Interactive Retouching of Images in the Frequency Domain

Hannes Jung*

School of Informatics, Communications and Media
Department of Digital Media†
Upper Austria University of Applied Sciences
Hagenberg, Austria

Abstract

This paper describes a method and implementation for eliminating repetitive 2D patterns in images by interactively editing their Fourier spectra. It includes a brief discussion of the basic mathematical prerequisites, in particular, the 2D Discrete Fourier Transform and its relevant properties. Based on this concept, "InSpectral" is an interactive software application that provides intuitive and easy-to-use image editing functionality in the spectral domain with immediate visual feedback in the spatial domain. Emphasis is placed on special features, such as real-time visualization of the edited images and the design of the corresponding tools. The efficiency and quality of the results are demonstrated on several examples.

Keywords: Discrete Fourier Transform, Repetitive Pattern Removal, Denoising, Frequency Domain Editing, Image Retouching, Java, ImageJ.

1 Introduction

Pictures have gained major relevance in our daily lives. They do not just capture important moments but also transport feelings and emotions. In former times, people used their analog cameras only for special occasions but, especially since the introduction of digital cameras, images made their way into everyday life and have became a social commodity. Nowadays, almost everyone owns a camera and tries to take a picture at every occasion. Of course, images of these special moments should be of high quality. Therefore people make an effort to improve them by editing with applications like Adobe Photoshop Lightroom[1] or Apple Aperture.[2] These applications provide various tools (e.g., noise removal, red eye correction, sharping, etc.) to help users improve the quality of their pictures.

Unfortunately, there is no easy to use solution for every problem available. For example, it is very difficult to remove the printing raster of a newspaper scan, as shown in Figure 1. This paper deals with an intuitive approach to correct exactly such repetitive patterns from images, based on their distinct effects in the frequency domain.

Figure 1: Scanned newspaper image with apparent printing raster (taken from [2, Chapter 14]).

1.1 Approach

To cope with the appearance of such periodic interferences, it is useful to know that repetitive image components correspond to local energy peaks in the image's Fourier spectrum. These energy peaks show up as bright spots in the corresponding power spectrum, as shown in Figure 2.

The calculation of such a power spectrum is possible through the use of the Fourier Transform. The two-dimensional power spectrum of an image is calculated with Discrete Fourier Transform (DFT) or the Fast Fourier Transform [2, Chapter 14]. Afterwards it is necessary to suppress the energy peaks caused by the periodic noise pattern to eliminate them in the original picture. When all essential modifications in the power spectrum are done, the inverse Fourier Transform is applied to obtain the cor-

*hannes.jung@fh-hagenberg.at
†http://dm.fh-hagenberg.at
[1] http://www.adobe.com/products/photoshoplightroom/
[2] http://www.apple.com/aperture/

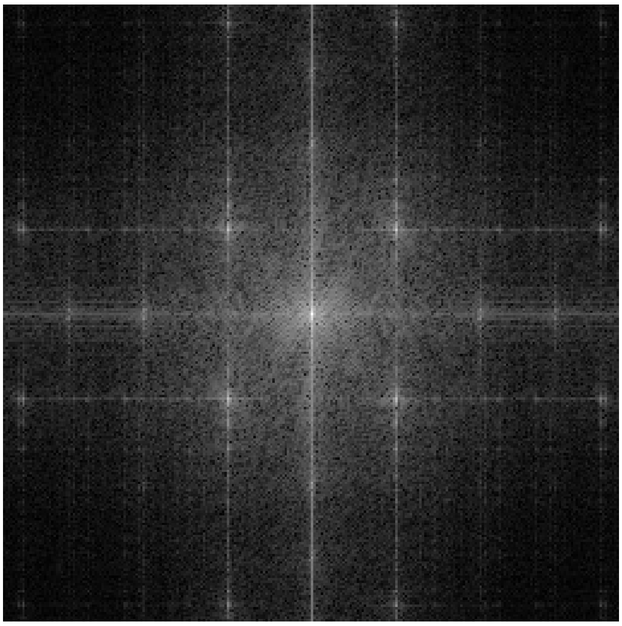


Figure 2: Logarithmic power spectrum of the image in Figure 1 (from [2, Chapter 14]). The repetitive print raster shows up as bright spots located symmetrically around the origin of the spectrum at the center.

rected version of the input image.

