Novel heuristics for deconvolution applied to picture deblurring

Evgeniya Olenuk, Maxim Gromov
Faculty of Radiophysics
Tomsk State University
Tomsk, Russia
Email: evgeniya_ok@mail.ru, gromov@sibmail.com

***Abstract*—In this paper we describe an implementation of a method for blurred pictures restoring. We consider uniform, linear photo-camera shift (movement) while picture is taken. In this case, a picture, formed on a chip of a camera, is not static, but moving along some straight line, and as a result photo is blurred. Blurred picture can be seen as a convolution of two functions – one describes initial picture, another is point-spread (or blurring) function (PSF) [1]–[3]. "Deblurred" picture can be found as a deconvolution of the given picture with known PSF [1], [2]. To do this it is convenient to use Fourier transform and apply the convolution theorem. But it leads to division by 0, and thus the method is unstable. We suggest two heuristics to avoid division by 0 and compare them with the known one [2].**

***Index Terms*—Digital picture processing, blurred picture, Fourier transform.**

I. Introduction

Rapid development of computers brings lots of new possibilities into the world of picture processing. Some operations become easier with a computer – like colour correction, or appliqué creation etc – some become possible by themselves. One of such operation, which has no "manual" implementation, is picture deblurring [1]–[3].

In this paper we tell about our program which restores blurred pictures, taken by a digital camera. We use well-known method of deconvolution of two signals [1], [2]. The idea of the method is to describe blurring as a convolution of two functions and then to use Fourier transform to solve corresponding equation. However, this equation does not have a solution in general, since at some points it requires division by 0. To overtake this hardship there are several approaches.

One is known as Wiener deconvolution [4]. It relies on the fact, that in practice there is no measurement of a signal which can be done without noise. So the noise-to-signal ratio is never 0 and being added to the denominator of the deconvolution fraction it helps to avoid division by 0.

Another one is suggested by the authors of [2] and actually is application of the regularizing theory by Tikhonov [5]. They also modify the denominator of the deconvolution fraction in such a way, that it never turns 0. The idea comes from the fact, that we can tolerate the solution to be not exact but approximate.

Also we should mention Richardson-Lucy deconvolution [6]. It does not use Fourier transformation and being an iterative method it is considered to be very slow. Its implementation is used, for example, in ASTRA IMAGE software [7].

All mentioned methods assume the point-spread (or blurring) function (PSF) to be known. However there are bunch of works, devoted to automatic PSF reconstruction [3], [8], [9].

In our work we assume PSF to be known and to be one of the simplest type: uniform, linear shift of a camera. We use Fourier transform based deconvolution and suggest two new heuristics to avoid division by 0, which prove to work well according to our experiments, and which, we believer, are a bit simpler then Wiener or Tikhonov deconvolution. The idea is to modify denominator of the deconvolution fraction only when there is a need in division by 0. Then we replace denominator with 1, so there is no any division (first heuristics) or with the value which was used in divison at previous point (second heuristics).

The rest paper is structured as follows. In section II brief description of the method is given. Section III tells about the application itself. In section IV we show some experimental results, and section V concludes the paper.

II. Method description

A. Picture and its blurring

A picture may be defined as a function $u(x, y)$, where x and y are coordinates on a plane of the camera sensor [4]. The value of u at a point (x, y) is intensity of the light beam which has reached this point.

If the camera or the picturing object is moving while shutter is open, then a light point, formed on the sensor by the camera lens, will travel along some line and as a resulting in a blurred picture (Figure 1).

In our work we assume, that the camera (or object) motion is straight-line and uniform (without acceleration). In this case the travelling line of light points will be rectilinear segment of a length L, and the light intensity of the point will be uniformly distributed along this line. Without loss of generality we can assume, that the movement occurs along the axis x. Then intensity of light $u(x, y)$, measured by the sensor at the point (x, y), can be counted by the following formula $u(x, y) = \frac{1}{L}v(x, y) + \frac{1}{L}v(x + \Delta x, y) + \frac{1}{L}v(x + 2\Delta x, y) +$

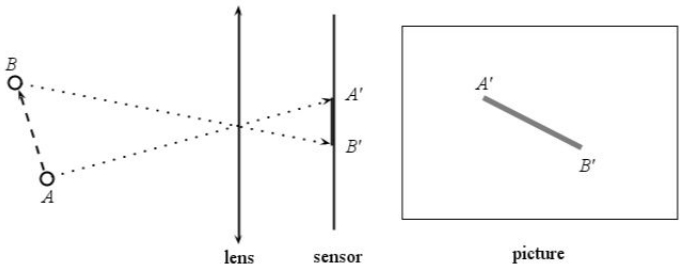


Figure 1. Scheme of formation blurred picture

$\ldots + \frac{1}{L}v(x+L,y)$, or in integral form

$$u(x,y) = \int_{-\infty}^{\infty} k(x-\xi)v(\xi,y)d\xi, \tag{1}$$

where $v(x,y)$ – picture which would be got, if there were no movement (and blurring),

$$k(x) = \begin{cases} \frac{1}{L}, & \text{if } x \leqslant L, \\ 0, & \text{otherwise} \end{cases}$$

is blurring function.

