损坏")  exit()  show\_image(file)  # 创建“打开文件”按键  open\_file\_button = ttk.Button(frame, text = "打开文件", command = open\_file) # 将按键的回调定位到open\_file函数  open\_file\_button.grid(row = 0, column = 0)  # 创建“quit”按键  quit\_button = ttk.Button(frame, text = "quit", command = root.destroy)  quit\_button.grid(row = 2, column = 0)  root.mainloop() |

代码一

代码一的作用是通过“打开文件”按键，选择图片，然后在由tkinter创建出来的图窗中显示。在运行代码一的时候，图片并没有显示。

在后续的实验过程中，也出现了类似的问题。

|  |
| --- |
| change\_image\_button = ttk.Button(frame, text = "根据顶部的提示，点击我切换图片")  change\_image\_button.bind("<Button-1>", img\_backward) # 左键显示上一张图片  change\_image\_button.bind("<Button-3>", img\_forward) # 右键显示下一张图片  if file\_num > 1:  # 显示切换图片的按键  change\_image\_button.grid(row = 6, column = 0)  else:  # 隐藏切换图片的按键  change\_image\_button.grid\_forget() |

代码二

代码二是一段位于一个子函数中的代码。代码二的作用是创建一个按键，该按键可以与鼠标的行为进行交互，鼠标左键对应显示上一张图片，鼠标右键对应显示下一张图片，然后根据打开图片的数量决定是否显示该按键。但是在运行代码二的时候，按键能显示，但是不能隐藏。

【**问题探究**】

在阅读了tkinter的官网文档和查阅资料之后，我了解到ttk.Label和ttk.Button在创建之后，如果使用了grid()，那么会直接与root绑定，由root全局管理。虽然代码二位于子函数中，change\_image\_button在子函数结束后会被释放，但是由于它已经与root绑定，虽然变量被释放了，但是这个部件还是与root绑定着，所以会显示在图窗中，只是由于变量被释放了，所以对按键施加的操作就无效了。

对于代码一中的ttk.Label，如果用ttk.Label来显示文本的话，文本会被存储在Label中，由于此时Label与root绑定，被root全局管理着，所以就算是写在子函数中，Label被释放了，文本也可以显示。但是图片不行，PhotoImage不会被ttk.Label存储，所以代码一中的PhotoImage类型的变量image\_tk在子函数结束，被释放之后，就真的不存在了，所以图片显示不了。

【**问题解决**】

将change\_image\_button和image\_tk设置为全局变量即可，即在子函数最上方使用global进行声明。

* 1. **问题四**

【**问题描述**】

data = struct.pack(f"ii{row \* column}B", column, row, array\_new)打包数据有问题，导致raw文件的图片有误。

【**问题解决**】

在array\_new前面加一个“\*”，即data = struct.pack(f"ii{row \* column}B", column, row, \*array\_new)。struct.pack需要的是数据，而不是数组，需要用“\*”来解包数组，将其变成一个个元素，不然数组将会被当作是元素。

6. 实验总结

在本次实验中，我了解到了python的struct库的用法，python文件操作。与此同时，对于已经接触过的tkinter，在本次实验中我又学到了新的用法，比如将鼠标的行为与按键绑定，在tkinter界面中进行文件交互。

