Image Local Polynomial Approximation (LPA) and its Applications.

(Draft, March 2011)

Guennadi Levkine (email: hlevkin@gmail.com)
Vancouver, Canada.

Abstract.

The original treatment of image local polynomial approximation in blocks 3x3, 4x4, 5x5 and 7x7 is proposed. An intention to demonstrate using of an alternate approach for different problem solving. Many new convolution kernels are presented. Difference with Haralick facet model is the use of ordinary polynomials of two variables instead of orthogonal polynomials, in considering rounded and cutting blocks in 3D polynomial approximation.

Key words: Local Polynomial Approximation, image processing, image filtration, gradient, image resize, video filtration, image convolution, convolution mask, noise filtration, facet model

1. Introduction.

Among different approaches to image processing some of the most popular are:
- Spectral transformation of images or local parts (blocks)
- Convolutions of images with square matrices of size 3x3, 4x4, 5x5, 7x7, 8x8. These square matrices are called convolution kernels.

In the first approach it was described in many books and articles (look [1], [2], [3], [4] for example). It is based on the theory of orthogonal/unitary transformations.
For the second approach we have to build convolution kernels of small sizes 3x3, 4x4, 5x5 and so on for problem solving. Usually it is done empirically. But for development of such kernels it is possible to locally approximate image intensity by polynomials of two variables (rows and columns) and after that use these polynomial coefficients for different problem solving.
Here we give kernels for calculation of polynomial coefficients of different size and shape. We demonstrate noise filtration, and gradient estimation, Laplace operator kernels, resizing, temporal video filtration and more by the use of coefficients. The coefficients are calculated using Local Polynomial Approximation (LPA) method, which will be described here.

2. Two dimensional polynomial approximations.

In this method of LPA for every pixel we calculate 2D polynomial, which approximate image intensity in some block around the pixel. A common view for such a polynomial is

$$I(x,y) = \sum_{i,j=0}^{n} c_{ij} x^i y^j$$

For the coefficient calculation we use standard Least Square Fit method. Common block sizes are 3,4,5,7,8,9.

For n=0, n=1 , n=2, n=3, n=4 we use special notation:

$$I_0(x,y) = \alpha$$

$$I_1(x,y) = \alpha + \beta * x + \gamma * y$$

$$I_2(x,y) = \alpha + \beta * x + \gamma * y + \delta * x^2 + \varepsilon * x * y + \zeta * y^2$$

$$I_3(x,y) = \alpha + \beta * x + \gamma * y + \delta * x^2 + \varepsilon * x * y + \zeta * y^2 + \eta * x^3 + \theta * x^2 * y + \iota * x * y^2 + \kappa * y^3$$

$$I_4(x,y) = I_3(x,y) + \lambda * x^4 + \mu * x^3 * y + \nu * x^2 * y^2 + \xi * x * y^3 + o * y^4$$

In this work we will use empirical "Interpolation Level" parameter for estimation to how close our approximation to the 2D interpolation of the image. This parameter is calculated by the formula

$$IL = \frac{NumberOfCoefficients\ InPolynomial}{NumberOfPixelsInBlock}$$

If this parameter is close to 1, it means that the approximation is close to an interpolation.

3. Linear local approximation in 3x3, 4x4 and 5x5 blocks.

In this case we use simple linear approximation

$$I_1(x,y) = \alpha + \beta * x + \gamma * y$$

For 3x3 blocks estimation of α, β, γ coefficients is obvious. Interpolation level is 3/9= 1/3 and convolution kernels for their calculation are:

$$\alpha : \frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} \qquad \beta : \frac{1}{6}\begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix} \qquad \gamma : \frac{1}{6}\begin{bmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ +1 & +1 & +1 \end{bmatrix}$$

For 4x4 blocks Interpolation level is 3/16 and convolution kernels are:

$$\alpha : \frac{1}{16}\begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \end{bmatrix} \qquad \beta : \frac{1}{80}\begin{bmatrix} -3 & -1 & 1 & 3 \\ -3 & -1 & 1 & 3 \\ -3 & -1 & 1 & 3 \\ -3 & -1 & 1 & 3 \end{bmatrix}$$

$$\gamma : \frac{1}{80}\begin{bmatrix} -3 & -3 & -3 & -3 \\ -1 & -1 & -1 & -1 \\ +1 & +1 & +1 & +1 \\ +3 & +3 & +3 & +3 \end{bmatrix}$$

For 5x5 blocks Interpolation level is 3/25 and convolution kernels are:

$$\alpha: \frac{1}{25}\begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \end{bmatrix} \qquad \beta: \frac{1}{50}\begin{bmatrix} -2 & -1 & 0 & 1 & 2 \\ -2 & -1 & 0 & 1 & 2 \\ -2 & -1 & 0 & 1 & 2 \\ -2 & -1 & 0 & 1 & 2 \\ -2 & -1 & 0 & 1 & 2 \end{bmatrix}$$

$$\gamma: \frac{1}{50}\begin{bmatrix} -2 & -2 & -2 & -2 & -2 \\ -1 & -1 & -1 & -1 & -1 \\ 0 & 0 & 0 & 0 & 0 \\ +1 & +1 & +1 & +1 & +1 \\ +2 & +2 & +2 & +2 & +2 \end{bmatrix}$$

When we calculate kernels for coefficient estimation, all the pixels in the block have the same weight. But it is possible to use different pixel weights: higher for pixels close to block center. We consider here weight matrix (Rosenfeld's weight matrix)

$$\begin{bmatrix} 1 & \sqrt{2} & 1 \\ \sqrt{2} & 2 & \sqrt{2} \\ 1 & \sqrt{2} & 1 \end{bmatrix}$$

and calculate for it coefficients α, β, γ . It was found that corresponding kernels are:

$$\alpha: \frac{1}{16}\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix} \qquad \beta: \frac{1}{8}\begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} \qquad \gamma: \frac{1}{8}\begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix}$$

These kernels are well known and widely used, kernel for α is used for noise filtration, kernels for β, γ , called Sobel's kernels, and are used for gradient estimation. We named kernel α as Rosenfeld's kernel.

This approach (using pixel weight matrices) gives opportunities for creating kernels. It is convenient for different tasks in image processing. For example it will be interesting to consider the following weight matrix

$$\begin{matrix} \frac{1}{\sqrt{2}} & 1 & \sqrt{2} & 1 & \frac{1}{\sqrt{2}} \\ 1 & 2 & 2\sqrt{2} & 2 & 1 \\ \sqrt{2} & 2\sqrt{2} & 4 & 2\sqrt{2} & \sqrt{2} \\ 1 & 2 & 2\sqrt{2} & 2 & 1 \\ \frac{1}{\sqrt{2}} & 1 & \sqrt{2} & 1 & \frac{1}{\sqrt{2}} \end{matrix}$$

Center of it equals to Rosenfeld's weight matrix 3x3.