The expression (1) is a convolution-like equation for function $v(x,y)$, which is a original unblurred picture. Applying Fourier transform to the both sides of the equation (1) and keeping in mind convolution theorem [10] we shall get algebraic equation for Fourier image $V(\omega,y)$ of the function $v(x,y)$: $U(\omega,y) = K(\omega)V(\omega,y)$. Unfortunately, $|K(\omega)|$ turns 0 (or, with no difference, very small) for some values of ω and in this points solution could not be found as

$$V(\omega,y) = \frac{U(\omega,y)}{K(\omega)} \equiv \frac{U(\omega,y)K^*(\omega)}{|K(\omega)|^2}. \tag{2}$$

B. Different approaches to avoid division by 0

There are several approaches to avoid division on 0 in expression (2). The historically first approach should be considered deconvolution, based on Wiener's filter [**?**]:

$$V(\omega,y) = \frac{U(\omega,y)K^*(\omega)}{|K(\omega)|^2 + \mathrm{NSR}(\omega)},$$

where $\mathrm{NSR}(\omega) > 0$ is noise-to-signal ratio, and since it is hard to have a measurement without noise, this function in practice is never 0. In most application it can be replaced with some a priory known constant.

In [2] it is suggested to find solution in the form

$$V(\omega,y) = \frac{U(\omega,y)K^*(\omega)}{|K(\omega)|^2 + \alpha\omega^{2p}},$$

where α and p are heuristics parameters. This formula comes from the regularizing theory by Tikhonov [5]. It is easy to see, that in certain cases Tikhonov deconvolution turns into Wiener deconvolution, for example, when $p = 0$ and $\alpha = \mathrm{NSR} = \mathrm{const}$.

We use a little bit different approach. We suggest to keep formula (2) intact while there is no division by 0 and only in cases of such ω, that $|K(\omega)|^2 < \varepsilon$ replace $|K(\omega)|^2$ with 1 (then there is no division), or with the value which was used obtained for previous ω. Here ε is a priory specified (by user) threshold.

C. Digital case

Since all pictures we deal with are taken by digital camera, function $u(x,y)$ (as well as $v(x,y)$ and $k(x)$) and its arguments x and y will take discrete values within certain limits. Usually, for 24-bit `.bmp` files, which we shall use for our application, value of intensity of a channel (red, green or blue) lies within 0 and 255, 0 stands for the lowest intensity (no light), 255 – the highest. Intensity, which is less then the lowest is considered to be 0, and which is more then the highest – 255.

Values are measured equidistantly all over camera's sensor, which is usually rectangular. Points of the measurement are called pixels. Upper left pixel has coordinates $(0,0)$, next right to it – $(1,0)$ etc; next bottom to the pixel $(0,0)$ is pixel $(0,1)$ etc. Thus, digital picture is a matrix $\mathbf{u}$, where $\mathbf{u}(x,y)$ is intensity, measured at pixel (x,y), $x \in \{0,1,\ldots m-1\}$, $y \in \{0,1,\ldots n-1\}$, m – width of the picture in pixels, n – its height. Note, that we consider first index of the matrix to be its column, and second – row.

In the digital case, the integral in (1) should be replaced by the sum:

$$\mathbf{u}(x,y) = \sum_{\xi=0}^{m-1} \mathbf{k}(x-\xi)\mathbf{v}(\xi,y), \tag{3}$$

and Fourier transform becomes the discrete Fourier transform, when we try to find a solution for (3):

$$\mathbf{V}(\omega,y) = \frac{\mathbf{U}(\omega,y)}{\mathbf{K}(\omega)}. \tag{4}$$

Here $\mathbf{V}$ – is the Fourier image of the unblurred digital picture $\mathbf{v}$, $\mathbf{U}$ – is the Fourier image of the blurred digital picture $\mathbf{u}$, taken by the camera, $\mathbf{K}$ – is the Fourier image of the blurring vector $\mathbf{k}$. Again, for some ω $|\mathbf{K}(\omega)|^2$ may turn 0 or become small enough to make division in (4) invalid. If that happens, our program replaces $|\mathbf{K}(\omega)|^2$ with 1 or replaces $\mathbf{K}(\omega)$ with the value $\mathbf{K}(\omega-1)$, depending on what heuristics was chosen by a user.

