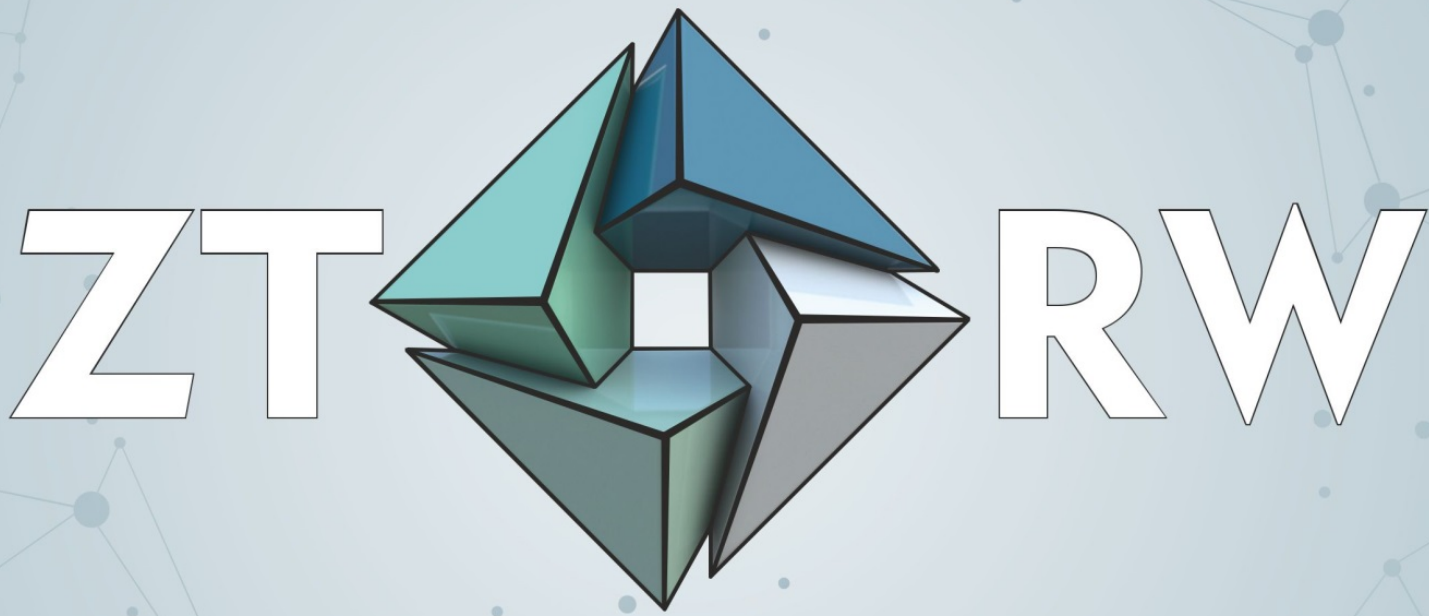
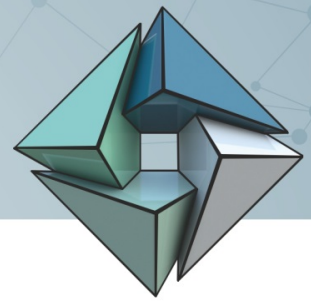

ZT
RW



Digital image processing #5

Basic neighborhood operations

Kamil Szeląg

Warsaw University of Technology
Institute of Micromechanics and Photonics
Virtual Reality Techniques Division

December 13, 2017

Basic neighborhood operations

Filtrations

- Task #35
  - Load color image
  - Perform blurring operations: mean blur, gaussian blur and median blur
  - Discuss diferences between results
- Task #37
  - Load color image
  - Perform gaussian blurring with changing sigma value and compare it with mean blurring
- Task #37
  - Load color image
  - Use unsharp mask filter on loaded image

Basic neighborhood operations

Filtrations

```
//Task #35
        cv::Mat img = cv::imread("lena.bmp");
        cv::Mat gaussian_result, mean_result, median_result;
        cv::GaussianBlur(img, gaussian_result, cv::Size(5, 5), 1, 1);
        cv::blur(img, mean_result, cv::Size(5, 5));
        cv::medianBlur(img, median_result,5);

//Task #36
        cv::Mat img = cv::imread("lena.bmp");
        cv::Mat gaussian_result, mean_result;
        cv::blur(img, mean_result, cv::Size(5, 5));
        for(int i=0;i<100;i++)
    {
    cv::GaussianBlur(img, gaussian_result, cv::Size(5, 5),0.1*i, 0.1*i);
    cv::imshow("Gaussian",gaussian_result);
    cv::imshow("Mean",mean_result);
    cv::waitKey(30);
    }
 //Task #37
        cv::Mat img = cv::imread("lena.bmp");
        cv::Mat gaussian_result,unsharp_mask;
        double alpha = 1;
        cv::GaussianBlur(img, gaussian_result, cv::Size(5, 5), 1, 1);
        cv::addWeighted(img, 1 + alpha, gaussian_result, -alpha, 0, unsharp_mask);
```

Basic neighborhood operations

0 sum filtrations

- Task #38
  - Load image
  - Calculate Sobel derivatives of image
- Task #39
  - Load image
  - Calculate Laplacian of image. What is Laplacian and why it is good for edge detection?