4. Quadric polynomial approximation in 3x3 block.

In this important case we use approximation

$$I_2(x,y) = \alpha + \beta * x + \gamma * y + \delta * x^2 + \varepsilon * x * y + \zeta * y^2$$

Interpolation level is 6/9 = 2/3, two times stronger then linear LPA.

Convolution kernels for $\alpha, \beta, \gamma, \delta, \varepsilon, \zeta$ coefficient calculation are:

$$\alpha : \frac{1}{9}\begin{bmatrix} -1 & 2 & -1 \\ +2 & 5 & +2 \\ -1 & 2 & -1 \end{bmatrix} \quad \beta : \frac{1}{6}\begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix}$$

$$\gamma : \frac{1}{6}\begin{bmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ +1 & +1 & +1 \end{bmatrix} \quad \delta : \frac{1}{6}\begin{bmatrix} 1 & -2 & 1 \\ 1 & -2 & 1 \\ 1 & -2 & 1 \end{bmatrix}$$

$$\varepsilon : \frac{1}{4}\begin{bmatrix} -1 & 0 & 1 \\ 0 & 0 & 0 \\ 1 & 0 & -1 \end{bmatrix} \quad \zeta : \frac{1}{6}\begin{bmatrix} +1 & +1 & +1 \\ -2 & -2 & -2 \\ +1 & +1 & +1 \end{bmatrix}$$

This LPA saves mush more information about the image when compared with linear LPA. First of all we see non-standard convolution kernel for coefficient α . It has negative elements. Usually for estimation of α kernels with positive elements are used. Coefficients β, γ have the same kernels as in linear LPA. Coefficients δ, ζ detect vertical and horizontal ridges / valleys on image. Coefficient ε detects saddle points.

5. Quadric Polynomial approximation in 5x5 block.

In this other important case we use approximation

$$I_2(x,y) = \alpha + \beta * x + \gamma * y + \delta * x^2 + \varepsilon * x * y + \zeta * y^2$$

Interpolation level is 6/25.

Convolution kernels for $\alpha, \beta, \gamma, \delta, \varepsilon, \zeta$ coefficients calculation are:

$$\alpha : \frac{1}{175}\begin{bmatrix} -13 & 2 & 7 & 2 & -13 \\ 2 & 17 & 22 & 17 & 2 \\ 7 & 22 & 27 & 22 & 7 \\ 7 & 17 & 22 & 17 & 2 \\ -13 & 2 & 7 & 2 & -13 \end{bmatrix} \quad \beta : \frac{1}{50}\begin{bmatrix} -2 & -1 & 0 & 1 & 2 \\ -2 & -1 & 0 & 1 & 2 \\ -2 & -1 & 0 & 1 & 2 \\ -2 & -1 & 0 & 1 & 2 \\ -2 & -1 & 0 & 1 & 2 \end{bmatrix}$$

$$\gamma:\frac{1}{50}\begin{bmatrix}-2&-2&-2&-2&-2\\-1&-1&-1&-1&-1\\0&0&0&0&0\\+1&+1&+1&+1&+1\\+2&+2&+2&+2&+2\end{bmatrix}\qquad \delta:\frac{1}{70}\begin{bmatrix}2&-1&-2&-1&2\\2&-1&-2&-1&2\\2&-1&-2&-1&2\\2&-1&-2&-1&2\\2&-1&-2&-1&2\end{bmatrix}$$

$$\varepsilon:\frac{1}{50}\begin{bmatrix}+4&+2&0&-2&-4\\+2&+1&0&-1&-2\\0&0&0&0&0\\-2&-1&0&+1&+2\\-4&-2&0&+2&+4\end{bmatrix}\qquad \zeta:\frac{1}{50}\begin{bmatrix}+2&+2&+2&+2&+2\\-1&-1&-1&-1&-1\\-2&-2&-2&-2&-2\\-1&-1&-1&-1&-1\\+2&+2&+2&+2&+2\end{bmatrix}$$

Interpretation of coefficients is the same as for 3x3 quadric LPA. Difference is only in the block size.

6. Cubic Polynomial approximation in 5x5 block.

In this interesting case we use approximation

$$I_3(x,y)=\alpha+\beta*x+\gamma*y+\delta*x^2+\varepsilon*x*y+\zeta*y^2+\eta*x^3+\theta*x^2*y+\iota*x*y^2+\kappa*y^3$$

Interpolation level is 10/25 = 2/5.

Convolution kernels for $\alpha,\beta,\gamma,\delta,\varepsilon,\zeta,\eta,\theta,\iota,\kappa$ coefficients calculation are:

$$\alpha:\frac{1}{175}\begin{bmatrix}-13&2&7&2&-13\\2&17&22&17&2\\7&22&27&22&7\\7&17&22&17&2\\-13&2&7&2&-13\end{bmatrix}\qquad \beta:\frac{1}{420}\begin{bmatrix}+31&-44&0&44&-31\\-5&-62&0&62&5\\-17&-68&0&68&17\\-5&-62&0&62&5\\+31&-44&0&44&-31\end{bmatrix}$$

$$\gamma:\frac{1}{420}\begin{bmatrix}31&-5&-17&-5&31\\-44&-62&-68&-62&-44\\0&0&0&0&0\\+44&+62&+68&+62&+44\\-31&+5&+17&+5&-31\end{bmatrix}\qquad \delta:\frac{1}{70}\begin{bmatrix}2&-1&-2&-1&2\\2&-1&-2&-1&2\\2&-1&-2&-1&2\\2&-1&-2&-1&2\\2&-1&-2&-1&2\end{bmatrix}$$

$$\varepsilon:\frac{1}{50}\begin{bmatrix}+4&+2&0&-2&-4\\+2&+1&0&-1&-2\\0&0&0&0&0\\-2&-1&0&+1&+2\\-4&-2&0&+2&+4\end{bmatrix}\qquad \zeta:\frac{1}{70}\begin{bmatrix}+2&+2&+2&+2&+2\\-1&-1&-1&-1&-1\\-2&-2&-2&-2&-2\\-1&-1&-1&-1&-1\\+2&+2&+2&+2&+2\end{bmatrix}$$

$$\eta:\frac{1}{60}\begin{bmatrix}-1 & 2 & 0 & -2 & 1\\ -1 & 2 & 0 & -2 & 1\\ -1 & 2 & 0 & -2 & 1\\ -1 & 2 & 0 & -2 & 1\\ -1 & 2 & 0 & -2 & 1\end{bmatrix}\qquad \theta:\frac{1}{140}\begin{bmatrix}+4 & -2 & -4 & -2 & 4\\ +2 & -1 & -2 & -1 & 2\\ 0 & 0 & 0 & 0 & 0\\ -2 & +1 & 2 & +1 & -2\\ -4 & +2 & 4 & +2 & -4\end{bmatrix}$$

$$\iota:\frac{1}{140}\begin{bmatrix}-4 & -2 & 0 & +2 & +4\\ +2 & +1 & 0 & -1 & -2\\ +4 & +2 & 0 & -2 & -4\\ +2 & +1 & 0 & -1 & -2\\ -4 & -2 & 0 & +2 & +4\end{bmatrix}\qquad \kappa:\frac{1}{60}\begin{bmatrix}+1 & +1 & +1 & +1 & +1\\ -2 & -2 & -2 & -2 & -2\\ 0 & 0 & 0 & 0 & 0\\ +2 & +2 & +2 & +2 & +2\\ -1 & -1 & -1 & -1 & -1\end{bmatrix}$$

Kernels for $\alpha,\delta,\zeta,\varepsilon$ are the same as corresponding kernels for quadric LPA.
We see here new non-trivial convolution masks for coefficients β,γ (gradient coefficients). They are the same as coefficients in tetra LPA with polynomial power equals 4.