2 Mathematical Background

To fully understand this process some mathematics is required. This section briefly describes the Discrete Fourier Transform, the power spectrum and its relevant characteristics (symmetry, periodicity), and the application of filters in the spectral domain. Further information on this topic can be found in [1, 2, 3, 4].

2.1 The Discrete Fourier Transform (DFT)

The goal of the *continuous* Fourier Transform

$$G(\omega) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\infty} g(x) \cdot e^{-i\omega x} \,\mathrm{d}x \tag{1}$$

is to calculate all existing frequencies of a given signal or function $g(x)$. It transforms a function into another complex-valued function which is usually called Fourier spectrum $G(\omega)$. Both the signal $g(x)$ and its Fourier spectrum $G(\omega)$ are complex-valued in general. It is possible to recompute the input signal or function $g(\omega)$ from the Fourier spectrum $G(\omega)$ by applying the inverse Fourier Transform,

$$g(x) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\infty} G(\omega) \cdot e^{i\omega x} \,\mathrm{d}\omega. \tag{2}$$

Since images are discrete functions, we need to use the Discrete Fourier Transform (DFT), defined for one-dimensional signals as

$$G(m) = \frac{1}{\sqrt{M}} \sum_{u=0}^{M-1} g(u) \cdot e^{-i2\pi \frac{mu}{M}}, \tag{3}$$

for $0 \leq m < M$, where M denotes the length of the discrete signal vector $g(u) \in \mathbb{C}$. The corresponding inverse transformation is defined as

$$g(u) = \frac{1}{\sqrt{M}} \sum_{m=0}^{M-1} G(m) \cdot e^{i2\pi \frac{mu}{M}}, \tag{4}$$

for $0 \leq u < M$. All described transformations so far are applicable to one-dimensional input signals only. To process two-dimensional functions like images it is necessary to compute the discrete Fourier Transform in both dimensions. Fortunately, the 2D DFT is separable along the two dimensions, i.e., it can be calculated line by line and then row by row (or in reverse order) as

$$G(m,n) = \frac{1}{\sqrt{MN}} \sum_{u=0}^{M-1} \sum_{v=0}^{N-1} g(u,v) \cdot e^{-i2\pi(\frac{mu}{M} + \frac{nv}{N})} \tag{5}$$

and, analogously, the inverse transform as

$$g(u,v) = \frac{1}{\sqrt{MN}} \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} G(m,n) \cdot e^{i2\pi(\frac{um}{M} + \frac{vn}{N})}. \tag{6}$$

2.2 Fast Fourier Transform (FFT)

Despite the fact that the processing power of today's computers accelerates the computation of the discrete Fourier Transform to a great extent, its calculation for long input signals is still expensive. Therefore, many efficient algorithms have been implemented—they are usually referred to as "fast" Fourier Transforms [1] or FFT for short. With the FFT it is possible to reduce the time complexity of the computation with an input signal of length N from $\mathcal{O}(N^2)$ to only $\mathcal{O}(N \cdot \log_2 N)$. As shown in Table 1, the FFT is at least $10\times$ faster than the discrete transform even for small images. Furthermore, the table shows the increasing gain for longer input signals. Further information on the fast Fourier Transform is available in [3].

image size	signal length DFT/FFT	acceleration
100×100	max. 100/128	$\sim 10\times$
320×240	max. 320/512	$\sim 20\times$
640×480	max. 640/1024	$\sim 40\times$
1024×768	max. 1024/1024	$\sim 100\times$

Table 1: Acceleration by using the fast Fourier Transform algorithm.

2.3 Padding

To reach such an increase of calculation speed as mentioned in Section 2.2 the FFT algorithms are optimized

(a)

(b)

Figure 3: Two possible methods to upscale an image. The content of the image is enlarged to the next power of two (a). This can lead to poor results due to the required periodicity of the input signal (see Section 2.6 for detailed information). Only the image is enlarged, the content keeps its original size (b).

on input signal lengths of 2^k, for $k \in \mathbb{N}$. Therefore, pictures whose width or height are not powers of two must be enlarged appropriately. Figure 3 shows two different ways to accomplish this. Method (a) enlarges the whole picture including it's content to the next power of two, but because of the necessary periodicity of the input signal/image there may appear large intensity differences at the image borders. A solution would be to use a windowing function but therefore image information would be lost at the boundaries. A good solution to upscale an image is described in Figure 3 (b). Yet there is still a problem with this approach. Because there is a noticeable difference in intensity at the opposite image borders, a wide-band energy distribution will appear in the Fourier spectrum along the main axes. Thus, several important signal components of the spectrum can be shadowed and are thus lost. To solve this problem, it is common practice to combine the image content borders with one half of a Hanning windowing function (Figure 4), defined as

$$f(x) = \cos^2\left(\frac{x\pi}{2D}\right), \tag{7}$$

where D is the width of the window function. Thereby the transition from image content to image background is smoothed (Figure 5). This method is called “padding” in general and works as following:

