Learn OpenCV by Examples

OpenCV simplified for beginners by the use of examples. Learn OpenCV with basic implementation of different algorithms.

Sobel Edge Detection

void **Sobel**(InputArray src, OutputArray dst, int ddepth, int dx, int dy, int ksize=3, double scale=1, double delta=0, int borderType=BORDER_DEFAULT)

Parameters:

- **src** – input image.
- **dst** – output image of the same size and the same number of channels as `src` .
- **ddepth** –

  output image depth; the following combinations of `src.depth()` and `ddepth` are supported:

  - `src.depth()` = `CV_8U`, `ddepth` = -1/`CV_16S`/`CV_32F`/`CV_64F`
  - `src.depth()` = `CV_16U`/`CV_16S`, `ddepth` = -1/`CV_32F`/`CV_64F`
  - `src.depth()` = `CV_32F`, `ddepth` = -1/`CV_32F`/`CV_64F`
  - `src.depth()` = `CV_64F`, `ddepth` = -1/`CV_64F`

  when `ddepth=-1`, the destination image will have the same depth as the source; in the case of 8-bit input images it will result in truncated derivatives.
- **xorder** – order of the derivative x.
- **yorder** – order of the derivative y.

SEARCH CONTENTS OF THIS BLOG

Search

POPULAR POSTS

Find Contour

Basic drawing examples

Line Detection by Hough Line Transform

Face Detection using Haar-Cascade Classifier

Perspective Transform

Sobel Edge Detection

- **ksize** – size of the extended Sobel kernel; it must be 1, 3, 5, or 7.
- **scale** – optional scale factor for the computed derivative values; by default, no scaling is applied (see `getDerivKernels()` for details).
- **delta** – optional delta value that is added to the results prior to storing them in `dst`.
- **borderType** – pixel extrapolation method (see `borderInterpolate()` for details).

Functions:
Sobel, Vec3b, convertTo, imshow, imread, namedWindow, waitKey.

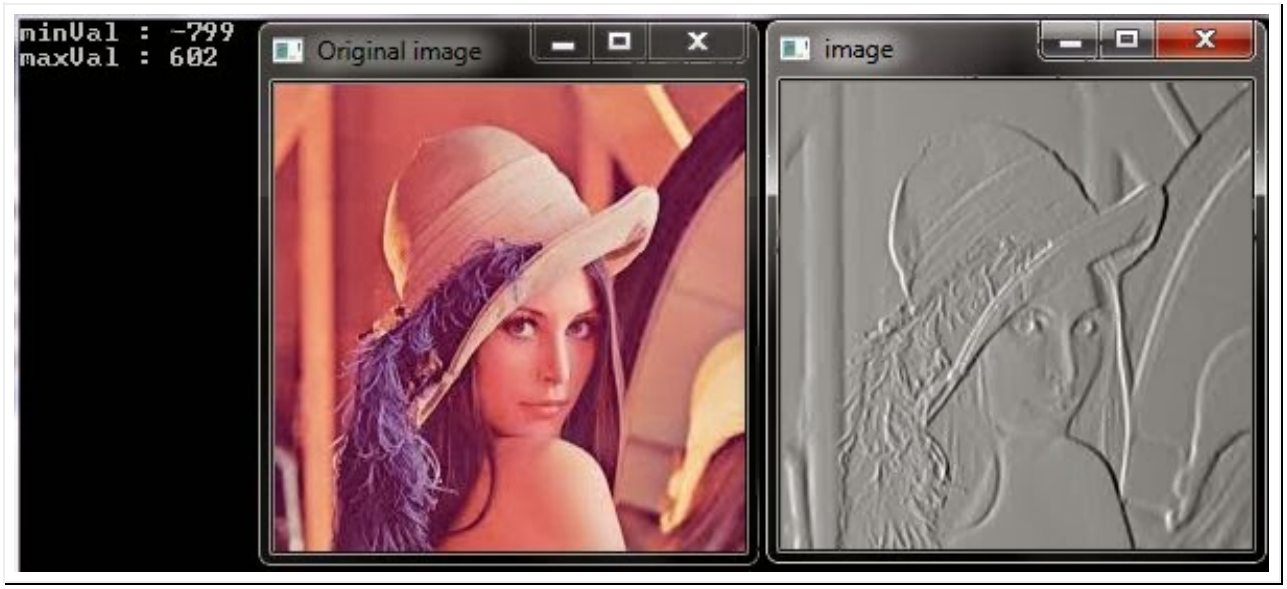
This is a code from OpenCV documentation. I have made some changes to it.

Example:

```
#include "opencv2/core/core.hpp"
#include "opencv2/highgui/highgui.hpp"
#include "opencv2/imgproc/imgproc.hpp"
#include "iostream"

using namespace cv;
using namespace std;

int main( )
{
    Mat src1;
    src1 = imread("lena.jpg", CV_LOAD_IMAGE_COLOR);
    namedWindow( "Original image", CV_WINDOW_AUTOSIZE );
    imshow( "Original image", src1 );

    Mat grey;
    cvtColor(src1, grey, CV_BGR2GRAY);

    Mat sobelx;
    Sobel(grey, sobelx, CV_32F, 1, 0);

    double minVal, maxVal;
    minMaxLoc(sobelx, &minVal, &maxVal); //find minimum and maximum
    cout << "minVal : " << minVal << endl << "maxVal : " << maxVal

    Mat draw;
    sobelx.convertTo(draw, CV_8U, 255.0/(maxVal - minVal), -minVal

    namedWindow("image", CV_WINDOW_AUTOSIZE);
    imshow("image", draw);

    waitKey(0);
    return 0;
}
```


CATEGORIES

Result:



Labels: Edge Detection

No comments:

Post a Comment