



19.04.2018

Вычислительные модели с использованием научных библиотек Python Обработка изображений

Фильтры

F_1	F_2	F_3
F_4	F_5	F_6
F_7	F_8	F_9

I(i-1,j-1)	I(i-1,j)	I(i-1,j+1)
I(i,j-1)	I(i,j)	I(i, j+1)
I(i+1,j-1)	I(i+1,j)	I(i+1,j+1)

<u>Изображение</u>

<u>Фильтр</u>

$$I_{new}(i,j) = F_1 * I(i-1,j-1) + F_2 * I(i-1,j) + F_3 * I(i-1,j+1)$$

$$+ F_4 * I(i,j-1) + F_5 * I(i,j) + F_6 * I(i,j+1)$$

$$+ F_7 * I(i+1,j-1) + F_8 * I(i+1,j) + F_9 * I(i+1,j+1)$$



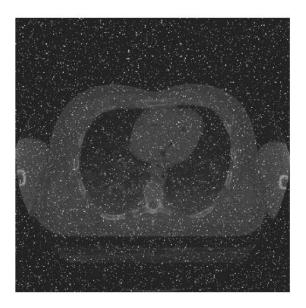


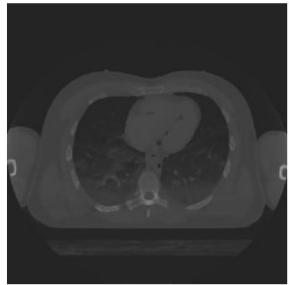
Медианный фильтр

import scipy.misc
import scipy.ndimage
from scipy.misc.pilutil import Image
opening the image and converting it to
grayscale
a =
Image.open(../Figures/ct_saltandpepper.png).
convert(L)

performing the median filter b = scipy.ndimage.filters.median_filter(a,size=5, footprint=None,output=None,mode='reflect', cval=0.0,origin=0)

b is converted from an ndarray to an image
b = scipy.misc.toimage(b)
b.save(../Figures/median_output.png)









Мах фильтр

import scipy.ndimage from scipy.misc.pilutil import Image

opening the image and converting it to grayscale

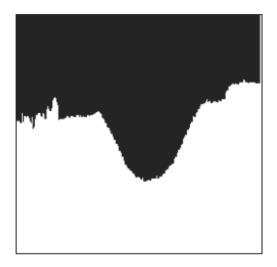
a = Image.open(../Figures/wave.png).convert(L)

performing maximum filter

b =

scipy.ndimage.filters.maximum_filter(a,size=5, footprint=None,output=None,mode=reflect, cval=0.0,origin=0)

b is converted from an ndarray to an image
b = scipy.misc.toimage(b)
b.save(../Figures/maxo.png)







Min фильтр

import scipy.misc import scipy.ndimage from scipy.misc.pilutil import Image

opening the image and converting it to grayscale a = Image.open(../Figures/wave.png).convert(L)

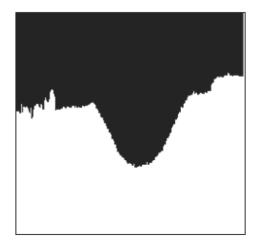
performing minimum filter

b =

scipy.ndimage.filters.minimum_filter(a,size=5, footprint=None,output=None,mode=reflect, cval=0.0,origin=0)

b is converted from an ndarray to an image b = scipy.misc.toimage(b)

saving b as mino.png
b.save(../Figures/mino.png)





