Technical University of Łódź
Institute of Electronics
Medical Electronics Division

Image Processing and Computer Graphics

3*D* images

Author: Marek Kociński

March 2010

1 Purpose

To get acquainted with basic manipulation on $3D$ raw image data. The matplotlib module will be used to create publication quality figures.

Time

3×45 minutes

2 Tasks

1. Open Python interpreter window (Start→ Programy→ EPD32-6.0.2 → IDLE)
2. Open new Editor Window (File → New Window) and write your code into it.
3. Import needed modules, e.g. *Image, array*. Use construction:

```
import scipy as sc
from pylab import *
import array
import Image
```

4. Read data from the flie. Each voxel is 8-bit unsigned integr. The size of the image ($3D$ matrix) is $256 \times 256 \times 256$ voxels.

```
tmpfile = "qinp01_3000_036_3_256.raw"

fileobj = open(tmpfile, mode='rb')
binvalues = array.array('B')
binvalues.read(fileobj, 256*256*256)
```

5. Convert loaded data to SciPy array structure:

```
data = sc.array(binvalues, dtype=sc.uint8)
data = sc.reshape(data, (256,256,256))
fileobj.close()
```

6. Print basic information abou array

```
print data.size
print data.shape
```

7. It is possoble to *connect* SciPy array structure with Image object of the PIL module. Select 127 slice in each direction 1, 2, 3, convert to Image structure and show it.
8. Print maximum, minimum and mean of the data:

(a) dir. 1 *(b) dir* 2 *(c) dir* 3

Figure 1: Selected slice from $3D$ image

```
print data.max()
print data.min()
print data.mean()
```

9. Create MIP (Maximum Intensity Projection) of the image in each direction. The idea of MIP is presented in the Figure 2.

```
mip1 = Image.fromarray (data.max(0))
mip2 = Image.fromarray (data.max(1))
mip3 = Image.fromarray (data.max(2))
```

10. Present 4 images on one figure. It is possible to write this figue in different file formats, like: pdf, eps, png, ps, emf, raw or even vecor graphics svg.

```
fig = figure()

subplot(221)
imshow(im2,cmap=cm.gray)
colorbar()
subplot(222)
imshow(im1)
subplot(223)
imshow(mip2,cmap=cm.gray)
subplot(224)
imshow(mip3)
colorbar()
show()
```

11. Create new Python script. Load $3D$ data as in the previous example. Create one MIP image:

```
im1 = Image.fromarray (data.max(0))
```

