數值方法 Numerical Method
期末專題報告

探討以線性代數為基礎之修圖軟體

指導教授：游濟華

學生：李云騫

學號：E94084016

一、 摘要

此期末專題運用從數值方法課程中獲取的知識，使用 Python 程式語言把理論轉換成實際應用，讓所學不僅僅是抽象的理論，更是能夠真實地運用在現實生活中，透過 Tkinter 模組構建 GUI 介面，實作一個簡易版軟圖軟體。

二、 研究動機與研究問題

充斥於我們生活中的各種圖片及影片，皆與數值方法中的線性代數息息相關。透過實作此修圖軟體，可以從不同的角度來探索線性代數的基本原理，以獲得更加深入的理解。

此期末專題實作簡易版修圖軟體，主要包含七大功能，即旋轉圖片，以及調整圖片之銳利度、陰影度、對比度、亮度、飽和度與模糊度，此外，在選擇好圖片並完成修圖後，如有需要，可將處理過後的圖片儲存下來，以供其他用途。而主要研究的問題在於運用 Python 中已經存在的模組和函式庫，進行組合與定制，以滿足我們特定的需求，如此我們便可建立功能強大的應用程式，且能夠站在前人的基礎上進一步發展。

三、 研究方法及步驟

步驟一：匯入所需模組

```
1  from tkinter import *
2  from tkinter import ttk
3  from tkinter import filedialog
4  from PIL import Image, ImageTk, ImageEnhance, ImageFilter
5  import os
```

首先，需要匯入所需的模組，包括 tkinter 用於構建 GUI 介面，PIL 用於編輯圖片，os 用於處理檔案和目錄。

步驟二：定義一個名為 PhotoEditor 的 class

```
# 定義一個叫做PhotoEditor的class
class PhotoEditor:
    def __init__(self, window):
        # 初始化PhotoEditor class，將傳入的window物件賦值給實例變數window
        self.window = window
        # 設定應用程式標題為"修圖神器"
        self.window.title("修圖神器")
        # 建立應用程式的所有widget
        self.setup_widgets()
```

在此處定義一個名為 PhotoEditor 的 class，包含用於圖片處理應用程式的所有功能和組件。

步驟三：構建 GUI 介面

```
def setup_widgets(self):
```

1. 建立頂部框架

```
# 在頂部建立一個Frame widget
top_frame = Frame(self.window)
top_frame.pack(side=TOP, fill=X, padx=10, pady=10)
```

在應用程式的頂部，建立一個框架來放置其他組件。

2. 添加標題標籤

```
# 在頂部frame中建立一個Label widget，用於顯示"修圖神器"
retouch_label = Label(top_frame, text="修圖神器", font=("Microsoft JhengHei UI", 30, "bold"))
retouch_label.pack(side=LEFT, anchor=N, padx=(30, 0), pady=(33, 0))
```

在頂部框架中，添加一個標籤來顯示應用程式的標題。

3. 添加「選擇圖片」按鈕

```
# 在頂部frame中建立一個Button widget，用於選擇圖片
self.load_button = Button(top_frame, text="選擇圖片", command=self.load_image, width=13, height=2, font=("Microsoft JhengHei UI", 18))
self.load_button.pack(side=RIGHT, anchor=N, padx=10, pady=10)
```

在頂部框架中，添加一個按鈕，可以讓使用者選擇要編輯的圖片。

4. 建立主框架

```
# 在主視窗中建立一個Frame widget
self.main_frame = Frame(self.window)
self.main_frame.pack(side=TOP, padx=10, pady=10)
```

建立一個主框架來放置圖片畫布和控制元件。

5. 建立畫布

```
# 在主視窗中建立一個Canvas widget，用於顯示圖片
self.canvas = Canvas(self.main_frame, width=500, height=500)
self.canvas.pack(side=LEFT)
```

在主框架中，建立一個畫布來顯示使用者選擇的圖片。

6. 建立控制框架

```
# 建立一個控制frame，用於放置調整圖片效果的滑動條
self.controls_frame = Frame(self.main_frame)
self.controls_frame.pack(side=LEFT, padx=20)
```

在主框架中建立另一個框架來放置圖片效果的控制元件，如滑動條。

7. 建立底部框架

```
# 在底部建立一個Frame widget
bottom_frame = Frame(self.window)
bottom_frame.pack(side=BOTTOM, fill=X, padx=10, pady=10)
```

在應用程式的底部，建立一個框架來放置其他組件。

8. 添加底部標籤

```
# 在底部frame中建立一個Label widget，用於顯示"人工智慧與多尺度模擬實驗室"
LAiMM_label = Label(bottom_frame, text="人工智慧與多尺度模擬實驗室", font=("Microsoft JhengHei UI", 24, "bold"))
LAiMM_label.pack(side=LEFT, anchor=N, padx=10, pady=21)
```

在底部框架中，添加一個標籤來顯示底部的文字。

9. 添加「儲存圖片」按鈕

```
# 在底部frame中建立一個Button widget，用於儲存圖片
self.save_button = Button(bottom_frame, text="儲存圖片", command=self.save_image, width=13, height=2, font=("Microsoft JhengHei UI", 18))
self.save_button.pack(side=RIGHT, anchor=N, padx=10, pady=10)
```