Here we have new coefficients η,θ,ι,κ. It will be interesting to find their geometrical interpretation and possible use.

7. Applications to noise filtration.

Noise filtration of images using Local Polynomial Approximation (LPA) based on assumption that the image can be presented by set of formulas (one formula per pixel)

$$I(x,y)=I_{ini}(x,y)+n(x,y)$$

Here $I(x,y)$ is an observable image, which is the sum of "ideal" image $I_{ini}(x,y)$ without noise and $n(x,y)$, which is the noise in pixel with coordinates (x,y).

We assume that "ideal" image can be represented locally by set of polynomials. But it is not an analytic function: changes in one part of image are very often not influential to other parts of the image. We also assume that the noise $n(x,y)$ is Gaussian with 0 mean value and standard deviation $\sigma(x,y)$ which can be vary from pixel to pixel. We also have to make assumption about the power of polynomial n, which can be 0, 1, 2, 3, 4, 5.

If we use LPA, that means that coefficient "α" for a selected pixel is a LSF approximation of image brightness for this pixel:

$$I_{ini}(0,0)=\alpha$$

It means that we can use for filtration kernels for the estimation of coefficient α in polynomial approximation.
For example if we make assumption that in blocks of 3x3 images can be described by linear polynomials, that means we can use the kernel

$$\frac{1}{9}\begin{bmatrix}1 & 1 & 1\\ 1 & 1 & 1\\ 1 & 1 & 1\end{bmatrix}$$

for noise filtration.
If we think that the best is a weighted (by Rosenfeld) linear approximation we have to use Rosenfeld's kernel

$$\frac{1}{16}\begin{bmatrix}1 & 2 & 1\\ 2 & 4 & 2\\ 1 & 2 & 1\end{bmatrix}$$

If locally image is more complicated, we can use for 3x3 blocks square LPA:

$$I_2(x,y)=\alpha+\beta*x+\gamma*y+\delta*x^2+\varepsilon*x*y+\zeta*y^2$$

It means that noise filter should use kernel

$$\frac{1}{9}\begin{bmatrix}-1 & 2 & -1\\ +2 & 5 & +2\\ -1 & 2 & -1\end{bmatrix}$$

Noise filtration based on quadric 5x5 LPA should use the kernel:

$$\frac{1}{175}\begin{bmatrix}-13 & 2 & 7 & 2 & -13\\ 2 & 17 & 22 & 17 & 2\\ 7 & 22 & 27 & 22 & 7\\ 7 & 17 & 22 & 17 & 2\\ -13 & 2 & 7 & 2 & -13\end{bmatrix}$$

Noise filtration based on tetra 5x5 LPA should use the kernel:

$$\frac{1}{825}\begin{bmatrix}+19 & -31 & 24 & -31 & +19\\ -31 & -56 & 174 & -56 & -31\\ +24 & 174 & 429 & 174 & +24\\ -31 & -56 & 174 & -56 & -31\\ 19 & -31 & 24 & -31 & +19\end{bmatrix}$$

And so on.

Filtration will be the best only in the case when the assumption about polynomial power is correct, polynomials with this power will describe the image properly. We need special investigation to find proper power of polynomial for any given image.

8. Exactly locally approximated images.

Obviously that image is a 3x3 linear LPA, if after filtration using kernel

$$\frac{1}{9}\begin{bmatrix}1 & 1 & 1\\ 1 & 1 & 1\\ 1 & 1 & 1\end{bmatrix} \qquad (8.1)$$

the image does not change.

Similarly image is 3x3 quadratic LPA if after filtration using kernel

$$\frac{1}{9}\begin{bmatrix}-1 & 2 & -1\\ +2 & 5 & +2\\ -1 & 2 & -1\end{bmatrix} \qquad (8.2)$$

the image does not change.

For any image it is possible to get linear LPA image most close to the first one by multiple filtration of it until result will not change, Let's use standard grey Lenna image from [10] (it is on Picture 1). After around 550 repetitions of filtration with kernel (8.1) result image does not change. It is in Picture 2. We see strong degradation of image after such a transformation. The difference between initial and transformed image is in Picture 3.
Now we show result of using kernel (8.2). After around 70 repetitions of filtration with kernel (8.2) we see that result image stops changing. It is in Picture 4. Compare this image with initial Lenna we can find only small changes in sharp edges. Difference of Lenna with result image is in picture 5.

We conclude that image can be dividing onto three parts:

- part which is linear LPA (regular linear pixels)
- part which is quadric LPA (regular quadric pixels)
- part which needs more complex approximation(complex pixels)

We did a program which for every point estimates the power of LPA. We generate image in which every linear point is black, every quadric point is grey, and every complex approximated point is white. The calculation result is in Picture 6.

Similar investigations can be conducted using 5x5 LPA convolution matrices: standard and rounded. In this case image is divided onto 4 parts:

- linear LPA pixels
- quadric LPA pixels
- tetra LPA pixels
- complex pixels

For Lenna image results are similar to previous one with some differences.

This pixel division can be used for fine image processing, where we first define the type of pixel and select the proper kernel.

9. Gradient calculations.

Gradient of function $I(x,y)$ is defined as a vector with coordinates:

$$G_x(x,y)=\frac{\partial I(x,y)}{\partial x};\qquad G_y(x,y)=\frac{\partial I(x,y)}{\partial y};$$

In polynomial approximation the components of gradient are:

$$G_x = \beta \qquad G_y = \gamma$$

Using different power of the approximating polynomial and different kernel mask we can generate many different convolution kernel matrices for gradient estimation. Here we summarise masks for 3x3, 4x4, and 5x5 blocks for gradient calculations.

