Дигитална обрада звука и слике 3

Саша Малков
Математички факултет
2023/2024

Дигитална обрада звука и слике

Тема 1
Дигитална обрада слике
(наставак)

Филтер *UnsharpMask*

- Слично као изоштравање
  - од оригиналне слике се одузима инвертована замућена слика
    - додаје се поређење тачке са њеном околином
    - израчуна се просек околине и пореди са тачком
- Додаје се поређење разлике оригиналне тачке од замућене околине
  - тачка се мења само ако се разликује од околине више од датог прага

Филтер *UnsharpMask* (2)

- Параметри
  - *radius* – полупречник круга чија се средња вредност рачуна
    - слично као полупречник код Гаусовог замућења
    - т.ј. величина матрице
  - *percent* – проценат повећавања разлике
    - као интензитет централне тачке код матрице изоштравања
  - *threshold* – праг повећавања разлике
    - ново – не ради се изоштравање фиких прелаза

UnsharpMask – Примери са различитим интензитетом операције (*percent*)

```
tile(
    img,
    img.filter( ImageFilter.UnsharpMask(3, 50, 0) ),
    img.filter( ImageFilter.UnsharpMask(3, 100, 0) ),
    img.filter( ImageFilter.UnsharpMask(3, 150, 0) ),
    img.filter( ImageFilter.UnsharpMask(3, 200, 0) )
).show()
```


UnsharpMask – Примери са различитим полупречником операције (*radius*)

```
tile(
    img,
    img.filter( ImageFilter.UnsharpMask( 2, 150,0) ),
    img.filter( ImageFilter.UnsharpMask( 3, 150,0) ),
    img.filter( ImageFilter.UnsharpMask( 5, 150,0) ),
    img.filter( ImageFilter.UnsharpMask( 10, 150,0) )
).show()
```


UnsharpMask – Примери са различитим прагом примене операције (*treshold*)

```
tile(
    img,
    img.filter( ImageFilter.UnsharpMask(3,150, 0 ) ),
    img.filter( ImageFilter.UnsharpMask(3,150, 10 ) ),
    img.filter( ImageFilter.UnsharpMask(3,150, 50 ) )
).show()
```


Филтер *UnsharpMask* (3)

- Ако се праг постави на 0, онда се практично не разликује од одговарајућег конволутивног филтера изоштравања
- Ако се постави умерено висок праг и висок проценат, онда операција поприма карактеристике детекције ивица

UnsharpMask – Примери издвајања ивица

```
tile(
    img,
    img.filter( ImageFilter.UnsharpMask(3, 2500, 0 ) ),
    img.filter( ImageFilter.UnsharpMask(3, 2500, 10 ) ),
    img.filter( ImageFilter.UnsharpMask(3, 2500, 15 ) ),
    img.filter( ImageFilter.UnsharpMask(3, 2500, 20 ) )
).show()
```

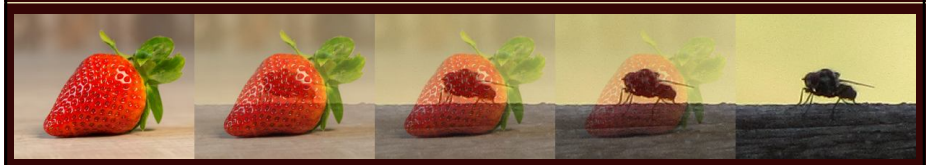

UnsharpMask – Примери издвајања ивица

```
tile(
    img,
    img.filter( ImageFilter.UnsharpMask(5, 2500, 0 ) ),
    img.filter( ImageFilter.UnsharpMask(5, 2500, 10 ) ),
    img.filter( ImageFilter.UnsharpMask(5, 2500, 15 ) ),
    img.filter( ImageFilter.UnsharpMask(5, 2500, 20 ) )
).show()
```


Комбиновање слика исте величине

- Комбиновање слика повезује садржаје двеју слика у једну
- Операција се примењује на сваки канал посебно
  - Стапање (енгл. *blend*): *pix1 * factor + pix2 * (1-factor)*, 0<= *factor* <=1
  - Избор тамнијих тачака: *min(pix1, pix2)*
  - Избор светлијих тачака: *max(pix1, pix2)*
  - Апсолутна разлика: *abs(pix1 - pix2)*
  - Сабирање: *(pix1 + pix2) / factor + offset*
  - Множење: *pix1 * pix2 / MAX*
  - *Screen*: *MAX – (MAX-pix1) * (MAX-pix2) / MAX*
  - *Overlay:* за тамне тачке прве слике као множење, иначе *Screen*
  - *Hard light*: као *Overlay* али у односу на другу слику
  - ...