在底部框架中，添加一個按鈕，可以讓使用者保存編輯過的圖片。

10. 建立滑動條來調整圖片效果

```
# 建立多個滑動條，用於調整圖片的效果
self.rotate_slider = self.create_slider("旋轉", from_=0, to=360, command=self.rotate_image)
self.sharpness_slider = self.create_slider("銳利度", from_=0, to=200, command=self.adjust_sharpness)
self.shadow_slider = self.create_slider("陰影度", from_=0, to=10, command=self.adjust_shadow)
self.contrast_slider = self.create_slider("對比度", from_=0, to=10, command=self.adjust_contrast)
self.brightness_slider = self.create_slider("亮度", from_=0, to=10, command=self.adjust_brightness)
self.saturation_slider = self.create_slider("飽和度", from_=0, to=10, command=self.adjust_saturation)
self.blur_slider = self.create_slider("模糊度", from_=0, to=100, command=self.adjust_blur)
# 保存原始圖片的副本
self.original_image = None
```

在控制框架中，建立多個滑動條，用於調整圖片的各種效果，包括旋轉、銳利度、陰影度、對比度、亮度、飽和度和模糊度，而每個滑動條都使用了一個通用的 create_slider 方法來建立。

此外，我保存了原始圖片的副本，以便在使用不同的效果時，可以讓原始圖片始終保持不變。

步驟四：建立滑動條

```
def create_slider(self, name, from_, to, command):
    # 建立一個滑動條widget，用於調整圖片效果
    frame = Frame(self.controls_frame)
    frame.pack(pady=10, padx=1)

    label = Label(frame, text=name, width=11, font=("Microsoft JhengHei UI", 16))
    label.pack(side=LEFT)

    slider = Scale(frame, from_=from_, to=to, orient=HORIZONTAL, command=command, showvalue=0)
    slider.set(0)
    slider.pack(side=LEFT)

    return slider
```

在 setup_widgets 方法中，使用 create_slider 方法來建立滑動條，這些滑動條可以讓使用者調整圖片的各種效果。

步驟五：載入圖片

```
def load_image(self):
    # 打開一個檔案選擇對話框，選擇要編輯的圖片
    self.image_path = filedialog.askopenfilename(filetypes=[("Image Files", "*.png *.jpg *.jpeg *.bmp")])
    if self.image_path:
        # 如果成功選擇了圖片，則讀取該圖片並顯示在canvas中
        self.image = Image.open(self.image_path)
        self.original_image = self.image.copy()
        self.image.thumbnail((500, 500))
        self.photo = ImageTk.PhotoImage(self.image)
        self.canvas.create_image(250, 250, image=self.photo)
        self.canvas.image = self.photo
```

定義一個 load_image 方法，用於打開檔案選擇對話框，讓使用者選擇要編輯的圖片。

步驟六：圖片處理

1. 旋轉圖片

```
def rotate_image(self, value):
    # 旋轉圖片
    if self.image:
        self.image = self.original_image.copy()
        self.image.thumbnail((500, 500))
        self.image = self.image.rotate(int(value), resample=Image.BICUBIC, expand=True)
        self.update_canvas()
```

此函式可以讓使用者旋轉圖片，使用 PIL 的 rotate 方法來實現，而 value 參數指定了圖片要旋轉的角度，函式會更新畫布以顯示旋轉後的圖片。

2. 調整圖片銳利度

```
def adjust_sharpness(self, value):
    # 調整圖片的銳利度
    self.image = self.original_image.copy()
    enhancer = ImageEnhance.Sharpness(self.image)
    self.image = enhancer.enhance(float(value) / 10)
    self.image.thumbnail((500, 500))
    self.update_canvas()
```

此函式可以讓使用者調整圖片的銳利度，使用 PIL 的 ImageEnhance 模組中的 Sharpness class 來增加或減少圖片的銳利效果。

3. 調整圖片陰影度

```
def adjust_shadow(self, value):
    # 調整圖片的陰影效果
    self.image = self.original_image.copy()
    enhancer = ImageEnhance.Brightness(self.image)
    self.image = enhancer.enhance(1 - float(value) / 10)
    self.image.thumbnail((500, 500))
    self.update_canvas()
```

此函式可以讓使用者調整圖片的陰影度，使用 PIL 的 ImageEnhance 模組中的 Brightness class 來增加或減少圖片的陰影效果。

4. 調整圖片對比度

```
def adjust_contrast(self, value):
    # 調整圖片的對比度
    self.image = self.original_image.copy()
    enhancer = ImageEnhance.Contrast(self.image)
    self.image = enhancer.enhance(1 + float(value) / 10)
    self.image.thumbnail((500, 500))
    self.update_canvas()
```

此函式可以讓使用者調整圖片的對比度，使用 PIL 的 ImageEnhance 模組中的 Contrast class 來增加或減少圖片的對比效果。

5. 調整圖片亮度

```
def adjust_brightness(self, value):
    # 調整圖片的亮度
    self.image = self.original_image.copy()
    enhancer = ImageEnhance.Brightness(self.image)
    self.image = enhancer.enhance(1 + float(value) / 10)
    self.image.thumbnail((500, 500))
    self.update_canvas()
```

