jpegtran-cffi Documentation

Release 0.3.1

Johannes Baiter

May 24, 2015

Contents

jpegtran-cffi is a Python package for fast JPEG transformations. Compared to other, more general purpose image processing libraries like wand-py or PIL/Pillow, transformations are generally more than twice as fast (see *Benchmarks*). In addition, all operations except for scaling are lossless, since the image is not being re-compressed in the process. This is due to the fact that all transformation operations work directly with the JPEG data.

This is achieved by using multiple C routines from the Enlightenment project's epeg library (for scaling) and *jpegtran* from the Independent JPEG Group's libjpeg library (for all other operations). These routines are called from Python through the CFFI module, i.e. no external processes are launched.

The package also includes rudimentary support for getting and setting the EXIF orientation tag, automatically transforming the image according to it and obtaining the JFIF thumbnail image.

jpegtran-cffi was developed as part of a web interface for the spreads project, where a large number of images from digital cameras had to be prepared for display by a Raspberry Pi. With the Pi's rather slow ARMv6 processor, both Wand and PIL were too slow to be usable.

Supported Python versions are CPython 2.6, 2.7 and 3.3, as well as PyPy.

The source code is under the MIT license and can be found on GitHub.

CHAPTER 1

Requirements

- CPython 2.6, 2.7, 3.3 or PyPy
- cffi
- **libjpeg8** with headers (v6 will not work) *or* **libjpeg-turbo** with *turbojpeg* headers

CHAPTER 2

Installation

```
$ pip install jpegtran-cffi
```

CHAPTER 3

Usage

```
from jpegtran import JPEGImage

img = JPEGImage('image.jpg')

# JPEGImage can also be initialized from a bytestring
blob = requests.get("http://example.com/image.jpg").content
from_blob = JPEGImage(blob=blob)

# Reading various image parameters
print img.width, img.height  # "640 480"
print img.exif_orientation  # "1" (= "normal")

# If present, the JFIF thumbnail can be obtained as a bytestring
thumb = img.exif_thumbnail

# Transforming the image
img.scale(320, 240).save('scaled.jpg')
img.rotate(90).save('rotated.jpg')
img.crop(0, 0, 100, 100).save('cropped.jpg')

# Transformations can be chained
data = (img.scale(320, 240)
           .rotate(90)
           .flip('horizontal')
           .as_blob())

# jpegtran can transform the image automatically according to the EXIF
# orientation tag
photo = JPEGImage(blob=requests.get("http://example.com/photo.jpg").content)
print photo.orientation  # "6" (= 270°)
print photo.width, photo.height # "4320 3240"
corrected = photo.exif_autotransform()
print corrected.orientation  # "1" (= "normal")
print corrected.width, corrected.height  # "3240 4320"
```

For more details, refer to the *API Reference*.

CHAPTER 4

Benchmarks

All operations were done on a 3.4GHz i7-3770 with 16GiB of RAM and a 7200rpm HDD with the following 2560x1920 8bit RGB JPEG:

http://upload.wikimedia.org/wikipedia/commons/8/82/Mandel_zoom_05_tail_part.jpg

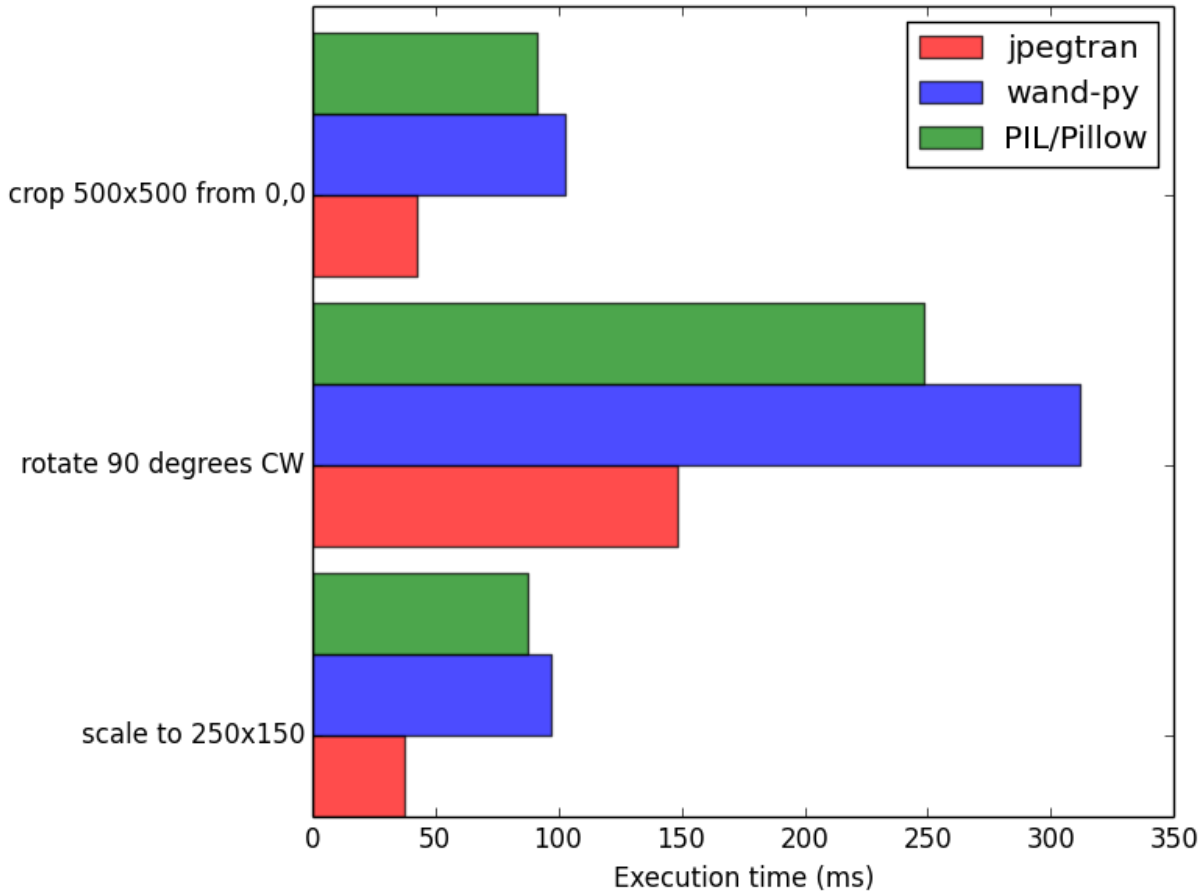

Fig. 4.1: Both wand-py and PIL were run with the fastest scaling algorithm available, for wand-py this meant using `Image.sample` instead of `Image.resize` and for PIL the nearest-neighbour filter was used for the `Image.resize` call.