在编程过程中，虽然遇到了不少问题，但通过查阅相关资料和阅读文档，最终都可以解决这些问题。这个过程让我对代码调试的能力有了很大的帮助。

附录：

【仿真程序】

|  |
| --- |
| import tkinter as tk  from tkinter import ttk  from tkinter import filedialog as fd  from PIL import Image  from PIL import ImageTk  import numpy as np  import struct  # 创建基本的图窗  root = tk.Tk()  root.title("数字图像处理实验二：图像文件读写") # 设置图窗标题  frame = ttk.Frame(root, padding = 10)  frame.grid()  file\_num = 0 # 选取的文件的数量  file\_index = 0 # 文件路径元组的索引  image\_label = ttk.Label(frame) # 用于显示图片的Label  '''  读取raw文件  '''  def read\_raw (file\_name):  # 获取raw文件表示的图片的数据  raw\_file = open(file\_name, "rb") # 打开文件，以只读、二进制的方式打开  raw\_width = struct.unpack("i", raw\_file.read(4))[0] # 获取raw文件表示的图片的宽度  raw\_height = struct.unpack("i", raw\_file.read(4))[0] # 获取raw文件表示的图片的高度  raw\_data = struct.unpack(f"{raw\_width \* raw\_height}B", raw\_file.read()) # 获取raw文件表示的图片的数组  raw\_file.close() # 关闭文件  # 将获取到的数组转换为二维数组  raw\_array = np.array(raw\_data).reshape(raw\_height, raw\_width)  return raw\_array  '''  写入raw文件  '''  def write\_raw (file\_name, array):  row, column = array.shape # 获取二维数组各维的长度  array\_new = array.reshape(row \* column)  file\_name\_new = file\_name[:len(file\_name) - 4] + "\_new" + file\_name[len(file\_name) - 4:] # 创建一个新的文件名  raw\_file\_new = open(file\_name\_new, "wb") # 写入新的文件，每次都先清空再写入  data = struct.pack(f"ii{row \* column}B", column, row, \*array\_new) # 数据转换  raw\_file\_new.write(data) # 写入数据  raw\_file\_new.close() # 关闭文件  '''  显示图片及其相关信息  '''  def show\_image (image\_path):  global image\_tk, image\_label # 将与图窗相关的变量设置为全局变量，使得它们在图窗的作用域内  image = Image.open(image\_path) # 通过图片路径打开图片  image\_tk = ImageTk.PhotoImage(image) # 将图像转化为tkinter可用的PhotoImage对象  # 显示图像  image\_label.config(image = image\_tk)  image\_label.grid(row = 3, column = 0)  # 显示图像大小  image\_size\_label = ttk.Label(frame, text = f"图像大小：{image.size}")  image\_size\_label.grid(row = 4, column = 0)  # 显示图像类型  image\_mode\_label = ttk.Label(frame)  if image.mode == "RGB":  image\_mode\_label.config(text = "图像类型：彩色图像")  elif image.mode == "L":  image\_mode\_label.config(text = "图像类型：灰度图像")  image\_mode\_label.grid(row = 5, column = 0)  '''  显示文件  '''  def show\_file ():  global file\_path, file, file\_num, file\_index, raw\_image\_tk, image\_label  # 获取要显示的文件  file = file\_path[file\_index]  # 获取文件格式  format = []  for i in reversed(range(len(file))): # 从后往前遍历  if file[i] == '.': # 由于只需要文件格式，所以遇到'.'就退出循环  break  format.append(file[i])  format.reverse() # 将列表反装  format\_str = "".join(format) # 将列表转化为字符串  # 判断是否为raw文件，并进行相应的操作  if format\_str == "raw":  raw\_array = read\_raw(file) # 获取raw文件的图片的二维数组  raw\_image = Image.fromarray(raw\_array) # 二维数组转换为图片  raw\_image\_tk = ImageTk.PhotoImage(raw\_image) # 转换为tkinter能够解析的PhotoImage类型  # 显示raw文件的图片  image\_label.config(image = raw\_image\_tk)  image\_label.grid(row = 3, column = 0)  # 显示图像大小  raw\_image\_size\_label = ttk.Label(frame, text = f"图像大小：{raw\_image.size}")  raw\_image\_size\_label.grid(row = 4, column = 0)  # 显示图像类型  raw\_image\_mode\_label = ttk.Label(frame, text = "图像类型：raw图像")  raw\_image\_mode\_label.grid(row = 5, column = 0)  write\_raw(file, raw\_array)  else:  show\_image(file)  '''  显示下一张图片  '''  def img\_forward (event):  global file\_path, file, file\_num, file\_index  file\_index = (file\_index + 1) % file\_num # 更新索引  file = file\_path[file\_index] # 获取要显示的文件  show\_file() # 显示获取到的文件  '''  显示上一张图片  '''  def img\_backward (event):  global file\_path, file, file\_num, file\_index  # 更新索引  if file\_index == 0:  file\_index = file\_num - 1  else:  file\_index = (file\_index - 1) % file\_num  # 显示获取到的文件  show\_file()  '''  打开文件  '''  def open\_file ():  global file\_path, file, file\_num, file\_index, change\_image\_button  file\_path = fd.askopenfilenames() # 获取文件路径  file\_num = len(file\_path) # 获取文件数量  file\_index = 0  if file\_num > 1:  # 显示切换图片的按键  change\_image\_button.grid(row = 6, column = 0)  else:  # 隐藏切换图片的按键  change\_image\_button.grid\_forget()  # 显示获取到的文件  show\_file()  # 图窗中的提示  tip\_1 = ttk.Label(frame, text = "如果只选了一张图像，那么只显示被选中的图像")  tip\_1.grid(row = 0, column = 0)  tip\_2 = ttk.Label(frame, text = "如果选中了多张图像，鼠标左键切换到上一张，右键切换到下一张，刚开始显示第一张")  tip\_2.grid(row = 1, column = 0)  # 创建“打开文件”按键  open\_file\_button = ttk.Button(frame, text = "打开文件", command = open\_file) # 将按键的回调定位到open\_file函数  open\_file\_button.grid(row = 2, column = 0)  # 创建用于鼠标交互的按键  change\_image\_button = ttk.Button(frame, text = "根据顶部的提示，点击我切换图片")  change\_image\_button.bind("<Button-1>", img\_backward) # 左键显示上一张图片  change\_image\_button.bind("<Button-3>", img\_forward) # 右键显示下一张图片  # 创建“quit”按键  quit\_button = ttk.Button(frame, text = "quit", command = root.destroy)  quit\_button.grid(row = 7, column = 0)  root.mainloop() |

指导老师评价

|  |  |  |
| --- | --- | --- |
|  | | |
| 仿真实验报告成绩 | 交流展示成绩 | 总评成绩 |
|  |  |  |

指导教师：

年 月 日

**实验报告提交说明:**

文件名 **姓名\_学号\_实验x.docx**

文件格式 Microsoft Word