We measure blurring distance L in pixels, which, we think, is the most convenient way. So, the value of L is integer and has the following meaning. Suppose $L = 5$. Then a light point, which in other circumstances would result as a pixel (x,y) with intensity $\mathbf{v}(x,y)$, "spreads" its intensity among L pixels (x,y), $(x+1,y)$, $(x+2,y)$, $\ldots(x+4,y)$ and each gets $1/L = 1/5$ part of the original intensity $\mathbf{v}(x,y)$, and thus $\mathbf{k} = \langle \frac{1}{5}, \frac{1}{5}, \frac{1}{5}, \frac{1}{5}, \frac{1}{5}, 0 \ldots, 0 \rangle$.

III. Method implementation

For implementation of the method, described in section II, we have chosen Java and NetBeans [11] as IDE. For the discrete Fourier transform we decided to use The Apache Software Foundation implementation [12] of Fast Fourier Transform algorithm [10]. This fact implies some restrictions

for the width m of a picture: it should be of an integral extent of 2. We decided m to be $2^{10} = 1024$.

The application works with files of 24-bit `.bmp` pictures of width 1024 pixels. Due to our assumption, that blurring is happening along axis x, each line of the picture (the row in terms of matrix) is processed separately (and the line has three independent channels: red, blue and green).

Interface of the program you can see in Figure 2. On the left

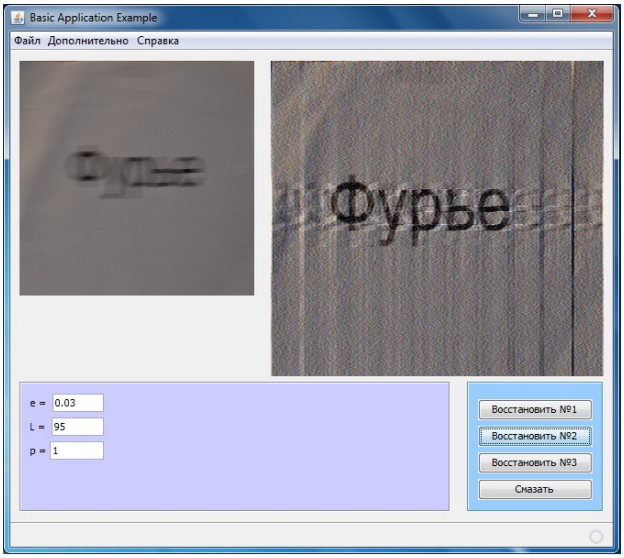


Figure 2. Interface

part of the window there is original picture and on the right – processed. User can chose either to restore the original picture (buttons "Restore No. 1", "Restore No. 2", "Restore No. 3") or blur it (the button "Blur").

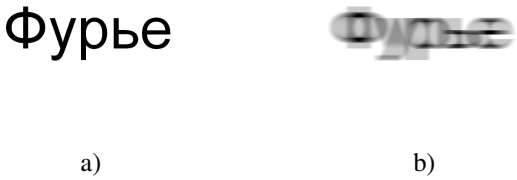
Value of e is the threshold for our heuristics. If the value of $|\mathbf{K}(\omega)|^2$ is less then e, program replaces $|\mathbf{K}(\omega)|^2$ with 1 (i.e. there is no division for this ω in (4)), when the first heuristics is chosen (button "Restore No. 1"), or replaces $\mathbf{K}(\omega)$ with the value $\mathbf{K}(\omega - 1)$, when the second heuristics is chosen (button "Restore No. 2"). In case the user presses button "Restore No. 3" Tikhonov deconvolution will be used with α and p taken from edit-fields e and p correspondingly.

Value of L stands for the blurring distance. The user can specify it by hand or by clicking mouse on the original picture.

IV. Some experiments with the program

Here we present some examples of pictures, restored by our program. First we use artificially blurred picture (Figure 3 b)), blurring distance L is 80 pixels. Restored picture is shown in Figure 4. There we used first heuristics: replacement of values of $|\mathbf{K}(\omega)|^2 < e$ with 1, where $e = 0.01$.