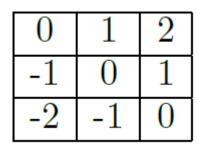




Фильтра Собеля

-1	-2	-1
0	0	0
1	2	1

-1	0	1
-2	0	2
-1	0	1

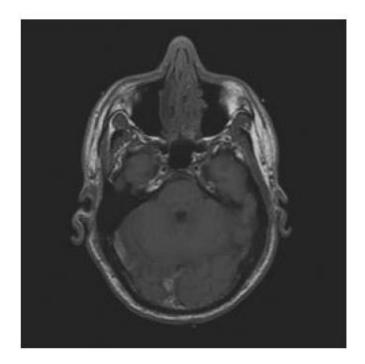


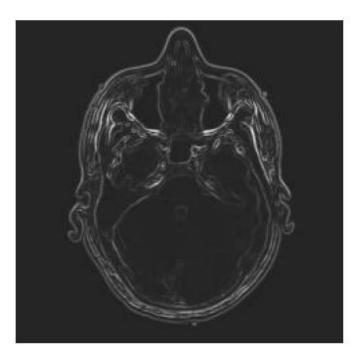
-2	-1	0
-1	0	1
0	1	2

<u>Горизонтальный</u>

<u>Вертикальный</u>

<u>Диагональный</u>





Определение границ по градиенту яркости

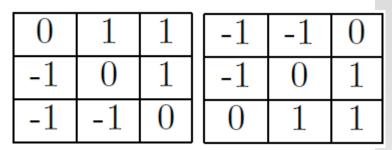




Фильтра Прюитта

-1	-1	-1
0	0	0
1	1	1

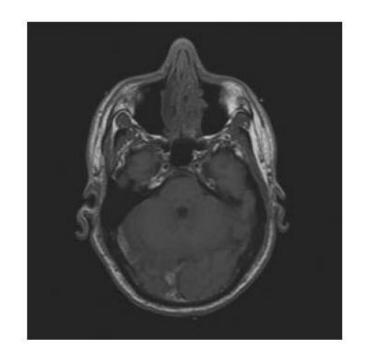
-1	0	1
-1	0	1
-1	0	1

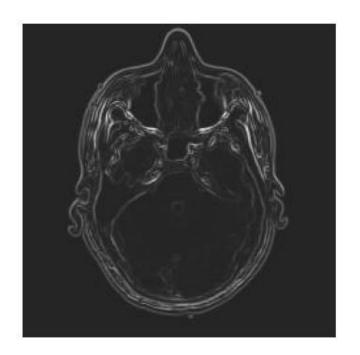


<u>Горизонтальный</u>

<u>Вертикальный</u>

<u>Диагональный</u>





Определение границ по градиенту яркости





Фильтра Лапласа

0	1	0
-1	4	-1
0	-1	0

-1	-1	-1
-1	8	1
-1	-1	- 1



b = scipy.ndimage.filters.gaussian_laplace(a,1, mode=reflect)

b is converted from an ndarray to an image b = scipy.misc.toimage(b)

b.save(../Figures/log_vh1.png)

0	ථ	®
0		θ
Q	\bigcirc	0
	\$\$\$	2

0	0	-1	0	0
0	-1	-2	-1	0
-1	-2	16	-2	-1
0	-1	-2	-1	0
0	0	-1	0	0

$$G(r) = -e^{rac{-r^2}{2\sigma^2}}$$
 - Сглаживание Гаусса



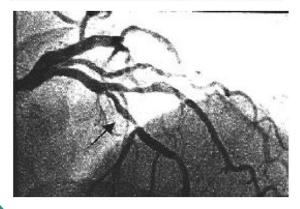


Гамма - коррекция





$$\gamma = 0.5$$



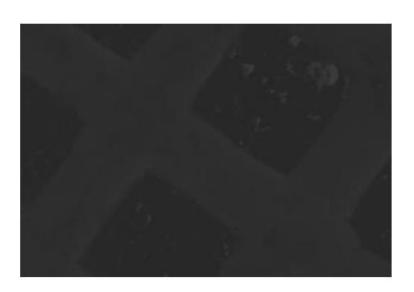
$$\gamma = 5$$

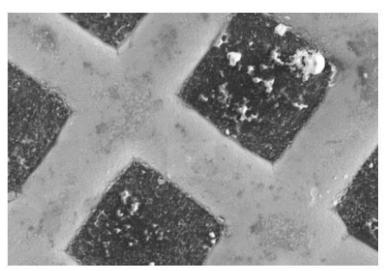
$$t(i,j) = e^{\gamma * \ln(I_{norm})} * 255.$$

opening the image and converting it to grayscale a = Image.open(../Figures/angiogram1.png). convert(L) # a is converted to an ndarray b = scipy.misc.fromimage(a) # gamma is initialized gamma = 0.5# b is converted to type float b1 = b.astype(float)# maximum value in b1 is determined b3 = numpy.max(b1)# b1 is normalized b2 = b1/b3# gamma-correction exponent is computed b3 = numpy.log(b2)*gamma# gamma-correction is performed c = numpy.exp(b3)*255.0# c is converted to type int c1 = c.astype(int)# c1 is converted from ndarray to image d = scipy.misc.toimage(c1)