Linear and quadric 3x3 LPA:

$$G_x : \frac{1}{6}\begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix} \qquad G_y : \frac{1}{6}\begin{bmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ +1 & +1 & +1 \end{bmatrix}$$

Rosenfeld weighted Linear LPA (Sobel kernels):

$$G_x : \frac{1}{8}\begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} \quad G_y : \frac{1}{8}\begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix}$$

Linear and quardic 5x5 LPA:

$$G_x : \frac{1}{50}\begin{bmatrix} -2 & -1 & 0 & 1 & 2 \\ -2 & -1 & 0 & 1 & 2 \\ -2 & -1 & 0 & 1 & 2 \\ -2 & -1 & 0 & 1 & 2 \\ -2 & -1 & 0 & 1 & 2 \end{bmatrix} \quad G_y : \frac{1}{50}\begin{bmatrix} -2 & -2 & -2 & -2 & -2 \\ -1 & -1 & -1 & -1 & -1 \\ 0 & 0 & 0 & 0 & 0 \\ +1 & +1 & +1 & +1 & +1 \\ +2 & +2 & +2 & +2 & +2 \end{bmatrix}$$

Cubic and tetric 5x5 LPA:

$$G_x : \frac{1}{420}\begin{bmatrix} +31 & -44 & 0 & 44 & -31 \\ -5 & -62 & 0 & 62 & 5 \\ -17 & -68 & 0 & 68 & 17 \\ -5 & -62 & 0 & 62 & 5 \\ +31 & -44 & 0 & 44 & -31 \end{bmatrix} \quad G_y : \frac{1}{420}\begin{bmatrix} 31 & -5 & -17 & -5 & 31 \\ -44 & -62 & -68 & -62 & -44 \\ 0 & 0 & 0 & 0 & 0 \\ +44 & +62 & +68 & +62 & +44 \\ -31 & +5 & +17 & +5 & -31 \end{bmatrix}$$

In appendices 2 and 3 there are gradient kernels for 5x5 rounded cubic LPA and for 5x5 corner-cutting cubic LPA.

The best gradient estimation should be started from pixel type estimation: linear LPA or quadric LPA or tetra LPA. And only after can one calculate the gradient, using the corresponding kernel. It will be the best estimation of the gradient.

10. Laplace operators for 3x3 and 5x5 blocks.

Laplace operator for function $I(x, y)$ is defined as a scalar field:

$$\frac{\partial^2 I(x,y)}{\partial x^2} + \frac{\partial^2 I(x,y)}{\partial y^2}$$

In polynomial approximation Laplace operator is equals to

$$\left(2*\delta + 2*\zeta\right).$$

Here we give examples of corresponding convolution kernels for 3x3 and 5x5 blocks.

Laplace operator for quadric 3x3 LPA:

$$\frac{1}{3}\begin{bmatrix} +2 & -1 & +2 \\ -1 & -4 & -1 \\ +2 & -1 & +2 \end{bmatrix}$$

Laplace operator for quadric 5x5 LPA:

$$\frac{1}{35}\begin{bmatrix} +4 & +1 & 0 & +1 & +4 \\ +1 & -2 & -3 & -2 & +1 \\ 0 & -3 & -4 & -3 & 0 \\ +1 & -2 & -3 & -2 & +1 \\ +4 & +1 & 0 & +1 & +4 \end{bmatrix}$$

It will be interesting to calculate Laplace operator convolution matrix for 5x5 tetra LPA.

11. Applications to image down-sampling.

Here we consider image Local Polynomial Approximation (LPA) method for a special case of resizing of the images : reducing image size twice for width and height. Usually it is called "down-sampling".

Standard down-sampling algorithm is based on substitution of 2x2 blocks by one pixel. Value of a pixel is a mean value of pixels in 2x2 block.

LPA based down-sampling uses 4x4 blocks for calculation of a coefficient "α". After that internal 2x2 block in 4x4 is substituted by pixel with "α" value.

If we use cubic LPA, that convolution matrix for estimation of "α" equals

$$\alpha : \frac{1}{32}\begin{bmatrix} -3 & 2 & 2 & -3 \\ +2 & 7 & 7 & +2 \\ +2 & 7 & 7 & +2 \\ -3 & 2 & 2 & -3 \end{bmatrix}$$

If we use tetra LPA with polynomial power equals 4, that convolution matrix for estimation of "α" equals

$$\alpha : \frac{1}{256}\begin{bmatrix} +1 & -9 & -9 & +1 \\ -9 & 81 & 81 & -9 \\ -9 & 81 & 81 & -9 \\ +1 & -9 & -9 & +1 \end{bmatrix}$$

Up-sampling is more complicated and needs in special investigation.

12. Applications to grey scale image compression.

There are many ways that are possible for using of Local Polynomial Approximation (LPA) in image compression. Here we shortly described one simple method, based on division of image onto 4x4 or 8x8 blocks.

For every block we do the following:

First of all we estimate coefficients of linear LPA

$$I_1(x,y) = \alpha + \beta * x + \gamma * y$$

and test the block on acceptance of this approximation. If exactness is acceptable, we mark this block as linear and save (α, β, γ).
If block is not linear we use quadratic LPA

$$I_2(x,y) = \alpha + \beta * x + \gamma * y + \delta * x^2 + \varepsilon * x * y + \zeta * y^2$$

and calculate and save coefficients $(\alpha, \beta, \gamma, \delta, \varepsilon, \zeta)$.
As a result we have 6 sparse (except α) small matrices with coefficients. We save them using lossless compression.
After that for every block we create a residual block as a difference of block and its LPA prediction. We compress it using DCT, quantization, zigzag rearrangement and entropy encoding.

13. Polynomial approximation in 3x3x3 cube and spatial temporal video filtration.

Here we consider image LPA method for video streams. Let us take 3 consecutive frames from the stream for the same scene.
In the middle frame we select one pixel and create a 3x3 pixel cube around it. For that in the left and right frames we find 3x3 square blocks which corresponds to 3x3 block in the middle frame. We can use for this any search algorithm for block matching for block motion estimation.
We have in result 3x3x3 cube of pixels, comprised from three vertical 3x3 square blocks, one block from one frame. Pixels in the left square block have usual x,y coordinates and common time coordinate equals -1. Pixels in the middle block have time coordinate t=0 and pixels in the right block have time coordinate t=1.
For approximation of pixels values in this cube we select quadric polynomial

$$I(x,y,t) = \alpha + \beta * x + \gamma * y + \delta * t + \varepsilon * x^2 + \zeta * x * y + \eta * x * t + \theta * y^2 + \iota * y * t + \kappa * t^2$$

For this 3 Dimensional LPA interpolation level equals 10/27.

Here convolution kernels are cubes not squares. We present each of them by three squares: one for the left frame, one for the middle frame and one for the right frame. And for every convolution cube we give a normalization multiplier.

