ENGG1003 - Thursday Week 10

Assignment 2: Image processing

Steve Weller

University of Newcastle

13 May 2021

Last compiled: May 11, 2021 6:31pm +10:00

Lecture overview

1. key dates, worth 15%, lab question sheet in BB > Assessment
2. images as 3D arrays
3. digital image formats
4. data types
5. structure of assignment
6. strategies for the assignment

1) images as 3D arrays

- raster images
  - gif, jpeg, png
  - contrast with vector images: svg

- review of Sarah's material from Thursday week 7

2) digital image formats

- colourspaces
  - RGB
  - HSL

- RGB and HSL are two different ways of representing the *same* colour
  - key theme of assignment: RGB <−−> HSL

- $[0, 1]$ and $[0, 255]$

- use colour images and links to colour picker

3) data types

- uint8
- uint16
- float32
- float64

- type conversions

4) structure of assignment

first 5 functions

- `loadImage`
  - read image file into 3D numpy array
- `saveImage`
  - save 3D numpy array as image file
- `rgb2hsl`
  - convert image in RGB format to HSL format
- `rgb2hsl`
  - convert image in HSL format to RGB format
- `showImage`
  - display image in window

5) how to go about the assignment

eight (8) functions to be graded in assignment

- `brightness`
  - adjust image brightness
- `contrast`
  - adjust image contrast
- `saturation`
  - adjust image saturation
- `toneMap`
  - adjust image by setting H and S channels of each pixel

eight (8) functions to be graded in assignment (ctd.)

- `crops`
  - crop image
- `histogram`
  - plot histogram of image
- `saturated`
  - compute percentage of pixels which have at least one RGB channel value which has undergone clipping saturation
- `unsharpMask`
  - sharpen image

strategies

- start small
- Lab sheet week 10 first
- test RGB/HSL conversion against colour picker
- remember first 5 functions: infinite help from discord, demonstrators, fellow students
    - no marks for these questions; required for later q's

strategies

- submission to BB will be a single file `imageProcessing.py` with definitions and code for up to eight (8) functions
  - you may implement <8 functions, for <15 marks

- strongly encouraged to develop and test as follows:
  1. each function's beheviour in its own script (test it)
  2. define code into function in same file (test it)
  3. copy/paste working function into `imageProcessing.py`
     - we'll be making test code available to students
     - you can check *in advance* if your code works correctly!

strategies

Step 1: in `square.py`

```
# square

x = 3
print('{} squared = {:.4f}'.format(x,x**2))
```

Step 2: in `square_fn.py`

```
# square_fn
def f(x):
    return x**2

x = 3
print('{} squared = {:.4f}'.format(x,f(x)))
```

Step 3: in `imageProcessing.py`

```
def f(x):
    return x**2
```

Lecture summary

- xxx