**Convolution Matrix**

[Emboss - Рельеф](http://developer.alexanderklimov.ru/android/catshop/convolution.php#emboss)  
[Gaussian Blur - Размытие по Гауссу](http://developer.alexanderklimov.ru/android/catshop/convolution.php#gauss)  
[Engraving Effect - Гравировка](http://developer.alexanderklimov.ru/android/catshop/convolution.php#engraving)  
[Sharpen - Резкость](http://developer.alexanderklimov.ru/android/catshop/convolution.php#sharpen)  
[Mean Removal Effect](http://developer.alexanderklimov.ru/android/catshop/convolution.php#meanremoval)  
[Smooth - Сглаживание](http://developer.alexanderklimov.ru/android/catshop/convolution.php#smooth)

Для различных стандартных эффектов (размытие по Гауссу, рельеф, гравировка) используются специальные функции, известные под именем Convolution. Подробнее можно узнать по ссылкам:

[Convolution on Wikipedia](http://en.wikipedia.org/wiki/Convolution)

[GIMP Documentation on Convolution Matrix](http://docs.gimp.org/en/plug-in-convmatrix.html)

[Convolution Study by Ahn Song Ho](http://www.songho.ca/dsp/convolution/convolution.html)

[VcsKicks.Com – Convolution Application on Box Blur effect](http://www.vcskicks.com/box-blur.php)

[CodeProject – Image Processing for Dummies (Part II – Convolution Filters)](http://www.codeproject.com/KB/GDI-plus/csharpfilters.aspx)

[RoboRealms – Short article on Convolution](http://www.roborealm.com/help/Convolution.php#1098)

Нас интересует практическая часть. Создадим новый класс, с помощью которого мы сможем создавать новые эффекты:

package ru.alexanderklimov.flip\_1098;

import android.graphics.Bitmap;

import android.graphics.Color;

public class ConvolutionMatrix {

public static final int SIZE = 3;

public double[][] Matrix;

public double Factor = 1;

public double Offset = 1;

public ConvolutionMatrix(int size) {

Matrix = new double[size][size];

}

public void setAll(double value) {

for (int x = 0; x < SIZE; ++x) {

for (int y = 0; y < SIZE; ++y) {

Matrix[x][y] = value;

}

}

}

public void applyConfig(double[][] config) {

for(int x = 0; x < SIZE; ++x) {

for(int y = 0; y < SIZE; ++y) {

Matrix[x][y] = config[x][y];

}

}

}

public static Bitmap computeConvolution3x3(Bitmap src, ConvolutionMatrix matrix) {

int width = src.getWidth();

int height = src.getHeight();

Bitmap result = Bitmap.createBitmap(width, height, src.getConfig());

int A, R, G, B;

int sumR, sumG, sumB;

int[][] pixels = new int[SIZE][SIZE];

for(int y = 0; y < height - 2; ++y) {

for(int x = 0; x < width - 2; ++x) {

// get pixel matrix

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

for(int j = 0; j < SIZE; ++j) {

pixels[i][j] = src.getPixel(x + i, y + j);

}

}

// get alpha of center pixel

A = Color.alpha(pixels[1][1]);

// init color sum

sumR = sumG = sumB = 0;

// get sum of RGB on matrix

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

for(int j = 0; j < SIZE; ++j) {

sumR += (Color.red(pixels[i][j]) \* matrix.Matrix[i][j]);

sumG += (Color.green(pixels[i][j]) \* matrix.Matrix[i][j]);

sumB += (Color.blue(pixels[i][j]) \* matrix.Matrix[i][j]);

}

}

// get final Red

R = (int)(sumR / matrix.Factor + matrix.Offset);

if(R < 0) { R = 0; }

else if(R > 255) { R = 255; }

// get final Green

G = (int)(sumG / matrix.Factor + matrix.Offset);

if(G < 0) { G = 0; }

else if(G > 255) { G = 255; }

// get final Blue

B = (int)(sumB / matrix.Factor + matrix.Offset);

if(B < 0) { B = 0; }

else if(B > 255) { B = 255; }

// apply new pixel

result.setPixel(x + 1, y + 1, Color.argb(A, R, G, B));

}

}

// final image

return result;

}

}

Теперь мы можем обращаться к созданному классу и реализовать новые эффекты. Все эффекты ресурсоемкие и требуют много времени для выполнения.