Then we took a picture of a sheet of paper with Figure 3a while moving the camera (see Figure 5). Result of restore is shown in Figure 6.



a) b)

Figure 3. Original picture (a) and blurred by PHOTOSHOP [13] (b)

Figure 4. Restored picture from Fig. 3b, $e = 0.01$, $L = 80$

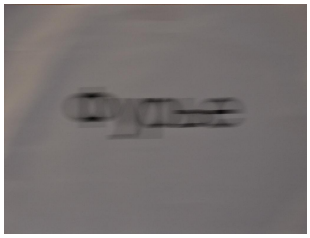
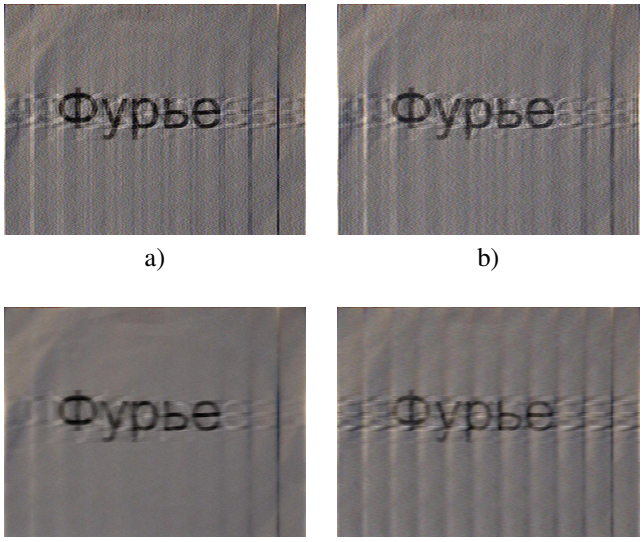


Figure 5. Really blurred picture



a) b)

c) d)

Figure 6. Restored picture from Fig. 5: a) "invalid" values of $\mathbf{K}(\omega)$ are replaced by 1, $e = 0.03$, $L = 95$; b) "invalid" values of $\mathbf{K}(\omega)$ are replaced by $\mathbf{K}(\omega - 1)$, $e = 0.04$, $L = 95$; c) using Tikhonov's method with $\alpha = 0.01$ and $p = 0$; d) using Tikhonov's method with $\alpha = 10^{-6}$ and $p = 1$

V. Conclusion

In this paper we have presented our implementation of the picture deblurring method, based on deconvolution. Blurring

process in most cases is a process with loss of data: at some points value of Fourier image $K(\omega)$ of PSF $k(x)$ becomes 0, causing loss of data. This means, that equation (1) has no solution for $v(x, y)$ (deblurred picture). To get some function, close enough to the solution of (1) authors of [2] suggested to use Tikhonov deconvolution, replacing $1/K(\omega)$ with $K^*(\omega)/(|K(\omega)|^2 + \alpha\omega^{2p})$, where α and p are parameters of their heuristics. Another well-known approach is to use Wiener's deconvolution [4] Here we have developed our own heuristics: replace $K(\omega)$ with 1 or replace $K(\omega)$ with the value from previous point whenever $|K(\omega)|^2$ becomes less then some predefined by a user threshold. Experiments have shown, that our approach is capable to work with artificially created blurred pictures as well as with blurred pictures taken by real camera.

REFERENCES

[1] V.S. Sizikov *Stable methods for the measurement results processing*, Tutorial, Saint-Petersburg: "SpetzLit", 1999 (in Russian).

[2] A.G. Yagola and N.A. Koshev *Restoring of blurred and defocused pictures*, Computational methods and programming, Moscow, Russia: Moscow University publishing house, Vol. 9, 2008, Pp.207-212 (in Russian).

[3] N. Joshi, S.B. Kang, C.L. Zitnick and R. Szeliski *Image deblurring using inertial measurement sensors*, ACM Transactions on Graphics (TOG) – Proceedings of ACM SIGGRAPH 2010, Volume 29 Issue 4, July 2010 ACM New York, NY, USA (available at http://research.microsoft.com/en-us/um/redmond/groups/ivm/imudeblurring/).

[4] R. Gonzalez and R. Woods *Digital Image Processing*, Moscow, Russia: Techosphere, 2005 (in Russian).

[5] A.N. Tikhonov and V.Ya. Arsenyev *Incorrect problems solving methods*, Moscow: Nauka, 1979 (in Russian).

[6] URL: http://en.wikipedia.org/wiki/Richardson-Lucy_deconvolution

[7] URL: http://www.phasespace.com.au/decon_ex.htm

[8] URL: http://en.wikipedia.org/wiki/Blind_deconvolution

[9] Y.-W. Tai, H. Du, M.S. Brown and S. Lin *Image/video deblurring using a hybrid camera*, In Computer Vision and Pattern Recognition, 2008.

[10] A.V. Aho, J.E. Hopcroft and J.D. Ullman *The Design and Analysis of Computer Algorithms*, Moscow, Russia: Mir, 1979 (in Russian).

[11] URL: http://netbeans.org/

[12] URL: http://commons.apache.org/math/

[13] URL: http://www.photoshop.com/