1. The input image size is increased to $2^k \times 2^l$ with $k, l \in \mathbb{N}$. The content of the picture stays unchanged in the center of the enlarged image.
2. Every pixel of the picture content is extended to the image borders so that no pixel is undefined any more.
3. A weight image W is calculated from one half of the windowing function as

$$W_x(u,v) = \cos^2\left(\frac{d_x\pi}{2D_x}\right), \tag{8}$$

$$W_y(u,v) = \cos^2\left(\frac{d_y\pi}{2D_y}\right), \tag{9}$$

   with $d_x = |u - I_w - D_x|$ and $d_y = |v - I_h - D_y|$. The computation of this weighting image is illustrated in Figure 6.
4. Finally, the modified input and the weight image get combined by a point-wise multiplication to the resulting image (Figure 5), i.e.,

$$W_{xy}(u,v) = W_x(u,v) \cdot W_y(u,v). \tag{10}$$

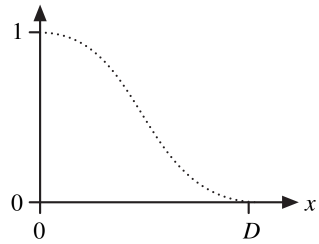


Figure 4: Plot of one half of the Hanning windowing function.

Figure 5: Image after padding.

2.4 Power Spectrum

Since the Fourier spectrum is complex-valued it is hard to visualize it as a two dimensional image. One possibility would be to illustrate the three-dimensional plots of the real and imaginary parts of the spectrum. This method of displaying the result is not very suitable in this case, because it would hardly be possible to work with them in an intuitive way.

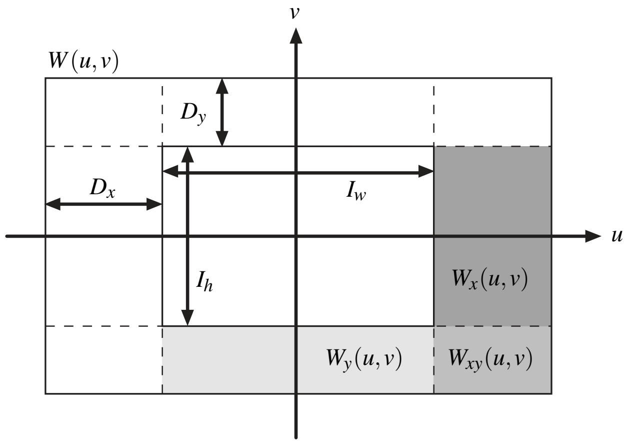


Figure 6: Schematic illustration of the weight image computation $W(u,v)$.

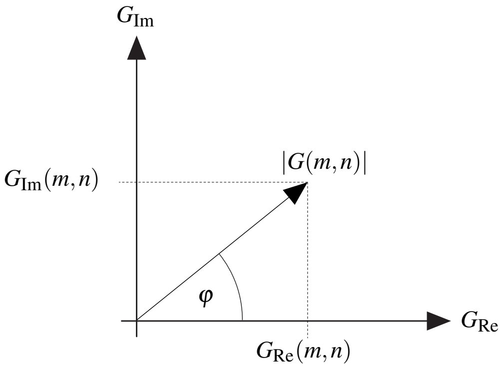


Figure 7: Illustration of the real and imaginary part of a Fourier spectrum $G(m,n)$ and how they relate to the power spectrum.