**Рельеф**

Для создания эффекта Emboss (Рельеф) нужно задать определенные параметры в матрице:

public static Bitmap emboss(Bitmap src) {

double[][] EmbossConfig = new double[][] {

{ -1 , 0, -1 },

{ 0 , 4, 0 },

{ -1 , 0, -1 }

};

ConvolutionMatrix convMatrix = new ConvolutionMatrix(3);

convMatrix.applyConfig(EmbossConfig);

convMatrix.Factor = 1;

convMatrix.Offset = 127;

return ConvolutionMatrix.computeConvolution3x3(src, convMatrix);

}



**Gaussian Blur - Размытие по Гауссу**

Размытие по Гауссу достигается следующим кодом:

public static Bitmap applyGaussianBlur(Bitmap src) {

double[][] GaussianBlurConfig = new double[][] {

{ 1, 2, 1 },

{ 2, 4, 2 },

{ 1, 2, 1 }

};

ConvolutionMatrix convMatrix = new ConvolutionMatrix(3);

convMatrix.applyConfig(GaussianBlurConfig);

convMatrix.Factor = 16;

convMatrix.Offset = 0;

return ConvolutionMatrix.computeConvolution3x3(src, convMatrix);

}



**Engraving Effect - Гравировка**

Код для эффекта гравировки:

public static Bitmap engrave(Bitmap src) {

ConvolutionMatrix convMatrix = new ConvolutionMatrix(3);

convMatrix.setAll(0);

convMatrix.Matrix[0][0] = -2;

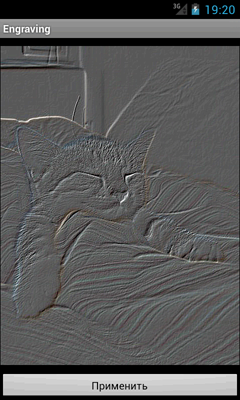
convMatrix.Matrix[1][1] = 2;

convMatrix.Factor = 1;

convMatrix.Offset = 95;

return ConvolutionMatrix.computeConvolution3x3(src, convMatrix);

}



**Sharpen - Резкость**

Для создания резкости создадим следующий метод:

public static Bitmap sharpen(Bitmap src, double weight) {

double[][] SharpConfig = new double[][] { { 0, -2, 0 },

{ -2, weight, -2 }, { 0, -2, 0 } };

ConvolutionMatrix convMatrix = new ConvolutionMatrix(3);

convMatrix.applyConfig(SharpConfig);

convMatrix.Factor = weight - 8;

return ConvolutionMatrix.computeConvolution3x3(src, convMatrix);

}

В моём примере использовался weight = 12.

**Mean Removal Effect**

Mean Removal Effect - это другой способ резкости.

public static Bitmap applyMeanRemoval(Bitmap src) {

double[][] MeanRemovalConfig = new double[][] { { -1, -1, -1 },

{ -1, 9, -1 }, { -1, -1, -1 } };

ConvolutionMatrix convMatrix = new ConvolutionMatrix(3);

convMatrix.applyConfig(MeanRemovalConfig);

convMatrix.Factor = 1;

convMatrix.Offset = 0;

return ConvolutionMatrix.computeConvolution3x3(src, convMatrix);

}



Этот же эффект на C# - [Image Processing in C#: Applying a mean removal - Smoky Cogs](http://www.smokycogs.com/blog/image-processing-in-c-sharp-applying-a-mean-removal/)

**Smooth - Сглаживание**

Эффект сглаживания. На используемой картинке эффект трудно заметить, поэтому скриншота не будет. Попробуйте использовать другие картинки для опытов:

public static Bitmap smooth(Bitmap src, double value) {

ConvolutionMatrix convMatrix = new ConvolutionMatrix(3);

convMatrix.setAll(1);

convMatrix.Matrix[1][1] = value;

convMatrix.Factor = value + 8;

convMatrix.Offset = 1;

return ConvolutionMatrix.computeConvolution3x3(src, convMatrix);

}

**Использованные материалы**

[Image Processing – Convolution Matrix](http://xjaphx.wordpress.com/2011/06/22/image-processing-convolution-matrix/)

[Image Processing – Emboss Effect](http://xjaphx.wordpress.com/2011/06/22/image-processing-emboss-effect/)

[Image Processing – Gaussian Blur](http://xjaphx.wordpress.com/2011/06/22/image-processing-gaussian-blur/)

[Image Processing – Engraving Effect « [ Android Newbie ]](http://xjaphx.wordpress.com/2011/06/22/image-processing-engraving-effect/)

[Image Processing – Sharpening Image « [ Android Newbie ]](http://xjaphx.wordpress.com/2011/06/22/image-processing-sharpening-image/)

[Image Processing – Mean Removal Effect | [ Android Newbie ]](http://xjaphx.wordpress.com/2011/06/22/image-processing-mean-removal-effect/)

[Image Processing – Smooth Effect | [ Android Newbie ]](http://xjaphx.wordpress.com/2011/06/22/image-processing-smooth-effect/#1098)