OpenCV – Multiple Inheritance and Polymorphism

Reading materials

- Class - Inheritance
- Class - Polymorphism

Exercise 1: Multiple inheritance and image processing

The goal of this practical work is to create 2 classes: ***EdgeDetector*** and ***BlurFiltering*** which inherit from the OpenCV class ***Mat*** and from a common class ***Base***.

The ***class Base*** is a base class which will contain data and methods common for these two classes.

The ***class EdgeDetector*** will implement various image processing methods for edge detections defined in OpenCV (you have to list and define the methods you want to implement in this class).

The ***class BlurFiltering*** will implement various filtering methods for image blurring defined in OpenCV (you have to list and define the methods you want to implement in this class).

Your code must be able to execute the following main function:

```
int main ( void )
{
  // read an image in source
  cv::Mat source = imread("test.png", CV_LOAD_IMAGE_UNCHANGED);

  cosi::BlurFiltering blurFiltering;
  cosi::EdgeDetector edgeDetector ( source, SOBEL );

  // display edgeDetector result
  namedWindow( edgeDetector.getOpenCVProcessName(), CV_WINDOW_AUTOSIZE);
  imshow( edgeDetector.getOpenCVProcessName(), edgeDetector);

  blurFiltering.setProcess ( GAUSSIAN );
  blurFiltering.process ( source );

  // display blurFiltering result
  namedWindow( blurFiltering.getOpenCVProcessName(), CV_WINDOW_AUTOSIZE);
  imshow(blurFiltering.getOpenCVProcessName(), blurFiltering);

  edgeDetector.setProcess ( LAPLACIAN );
  edgeDetector.process ( source );
  // display edgeDetector result
  namedWindow( edgeDetector.getOpenCVProcessName(), CV_WINDOW_AUTOSIZE);
  imshow( edgeDetector.getOpenCVProcessName(), edgeDetector);
```

```
  waitKey(0); //wait infinite time for a keypress

  return 0;
}
```

Exercise 2: Virtual members

Transform your code in order to execute the following code:

```
void globalProcess ( base& imageProcessing, const cv::Mat& imageToProcess )
{
   imageProcessing.process ( imageToProcess );
}

int main ( void )
{
  // read an image in source
  cv::Mat source = imread("test.png", CV_LOAD_IMAGE_UNCHANGED);

  cosi::BlurFiltering blurFiltering;
  blurFiltering.setProcess ( GAUSSIAN );

  globalProcess ( blurFiltering, source );

  // display blurFiltering result
  namedWindow( blurFiltering.getOpenCVProcessName(), CV_WINDOW_AUTOSIZE);
  imshow(blurFiltering.getOpenCVProcessName(), blurFiltering);

  waitKey(0); //wait infinite time for a keypress

  return 0;
}
```

Exercise 3: Abstract base classes

Make the ***class Base*** an abstract class.