Coefficient α : normalization multiplier $1/27$, convolution cube for coefficient calculation:

$$\begin{bmatrix} -2 & 1 & -2 \\ +1 & 4 & +1 \\ -2 & 1 & -2 \end{bmatrix} \quad \begin{bmatrix} 1 & 4 & 1 \\ 4 & 7 & 4 \\ 1 & 4 & 1 \end{bmatrix} \quad \begin{bmatrix} -2 & 1 & -2 \\ +1 & 4 & +1 \\ -2 & 1 & -2 \end{bmatrix}$$

Coefficient β : normalization multiplier $1/18$, convolution cube for coefficient calculation:

$$\begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix} \quad \begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix} \quad \begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix}$$

Coefficient γ : normalization multiplier $1/18$, convolution cube for coefficient calculation:

$$\begin{bmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ +1 & +1 & +1 \end{bmatrix} \quad \begin{bmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ +1 & +1 & +1 \end{bmatrix} \quad \begin{bmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ +1 & +1 & +1 \end{bmatrix}$$

Coefficient δ : normalization multiplier $1/18$, convolution cube for coefficient calculation:

$$\begin{bmatrix} -1 & -1 & -1 \\ -1 & -1 & -1 \\ -1 & -1 & -1 \end{bmatrix} \quad \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \quad \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

Coefficient ε : normalization multiplier $1/18$, convolution cube for coefficient calculation:

$$\begin{bmatrix} 1 & -2 & 1 \\ 1 & -2 & 1 \\ 1 & -2 & 1 \end{bmatrix} \quad \begin{bmatrix} 1 & -2 & 1 \\ 1 & -2 & 1 \\ 1 & -2 & 1 \end{bmatrix} \quad \begin{bmatrix} 1 & -2 & 1 \\ 1 & -2 & 1 \\ 1 & -2 & 1 \end{bmatrix}$$

Coefficient ζ : normalization multiplier $1/12$, convolution cube for coefficient calculation:

$$\begin{bmatrix} +1 & 0 & -1 \\ 0 & 0 & 0 \\ -1 & 0 & +1 \end{bmatrix} \quad \begin{bmatrix} +1 & 0 & -1 \\ 0 & 0 & 0 \\ -1 & 0 & +1 \end{bmatrix} \quad \begin{bmatrix} +1 & 0 & -1 \\ 0 & 0 & 0 \\ -1 & 0 & +1 \end{bmatrix}$$

Coefficient η : normalization multiplier $1/12$, convolution cube for coefficient calculation:

$$\begin{bmatrix} 1 & 0 & -1 \\ 1 & 0 & -1 \\ 1 & 0 & -1 \end{bmatrix} \quad \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \quad \begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix}$$

Coefficient θ : normalization multiplier $1/18$, convolution cube for coefficient calculation:

$$\begin{bmatrix} +1 & +1 & +1 \\ -2 & -2 & -2 \\ +1 & +1 & +1 \end{bmatrix} \quad \begin{bmatrix} +1 & +1 & +1 \\ -2 & -2 & -2 \\ +1 & +1 & +1 \end{bmatrix} \quad \begin{bmatrix} +1 & +1 & +1 \\ -2 & -2 & -2 \\ +1 & +1 & +1 \end{bmatrix}$$

Coefficient ι: normalization multiplier $1/12$, convolution cube for coefficient calculation:

$$\begin{bmatrix} +1 & +1 & +1 \\ 0 & 0 & 0 \\ -1 & -1 & -1 \end{bmatrix} \quad \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \quad \begin{bmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ +1 & +1 & +1 \end{bmatrix}$$

Coefficient κ: normalization multiplier $1/18$, convolution cube for coefficient calculation:

$$\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} \quad \begin{bmatrix} -2 & -2 & -2 \\ -2 & -2 & -2 \\ -2 & -2 & -2 \end{bmatrix} \quad \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

For video noise filtration convolution cube for coefficient α calculation can be used.

14. Conclusions.

Local polynomial approximation (LPA) of images is a very powerful method for the solving of different problems in image processing. Pixel classification onto regular and complex pixels is useful for image processing without degradation.
It is regrettable that this approach is rarely and incompletely described in literature.

15. Future directions.

First of all, the question of how to process parts of the image with complex pixels is very important and needs a special investigation.

Second, it will be useful to calculate all kernels for LPA coefficients in blocks 5x5, 7x7 and 8x8 for polynomials up to power equals 5. It also will be interesting to get LPA coefficients for rounded 4x4 blocks. Calculation of 3D LPA coefficients with different convolution cube size, geometry and polynomial power will be useful for video stream filtration.

Third, to review the possible LPA using in different applications in image/video processing and compression.

Fourth, it will be interesting to investigate other function types for local image approximation. For example, investigate functions which are described by the formula

$$F(x,y) = \frac{a + b*x + c*y}{1 + d*x + e*y}$$

And by the formula

$$G(x,y) = \frac{a + b*x + c*y + d*x^2 + e*x*y + f*y^2}{1 + g*x + h*y}$$

Because we are using here rational polynomial functions, it can be called LRA (local rational approximation).

We used for LPA two discrete functions

$$X(x,y)=\begin{bmatrix}-2 & -1 & 0 & 1 & 2\\ -2 & -1 & 0 & 1 & 2\\ -2 & -1 & 0 & 1 & 2\\ -2 & -1 & 0 & 1 & 2\\ -2 & -1 & 0 & 1 & 2\end{bmatrix} \qquad Y(x,y)=\begin{bmatrix}-2 & -2 & -2 & -2 & -2\\ -1 & -1 & -1 & -1 & -1\\ 0 & 0 & 0 & 0 & 0\\ +1 & +1 & +1 & +1 & +1\\ +2 & +2 & +2 & +2 & +2\end{bmatrix}$$

But it possible to use other couples of functions of two variables

$$U(x,y);V(x,y)$$

for local approximation. Image brightness is represented by formula

$$I(x,y)=\alpha+\beta*U(x,y)+\gamma*V(x,y)+\delta*U^2(x,y)+\varepsilon*U(x,y)*V(x,y)+\zeta*(V(x,y)+...$$

For example, it may be interesting use of step functions

$$S_x(x,y)=\begin{bmatrix}-3 & -2 & 0 & 2 & 3\\ -3 & -2 & 0 & 2 & 3\\ -3 & -2 & 0 & 2 & 3\\ -3 & -2 & 0 & 2 & 3\\ -3 & -2 & 0 & 2 & 3\end{bmatrix} \qquad S_y(x,y)=\begin{bmatrix}-3 & -3 & -3 & -3 & -3\\ -2 & -2 & -2 & -2 & -2\\ 0 & 0 & 0 & 0 & 0\\ +2 & +2 & +2 & +2 & +2\\ +3 & +3 & +3 & +3 & +3\end{bmatrix}$$

Appendix 1. Cubic polynomial approximation in 4x4 block.