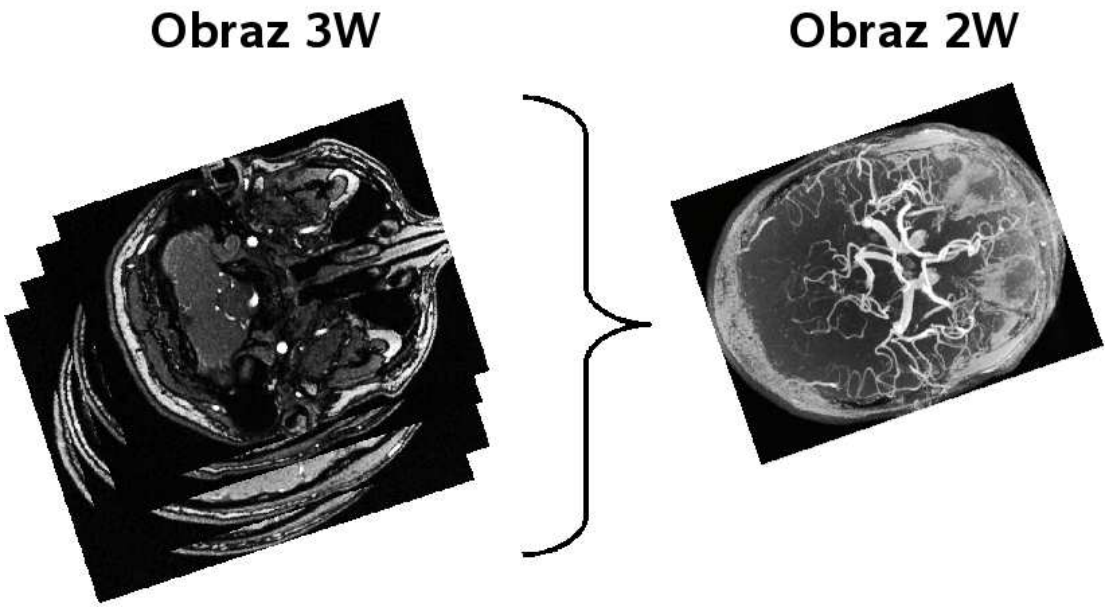


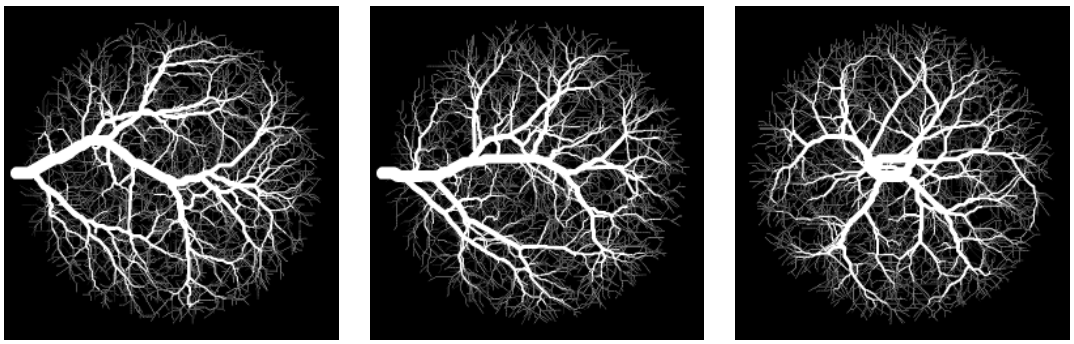
Figure 2: Four images on one figure

12. Invert $3D$ image and create mIP (minimum Intensity Projection)

```
im4 = Image.fromarray (d.min(0))
```

Hint 1: It is not good idea to use three for loops ;-). But if you decide so, the *Ctrl+z* key sequence may occur to be helpful...
Hint 2: Change data precision to *int16*, do invertion, and back to *uint8*.

```
a = sc.int16 (data)
...
...here invert image...
```



(a) dir. 1 *(b) dir* 2 *(c) dir* 3

Figure 3: MIP images of $3D$ data

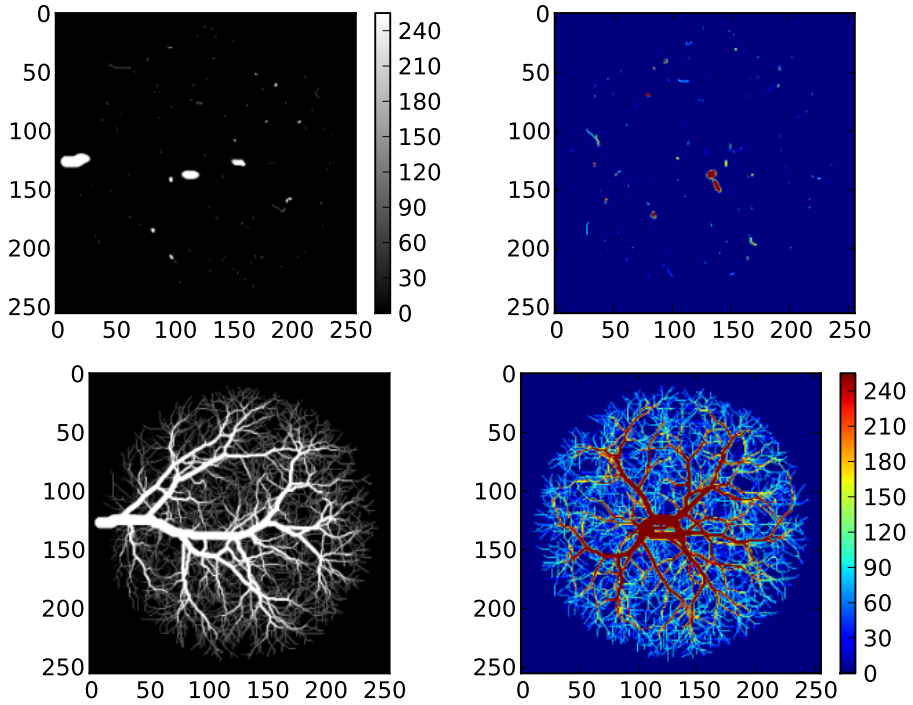


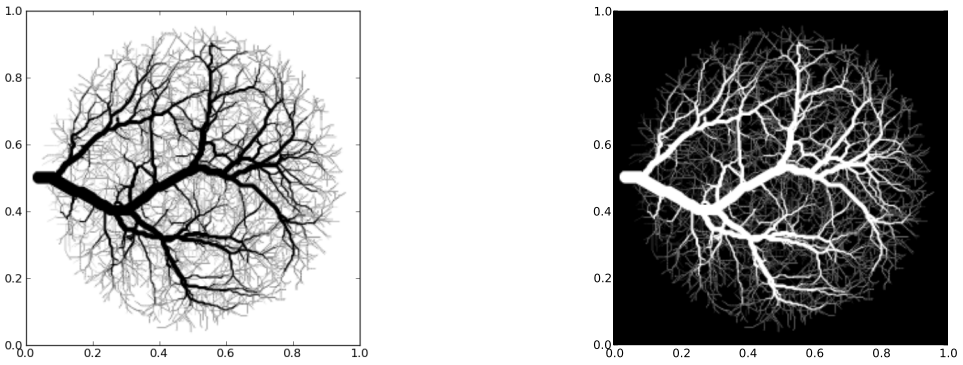
Figure 4: Four images on one figure

```
...
d = sc.uint8 (c)
```

13. By pressing 't' letter toogle between two images showed on one fiugure.

```
print "Press t ... :-)"
extent = (0,1,0,1)
img1 = imshow(im1, extent=extent, cmap=cm.gray )
img2 = imshow(im4, extent=extent, hold=True, cmap=cm.gray )
img2.set_visible(False)

def toggle_images(event):
    'toggle the visible state of the two images'
    if event.key != 't': return
    b1 = img1.get_visible()
    b2 = img2.get_visible()
    img1.set_visible(not b1)
    img2.set_visible(not b2)
```



(a) mIP *(b)MIP*

Figure 5: Toogle between MIP and mIP imges

```
    draw()

connect('key_press_event', toggle_images)
show()
```