Project2 Write-up

Yue Pan

1063965

04/06/2020

Python Imaging library Write-up

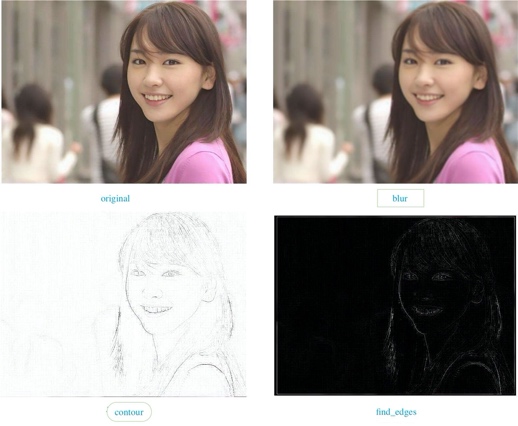
***1.Purpose***

The Python Imaging Library adds image processing capabilities to people Python interpreter. This library provides extensive file format support, an efficient internal representation, and fairly powerful image processing capabilities. The core image library is designed for fast access to data stored in a few basic pixel formats. It should provide a solid foundation for a general image processing tool.

***2.Examples***

1)The first example is a useful one, which shows some basic functions in the Python Imaging Library. First, I use function Image.open(‘path’) to open an image file. Then I read the image size through the size attribute. This is a 2-tuple, containing the horizontal and vertical size in pixels. I used the img.size method to see the size of the image, and used img.mode to see the mode of the image (the P, L, RGB are common one), and img.format to see the format of the image(png and jpg are common). img.thumbnail((w//2, h//2)) is a useful method which can help me to reduce the picture by 50%. Then there’s an interesting function to help me change the mode, which is img.convert('mode'). At the beginning, I read and open AragakiYui’s photo, because she’s very beautiful who is an actress and singer from Japan. First I changed it to L mode, which is 8-bit pixels, and the color is black and white. Then I changed it to P mod, which is also 8-bit pixels, but now the new image is mapped to any other mode using a colour palette.

2) The second example I wrote is more complicated. I imported ImageFilter from the PIL. I used 3 methods to change the image which are imgF.filter(ImageFilter.BLUR), imgF.filter(ImageFilter.CONTOUR) and imgF.filter(ImageFilter. FIND\_EDGES). The first one is to blur it, to make the picture hazy and make people feel like it's raining. The second one the to find the contour of the image and the third one is to find the edge of the picture and almost is black. I put the original image and the 3 new one together below here. Then I use the blend function(Image.blend(image1, image2, alpha)) to blend 2 images together.



***3. high-level overview***

1)Open function : Image.open(infile) or Image.open(infile,mode)

This is a function for opening and identifiing the given image file. This is a lazy operation; the actual image data is not read from the file until you try to process the data (or call the load method).

2)Blend function: Image.blend(image1, image2, alpha)

This function is used to blend 2 images. It can create a new image by interpolating between the given images, usinga constant alpha. Both images must have the same size and mode.out = image1 \* (1.0 - alpha) + image2 \* alpha

Link: http://www.pythonware.com/media/data/pil-handbook.pdf

Turtle Library

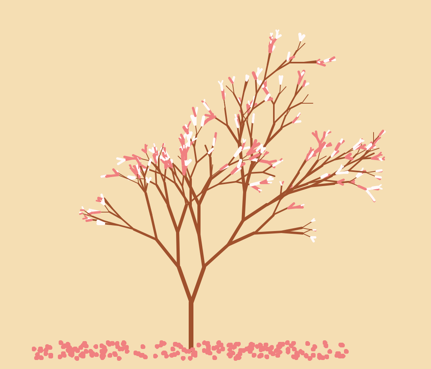
***1.Purpose***

“Turtle” is a python feature like a drawing board, which lets you command a turtle to draw all over it. Internally, the library stores raster pixel data in a simple array, tracking turtle location and drawing parameters. The library can export results to .bmp (bitmap) files, and can optionally export frames suitable for compilation into an animation.

***2.Eamples***

1)In the first example, I use the turtle library to draw a sakura tree. I knew about turtle library in previous class, and I can draw some simple pictures, like a filled-color triangle. But some time ago, I found that people can also use the turtle library to draw some beautiful and complex pictures and even generate dynamic pictures. I think it’s really interesting, so I explored the turtle library deeper. First, I create the method called Tree(branch, t), and import turtle, random and time. Then I use different time in the if loop to control which color to use and which direction to go. I also use random.random() method which make the results are from the “continuous uniform” distribution over the stated interval. So a beautiful sakura tree is drawn just like this.

2)In the second example, I developed the example one, I tried to make the petals falling down. I tried a lot of times. First, I create a method called Petal(m, t), I use the for loop to control the petal in which region will fall and when will the petal fall and how long will the petals falling. In both examples, I use turtle(), forward(), backward(), right(), left(), up(), down(), color(),circle()function to draw the tree. These 2 pictures are below there.



***3. Functions***

Turtle(): Creates and returns a new tutrle object

forward(): Moves the turtle forward by the specified amount

backward(): Moves the turtle backward by the specified amount

right(): Turns the turtle clockwise

left(): Turns the turtle counter clockwise

penup(): Picks up the turtle’s Pen

pendown() : Puts down the turtle’s Pen

up(): Picks up the turtle’s Pen

down() : Puts down the turtle’s Pen

color() : Changes the color of the turtle’s pen

link: https://docs.python.org/3.3/library/turtle.html?highlight=turtle

Difflib library Write-up

***1.Purpose***

The difflib library provides classes and functions for comparing sequences. It can be used for example, for comparing files, and can produce difference information in various formats, including HTML and context and unified diffs.

2.Examples

1)The first example is used to compare the differences between 2 texts and print out the result. It’s very useful and easy to command. Besides, it doesn’t need to download In my program, first, I entered 4 sentences in each text, and import the difflib library, then I use the difflib.Differ() function to compare the differences between the 2 texts, and I print out the result. The result is easy to read. The code ‘+’ means that the line is unique to sequence 1. The code ‘-’ means that the line is unique to sequence 2. The code ‘ ’ means that line common to both sequences. The code ‘?’ means that line not present in either input sequence. This is a basic example for exploring the difflib library, so I wrote the second example which is more complex.

2)The second example is used to compare the same items in the 2 shopping lists. First, it read and opened 2 txt files which are the shopping lists, then I use the difflib.HtmlDiff() and make\_file(content1, content2) functions to compare the differences between 2 txt files and show the results in the html document. The html document is nice looking and easy to read. in the html file the red one means the the distinct one in the first file, the green one means the the distinct one in the second file, the data with no color means the same one in both 2 files. I think this is useful, for example, it can be used in the grocery store. Before customers go, they can make a shopping list and the store can list what are in storage, then customers check the stores list with theirs, then they can decide which store to go which can help them save time and ensure the high effiency.

3. high-level overview

Function 1: difflib.**Differ**

This is a function for comparing sequences of lines of text, and producing human-readable differences or deltas. Differ uses SequenceMatcher both to compare sequences of lines, and to compare sequences of characters within similar (near-matching) lines.

Function 2: difflib.HtmlDiff

This class can be used to create an HTML table (or a complete HTML file containing the table) showing a side by side, line by line comparison of text with inter-line and intra-line change highlights. The table can be generated in either full or contextual difference mode.

Link: https://docs.python.org/3/library/difflib.html#difflib.SequenceMatcher