Log – преобразование





$$t(i,j) = k \log(1 + I(i,j))$$
$$k = \frac{L - 1}{\log(1 + |I_{max}|)}$$

a = Image.open(../Figures/bse.png).convert(L)
b = scipy.misc.fromimage(a)
b1 = b.astype(float)
maximum value in b1 is determined
b2 = numpy.max(b1)
performing the log transformation
c = (255.0*numpy.log(1+b1))/numpy.log(1+b2)
c is converted to type int
c1 = c.astype(int)
c1 is converted from ndarray to image
d = scipy.misc.toimage(c1)





Выравнивание гистограммы

$$C(i) = \sum_{j=0}^{i} p_x(j)$$
 $h(u) = round\left(\frac{C(u) - C_{min}}{1 - C_{min}} * (L-1)\right)$

```
# img is converted to an ndarray
img1 = scipy.misc.fromimage(img)
# 2D array is convereted to an 1D
fl = img1.flatten()
# histogram and the bins of the image are
computed
hist, bins = np.histogram(img1,256,[0,255])
# cumulative distribution function is computed
cdf = hist.cumsum()
# places where cdf=0 is masked or ignored and
# rest is stored in cdf m
cdf_m = np.ma.masked_equal(cdf,0)
# histogram equalization is performed
num\_cdf\_m = (cdf\_m - cdf\_m.min())*255
den_cdf_m = (cdf_m.max()-cdf_m.min())
cdf_m = num_cdf_m/den_cdf_m
```

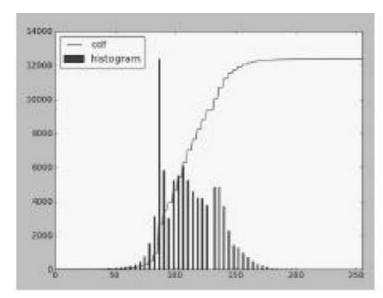
the masked places in cdf_m are now 0
cdf = np.ma.filled(cdf_m,0).astype(uint8)
cdf values are assigned in the flattened array
im2 = cdf[fl]
im2 is 1D so we use reshape command to
make it into 2D
im3 = np.reshape(im2,img1.shape)
converting im3 to an image
im4 = scipy.misc.toimage(im3)
saving im4 as hequalization_output.png
in Figures folder
im4.save(../Figures/hequalization_output.png)



