Local Polynomial Approximation (LPA) of images and some of its Applications.

Guennadi Henry Levkine (email: hlevkin@gmail.com)
Gibsons, BC, Canada.

Abstract. The original treatment of image local polynomial approximation in blocks 3x3 and 5x5 is proposed. They use it for convolution kernel building. These kernels are using for different problem solving in image processing.

Key words: image processing, convolution kernel, noise filtration, image gradient, image resize, feature points, Laplace kernel

1. Introduction.

Among different approaches to image processing some of the most popular are:
- Spectral transformation of images or their local parts (blocks)
- Convolutions of images with square matrices of size 3x3, 5x5, 7x7. 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 image processing. Usually it is doing empirically. But for building of such kernels it is possible locally approximate image intensity by polynomials of two variables (rows and columns) and after that uses these coefficients for kernel building.

Here we give kernels for calculation of polynomial coefficients of different size and shape. We demonstrate how to build these kernels for noise filtration, and gradient estimation, Laplace operator kernels, resizing and more by the use of coefficients from LPA. The coefficients are calculated using the linear least squares fitting technique.

2. Two dimensional polynomial approximations and pixel enumerating.

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

$$A(x,y) = a + b*x + c*y + d*x*x + e*x*y + f*y*y \qquad (1)$$

For the coefficients a, b, c, d, e, f calculation we use standard Least Square Fit method for minimization of difference between our approximation A(x, y) of brightness and real brightness in image.
We do it for blocks 3x3, 5x5 and 7x7, surrounding every pixel in image. Coordinates of pixels in block are (x,y)

For block 3x3 pixel coordinates are

(0, 0)	(1, 0)	(2, 0)
(0, 1)	(1, 1)	(2,1)
(0,2)	(1, 2)	(2, 2)

For block 5x5 pixel coordinates are

(0, 0)	(1, 0)	(2, 0)	(3,0)	(4,0)
(0, 1)	(1, 1)	(2,1)	(3,1)	(4,1)
(0,2)	(1, 2)	(2, 2)	(3,2)	(4,2)
(0,3)	(1,3)	(2,3)	(3,3)	(4,3)
(0,4)	(1,4)	(2,4)	(3,4)	(4,4)

3. Linear local approximation in 3x3 and 5x5 blocks.

In this case we use simple linear approximation

$$I(x, y) = a + b * x + c * y \qquad (2)$$

For calculation of 'a, b, c' coefficients convolution 3x3 kernels are (Weights are 1/6, 1/6, 1/6) :

1	1	1
1	1	1
1	1	1

-1	0	1
-1	0	1
-1	0	1

-1	-1	-1
0	0	0
1	1	1

Using 5x5 blocks, corresponding convolution kernels are (with weights 1/25, 1/50, 1/50):

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

-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

-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

4. Quadric polynomial approximation in 3x3 and 5x5 blocks

In this case we use approximation [1]

Coefficients "a, b, c, d, e, f" are estimated by Kernels with weights 1/9, 1/6, 1/6, 1/6, 1/4, 1/6

-1	2	-1
2	5	2
-1	2	-1

-1	-1	-1
0	0	0
1	1	1

-1	-1	-1
0	0	0
1	1	1

1	-2	1
1	-2	1
1	-2	1

-1	0	1
0	0	0
1	0	-1

1	1	1
-2	-2	-2
1	1	1

For 5x5 blocks convolution kernels are (with weights 1/175, 1/50, 1/50, 1/70, 1/50, 1 /70):

-13	2	7	2	-13
2	17	22	17	2
7	22	27	22	7
2	17	22	17	2
-13	2	7	2	-13

-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

-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

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

+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

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

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

5. 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)

$$A\,(x, y) = I\,(x, y) + n(x, y)$$

Here $A\,(x, y)$ is an observable image, which is the sum of "ideal" image $I\,(x, y)$ *ini* 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 • • (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 'a' for a selected pixel is a LSF approximation of image brightness for this pixel:

$$I(x, y) \sim a$$

It means that we can use for filtration kernels for the estimation of coefficient 'a' 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 this kernel for noise filtration:

1	1	1
1	1	1
1	1	1

If we think that the best is a weighted (by Rosenfeld) linear approximation if we have used Rosenfeld's kernel (look at [7]):