To cope with this issue, the power spectrum (i.e., the absolute values of the Fourier spectrum) is commonly used. Here we use the logarithmic power spectrum, defined as

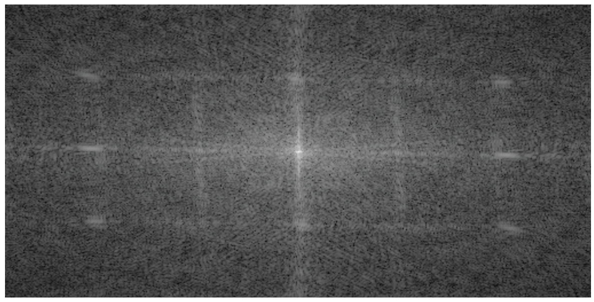
$$P(m,n) = \log\left[|G(m,n)|\right] \tag{11}$$

$$= \log\left[\sqrt{G_{\mathrm{Re}}^2(m,n) + G_{\mathrm{Im}}^2(m,n)}\right] \tag{12}$$

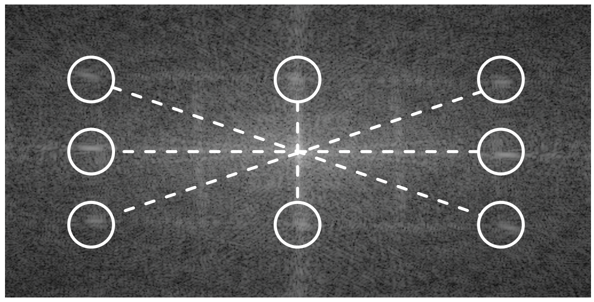
(see Figure 7). As the name tells, the power spectrum represents the power (energy) which the single frequency components of the Fourier spectrum contribute to the input signal. Two important characteristics of the power spectrum are its symmetry around the origin – just like the Fourier spectrum – and that it is a periodic function. This means that the origin is located at all four corners of the spectrum. It is common practice to exchange the quadrants in pairs so that the origin is in the center of the power spectrum.

2.5 Symmetry

One of the most important properties of the DFT is the symmetry of the resulting spectrum. Hence it is possible



(a)



(b)

Figure 8: Symmetry of the power spectrum.

to mirror refinement steps with respect to the spectrum's origin as shown in Figure 8 and thereby to reduce the required time and effort needed to edit the power spectrum.

2.6 Periodicity

To process a Fourier Transform or the resulting Fourier spectrum numerically with a computer a discrete and periodic input signal is needed. Only such an input generates a discrete and periodic spectrum. In addition to this fact the discrete and the fast Fourier Transform require the input signal to be discrete and periodic in both dimensions.

Considering such a periodic input image like Figure 9 enormous intensity differences are noticeable at the adjacent image borders. As already mentioned in Section 2.3, these intensity changes can result in wide-banded energy contributions which distort the signal components along the main axis. A practical solution to this problem is the use of windowing functions, which is described precisely in [2, 3].

2.7 Linear filtering in the frequency domain

One of the main application areas of the Fourier transform, especially of the fast Fourier Transform, in the field of image processing is the implementation of linear filters. The main advantage of using the Fourier transform at filtering images is the computational efficiency of applying filters of larger sizes.

The reason is the convolution property of the Fourier transform. This means that a linear convolution in the spatial domain, $g * h$, corresponds to a point-wise multiplica-

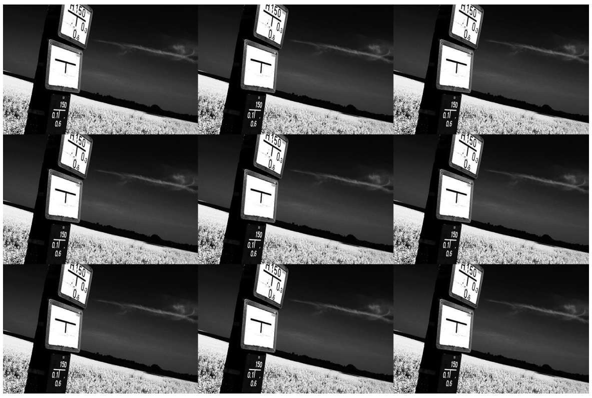


Figure 9: Input image with horizontal and vertical periodicity. The bright field at the bottom and the dark sky on the top create a major intensity difference which causes troublesome leaps in the input signal.

tion of the corresponding spectra in the frequency domain, $G \cdot H$. Considering the fact that a convolution of an image of size $M \times M$ with a filter matrix of size $N \times N$ has a time complexity of $\mathcal{O}(M^2N^2)$ but only $\mathcal{O}(M \log_2 M)$ when using a fast Fourier Transform we can say that it is almost a requisite to use a FFT for filtering images because of this enormous time saving. This process can be illustrated in the following way:

$$\begin{array}{ccccc} g(u,v) & * & h(u,v) & = & g'(u,v) \\ \downarrow & & \downarrow & & \uparrow \\ \text{DFT} & & \text{DFT} & & \text{DFT}^{-1} \\ \downarrow & & \downarrow & & \uparrow \\ G(m,n) & \cdot & H(m,n) & \rightarrow & G'(m,n) \end{array} \tag{13}$$

