

Отчет о прохождении практики по теме:

Программная реализация матричных фильтров
изображений с помощью свободной библиотеки
компьютерного зрения OpenCV.

Студент 1 курса _____ Чарзавакян М.Э.

Руководитель практики
доцент _____ Яковлев К.С.

Москва 2018г.

Оглавление

1	Введение	2
2	Теоретическая часть	3
2.1	Описание операции свертки	3
2.2	Математическая формула	4
3	Реализация	5
3.1	Пример программы	5
3.2	Примеры использования фильтров	6
4	Заключение	7
5	Список литературы	8

1. Введение

Цель практики:

1. Изучить библиотеку *OpenCV*.
2. Изучить технологию обработки фотографий с помощью матричных фильтров.
3. Реализовать матричные фильтры изображения.

OpenCV — библиотека алгоритмов компьютерного зрения, обработки изображений и численных алгоритмов общего назначения с открытым кодом.

2. Теоретическая часть

Множество фильтров таких как: размытие, повышение резкости, нахождение краев и множество других основаны на технологии "свертка". В данной практике именно рассматривается этот метод.

2.1 Описание операции свертки

Свертка (англ. convolution) — это операция, показывающая «схожесть» одной функции с отражённой и сдвинутой копией другой.

Свертка — это операция вычисления нового значения выбранного пикселя, учитывающая значения окружающих его пикселей. Для вычисления значения используется матрица, называемая ядром свертки. Обычно ядро свертки является квадратной матрицей $n \cdot n$, где n — нечетное. Во время вычисления нового значения выбранного пикселя ядро свертки как бы «прикладывается» своим центром к данному пикселю. Для пикселя вычисляется сумма его соседних пикселей умноженных на соответствующий коэффициент в матрицы - ядра свертки. Полученное число - новое значение пикселя.

Мы работали с черно белыми изображениями - матрицы состоящие из чисел от 0 до 255, однако операции свертки можно также применять и для цветных изображений.

При изменении значения пикселя в фотографии встает вопрос: что делать с границами. Моя реализация фильтров оставляет без изменения значения этих пикселей.

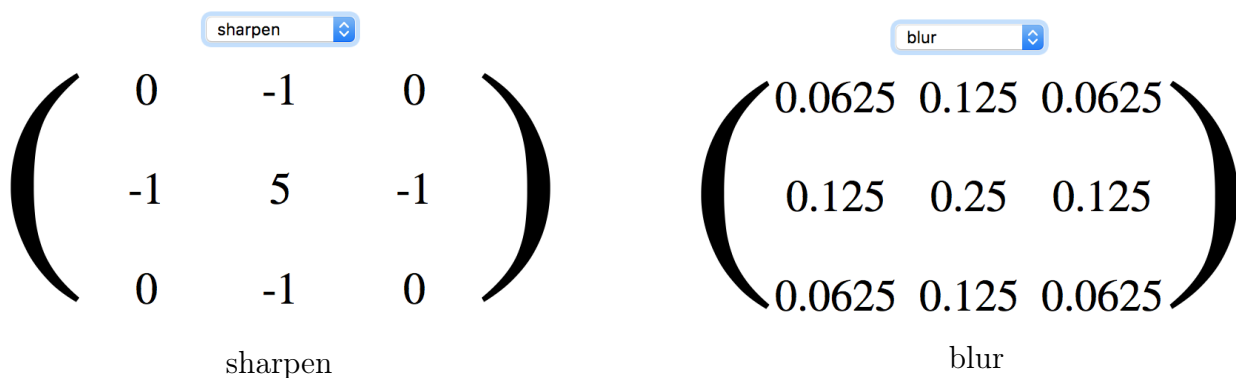

$$\begin{array}{c} \text{sharpen} \\ \begin{pmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{pmatrix} \\ \text{sharpen} \end{array} \qquad \begin{array}{c} \text{blur} \\ \begin{pmatrix} 0.0625 & 0.125 & 0.0625 \\ 0.125 & 0.25 & 0.125 \\ 0.0625 & 0.125 & 0.0625 \end{pmatrix} \\ \text{blur} \end{array}$$

Рис. 2.1: Примеры матриц

2.2 Математическая формула

Формула для вычисления нового значения пикселя:

$$image[x][y] = \sum_{i=0}^n \sum_{j=0}^n kernel[i][j] \cdot image \left[x - \frac{n}{2} + i \right] \left[y - \frac{n}{2} + j \right]$$

n - размер матрицы свертки

x и y - координаты пикселя изображения

3. Реализация

3.1 Пример программы

Данная функция принимает на вход путь к изображению и ядро свертки и ничего не возвращает. При работе данной функции в консоль выводиться изображение к которому применили данный фильтр.

```
void applyFilter(const string& fileName, const Mat& kernel) {
    Mat image = imread(fileName, 0);
    Mat newImage = image.clone();
    int band = kernel.cols / 2;
    for (int i = band; i < image.rows - band; ++i)
        for (int j = band; j < image.cols - band; ++j) {
            float newVal = 0;
            for (int g = 0; g < kernel.rows; ++g)
                for (int k = 0; k < kernel.cols; ++k) {
                    newVal += kernel.at<float>(g, k) *
                        image.at<uchar>(i + g - band, j + k - band);
                }
            if (newVal < 0)
                newImage.at<uchar>(i, j) = 0;
            else if (newVal > 255)
                newImage.at<uchar>(i, j) = 255;
            else
                newImage.at<uchar>(i, j) = newVal;
        }
    namedWindow("My image", WINDOW_AUTOSIZE);
    imshow("My image", newImage);
    waitKey(0);
}
```

3.2 Примеры использования фильтров

Применим несколько фильтров к изображению при помощи написанной программы.



original



blur



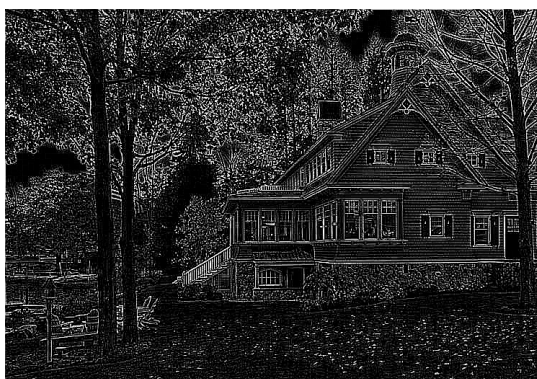
original



sharpen



original



edge detection

4. Заключение

В результате работы я ознакомился с библиотекой *OpenCV* и её функциями позволяющими загружать и обрабатывать изображения. Также я научился применять фильтры к изображениям методом свертки. Написал программу, реализующую матричные фильтры с произвольным ядром.

5. Список литературы

1. <http://setosa.io/ev/image-kernels/>
2. <http://aishack.in/tutorials/image-convolution-examples/>
3. <https://habr.com/post/62738/>