Стапање



- Стапање (енгл. *blend*): *pix1 * factor + pix2 * (1-factor)*, 0<= *factor* <=1

```
tile(
    img1,
    PIL.ImageChops.blend( img1, img2, 0.25 ),
    PIL.ImageChops.blend( img1, img2, 0.5 ),
    PIL.ImageChops.blend( img1, img2, 0.75 ),
    img2
).show()
```

Тамније, Светлије, Апс.разлика

- Избор тамнијих тачака: *min(pix1, pix2)*
- Избор светлијих тачака: *max(pix1, pix2)*
- Апсолутна разлика: *abs(pix1 - pix2)*

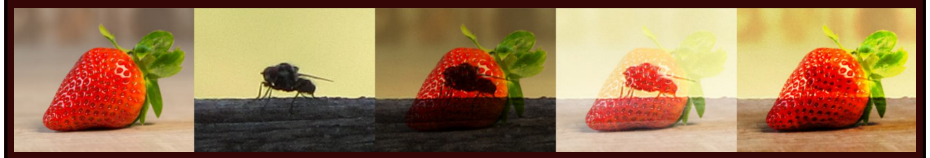
```
tile(
    img1, img2,
    PIL.ImageChops.darker( img1, img2 ),
    PIL.ImageChops.lighter( img1, img2 ),
    PIL.ImageChops.difference( img1, img2 )
).show()
```


Сабирање

- Сабирање: *(pix1 + pix2) / factor + offset*

```
tile(
    img1,
    img2,
    PIL.ImageChops.add( img1, img2 ),
    PIL.ImageChops.add( img1, img2, 1.5 ),
    PIL.ImageChops.add( img1, img2, 2 )
).show()
```


Множење и сличне операције



- Множење: *pix1 * pix2 / MAX*
- *Screen*: *MAX – (MAX-pix1) * (MAX-pix2) / MAX*
- *Overlay:* за тамне тачке прве слике као множење, иначе *Screen*

```
tile(
    img1, img2,
    PIL.ImageChops.multiply( img1, img2 ),
    PIL.ImageChops.screen( img1, img2 ),
    PIL.ImageChops.overlay( img1, img2 )
).show()
```


Остало...

- *Hard light*: као *Overlay* али у односу на другу слику

```
tile(
    img1, img2,
    PIL.ImageChops.soft_light( img1, img2 ),
    PIL.ImageChops.hard_light( img1, img2 )
).show()
```

Лепљење слика

- На једну слику се лепи друга:
  - img1.paste(img2)
  - img1.paste(img2, (x,y))
  - img1.paste(img2, (x1,y1,x2,y2))

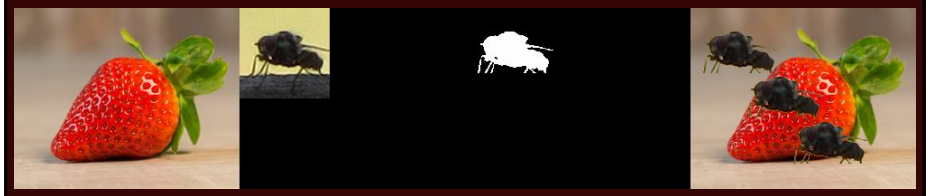
Лепљење слика

```
img = img1.copy()
for i in [ 10, 50, 90 ]:
    img.paste( img2, (i,i) )
tile( img1, img2, img ).show()
```


Лепљење слика са маском

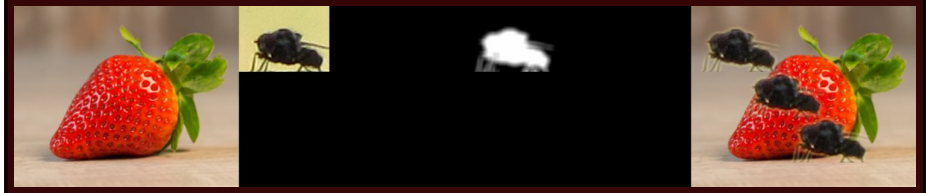
- Лепљење уз употребу маске:
  - img1.paste(img2, mask = imgm)
  - img1.paste(img2, (x,y), mask = imgm)
  - img1.paste(img2, (x1,y1,x2,y2), mask = imgm)
- Маска одређује начин лепљења
  - маска мора да има исте димензије као друга слика
    - и само један канал, “Л”
  - црна боја маске бира само прву слику
  - бела боја маске бира само другу слику
  - сива боја маске их уклапа