Here we give convolution kernels for coefficients calculation using standard 4x4 mask
Corresponding polynomial is:

$$I_3(x,y)=\alpha+\beta*x+\gamma*y+\delta*x^2+\varepsilon*x*y+\zeta*y^2+\eta*x^3+\theta*x^2*y+\iota*x*y^2+\kappa*y^3$$

Interpolation level is 10/16=5/8. Kernels for coefficients calculations are:

$$\alpha:\frac{1}{32}\begin{bmatrix}-3 & 2 & 2 & -3\\ +2 & 7 & 7 & +2\\ +2 & 7 & 7 & +2\\ -3 & 2 & 2 & -3\end{bmatrix} \qquad \varepsilon:\frac{1}{400}\begin{bmatrix}-9 & -3 & +3 & +9\\ -3 & -1 & +1 & +3\\ +3 & +1 & -1 & -3\\ +9 & +3 & -3 & -9\end{bmatrix}$$

$$\beta:\frac{1}{96}\begin{bmatrix}+5 & -12 & 12 & -5\\ -4 & -15 & 15 & +4\\ -4 & -15 & 15 & +4\\ +5 & -12 & 12 & -5\end{bmatrix} \qquad \gamma:\frac{1}{96}\begin{bmatrix}+5 & -4 & -4 & +5\\ -12 & -15 & -15 & -12\\ +12 & +15 & +15 & +12\\ -5 & +4 & +4 & -5\end{bmatrix}$$

$$\delta: \frac{1}{64}\begin{bmatrix} 1 & -1 & -1 & 1 \\ 1 & -1 & -1 & 1 \\ 1 & -1 & -1 & 1 \\ 1 & -1 & -1 & 1 \end{bmatrix} \qquad \zeta: \frac{1}{64}\begin{bmatrix} +1 & +1 & +1 & +1 \\ -1 & -1 & -1 & -1 \\ -1 & -1 & -1 & -1 \\ +1 & +1 & +1 & +1 \end{bmatrix}$$

$$\eta: \frac{1}{192}\begin{bmatrix} -1 & 3 & -3 & 1 \\ -1 & 3 & -3 & 1 \\ -1 & 3 & -3 & 1 \\ -1 & 3 & -3 & 1 \end{bmatrix} \qquad \kappa: \frac{1}{192}\begin{bmatrix} -1 & -1 & -1 & -1 \\ +3 & +3 & +3 & +3 \\ -3 & -3 & -3 & -3 \\ +1 & +1 & +1 & +1 \end{bmatrix}$$

$$\theta: \frac{1}{192}\begin{bmatrix} -3 & +3 & +3 & -3 \\ -1 & +1 & +1 & -1 \\ +1 & -1 & -1 & +1 \\ +3 & -3 & -3 & +3 \end{bmatrix} \qquad \iota: \frac{1}{192}\begin{bmatrix} -3 & -1 & +1 & +3 \\ +3 & +1 & -1 & -3 \\ +3 & +1 & -1 & -3 \\ -3 & -1 & +1 & +3 \end{bmatrix}$$

Appendix 2. Linear, quadric and cubic LPA in rounded 5x5 block.

Here we give convolution kernels for coefficients calculation using geometry mask

$$\begin{bmatrix} & \# & \# & \# & \\ \# & \# & \# & \# & \# \\ \# & \# & \# & \# & \# \\ \# & \# & \# & \# & \# \\ & \# & \# & \# & \end{bmatrix}$$

Linear LPA polynomial is:

$$I_1(x, y) = \alpha + \beta * x + \gamma * y$$

Interpolation level is 1/7. Kernels for coefficients calculations are:

$$\alpha: \frac{1}{21}\begin{bmatrix} & 1 & 1 & 1 & \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ & 1 & 1 & 1 & \end{bmatrix}$$

$$\beta : \frac{1}{34}\begin{bmatrix} & -1 & 0 & 1 & \\ -2 & -1 & 0 & 1 & 2 \\ -2 & -1 & 0 & 1 & 2 \\ -2 & -1 & 0 & 1 & 2 \\ & -1 & 0 & 1 & \end{bmatrix} \qquad \gamma : \frac{1}{34}\begin{bmatrix} & -2 & -2 & -2 & \\ -1 & -1 & -1 & -1 & -1 \\ 0 & 0 & 0 & 0 & 0 \\ +1 & +1 & +1 & +1 & +1 \\ & +2 & +2 & +2 & \end{bmatrix}$$

Quadric LPA polynomial is

$$I_2(x,y) = \alpha + \beta * x + \gamma * y + \delta * x^2 + \varepsilon * x * y + \zeta * y^2$$

Interpolation level is 2/7. Kernels for coefficients calculations are:

$$\alpha : \frac{1}{335}\begin{bmatrix} & -14 & 3 & -14 & \\ -14 & 37 & 54 & 37 & -14 \\ 3 & 54 & 71 & 54 & 3 \\ -14 & 37 & 54 & 37 & -14 \\ & -14 & 3 & -14 & \end{bmatrix} \qquad \varepsilon : \frac{1}{36}\begin{bmatrix} & -2 & 0 & +2 & \\ -2 & -1 & 0 & +1 & +2 \\ 0 & 0 & 0 & 0 & 0 \\ +2 & +1 & 0 & -1 & -2 \\ & +2 & 0 & -2 & \end{bmatrix}$$

$$\beta : \frac{1}{34}\begin{bmatrix} & -1 & 0 & 1 & \\ -2 & -1 & 0 & 1 & 2 \\ -2 & -1 & 0 & 1 & 2 \\ -2 & -1 & 0 & 1 & 2 \\ & -1 & 0 & 1 & \end{bmatrix} \qquad \gamma : \frac{1}{34}\begin{bmatrix} & -2 & -2 & -2 & \\ -1 & -1 & -1 & -1 & -1 \\ 0 & 0 & 0 & 0 & 0 \\ +1 & +1 & +1 & +1 & +1 \\ & +2 & +2 & +2 & \end{bmatrix}$$

$$\delta : \frac{1}{4690}\begin{bmatrix} & +29 & -78 & +29 & \\ 230 & -91 & -198 & -91 & 230 \\ 190 & -131 & -238 & -131 & 190 \\ 230 & -91 & -198 & -91 & 230 \\ & +29 & -78 & +29 & \end{bmatrix}$$

$$\zeta : \frac{1}{4690}\begin{bmatrix} & 230 & 190 & 230 & \\ +29 & -91 & -131 & -91 & +29 \\ -78 & -198 & -238 & -198 & -78 \\ +29 & -91 & -131 & -91 & +29 \\ & 230 & 190 & 230 & \end{bmatrix}$$

Cubic LPA polynomial is:

$$I_3(x,y) = \alpha + \beta * x + \gamma * y + \delta * x^2 + \varepsilon * x * y + \zeta * y^2 + \eta * x^3 + \theta * x^2 * y + \iota * x * y^2 + \kappa * y^3$$