此函式可以讓使用者調整圖片的亮度，使用 PIL 的 ImageEnhance 模組中的 Brightness class 來增加或減少圖片的明亮效果。

6. 調整圖片飽和度

```
def adjust_saturation(self, value):
    # 調整圖片的飽和度
    self.image = self.original_image.copy()
    enhancer = ImageEnhance.Color(self.image)
    self.image = enhancer.enhance(1 + float(value) / 10)
    self.image.thumbnail((500, 500))
    self.update_canvas()
```

此函式可以讓使用者調整圖片的飽和度，使用 PIL 的 ImageEnhance 模組中的 Color class 來增加或減少圖片的飽和效果。

7. 調整圖片模糊度

```
def adjust_blur(self, value):
    # 調整圖片的模糊度
    self.image = self.original_image.copy()
    self.image = self.image.filter(ImageFilter.GaussianBlur(radius=float(value) / 10))
    self.image.thumbnail((500, 500))
    self.update_canvas()
```

此函式可以讓使用者調整圖片的模糊度，使用 PIL 的 ImageFilter 模組中的 GaussianBlur class 來增加或減少圖片的模糊效果。

步驟七：更新畫布

```
def update_canvas(self):
    # 更新canvas中的圖片
    self.photo = ImageTk.PhotoImage(self.image)
    self.canvas.create_image(250, 250, image=self.photo)
    self.canvas.image = self.photo
```

當使用者調整圖片效果時，定義一個 update_canvas 方法來更新畫布上的圖片。

步驟八：儲存圖片

```
def save_image(self):
    # 儲存圖片到檔案
    if self.image:
        file_path = filedialog.asksaveasfilename(defaultextension=".png")
        if file_path:
            self.image.save(file_path)
```

定義一個 save_image 方法，可以讓使用者將編輯後的圖片儲存到檔案。

步驟九：主函數

```
if __name__ == "__main__":
    # 創建一個Tkinter的root widget
    root = Tk()
    # 創建一個PhotoEditor的實例
    app = PhotoEditor(root)
    # 開始Tkinter的主循環
    root.mainloop()
```

在主函數中，創建 Tkinter 的 root widget，實例化 PhotoEditor class，並開始 Tkinter 的主循環。

四、 成果與討論

最後的使用者介面，左上角和左下角分別顯示著應用程式的標題「修圖神器」和本學期課程的支柱「人工智慧與多尺度模擬實驗室」，右上角的「選擇圖片」按鈕可選擇要編輯的圖片，左中間的矩形框就是用來顯示圖片的 canvas，右邊的七個滑動條可以對圖片進行各種效果的調整，分別是旋轉、銳利度、陰影度、對比度、亮度、飽和度與模糊度，最後是右下角的「儲存圖片」按鈕，可以將編輯過後的圖片儲存下來。

期末專題之 GitHub 連結：https://github.com/YunChianLee2001/Final_Project
Demo 影片連結：https://youtu.be/vADyDHqdxHk

圖一、使用者介面

圖二、經過銳利度調整後的圖片

圖三、經過對比度調整後的圖片

圖四、經過模糊度調整後的圖片

圖五、經過亮度調整後的圖片

隨著科技的日新月異，修圖軟體和數值方法的結合無疑開闢了無限的可能。修圖軟體不僅具有調整色彩、濾鏡等基本功能，隨著技術的進步，其潛力和應用領域將持續擴大。本次的期末專題就是一個典型的例子，透過從數值方法課程中獲取的知識，我們使用 Python 程式語言把抽象的理論轉換成實際的應用。不僅如此，透過 Tkinter 模組，我們還能夠構建 GUI，讓使用者更加直觀地操作此修圖軟體。

進一步來說，修圖軟體背後所運用的各種技術，與數值方法的線性代數息息相關。圖片和影片作為我們日常生活中不可或缺的元素，其實是由大量的像素和數值數據組成，而線性代數則在處理這些數據時起著至關重要的作用。實作修圖軟體，我們得以從多個角度來探索和理解線性代數的基本原理，也使抽象的概念變得更加具體和實用。

在完成期末專題後，我深刻體會到數值方法並非遙不可及的抽象學問，而是緊密地與我們的生活相連。從圖片處理到各種數據分析，數值方法的蹤跡無處不在，且提供了豐富的應用場景。

當前，修圖軟體已經在許多領域中發揮作用，且有著相當高的普及程度。而我於本次期末專題中所開發出的簡易版修圖軟體，雖然功能不盡完善，但這也代表其有著巨大的進步空間，希望之後可以進一步開發出能夠在圖片上添加文字的功能，這不僅增添了創意元素，也可以讓圖片更加生動。

最後，我要特別感謝游濟華老師，以及本學期課程的所有助教們。這門課程不僅教會我如何運用數學知識來解決實際問題，更讓我學會從不同的角度看待事物，開闊視野。我相信，隨著時間的推移，數值方法將會發揮出更大的潛力，並在眾多領域中繼續促進創新和進步。