At first the input image $g(u,v)$ and the filter matrix $h(u,v)$ get transformed into the frequency domain by the Fourier transform DFT. Now it is possible to combine the spectrum of the image $G(m,n)$ with a point-wise multiplication with the spectrum of the filter matrix $H(m,n)$,

$$\begin{aligned} G'(m) &= G(m) \cdot H(m) \qquad (14) \\ &= (G_{\mathrm{Re}}(m) + \mathrm{i} \cdot G_{\mathrm{Im}}(m)) \cdot (H_{\mathrm{Re}}(m) + \mathrm{i} \cdot H_{\mathrm{Im}}(m)), \end{aligned}$$

where $\mathrm{i} = \sqrt{-1}$ is the imaginary unit. The resulting spectrum $G'(m,n)$ can be transformed to the spacial domain by applying an inverse Fourier transform DFT^{-1}. The outcome of this transformation is the filtered version of the input image $g'(u,v)$. Figure 10 shows this process for a one-dimensional signal.

This application uses the multiplication of a complex-valued Fourier spectrum $G(m,n)$ with a real-valued filter matrix $H(m,n)$

$$\begin{aligned} G_{\mathrm{Re}}(m,n) &= G_{\mathrm{Re}}(m,n) \cdot H(m,n) \qquad (15) \\ G_{\mathrm{Im}}(m,n) &= G_{\mathrm{Im}}(m,n) \cdot H(m,n) \\ |G(m,n)| &= \sqrt{G_{\mathrm{Re}}^2 + G_{\mathrm{Im}}^2} \end{aligned}$$

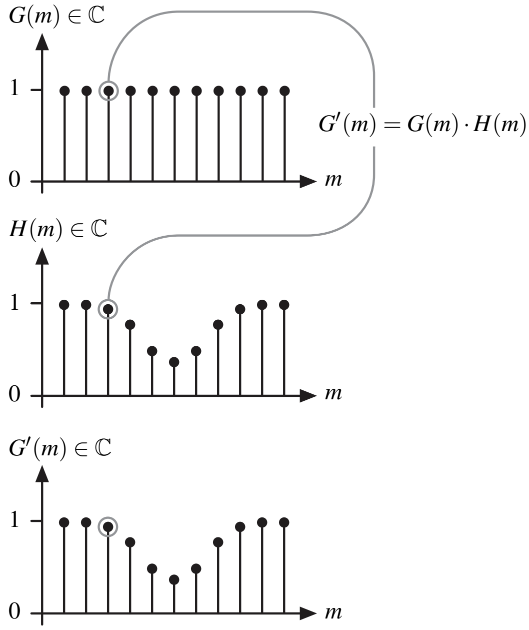


Figure 10: Point-wise multiplication of a one dimensional Fourier spectrum with a one dimensional filter.

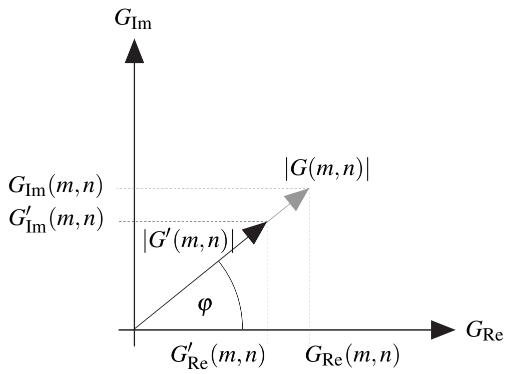


Figure 11: Point-wise multiplication of a complex value of the Fourier spectrum $G(m,n)$ with a coefficient of a real-valued filter matrix $H(m,n)$. Note that both components (real and imaginary part) are scaled to the same extent so that the angle φ stays the same.

to apply a filter to a spectrum. Hereby the real and imaginary part of the spectrum are multiplied with the same filter value so that the angle φ of the resulting vector is not changed as illustrated in figure 11.

