Laboratory 2

Machine Vision

Groups & points

► Create 6 groups: each consists 5 people

► Each group: one team leader

► Extra 10 points: based on overall group performance

► Laboratory: most of the lab session will be performed in c++, python is also accepted.

► Tasks and reports: each group must submit the given tasks and reports on time (team leader will take responsibility)

► Projects and reports:

1st project deadline: 22nd of December 2020

2nd project deadline: 31st of January 2021

OpenCV hello world
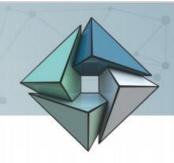## Load and display an image

- Task #1
    - Load image from filesystem
    - Create OpenCV named window
    - Show an image
- Task #2
    - Test different image load options
    - Display image in different window types
    - Perform memory cleanup

```cpp
//TASK #1
        cv::Mat img = cv::imread("lena.bmp",0);   //load image
        cv::namedWindow("Test"); //create window
        cv::imshow("Test", img); // show image
        cv::waitKey(0); //wait for user response


//TASK #2
        cv::Mat img = cv::imread("lena.bmp",1);

//Why memory cleanup is important - 1000 images loaded into RAM memory
        std::vector<cv::Mat> m;
        for (int i = 0;i<1000;i++)
        {
                m.push_back(img.clone());
        }
// release memory
        for (int i = 0;i<1000;i++)
        {
                m[i].release();
        }

        cv::Mat as a smart pointer example
        memory(img);
// what happened?
```

cv::Mat
cv::Mat as a pointer

- Task #3
    - Load an image
    - Create empty cv::Mat container
    - Assign ROI (region of interest) of loaded image to that container
    - Perform exemplary action on selected ROI - what happened to original image?
- Task #4
    - Perform Task #3 - use image clone - what happened to original image
- Task #5
    - Load image
    - Create second image with size equal to image size
    - Define 4 ROI's as a four quarters of image.
    - Compose four images into single mozaic image (full deep copy).

```cpp
//TASK #3
        cv :: Mat img = cv :: imread ("lena.bmp",1);
        cv :: Rect r(20, 20, 200, 300); //rectangle definition
        cv :: Mat tmpImg = img(r);
        tmpImg = 0.5*tmpImg; // lower image brightness
        cv :: namedWindow("Test", CV_WINDOW_NORMAL);

        cv :: imshow("Test", img);
        cv :: waitKey(0);

//TASK #4
        cv :: Mat img = cv :: imread ("lena.bmp",1);
        cv :: Rect r(20, 20, 200, 300);
        cv :: Mat tmpImg = img(r).clone (); //or with img.copyTo(tmpImg);
        tmpImg = 0.5*tmpImg;
        cv :: namedWindow("Test", CV_WINDOW_NORMAL);
        cv :: imshow("Test", img);
        cv :: waitKey(0);

//TASK #5
        cv :: Mat img = cv :: imread ("lena.bmp", 1);
        cv :: Mat tmpImg = cv :: Mat(img.size (), img.type ());

        cv :: Rect LU(0, 0, img.cols / 2, img.rows / 2);
        cv :: Rect RU(img.cols / 2, 0, img.cols / 2, img.rows / 2);
        cv :: Rect LD(0, img.rows / 2, img.cols / 2, img.rows / 2);
        cv :: Rect RD(img.cols / 2, img.rows / 2, img.cols / 2, img.rows / 2);

        img(LU).copyTo(tmpImg(RU));
        img(RU).copyTo(tmpImg(LD));
        img(LD).copyTo(tmpImg(RD));
        img(RD).copyTo(tmpImg(LU));
```

cv::Mat
Access pixel values

- Task #6
    - Load color image
    - Change pixel values to red ((0,0,255) - OpenCV uses BGR color space) inside 10x10 rectangle at image center
    - Use cv::Mat::at<> method
    - What does 'datatype' means in cv::Mat::at<'datatype'> method?

```cpp
// cv::Mat different data types

        cv::Mat ucharMat(10, 10, CV_8U);
        ucharMat.at<unsigned char>(5/*row number*/, 5/*column number*/) = 50;
        auto x = ucharMat.type();

        cv::Mat charMat(10, 10, CV_8S);
        charMat.at<char>(5/*row number*/, 5/*column number*/) = 50;
        x = charMat.type();

        cv::Mat floatMat(10, 10, CV_32F);
        floatMat.at<float>(5/*row number*/, 5/*column number*/) = 50;
        x = floatMat.type();

        cv::Mat doubleMat(10, 10, CV_64F);
        doubleMat.at<double>(5/*row number*/, 5/*column number*/) = 50;
        x = doubleMat.type();

        cv::Mat ucharMat3CH(10, 10, CV_8UC3);
        ucharMat3CH.at<cv::Vec3b>(5, 5) = cv::Vec3b( 50,100,20);
        x = ucharMat3CH.type();


//Task #6
        cv::Mat img = cv::imread("lena.bmp", 1);
        for(int i=img.cols/2-5;i<img.cols/2+5;i++)
                for(int j=img.rows/2-5;j<img.rows/2+5;j++)
                        img.at<cv::Vec3b>(j, i) = { 0,0,255 };
```

```cpp
//Task #6
    cv::Mat img = cv::imread("lena.bmp", 1);
    for(int i=img.cols/2-5;i<img.cols/2+5;i++)
        for(int j=img.rows/2-5;j<img.rows/2+5;j++)
            img.at<cv::Vec3b>(j, i) = { 0,0,255 };
```

```cpp
Mat image = img;
for(int y=0;y<img.rows;y++)
{
    for(int x=0;x<img.cols;x++)
    {
        // get pixel
        Vec3b & color = image.at<Vec3b>(y,x);

        // ... do something to the color ....
        color[0] = 13;
        color[1] = 13;
        color[2] = 13;

        // set pixel
        //image.at<Vec3b>(Point(x,y)) = color;
        //if you copy value
    }
}
```

cv::Mat
Homework

- Task #7
  - Perform Task #6 with usage of cv::Ptr
- Task #8
  - Perform Task #6 with usage of cv::Mat::data
- Discussion
  - What are advantages and disadvantages of each method?

- Task #9
  - Load color image
  - Perform color inversion of upper half of image and store it in image copy
  - Perform color inversion of left half of image and store it in image copy
  - Create third image, which has pixel values equal to mean of two previous images