Interpolation level is 10/21. Kernels for coefficients calculations are:

$$\alpha:\frac{1}{335}\begin{bmatrix} & -14 & 3 & -14 & \\ -14 & 37 & 54 & 37 & -14 \\ 3 & 54 & 71 & 54 & 3 \\ -14 & 37 & 54 & 37 & -14 \\ & -14 & 3 & -14 & \end{bmatrix} \qquad \varepsilon:\frac{1}{36}\begin{bmatrix} & -2 & 0 & +2 & \\ -2 & -1 & 0 & +1 & +2 \\ 0 & 0 & 0 & 0 & 0 \\ +2 & +1 & 0 & -1 & -2 \\ & +2 & 0 & -2 & \end{bmatrix}$$

$$\beta:\frac{1}{300}\begin{bmatrix} & +4 & 0 & -4 & \\ +23 & -62 & 0 & 62 & -23 \\ -21 & -84 & 0 & 84 & +21 \\ +23 & -62 & 0 & 62 & -23 \\ & 4 & 0 & -4 & \end{bmatrix} \qquad \gamma:\frac{1}{300}\begin{bmatrix} & +23 & -21 & +23 & \\ +4 & -62 & -84 & -62 & +4 \\ 0 & 0 & 0 & 0 & 0 \\ -4 & +62 & +84 & +62 & +4 \\ & -23 & +21 & -23 & \end{bmatrix}$$

$$\delta:\frac{1}{4690}\begin{bmatrix} & +29 & -78 & +29 & \\ 230 & -91 & -198 & -91 & 230 \\ 190 & -131 & -238 & -131 & 190 \\ 230 & -91 & -198 & -91 & 230 \\ & +29 & -78 & +29 & \end{bmatrix}$$

$$\zeta:\frac{1}{4690}\begin{bmatrix} & 230 & 190 & 230 & \\ +29 & -91 & -131 & -91 & +29 \\ -78 & -198 & -238 & -198 & -78 \\ +29 & -91 & -131 & -91 & +29 \\ & 230 & 190 & 230 & \end{bmatrix}$$

$$\eta:\frac{1}{300}\begin{bmatrix} & 2 & 0 & -2 & \\ -11 & 14 & 0 & -14 & 11 \\ -3 & 18 & 0 & -18 & 3 \\ -11 & 14 & 0 & -14 & 11 \\ & 2 & 0 & -2 & \end{bmatrix} \qquad \kappa:\frac{1}{300}\begin{bmatrix} & -11 & -3 & -11 & \\ +2 & +14 & +18 & +14 & +2 \\ 0 & 0 & 0 & 0 & 0 \\ -2 & -14 & -18 & -14 & -2 \\ & +11 & +3 & +11 & \end{bmatrix}$$

$$\theta:\frac{1}{100}\begin{bmatrix} & +2 & -4 & +2 & \\ +6 & -3 & -6 & -3 & +6 \\ 0 & 0 & 0 & 0 & 0 \\ -6 & +3 & +6 & +3 & -6 \\ & -2 & +4 & -2 & \end{bmatrix} \qquad \iota:\frac{1}{100}\begin{bmatrix} & -6 & 0 & +6 & \\ -2 & +3 & 0 & -3 & +2 \\ +4 & +6 & 0 & -6 & -4 \\ -2 & +3 & 0 & -3 & +2 \\ & -6 & 0 & +6 & \end{bmatrix}$$

Appendix 3. Linear, quadratic and cubic LPA in edge cutting 5x5 block.

Here we give convolution kernels for coefficients calculation using geometry mask

$$\begin{bmatrix} & & \# & & \\ & \# & \# & \# & \\ \# & \# & \# & \# & \# \\ & \# & \# & \# & \\ & & \# & & \end{bmatrix}$$

Linear LPA polynomial is:

$$I_1(x,y) = \alpha + \beta * x + \gamma * y$$

Interpolation level is 3/13 (low). Kernels for coefficients calculations are:

$$\alpha : \frac{1}{13}\begin{bmatrix} & & 1 & & \\ & 1 & 1 & 1 & \\ 1 & 1 & 1 & 1 & 1 \\ & 1 & 1 & 1 & \\ & & 1 & & \end{bmatrix}$$

$$\beta : \frac{1}{14}\begin{bmatrix} & & 0 & & \\ & -1 & 0 & 1 & \\ -2 & -1 & 0 & 1 & 2 \\ & -1 & 0 & 1 & \\ & & 0 & & \end{bmatrix} \qquad \gamma : \frac{1}{14}\begin{bmatrix} & & -2 & & \\ & -1 & -1 & -1 & \\ 0 & 0 & 0 & 0 & 0 \\ & +1 & +1 & +1 & \\ & & 2 & & \end{bmatrix}$$

Quadric LPA polynomial is

$$I_2(x,y) = \alpha + \beta * x + \gamma * y + \delta * x^2 + \varepsilon * x * y + \zeta * y^2$$

Interpolation level is 6/13 (middle). Kernels for coefficients calculations are:

$$\alpha : \frac{1}{11}\begin{bmatrix} & & -1 & & \\ & +1 & +2 & +1 & \\ -1 & +2 & +3 & +2 & -1 \\ & +1 & +2 & +1 & \\ & & -1 & & \end{bmatrix} \qquad \varepsilon : \frac{1}{4}\begin{bmatrix} & & 0 & & \\ & -1 & 0 & +1 & \\ 0 & 0 & 0 & 0 & 0 \\ & +1 & 0 & -1 & \\ & & 0 & & \end{bmatrix}$$

$$\beta:\frac{1}{14}\begin{bmatrix} & & 0 & & \\ & -1 & 0 & 1 & \\ -2 & -1 & 0 & 1 & 2 \\ & -1 & 0 & 1 & \\ & & 0 & & \end{bmatrix} \qquad \gamma:\frac{1}{14}\begin{bmatrix} & & -2 & & \\ & -1 & -1 & -1 & \\ 0 & 0 & 0 & 0 & 0 \\ & +1 & +1 & +1 & \\ & & 2 & & \end{bmatrix}$$

$$\delta:\frac{1}{2618}\begin{bmatrix} & & +50 & & \\ & -17 & -166 & -17 & \\ 358 & -89 & -238 & -89 & 358 \\ & -17 & -166 & -17 & \\ & & +50 & & \end{bmatrix}$$

$$\zeta:\frac{1}{2618}\begin{bmatrix} & & 358 & & \\ & -17 & -89 & -17 & \\ 50 & -166 & -238 & -166 & 50 \\ & -17 & -89 & -17 & \\ & & 358 & & \end{bmatrix}$$

Cubic LPA polynomial is:

$$I_3(x,y)=\alpha+\beta*x+\gamma*y+\delta*x^2+\varepsilon*x*y+\zeta*y^2+\eta*x^3+\theta*x^2*y+\iota*x*y^2+\kappa*y^3$$

