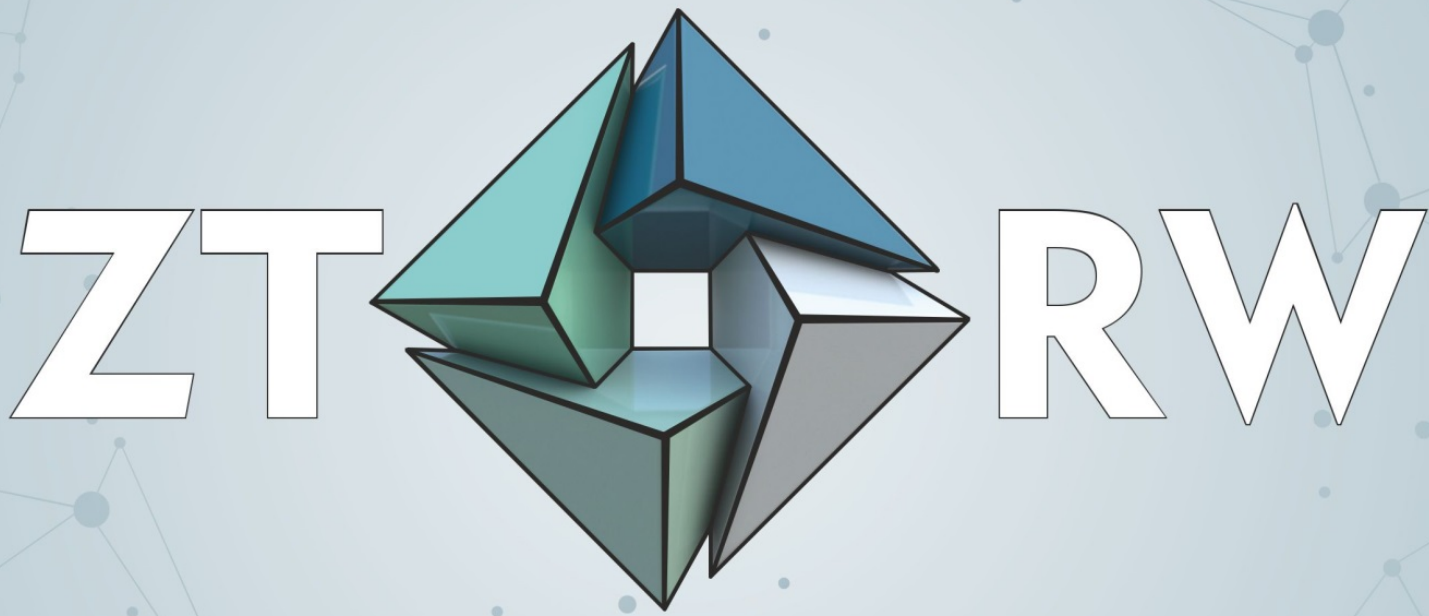
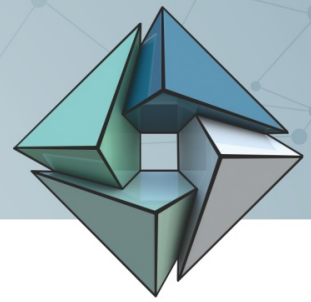

ZT
RW



Digital image processing #2

Pixelwise operations

Kamil Szeląg

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

November 3, 2017

Pixel arithmetics

Add, sub, mul, div

- Task #10
  - Load two images from filesystem
  - Adjust image sizes
  - Create 2 new output images
  - Perform '+','-' operations on that images
- Task #11
  - Perform Task #10 with usage of OpenCV methods. Is there any difference?

Pixel arithmetics

Misc operation

```
//Load images and adjust image sizes and types
    cv::Mat img1 = cv::imread("lena.bmp"); //load first image
        cv::Mat img2 = cv::imread("dancer.jpg"); //load second image

        if(img1.type()!=img2.type()) //if types aren't equal
        {
                img2.convertTo(img2, img1.type(), 1, 0); //convert second image type
        }
    //new size as mean value of image sizes - one of possible solutions
        cv::Size size = (img1.size() + img2.size()) / 2;

        cv::resize(img1, img1, size); // resize 1st image
        cv::resize(img2, img2, size); //resize 2nd image

    //preapre output
     cv::Mat res1;
        res1 = cv::Mat::zeros(size, img1.type());
```

Pixel arithmetics

Create and prepare outputs

```
// Task #10
      //Load images and prepare outputs
      res1 = img1 + img2;
      res2 = img1 - img2;

// Task #11
//Load images and prepare outputs
      //adjust image types in torder to have better results
      img1.convertTo(img1, CV_32FC3, 1.0 / 255.0, 0);
      img2.convertTo(img2, CV_32FC3, 1.0 / 255.0, 0);
      cv::add(img1, img2, res1); //add images
      cv::normalize(res1, res1, 0, 1, cv::NORM_MINMAX); //normalize image

      cv::subtract(img1, img2, res2); //subtract images
      cv::multiply(img1, img2, res3); //multiply images
      cv::divide(img1, img2, res4); //divide
      cv::absdiff(img1, img2, res5); // absolute difference |x1-x2|
```

Pixel arithmetics

Mask

- Task #12
  - Load two images from filesystem
  - Adjust image sizes
  - Create new output image
  - Perform single arithmetic operation on that images - use mask

Pixel arithmetics

Create and prepare outputs

```
// Task #12
//Load images and prepare outputs
  mask = cv::Mat::zeros(size, CV_8U); //create mask
       cv::circle(mask, cv::Point(200, 200), 100, cv::Scalar(255), -1); //add white circle
       cv::add(img1, img2, res1); //add without mask
       cv::add(img1, img2, res2,mask); //add with mask - what is the difference?
       img1.setTo(cv::Scalar(0, 0, 255),mask); //bonus: setTo with mask
```

Pixel arithmetics

Constant values

- Task #13
  - Load color image
  - Change all pixel values to 50,150,250
  - Use cv::Scalar
- Task #14
  - Load color image
  - Change all pixel values to 0,0,255 inside 80x80 rectangle inside image
- Task #15
  - Load grayscale image
  - Decrease all pixel values by 40
- Task #16
  - Load color image
  - Multiply image red channel by 2.