Лепљење слика са црно белом маском



```
img = img1.copy()
for i in [ 0, 50, 100 ]:
    img.paste( img2, (i,i), mask=mask )
tile( img1, img2, mask, img ).show()
```

Лепљење слика са маском са прелазима



```
img = img1.copy()
mask = mask.filter( ImageFilter.BoxBlur(5) )
for i in [ 0, 50, 100 ]:
    img.paste( img2, (i,i), mask=mask )
tile( img1, img2, mask, img ).show()
```


Лепљење слика са маском са прелазима

```
bmask1 = mask.filter( ImageFilter.BoxBlur(1) )
bmask3 = mask.filter( ImageFilter.BoxBlur(3) )
bmask5 = mask.filter( ImageFilter.BoxBlur(5) )
... :
    img.paste( img2, (i,i), mask=bmask... )
```


Лепљење слика са маском са прелазима

```
bmask1 = mask.filter( ImageFilter.BoxBlur(1) )
bmask3 = mask.filter( ImageFilter.BoxBlur(3) )
bmask5 = mask.filter( ImageFilter.BoxBlur(5) )
... :
    img.paste( img2, (i,i), mask=bmask... )
```


Цртање по слици

- Библиотека *Pillow* подржава и цртање
- Припрема објекта за цртање:

```
d = ImageDraw.Draw( img )
```

- Цртање:

```
d.rectangle( (50,50,100,100 ), outline='black', fill='white' )
d.line((50,50,100,100), fill=128)
```

Цртање по слици


```
d = ImageDraw.Draw( img )

d.rectangle( (50,50,100,100 ), outline='black', fill='yellow' )
d.line((50,50,100,100), fill=128)
d.line((50,100,100,50), fill=128)

fnt = ImageFont.truetype( "C:\\Windows\\Fonts\\cambriab.ttf", 24 )
d.text( (10, 20), "Zdravo!!!", font=fnt, fill=(0, 0, 192) )
d.text( (10, 110), "Здраво!!!", font=fnt, fill=(192, 0, 0) )
```


Цртање векторских слика

- Библиотека *drawsvg* подржава цртање векторских слика
- Цртеж се чува у формату *SVG*
- Цртеж представља колекцију елементарних објеката
- Гради се додавањем нових елемената

Прављење новог цртежа

- Нови цртеж се прави помоћу:

```
d = dw.Drawing( 800, 800, origin=(-400, -400) )
```

- Прва два аргумента су ширина и висина цртежа у пикселима
- Аргумент *origin* одређује координате горњег левог темена
  - у примеру, цртеж је област од (-400,-400) = лево горње теме, па до (399,399) = десно доње теме

Додавање елемената

- Нови елементи се додају са:

```
d.Append( ...нови елемент... )
```

- А претходно се праве методима за цртање:

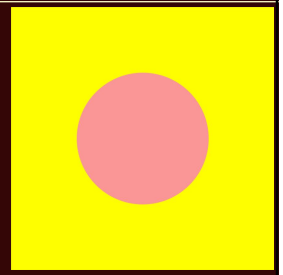
```
dw.Line( ... )
dw.Rectangle( ... )
dw.Circle( ... )
dw.Ellipse( ... )
...
```

Преузимање записа у формату *SVG*

- Цртеж у формату *SVG* се добија са:

```
d.as_svg()
```


Пример векторског цртежа



```
import drawsvg as dw

d = dw.Drawing( 800, 800, origin=(-400, -400) )
d.append( dw.Rectangle( -400, -400, 800, 800, fill='yellow', stroke='black' ))
d.append( dw.Circle( 0, 0, 200, fill='rgb(250,150,150)' ))

print(d.as_svg())

filename = "my.svg"
svgfile = open( filename, "w" )
svgfile.write( d.as_svg() )
svgfile.close()
```


Конверзија *SVG* у *PNG*

- Има више начина
- Најбоље то ради *Inkscape*
- Програмски:

```
cmd_list = [ 'C:\\Program Files\\Inkscape\\bin\\inkscape.exe',
             '--export-filename=' + outfilename,
             infilename ]
p = subprocess.run( cmd_list, capture_output=True )
```


Литература за тему

- **Pillow Documentation** https://pillow.readthedocs.io/en/stable/index.html
- **drawsvg Documentation** https://pypi.org/project/drawsvg/
- Stephen Gruppetta, **Image Processing With the Python Pillow Library,** https://realpython.com/image-processing-with-the-python-pillow-library/