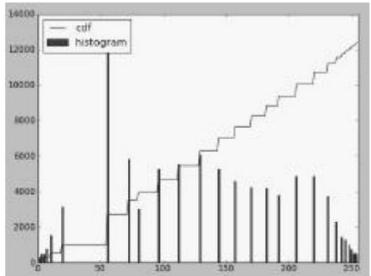


Выравнивание гистограммы





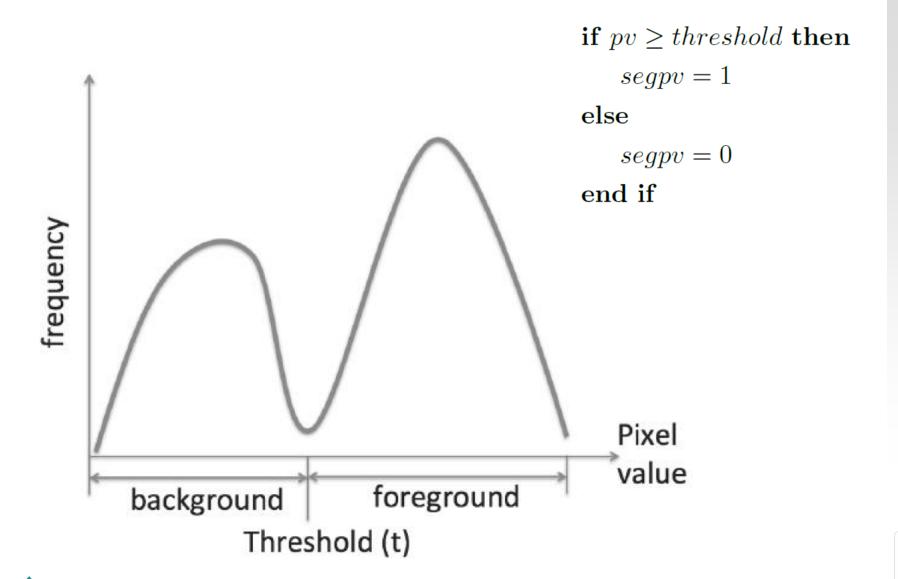








Сегментация на основе гистограммы

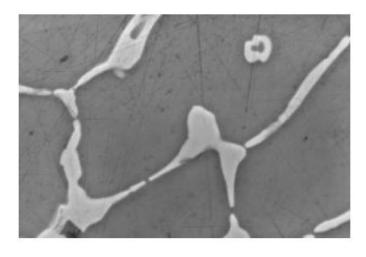




Сегментация. Метод Оцу

$$v_{within} = P_b(t)v_b + P_f(t)v_f$$

$$v_{inbetween} = v - v_{within} = P_b P_f (m_b - m_f)^2.$$





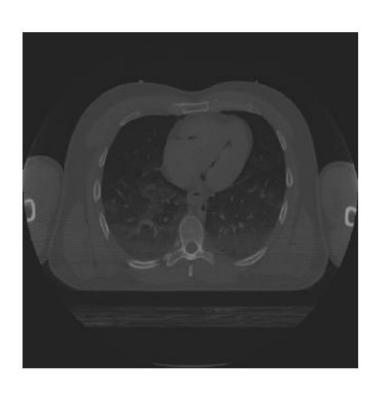
a =
Image.open(../Figures/sem3.png).convert("L")
a is converted to an ndarray
a = scipy.misc.fromimage(a)
performing Otsus thresholding
thresh = threshold_otsu(a)
pixels with intensity greater than
theshold are kept
b = a > thresh
b is converted from ndimage to
b = scipy.misc.toimage(b)





Сегментация. Метод Реньи

$$H_{\alpha}(x) = \frac{1}{1-\alpha} \log_a \left(\sum_{i=1}^n (p(x_i))^{\alpha} \right)$$





Сегментация небольших объектов





Адаптивная сегментация.

What is Para View?

and the season of the season o

The free and graph column interface (CCCI) has an open. Destries and resultive user interface that still gives you five free and open makes and display processing needed to explore and present the first f

Per View has programming long map for sampled control. As with the GUI, the profess company street are the windly an expending long map for sampled control. As with the GUI, the profess compand control is easy to make a solid, including the shifter account in the GUI and nove there our as account forman resoluble programming and provention excluding the shifter in write sampled filters that turn on the server that have access to many lot of poor these on a long possible machine.

Исходное изображение

•

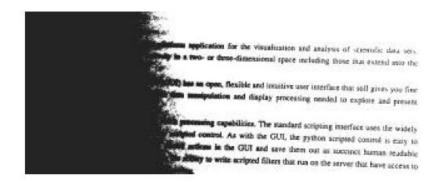
What is ParaView?

ParaView is an open-source, multi-platform application for the visualization and analysis of scientific data sets, primarily those that are defined natively in a two- or date-dimensional space including those that extend into the temporal dimension.

The front end graphical user interface (GUI) has an open, flexible and intuitive user interface that sall gives you fine grained and open ended occurol of the data manipulation and display processing needed to explore and present complex data as you see fit.

ParaView has extensive scripting and batch processing capabilities. The standard scripting interface uses the widely used python programming language for scripted control. As with the GUI, the python scripted control is easy to fearn, notably including the ability to record actions in the GUI and save them out as succinct human madable python programs and powerful, including the ability to write scripted filters that run on the server that have access to every bit of your data on a large parallel machine.

Адаптивная сегментация



<u>Метод Оцу</u>

a = Image.open(.example1.png).convert(L)

a is converted to an ndarray

a = scipy.misc.fromimage(a)

performing adaptive thresholding

b =threshold_local(a,

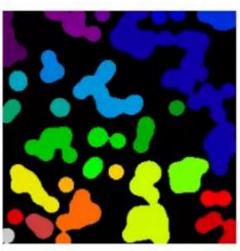
block_size=11,offset=0)





Сегментация. Связанные области





```
>>> n = 20
>>> 1 = 256
\rightarrow > im = np.zeros((I, I))
>>> points = I * np.random.random((2, n
2))
>>> im[(points[0])_astype(np_int),
(points[1]).astype(np.int)] = 1
>>> im = filters_gaussian_filter(im, sigma=1
/ (4. * n))
>>> blobs = im > im.mean()
>>> from skimage import measure
>>> all_labels = measure.label(blobs
>>> blobs_labels = measure.label(blobs,
background=0)
```





Задание





<u>Определить птиц на</u> <u>изображении</u>