Interpolation level is 10/13 (high). Kernels for coefficients calculations are:

$$\alpha:\frac{1}{11}\begin{bmatrix} & & -1 & & \\ & +1 & +2 & +1 & \\ -1 & +2 & +3 & +2 & -1 \\ & +1 & +2 & +1 & \\ & & -1 & & \end{bmatrix} \qquad \varepsilon:\frac{1}{4}\begin{bmatrix} & & 0 & & \\ & -1 & 0 & 1 & \\ 0 & 0 & 0 & 0 & 0 \\ & 1 & 0 & -1 & \\ & & 0 & & \end{bmatrix}$$

$$\beta:\frac{1}{12}\begin{bmatrix} & & 0 & & \\ & 0 & 0 & 0 & \\ 1 & -8 & 0 & 8 & -1 \\ & 0 & 0 & 0 & \\ & & 0 & & \end{bmatrix} \qquad \gamma:\frac{1}{12}\begin{bmatrix} & & +1 & & \\ & 0 & -8 & 0 & \\ 0 & 0 & 0 & 0 & 0 \\ & 0 & +8 & 0 & \\ & & -1 & & \end{bmatrix}$$

$$\delta : \frac{1}{2618}\begin{bmatrix} & & +50 & & \\ & -17 & -166 & -17 & \\ 358 & -89 & -238 & -89 & 358 \\ & -17 & -166 & -17 & \\ & & +50 & & \end{bmatrix}$$

$$\zeta : \frac{1}{2618}\begin{bmatrix} & & 358 & & \\ & -17 & -89 & -17 & \\ 50 & -166 & -238 & -166 & 50 \\ & -17 & -89 & -17 & \\ & & 358 & & \end{bmatrix}$$

$$\eta : \frac{1}{12}\begin{bmatrix} & & 0 & & \\ & 0 & 0 & 0 & \\ -1 & 2 & 0 & -2 & 1 \\ & 0 & 0 & 0 & \\ & & 0 & & \end{bmatrix} \qquad \kappa : \frac{1}{12}\begin{bmatrix} & & -1 & & \\ & 0 & +2 & 0 & \\ 0 & 0 & 0 & 0 & 0 \\ & 0 & -2 & 0 & \\ & & +1 & & \end{bmatrix}$$

$$\theta : \frac{1}{4}\begin{bmatrix} & & 0 & & \\ & +1 & -2 & +1 & \\ 0 & 0 & 0 & 0 & 0 \\ & -1 & +2 & -1 & \\ & & 0 & & \end{bmatrix} \qquad \iota : \frac{1}{4}\begin{bmatrix} & & 0 & & \\ & -1 & 0 & +1 & \\ 0 & +2 & 0 & -2 & 0 \\ & -1 & 0 & +1 & \\ & & 0 & & \end{bmatrix}$$

References

[1] A. Rosenfeld, “Picture Processing for Computer”, Academic Press, NY, 1969

[2] W. Pratt “Digital Image Processing”, Wiley-Interscience, 2007

[3] M. Sonka, V. Hlavac, R. Boyle “Image processing, analysis and Machine Vision”,

[4] K. Castleman, Digital Image Processing, Prentice-Hall, 1979

[5] Handbook of Image and Video Processing, editor Al Bovik, Elsevier, 2005

[6] A.M. Tekalp, Digital Video Processing, Prentice-Hall, 1995

[7] R. Haralick, “7.3. The Digital Step Edge”, NASA report

[8] Quang Ji, R. Haralick, “Efficient facet edge detection and quantitative performance evaluation”, Pattern Recognition 35 (2002), 689-700

[9] R. van den Boomgaard, “The image Facet model”, Informatics Institute, University of Amsterdam

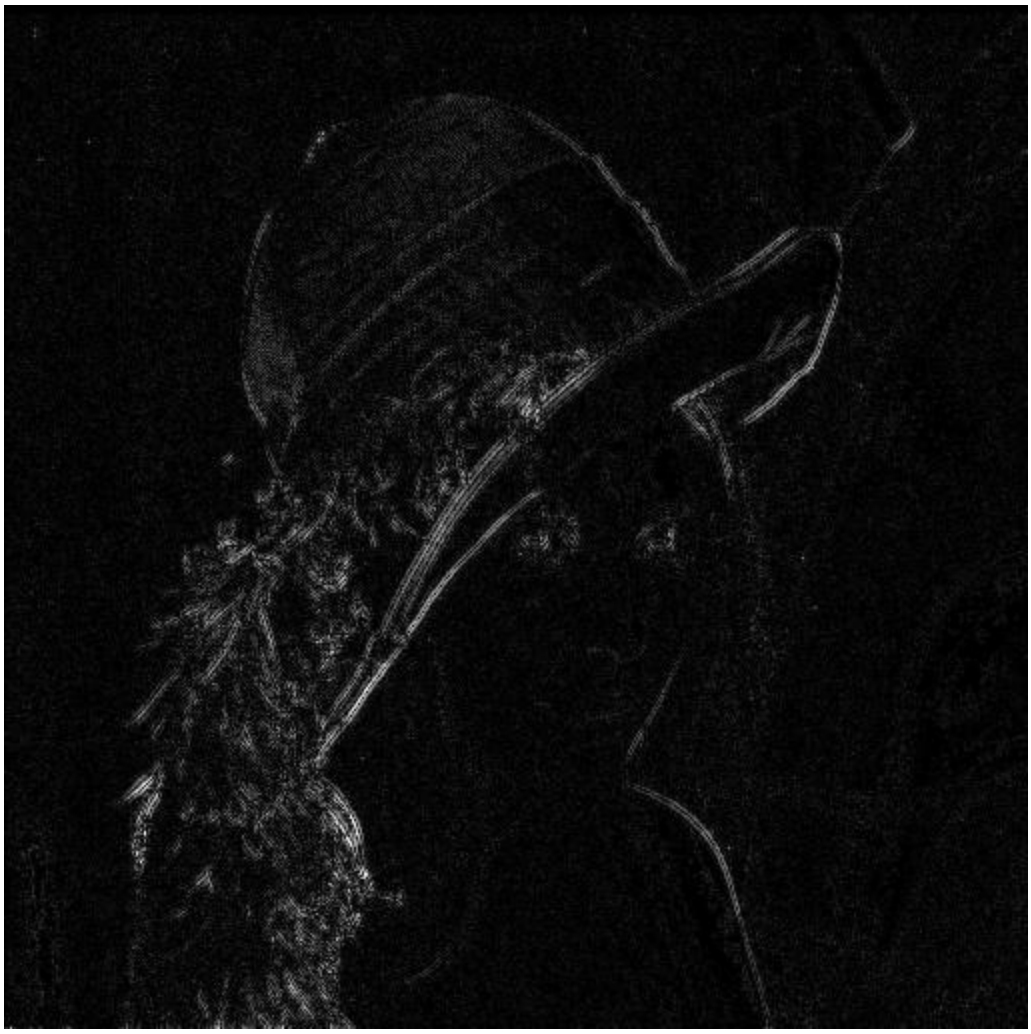
[10] www.hlevkin.com/TestImages/lenna.bmp

Picture 1. Lenna image

Picture 2. Linear 3x3 approximated Lenna image

Picture 3.Difference of Picture 1 and Picture 2, multiplied by 2

Picture 4. Quadratic 3x3 approximated Lenna.



Picture 5. Difference between Picture 1 and Picture 4, multiplied by 4

Picture 6. Segmentation of Lenna image.