3 Implementation

The main purpose of this work was the implementation of an image processing application which provides an intuitive approach of retouching images in the frequency domain. The application itself is a Java application and fully developed in Eclipse.[3] All the management of the image

[3] http://www.eclipse.org

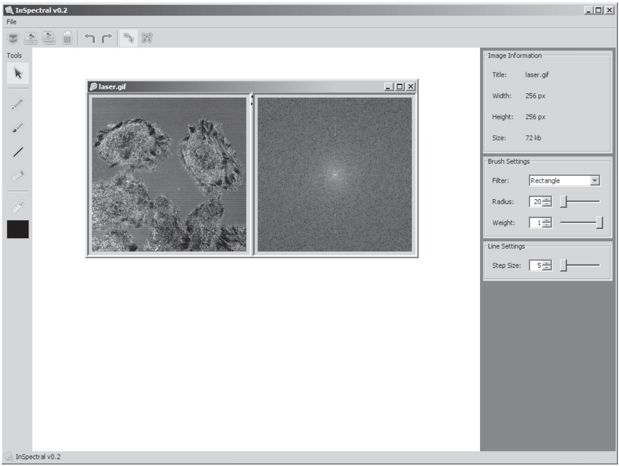


Figure 12: Graphical user interface of “InSpectral”.

data is controlled by the use of ImageJ[4] which is a free Java image processing library from National Institutes of Health[5]. The used classes and other components were adapted to fit the requirements and further criteria such as run time optimization. Furthermore, two available implementations[6] were used for computing the forward and inverse FFT. Thereby, the classes of both algorithms were adapted accordingly.

With InSpectral it is possible to work with gray scale images only – color images will be converted after opening.

3.1 User Interface

Figure 12 shows the graphical user interface of the realized application InSpectral. It consists of three main parts:

- working area,
- toolbars (main menu and toolset),
- adjustments panel.

All opened images are shown in the working area. From each of them the power spectrum is computed instantly and shown besides. Furthermore, it is possible to edit every power spectrum in this windows by simply painting in them with several tools. This works just like every other image processing software like Adobe Photoshop[7] or CorelDraw.[8] Both toolbars provide functional editing instruments that ease the way of image retouching to a large extent (e.g., the symmetrical editing option allows the user to suppress one energy peak and mirrors this editing step). In addition, it is possible to adjust almost every available tool by changing properties in the adjustment panel (e.g. filter radius and weights).

[4] http://rsbweb.nih.gov/ij
[5] http://www.nih.gov
[6] http://local.wasp.uwa.edu.au/~pbourke/other/dft/ and http://www.imagingbook.com/fileadmin/en/java/ch14.zip
[7] http://www.adobe.com/photshop
[8] http://www.coreldraw.com

3.2 Interaction

This section deals with the basic process of an editing step and is also shown in Figure 13. The interactive editing process can be summarized as follows:

1. After opening the image, the Fourier spectrum of the input image is calculated by the fast Fourier Transform. The spectrum is generally complex-valued, which means that it consists of a real and a imaginary part.
2. Since it is difficult to display a Fourier spectrum as a two dimensional image, both parts (real and imaginary) are combined into the power spectrum which can be displayed easily. The power spectrum itself is not used for editing; it is just a visual aid to illustrate the structure of the Fourier spectrum. It may look like the user is working directly on the power spectrum but the actual editing process is done in the original (complex-valued) Fourier spectrum.
3. A weight image is generated in the background when a user applies one of the provided tools to the power spectrum.
4. After each editing step (triggered by releasing the mouse button), this weight image is combined with the real and imaginary part of the Fourier spectrum by a point-wise multiplication.
5. Finally, the edited input image and the updated power spectrum are reconstructed by an inverse FFT.

3.3 Implemented Filters

To cope with all characteristics of a Fourier spectrum several filter forms were implemented:

- quadratic filter
- circular filter
- conic filter
- cosine filter
- Hanning filter
- Hamming filter
- Gauß filter

All of these filters can be adapted to the present properties of the current spectrum by adjusting the correspondent parameter like the radius or the strength of application. Furthermore there are two different tools available to apply one of these filters. At first there is the brush tool which enables the user to apply a filter at an exact location or arranged along a custom brush stroke. The alternative to the brush tool is the line tool. With this instrument the current filter can be applied on a defined line with a specified spacing.

