package org.bytedeco.javacv.android.recognize.technowings;

import com.constant.ServerConstants;

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

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

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

import org.bytedeco.javacpp.indexer.FloatRawIndexer;

import org.bytedeco.javacpp.indexer.Indexer;

import org.bytedeco.javacpp.indexer.UByteRawIndexer;

import java.util.ArrayList;

import org.bytedeco.javacpp.opencv\_core.Mat;

public class CommonFunctions {

public static void main(String[] args) {

float f = 12f;

double m[][] = new double[][] { new double[] { f, f + 1 }, new double[] { f, f }, new double[] { f, f } };

Mat m1 = new Mat(new float[] { f, f + 1, f + 2, f + 4 });

Mat M = new Mat(3, 1, CV\_32FC2);

// m1=m1.reshape(-1, 2);

FloatRawIndexer imagePixels2 = M.createIndexer();

imagePixels2.put(0, 0, 0, f + 1);

imagePixels2.put(0, 0, 1, f + 1);

imagePixels2.put(1, 0, 0, f + 2);

imagePixels2.put(1, 0, 1, f + 3);

imagePixels2.put(2, 0, 0, f + 4);

imagePixels2.put(2, 0, 1, f + 5);

// org.bytedeco.javacpp.opencv\_core$Mat[width=1,height=11,depth=32,channels=2]

ArrayList<Float[]> arr = new CommonFunctions().getImageData(M);

System.out.println("hi " + arr.size());

}

public Mat createMat(ArrayList<ArrayList<Float[]>> arr) {

Mat M = new Mat(arr.size() , 1, CV\_32FC2);

FloatRawIndexer imagePixels2 = M.createIndexer();

for (int i = 0; i < arr.size(); i++) {

ArrayList<Float[]> a = arr.get(i);

Float[] data=a.get(a.size()-1);

for (int j = 0; j < data.length; j++) {

imagePixels2.put(i, 0, j, data[j].floatValue());

}

}

return M;

}

public ArrayList getImageData(Mat mat) {

// int oldValue = 127;

// int newValue = 0;

int channels = mat.channels();

// System.out.println("mat " + mat+" mat.depth() "+mat.depth());

Indexer indexer = null;

if (mat.arrayDepth() > 8) {

ArrayList<Float[]> arr = new ArrayList<Float[]>();

FloatRawIndexer imagePixels2 = mat.createIndexer();

// UByteRawIndexer imagePixels2 = mat.createIndexer();

// System.out.println();

for (int col = 0; col < mat.cols(); col++) {

for (int row = 0; row < mat.rows(); row++) {

Float[] f = new Float[channels];

for (int i = 0; i < channels; i++) {

float pix = imagePixels2.get(row, col, i);

f[i] = pix;

// System.out.print(pix + ",");

}

arr.add(f);

}

// System.out.println();

}

return arr;

} else {

ArrayList<Integer> arr = new ArrayList<Integer>();

UByteRawIndexer imagePixels2 = mat.createIndexer();

// UByteRawIndexer imagePixels2 = mat.createIndexer();

// System.out.println();

for (int col = 0; col < mat.cols(); col++) {

for (int row = 0; row < mat.rows(); row++) {

for (int i = 0; i < channels; i++) {

int pix = imagePixels2.get(row, col, i);

arr.add(pix);

// System.out.print(pix + ",");

}

}

// System.out.println();

}

return arr;

}

}

public void printImagePixels(Mat mat) {

// int oldValue = 127;

// int newValue = 0;

FloatRawIndexer imagePixels2 = mat.createIndexer();

// UByteRawIndexer imagePixels2 = mat.createIndexer();

// System.out.println();

for (int col = 0; col < mat.cols(); col++) {

for (int row = 0; row < mat.rows(); row++) {

float pix = imagePixels2.get(row, col, 2);

System.out.print(pix + ",");

}

System.out.println();

}

}

public static void log(String message) {

System.out.println(message);

}

public void showWindow(String name, Mat mat) {

imshow(name, mat);

}

public void drawEnterExitLine(Mat newMat) {

Point start = new Point(ServerConstants.enterTop.x(), ServerConstants.enterTop.y() + 20);

putText(newMat, "Enter(" + ServerConstants.enterTop.x() + "," + ServerConstants.enterTop.y() + ")", start, FONT\_HERSHEY\_PLAIN, 1,

new Scalar(0, 255, 0, 0));

// putText(newMat, "Exit (" + ServerConstants.exitBottom.x() + "," +

// ServerConstants.exitBottom.y() + ")", ServerConstants.exitBottom,

// FONT\_HERSHEY\_PLAIN, 1, new opencv\_core.Scalar(0, 255, 0, 0));

// Exit Line

line(newMat, ServerConstants.enterTop, ServerConstants.enterBotton, new Scalar(0, 0, 255, 0));

// Entry Line

// line(newMat, ServerConstants.exitTop, ServerConstants.exitBottom, new

// opencv\_core.Scalar(0, 0, 255, 0));

}

public void highlightEnter(Mat newMat) {

line(newMat, ServerConstants.enterTop, ServerConstants.enterBotton, new Scalar(0, 255, 0, 0));

}

public void highlightExit(Mat newMat) {

line(newMat, ServerConstants.exitTop, ServerConstants.exitBottom, new Scalar(0, 255, 0, 0));

}

}