Part I

Publications IPOL

0.1 Measures - Measures - Measures

Considering an input discrete signal u_{orig} and its noised version u_{noisy}, both of M samples. We define the signal to noise ratio as follows:

$$\text{SNR}(u_{\text{orig}}, u_{\text{noisy}}) = \frac{\sqrt{\frac{1}{M}\sum_{m=0}^{M}(u_{\text{orig}}[m] - u_{\text{noisy}}[m])^2}}{\sqrt{\frac{1}{M}\sum_{m=0}^{M} u_{\text{orig}}[m]^2}} \tag{1}$$

We define the root mean square error as follows:

$$\text{RMSE }(u_{\text{orig}}, u_{\text{noisy}}) = \sqrt{\frac{1}{M}\sum_{m=0}^{M}(u_{\text{orig}}[m] - u_{\text{noisy}}[m])^2} \tag{2}$$

Chapter 1

The heat equation

Introduction

The classical Gaussian Smoothing Operator and its implementation for image processing is the object of this online IPOL publication. It is used extensively, either as a simple smoothing preliminary step to increase noise robustness of an algorithm (for instance edge detectors) or either as the fundamental *scale-space* operator.

Being the only operator that satisfies a series of natural requirements (linearity, translation invariance, rotation invariance, scale invariance...) it plays a key role in *scale-space* theory. By simulating a series of blurs, it allows to build a scale invariant representation of the image. This *scale-space* representation is at the very heart of famous registration algorithm such as SIFT, PCA SIFT, ASIFT.

Also the Gaussian is the fundamental solution of the heat equation. This link with the diffusion equation is exploited in many ways. It provide us for instance with a way to approximate the laplacian operator (as in SIFT with the DoG operator.)

The popular implementation technique is to substitute the continuous convolution by a discrete convolution with a truncated discretized Gaussian kernel. Instead of this approximations, one can also apply the continuous smoothing operator to a continuous representation of the signal. We will consider this approach with the *Discrete Fourier Transform* interpolate of the digital image. We will see that it provides a fast yet exact implementation of the Gaussian smoothing operator, only requiring 2 FFTs.

In the following we will first expose the DFT interpolate and then detail how the Gaussian Smoothing Operator should be implemented, exactly, by a simple wheighting of the image DFT coefficients. In the demo accompanying this article, one can study how an important requirement of *scale-space* theory, namely the semi-group property, holds with this exact implementation and with its discrete approximation.

1.1 Notations, Definitions and Conventions

In the following, we will consider the floor function on $\mathbb{R}$ defined as:

$$\forall a \in \mathbb{R}, \lfloor a \rfloor = \max\{n \leq a, n \in \mathbb{Z}\} \tag{1.1}$$

The Fourier Transform

Definition 1. *For every $f \in \mathcal{L}^1(\mathbb{R}^2)$, we define the* Fourier Transform *of f as the function $\hat{f}$, defined for all $(\xi, \eta) \in \mathbb{R}^2$ by*

$$\hat{f}(\xi, \eta) = \int_{(x,y)\in\mathbb{R}^2} f(x, y)e^{-i(x\xi+y\eta)}dxdy$$

The Fourier Transform of the Gaussian function

The Fourier Transform of Gaussian function is a classic result of Fourier Analysis. [**?**].

The Normalized Isotropic Bi-dimensional Gaussian function of standard deviation σ is for all $(x, y) \in \mathbb{R}^2$

$$G_\sigma(x, y) = \frac{1}{2\pi\sigma^2} \exp\left(-\frac{x^2 + y^2}{2\sigma^2}\right)$$

Its Fourier Transform is for all $(\xi, \mu) \in \mathbb{R}^2$ by

$$\hat{G_\sigma}(\xi, \mu) = \exp\left(-\sigma^2\frac{\xi^2 + \mu^2}{2}\right) \tag{1.2}$$

The Discrete Fourier Transform

Here, we will fix integers M and N and consider elements of $\mathbb{C}^{MN}$ noted

$$(u_{k,l})_{\substack{0\leq k\leq M-1\\ 0\leq l\leq N-1}}$$

.

Definition 2. *We define the* Discrete Fourier Transform *(DFT) of vector $(u_{k,l})_{k,l}$ of $\mathbb{C}^{MN}$ as the vector $(\hat{u}_{m,n})_{m,n}$ verifying:*

$$\tilde{u}_{m,n} = \frac{1}{MN}\sum_{k=0}^{M-1}\sum_{l=0}^{N-1} u_{k,l} \exp\left(-\frac{2i\pi mk}{M}\right)\exp\left(-\frac{2i\pi nl}{N}\right) \tag{1.3}$$

for all $m = -\lfloor\frac{M}{2}\rfloor, ..., -\lfloor\frac{M}{2}\rfloor + M - 1$ and $n = -\lfloor\frac{N}{2}\rfloor, ..., -\lfloor\frac{N}{2}\rfloor + N - 1$.