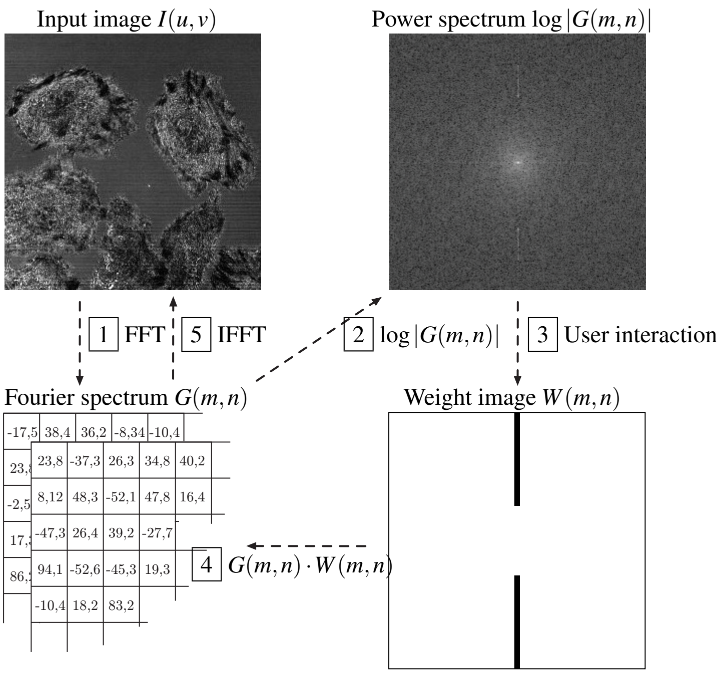


Figure 13: Process of a user interaction. A FFT is applied to the input image $I(u,v)$ to compute the complex-valued Fourier spectrum $G(m,n)$ $\boxed{1}$. The logarithmic power spectrum $\log|G(m,n)|$ is calculated $\boxed{2}$. A weight image $W(m,n)$ is created by user interaction $\boxed{3}$. The Fourier spectrum and the weight image are combined by a pointwise multiplication to the new/edited Fourier spectrum $\boxed{4}$. Finally the refined input picture is computed by an IFFT $\boxed{5}$. Furthermore the updated power spectrum is recalculated too.

4 Results

This section provides a listing of several test images and the corresponding results. The approach of the image enhancement and the quality of the results are explained on the basis of some test images.

4.1 Print raster removal

Figure 14 (a) demonstrates a cutout of an image that shows a car with a very rough printer raster. It is very unattractive to use such a picture for digital works (e.g. on websites). The periodic printer raster manifests as multiple energy peaks symmetrically located around the origin of the power spectrum. To minimize the impacts caused by this apexes a Gaussian filter was applied with double symmetry. Although the raster is still visible after the editing the image is now much clearer and small details seem to have appeared again. In addition to the refinement of the raster, the image got smoothed too. Unfortunately not only the important image parts got enhanced also disturbing elements are pointed out (especially in the shadow of the car). Another disadvantage is the loss of contrast in the picture that has occured.

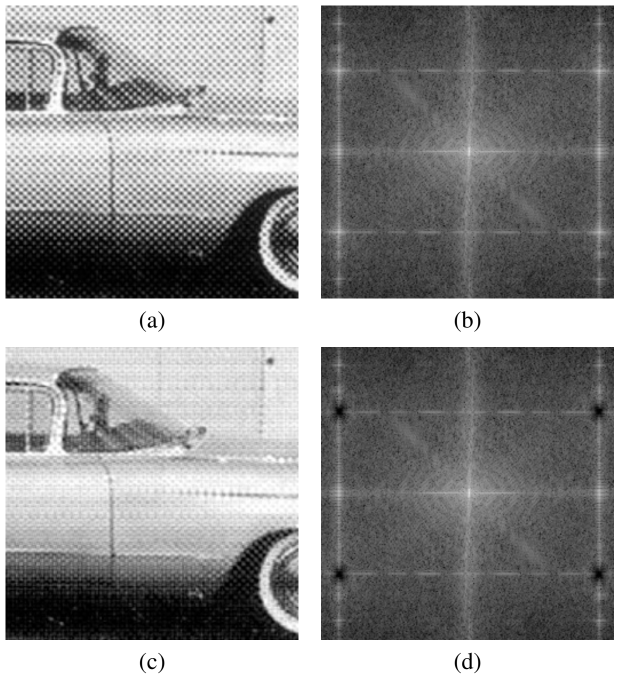
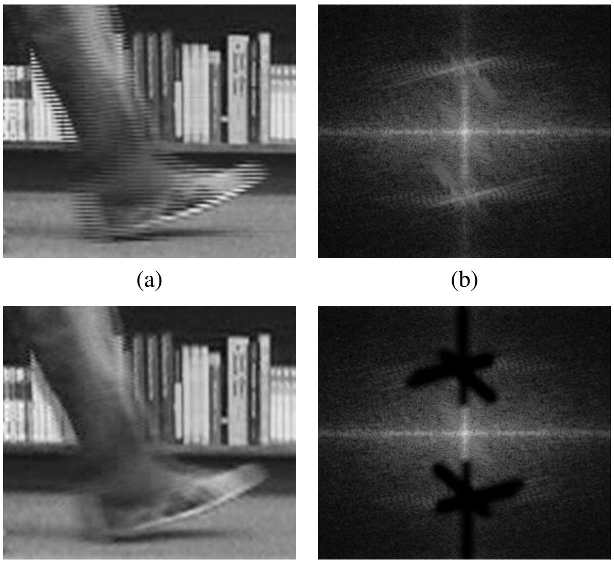