Basic neighborhood operations

Complex transformations

```
//Task #38
        cv::Mat img = cv::imread("lena.bmp");
        cv::Mat tmp, result;
        cv::Sobel(img, tmp, CV_16S, 1, 0, 3, 1, 0);
        cv::normalize(tmp, tmp, -255, 256, cv::NORM_MINMAX);
        tmp.convertTo(result, CV_8U);

//Task #39

        cv::Mat img = cv::imread("lena.bmp");
        cv::Mat tmp, result;
        cv::Laplacian(img, tmp, CV_16S, 3, 1, 0);
        cv::normalize(tmp, tmp, -255, 256, cv::NORM_MINMAX);
        tmp.convertTo(result, CV_8U);
```

Basic neighborhood operations

Own kernel filtrations

- Task #40
  - Load image
  - Use own kernel in orged to perform filtration operation.

Basic neighborhood operations

Complex transformations

```
//Task #39

        cv::Mat img = cv::imread("lena.bmp");
        cv::Mat result, tmp;
        cv::Mat kernel = (cv::Mat_<double>(3, 3) << -2, -1, 0, -1, 0, 1, 0, 1, 2);

        cv::filter2D(img, tmp, CV_16S,kernel);
        cv::normalize(tmp, tmp, -255, 256, cv::NORM_MINMAX);
        tmp.convertTo(result, CV_8U);
```

Basic neighborhood operations

Morphological transformations

- Task #41
  - Load color image
  - Perform erosion and dilation on image
- Task #42
  - Load color image
  - Perform open and close operations on image
- Task #43
  - Load color image
  - Perform morphological gradient, top hat and black hat operations

Basic neighborhood operations

Morphological transformations

```
//Task #41
        cv::Mat img = cv::imread("lena.bmp");
        cv::Mat gray,binary;
        cv::cvtColor(img, gray, CV_BGR2GRAY);
        cv::threshold(gray, binary, 100, 255, CV_THRESH_BINARY);
        cv::Mat eroded, dilated;
        cv::erode(binary, eroded, cv::Mat()); //default kernel
        cv::Mat customKernel = cv::getStructuringElement(CV_SHAPE_CROSS, cv::Size(5,5));
        cv::dilate(binary, dilated, customKernel); //custom kernel

//Task #42
        cv::Mat img = cv::imread("lena.bmp");
        cv::Mat gray,binary,open,close;
        cv::cvtColor(img, gray, CV_BGR2GRAY);
        cv::threshold(gray, binary, 100, 255, CV_THRESH_BINARY);
        cv::Mat eroded, dilated;
        cv::erode(binary, eroded, cv::Mat());
        cv::dilate(eroded,open,cv::Mat());
        cv::dilate(binary, dilated, cv::Mat());
        cv::erode(dilated, close, cv::Mat());

//Task #43
        cv::Mat img = cv::imread("lena.bmp");
        cv::Mat gray,binary;
        cv::cvtColor(img, gray, CV_BGR2GRAY);
        cv::threshold(gray, binary, 100, 255, CV_THRESH_BINARY);
        cv::Mat blackhat,tophat,gradient;
        cv::morphologyEx(img, blackhat, cv::MORPH_BLACKHAT, cv::Mat());
        cv::morphologyEx(img, tophat, cv::MORPH_TOPHAT, cv::Mat());
        cv::morphologyEx(img, gradient, cv::MORPH_GRADIENT, cv::Mat());
```

Basic neighborhood operations

Morphological transformations

- Task #44
  - Load color image
  - Perform own morphological operation using custom kernel (e.g. hit-or-miss)

Basic neighborhood operations

Morphological transformations

```
//Task #40
cv::Mat img = cv::imread("lena.bmp");
cv::Mat gray,binary;
cv::cvtColor(img, gray, CV_BGR2GRAY);
cv::threshold(gray, binary, 100, 255, CV_THRESH_BINARY);
cv::Mat hitmiss;

cv::Mat kernel = (cv::Mat_<double>(5, 5) << 1, 1, 1, 1, 1,
        0, 0, 0, 0, 1,
        0, 0, 0, 0, 1,
        0, 0, 0, 0, 1,
        0, 0, 0, 0, 1);
cv::morphologyEx(img, hitmiss, cv::MORPH_HITMISS, kernel);
```

The end

http://ztrw.mchtr.pw.edu.pl