Benchmark source: https://gist.github.com/jbaiter/8596064

CHAPTER 5

Example Output

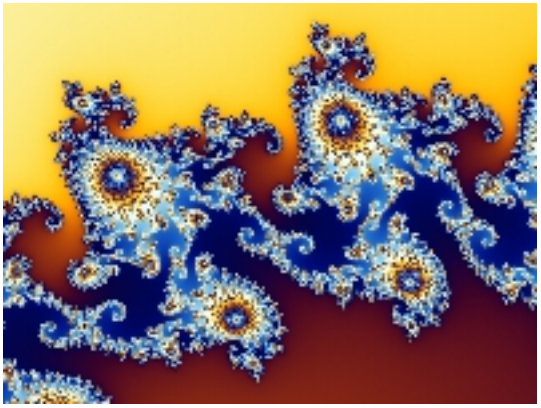

Fig. 5.1: Wand-Py `Image.sample(200, 150)`, filtering was nearest neighbour

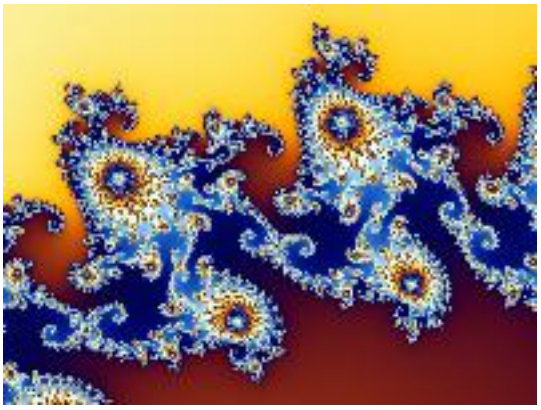

Fig. 5.2: PIL `Image.resize((200, 150))`

On imgur: http://imgur.com/a/JvAtM

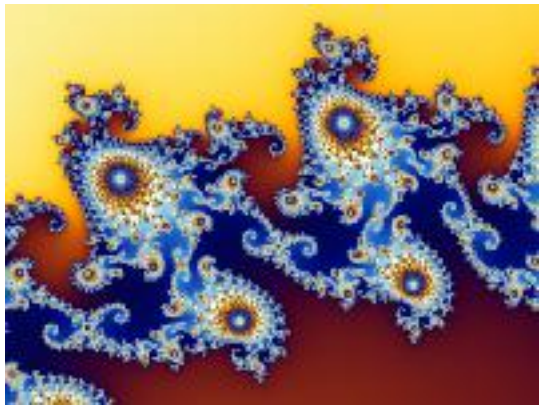

Fig. 5.3: jpegtran-cffi `JPEGImage.scale(200, 150, quality=75)`

CHAPTER 6

Change Log

6.1 0.5.1

- Fix for a memory leak (Thanks to Stephane Boisson)

6.2 0.5

- Support for libjpeg-turbo
- EXIF thumbnails are automatically updated on transformations
- Don't raise an error when doing no-op transformations

6.3 0.4

- EXIF thumbnail parsing is now much more stable
- *get_exif_thumbnail* returns a *JPEGImage* object instead of a *str*

CHAPTER 7

API Reference

class jpegtran.**JPEGImage**(*fname=None*, *blob=None*)

__init__(*fname=None*, *blob=None*)
Initialize the image with either a filename or a string or bytearray containing the JPEG image data.

Parameters

- **fname** (*str*) – Filename of JPEG file
- **blob** (*str/bytearray*) – JPEG image data

as_blob()
Get the image data as a string

Returns Image data

Return type bytes

crop(*x*, *y*, *width*, *height*)
Crop a rectangular area from the image.

Parameters

- **x** (*int*) – horizontal coordinate of upper-left corner
- **y** (*int*) – vertical coordinate of upper-left corner
- **width** (*int*) – width of area
- **height** (*int*) – height of area

Returns cropped image

Return type *jpegtran.JPEGImage*

downscale(*width*, *height*, *quality=75*)
Downscale the image.

Parameters

- **width** (*int*) – Scaled image width
- **height** (*int*) – Scaled image height
- **quality** (*int*) – JPEG quality of scaled image (default: 75)

Returns downscaled image

Return type *jpegtran.JPEGImage*

exif_autotransform()
Automatically transform the image according to its EXIF orientation tag.

Returns transformed image

Return type *jpegtran.JPEGImage*

flip(*direction*)
Flip the image in horizontal or vertical direction.

Parameters **direction** (*'vertical' or 'horizontal'*) – Flipping direction

Returns flipped image

Return type *jpegtran.JPEGImage*

rotate(*angle*)
Rotate the image.

Parameters **angle** (*-90, 90, 180 or 270*) – rotation angle

Returns rotated image

Return type *jpegtran.JPEGImage*

save(*fname*)
Save the image to a file

Parameters **fname** (*unicode*) – Path to file

transpose()
Transpose the image (across upper-right -> lower-left axis)

Returns transposed image

Return type *jpegtran.JPEGImage*

transverse()
Transverse transpose the image (across upper-left -> lower-right axis)

Returns transverse transposed image

Return type *jpegtran.JPEGImage*

exif_orientation
Exif orientation value as a number between 1 and 8.

Property is read/write

exif_thumbnail
EXIF thumbnail.

Returns EXIF thumbnail in JPEG format

Return type str

height
Height of the image in pixels.

width
Width of the image in pixels.

Symbols

A

C

D

E

F

H

J

R

S

T

W