Figure 14: Picture of a car with very strong and rough printer raster (a) and its power spectrum (b). The resulting image after applying a Gaussian filter (c) is much finer and clearer. The edited power spectrum is shown in (d). All these images are just small cutouts of the original data.



(a) (b) (c) (d)

Figure 15: Picture of a moving foot with strong "interlacing" (a) and its power spectrum (b). The improved result image (c) and the edited power spectrum (d) are shown as well.

4.2 Fields/Interlacing

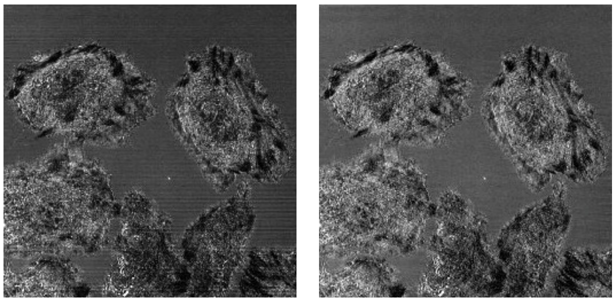
Video snapshots influenced by interlaced scanning picture another possible use of the spectral editing. The distracting scan lines displayed in figure 15 (a) show up in the power spectrum as bright fields. After cushioning them with a Gaussian filter and the line tool they completely disappear in the result image. Furthermore it can be observed that static, not moving image parts (e.g. books in the background) are not affected by the spectral retouching.

4.3 Interference lines

A less common problem are interference lines in pictures, as Figure 16 shows as horizontal lines. These lines lead to very wide-banded signal components that appear as bright lines along the vertical axis. By applying a Gaussian filter with the line tool and single symmetry to the power spectrum, the horizontal interferences vanish completely; the background seems to be smoothed. Due to the critical editing that took place, the resulting image also looses some contrast along with a slight increase of intensity.

4.4 Performance

Table 2 gives an overview of the image resolutions and milliseconds needed for the forward and inverse transformation. All these tests were done on a MacBook Pro 2.33 GHz Intel Core 2 Duo with 2 GB 667 MHz DDR2 SDRAM and an ATI Radeon X1600 256 MB graphic card.



(a) (b)

Figure 16: Picture taken by a confocal microscope with horizontal interference lines (a). After applying Gaussian filters to the power spectrum the lines disappear completely (b).

resolution	forward FFT	inverse FFT
1024×512	169 ms	156 ms
512×256	36 ms	36 ms
256×256	17 ms	17 ms

Table 2: Listing of image resolutions and time needed for the computation of a forward and inverse FFT.

5 Conclusions

After the analysis of several test images and the achieved results it is obvious that image retouching in the frequency domain is very efficient at removing repetitive image noise. Moreover, the needed run time is very short and almost real time, even when working with larger input images. The "InSpectral" tool is well suited for the use in spectral image retouching because there is no preliminary knowledge required to use the application and good results can be achieved in a very short amount of time.

6 Acknowledgements

The author would like to thank Dr. Wilhelm Burger for his support in developing this application and for helping to write this paper.

References

[1] Elbert O. Brigham. *FFT – Schnelle Fourier-Transformation*. Oldenbourg, Munich, 6th edition, 1995.

[2] Wilhelm Burger and Mark Burge. *Digital Image Processing – An Algorithmic Introduction Using Java*. Springer, New York, 2008.

[3] Tilman Butz. *Fourier Transform for Pedestrians*. Springer, Berlin, 2007.

[4] Ernest L. Hall. *Computer Image Processing and Recognition*. Academic Press, New York, 1979.