1	2	1
2	4	2
1	2	1

If locally image is more complicated, we can use for 3x3 blocks square LPA (Levkin's 3x3 filtration):

-1	2	-1
2	5	2
-1	2	-1

Noise filtration based on quadric 5x5 LPA should use the kernel (Levkin's 5x5 filtration):

-13	2	7	2	-13
2	17	22	17	2
7	22	27	22	7
2	17	22	17	2
-13	2	7	2	-13

Noise filtration kernel based on tetra 5x5 LPA is in [7]

6. Image segmentation to linear and quadric parts.

Obviously that image brightness is a 3x3 linear, if after line LPA filtration using kernel (W=1/6) it does not changed (human eye can't detect changes).

1	1	1
1	1	1
1	1	1

Similarly image is 3x3 quadratic LPA, if after filtration using kernel (W=1/9) image practically does not change , but linear filtration significantly change the image.

-1	2	-1
2	5	2
-1	2	-1

For any image it is possible to get part most close to the first one by multiple filtration of it until result will not change, Let's use standard grey Lenna image from [6] . After around 550 repetitions of filtration with the first kernel here result image does not change. It is in Picture 1. We see strong degradation of image after such a transformation..
Now we show result of using kernel (8.2). After around 70 repetitions of filtration with the second kernel here we see that result image stops changing. It is in Picture 2. Compare this image with initial Lenna we can find only small changes in sharp edges. Difference of Lenna with result image is in picture 3.

We conclude that image can be dividing onto three parts:
- part which is linear
- part which is quadric
- part which needs more complex approximation

We did a program which for every point estimates type of surrounding blocks. It generates the image in which every linear point is black, every quadric point is grey, and every complex approximated point is white. We found

that complex approximated point can be treated as feature points. (Our definition of feature points: they are in corners, ridges, ravines and so on ...)
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 (very complex feature points)

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

7. **Image Gradient calculation using LPA.**

Gradient of image A(x, y) is defined as a vector with coordinates

$$G_x = \frac{\partial A}{\partial x}, \qquad G_y = \frac{\partial \mathbf{A}}{\partial \mathbf{y}}$$

In polynomial approximation the components of gradient are: $G_x = b$, $G_y = c$

Using different power of the approximating polynomial and different kernel mask we can generate many different convolution kernel matrices for gradient estimation. Here we show kernels for gradient calculations for 3x3 and 5x5 image blocks .

Linear and quadric 3x3 LPA (weight $\frac{1}{6}$):

- 1	0	1
- 1	0	1
- 1	0	1

- 1	- 1	- 1
0	0	0
1	1	1

Rosenfeld weighted kernel (weight $\frac{1}{8}$):

- 1	0	1
- 2	0	2
- 1	0	1

- 1	- 2	- 1
0	0	0
1	1	1

Linear and quadric 5x5 LPA (weight $\frac{1}{50}$):

- 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

- 2	- 2	- 2	- 2	- 2
- 1	- 1	- 1	- 1	- 1
0	0	0	0	0
1	-1	-2	-1	2
2	-1	-2	-1	2

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

Laplace transformation for image A(x, y) is defined as a scalar field:

$$L(x,y) = \frac{\partial^2 A}{\partial x^2} + \frac{\partial^2 A}{\partial y^2}$$

In polynomial approximation Laplace operator is equals to (2*d+2*f)

Here are convolution kernels for its calculation for 3x3 and 5x5 image blocks (W=1/3 and W=1/35):

1	0	-1
0	0	0
-1	0	1

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

Second kernel is really non trivial. It will be interesting to investigate its properties.

9. 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

Mean value of pixels in 2x2 block.

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

If we use cubic LPA, that convolution matrix for estimation of "a" equals (W=1/32)

- 3	2	2	- 3
2	7	7	2
2	7	7	2
- 3	2	2	- 3

10. Conclusions.

Local polynomial approximation (LPA) of images is a powerful method for the solving of different problems in image processing. Pixel classification onto regular and complex pixels is very useful for image processing without degradation and for image feature point detection.

11. Acknowledgements.

I gratefully acknowledge V. Levkine for the support in writing and reviewing of this article.

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

[7] H. Levkine, “Image Local Polynomial Approximation (LPA) and its Applications”, 2011

Picture 1.linear 3x3 approximated Lenna.

Picture 2. Quadratic 3x3 approximated Lenna