/\*

\* To change this template, choose Tools | Templates

\* and open the template in the editor.

\*/

package com.helper;

import java.awt.image.BufferedImage;

import java.io.File;

import java.io.IOException;

import org.bytedeco.javacv.Frame;

import org.bytedeco.javacv.Java2DFrameConverter;

import org.bytedeco.javacv.OpenCVFrameConverter;

import static org.bytedeco.javacpp.opencv\_core.\*;

import static org.bytedeco.javacpp.opencv\_imgcodecs.\*;

import static org.bytedeco.javacpp.opencv\_highgui.\*;

import static org.bytedeco.javacpp.opencv\_core.\*;

import static org.bytedeco.javacpp.opencv\_imgcodecs.\*;

import static org.bytedeco.javacpp.opencv\_imgproc.\*;

/\*\*

\*

\* @author

\*/

public class OpenCVHelper {

public static Frame mat2frame(Mat img1) {

OpenCVFrameConverter.ToMat converter = new OpenCVFrameConverter.ToMat();

Frame img1f = converter.convert(img1);

return img1f;

}

// Image has to be a color image

public static void drawPolyLine(IplImage MyImage, int[] cordinatesXY) {

// cvPolyLine(MyImage, pts, new int[]{pts.length/2}, 1, 1, CV\_RGB(255,

// 255, 255), 5, CV\_AA, 8);

cvPolyLine(MyImage, cordinatesXY, new int[] { cordinatesXY.length / 2 }, 1, 1, CV\_RGB(255, 255, 255));

}

public static IplImage mat2ipl(Mat img1) {

OpenCVFrameConverter.ToMat converterMat = new OpenCVFrameConverter.ToMat();

OpenCVFrameConverter.ToIplImage converter = new OpenCVFrameConverter.ToIplImage();

Frame img1f = converterMat.convert(img1);

IplImage ipl = converter.convert(img1f);

return ipl;

}

public static Mat file2mat(String fileName) {

try {

File f = new File(fileName);

System.out.println("Image Path " + f.getCanonicalPath());

Mat m = imread(fileName);

return m;

} catch (IOException ex) {

ex.printStackTrace();

}

return null;

}

public static Mat ipl2mat(IplImage ipl) {

try {

OpenCVFrameConverter.ToIplImage converter = new OpenCVFrameConverter.ToIplImage();

Frame frame = converter.convert(ipl);

Mat m = converter.convertToMat(frame);

return m;

} catch (Exception ex) {

ex.printStackTrace();

}

return null;

}

public static Frame ipl2frame(IplImage ipl) {

try {

OpenCVFrameConverter.ToIplImage converter = new OpenCVFrameConverter.ToIplImage();

Frame m = converter.convert(ipl);

return m;

} catch (Exception ex) {

ex.printStackTrace();

}

return null;

}

public static Mat frame2mat(Frame frame) {

try {

OpenCVFrameConverter.ToIplImage converter = new OpenCVFrameConverter.ToIplImage();

Mat m = converter.convertToMat(frame);

return m;

} catch (Exception ex) {

ex.printStackTrace();

}

return null;

}

public static void frame2File(Frame frame, String fileName) {

try {

cvSaveImage(fileName, frame2ipl(frame));

} catch (Exception ex) {

ex.printStackTrace();

}

}

public static IplImage frame2ipl(Frame frame) {

try {

OpenCVFrameConverter.ToIplImage converter = new OpenCVFrameConverter.ToIplImage();

IplImage ipl = converter.convert(frame);

return ipl;

} catch (Exception ex) {

ex.printStackTrace();

}

return null;

}

public static IplImage file2ipl(String fileName) {

try {

File f = new File(fileName);

System.out.println("Image Path " + f.getCanonicalPath());

Mat m = imread(fileName);

IplImage i = mat2ipl(m);

// m.release();

// System.out.println(i);

return i;

} catch (IOException ex) {

ex.printStackTrace();

}

return null;

}

public static BufferedImage ipl2buffered(IplImage bi) {

try {

Java2DFrameConverter j2d = new Java2DFrameConverter();

BufferedImage f = j2d.convert(ipl2frame(bi));

return f;

} catch (Exception ex) {

ex.printStackTrace();

}

return null;

}

public static IplImage buffered2ipl(BufferedImage bi) {

try {

Java2DFrameConverter j2d = new Java2DFrameConverter();

Frame f = j2d.convert(bi);

return frame2ipl(f);

} catch (Exception ex) {

ex.printStackTrace();

}

return null;

}

public static BufferedImage frame2buffered(Frame f) {

try {

Java2DFrameConverter j2d = new Java2DFrameConverter();

BufferedImage bi = j2d.convert(f);

return bi;

} catch (Exception ex) {

ex.printStackTrace();

}

return null;

}

public static Mat buffered2mat(BufferedImage bi) {

try {

Java2DFrameConverter j2d = new Java2DFrameConverter();

Frame f = j2d.convert(bi);

return frame2mat(f);

} catch (Exception ex) {

ex.printStackTrace();

}

return null;

}

public static IplImage resizeImage(IplImage inputImage, int resizeWidth, int resizeHeight) {

IplImage ColorImage2 = cvCreateImage(new CvSize(100, 80), IPL\_DEPTH\_8U, 3);

cvResize(inputImage, ColorImage2, CV\_INTER\_NN);

// imwrite(System.currentTimeMillis() + ".jpg",

// OpenCVHelper.ipl2mat(ColorImage2));

return ColorImage2;

}

public static void writeImage(IplImage inputImage) {

imwrite(System.currentTimeMillis() + ".jpg", OpenCVHelper.ipl2mat(inputImage));

}

public static void writeImage(IplImage inputImage, String name) {

imwrite(name + "\_" + System.currentTimeMillis() + ".jpg", OpenCVHelper.ipl2mat(inputImage));

}

public static void writeImage(Mat inputImageMat) {

imwrite(System.currentTimeMillis() + ".jpg", inputImageMat);

}

}