Pixel arithmetics

Create and prepare outputs

```
//Task #13
    cv::Mat img1 = cv::imread("lena.bmp");
    img1.setTo(cv::Scalar(50, 100, 200));
    img1 = cv::Scalar(50, 150, 250);
    img1 = 255; //what happend?
    img1 = cv::Scalar(255); //any difference?
    img1 = cv::Scalar::all(255); // is it white already?

//Task #14
    cv::Mat img1 = cv::imread("lena.bmp");
    //rectangular mask
    cv::Mat mask = cv::Mat::zeros(img1.size(), CV_8U);
    cv::rectangle(mask,cv::Point(210,210),cv::Point(290,290), cv::Scalar(255),-1);
    img1.setTo(cv::Scalar(0, 0, 255),mask); //setTo with mask
    img1(cv::Rect(210, 210, 80, 80)).setTo(cv::Scalar(0, 255, 255)); //ROI approach

//Task #15
    cv::Mat img1 = cv::imread("lena.bmp",0); //load grayscale
    //Four different ways to solve a problem - possible only with grayscale
    img1 = img1 - cv::Scalar(40); //subtract
    img1 = img1 + 40; //add
    cv::subtract(img1, cv::Scalar(40), img1); //subtract
    cv::add(img1, 40, img1); //add

//Task #16
    cv::Mat img1 = cv::imread("lena.bmp");
    cv::multiply(img1, cv::Scalar(1, 1, 2), img1);
```

Pixel arithmetics

Weighted sum

- Task #17
  - Load two color image
  - Perform weighted sum between images

Pixel arithmetics

Create and prepare outputs

```
// Task 17
    cv::Mat img1 = cv::imread("lena.bmp");
    cv::Mat img2 = cv::imread("dancer.jpg");
    cv::Mat res1,res2;
    cv::Mat mask;
    if(img1.type()!=img2.type())
    {
        img2.convertTo(img2, img1.type(), 1, 0); //adjust image types
    }
    cv::Size size = (img1.size() + img2.size()) / 2; //new size as mean value
    cv::resize(img1, img1, size); // resize 1st image
    cv::resize(img2, img2, size); //resize 2nd image
    cv::addWeighted(img1, 0.4, img2, 0.6, -30, res1); // res = 0.4*img1+0.6*img2-30

    //BONUS: Crossfade between images
    for(int i=0;i<100;i++)
    {
        cv::addWeighted(img1, 0.01*i, img2, 1- 0.01*i, 0, res1);
        cv::imshow("test",res1);
        cv::waitKey(50); //20fps
    }
```

Pixel arithmetics

Logic operations

- Task #18
  - Load grayscale image
  - Make image negation
- Task #19
  - Load two images with filled circles
  - Perform AND, OR, XOR, NAND, NOR operations on that images - store each result in separate array
- Task #20
  - Create 5x5 matrix with values 2
  - Create 5x5 mask matrix
  - Perform copyTo, setTo with mask

Pixel arithmetics

Create and prepare outputs

```
//Task #18
    cv::Mat img1 = cv::imread("lena.bmp");
    cv::bitwise_not(img1, img1);

//Task #19
    cv::Mat img1 = cv::Mat::zeros(500, 500, CV_8U);
    cv::Mat img2 = cv::Mat::zeros(500, 500, CV_8U);

    cv::circle(img1, cv::Point(200, 250), 100, cv::Scalar::all(255), -1);
    cv::circle(img2, cv::Point(300, 250), 100, cv::Scalar::all(255), -1);

    //Prepare outputs res1,res2,res3,res

    cv::bitwise_and(img1, img2, res1);
    cv::bitwise_or(img1, img2, res2);
    cv::bitwise_xor(img1, img2, res3);
    cv::bitwise_and(img1, img2, res4);
    cv::bitwise_not(res4, res4);

//Task #20
    cv::Mat M0 = cv::Mat::zeros(10, 10, CV_8U);
    cv::Mat M1 = cv::Mat::zeros(10, 10, CV_8U);
    M1 = cv::Scalar(160);
    cv::Mat mask = cv::Mat::zeros(10, 10, CV_8U);
    mask(cv::Rect(3,3,1,1)) =  255;
    M1.copyTo(M0, mask);
```

Pixel arithmetics

Simple thresholding

- Task #21
  - Load color image
  - Convert image to grayscale
  - Perform simple threshold

Pixel arithmetics

Create and prepare outputs

```
//Task #21
      cv::Mat img1 = cv::imread("lena.bmp");
      cv::cvtColor(img1, img1, CV_BGR2GRAY);
      cv::threshold(img1, img1, 100, 255, CV_THRESH_BINARY);
```

Pixel arithmetics

Homework

- Task #22
  - Load color image (dancer image provided by instructor)
  - Perform pixel operations on that image to extract only dancer (make background black)
  - Perform threshold to get binary mask
  - Create third image with some operation with usage of mask created in previous step.

The end

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