Proposition 1. *If coefficients $u_{k,l}$ are real, then the DFT output vector $(\tilde{u}_{m,n})_{m,n}$ is conjugate symmetric*

$$\hat{u}_{-m,-n} = \hat{u}^*_{m,n} \tag{1.4}$$

for all appropriate indexes m and n.

Definition 3. *We define the* Inverse Discrete Fourier Transform *(noted IDFT) of vector*

$$(\tilde{u}_{m,n})_{\substack{-\lfloor\frac{M}{2}\rfloor\leq m\leq -\lfloor\frac{M}{2}\rfloor+M-1\\ -\lfloor\frac{N}{2}\rfloor\leq n\leq -\lfloor\frac{N}{2}\rfloor+N-1}}$$

as the vector of $\mathbb{C}^{MN}$ $(u_{k,l})_{k,l}$ *verifying:*

$$u_{k,l}=\sum_{m=-\lfloor\frac{M}{2}\rfloor}^{(-\lfloor\frac{M}{2}\rfloor+M-1)}\sum_{n=-\lfloor\frac{N}{2}\rfloor}^{(-\lfloor\frac{N}{2}\rfloor+N-1)}\tilde{u}_{m,n}\exp\left(\frac{2i\pi mk}{M}\right)\exp\left(\frac{2i\pi nl}{N}\right) \tag{1.5}$$

for all $k\in\{0,..,M-1\}$ *and* $l\in\{0,..,N-1\}$.

Proposition 2. *For all vector* $(u_{k,l})_{k,l}$ *with* $k=0,...,M-1$ *and* $l=0,...,N-1$.

$$IDFT(DFT((u_{k,l})_{k,l}))=(u_{k,l})_{k,l} \tag{1.6}$$

For all vector $(\tilde{u}_{m,n})_{m,n}$ *with* $m=-\lfloor\frac{M}{2}\rfloor,...,-\lfloor\frac{M}{2}\rfloor+M-1$ *and* $n=-\lfloor\frac{N}{2}\rfloor,...,-\lfloor\frac{N}{2}\rfloor+N-1$.

$$DFT(IDFT((\tilde{u}_{m,n})_{m,n}))=(\tilde{u}_{m,n})_{m,n} \tag{1.7}$$

1.2 Definition of the Continuous Image

First of all, we need to set the dimensions of the image via dimensionless real positives a and b and the sampling rate in both directions via integers M and N.

The vector

$$(u_{k,l})_{\substack{0\leq k\leq M-1\\ 0\leq l\leq N-1}}$$

of $\mathbb{C}^{MN}$ will then designate the MN samples of a digital image relative to this sampling grid on $\mathbb{R}^2$

$$\left(\frac{ka}{M},\frac{lb}{N}\right)_{\substack{0\leq k\leq M-1\\ 0\leq l\leq N-1}}$$

.

Trigonometric polynomial: We're looking for a trigonometric polynomial P of degrees $\lfloor\frac{M}{2}\rfloor$ and $\lfloor\frac{N}{2}\rfloor$, of periodicities a and b, with MN frequencies, and verifying the exact interpolation property

$$P\left(\frac{ka}{M},\frac{lb}{N}\right)=u_{k,l}$$

for all $l=0,...,M-1$ and for all $k=0,...,N-1$.

Formally this can be expressed as

$$P(x,y)=\sum_{m=-\lfloor\frac{M}{2}\rfloor}^{(-\lfloor\frac{M}{2}\rfloor+M-1)}\sum_{n=-\lfloor\frac{N}{2}\rfloor}^{(-\lfloor\frac{N}{2}\rfloor+N-1)}c_{m,n}\exp\left(\frac{2i\pi mx}{a}\right)\exp\left(\frac{2i\pi ny}{b}\right)$$

where $(c_{m,n})_{m,n}$ designates the Fourier coefficients of polynomial P, i.e

$$c_{m,n}(P) = \frac{1}{ab}\int_{x=0}^{a}\int_{y=0}^{b} P(x,y)\exp\left(-\frac{2i\pi mx}{a}\right)\exp\left(-\frac{2i\pi ny}{b}\right)dxdy$$

In the following we will consider vector $(\tilde{u}_{m,n})_{m,n}$, the DFT of sample vector $(u_{k,l})_{k,l}$. We'll show here that the DFT constitutes an exact representation of the interpolating polynomial.

Proposition 3. *We have the equivalence*

$$P\left(\frac{ka}{M}, \frac{lb}{N}\right) = u_{k,l}$$

for all $k = 0, ..., M-1$ *and for all* $l = 0, ..., N-1$.

$$\Leftrightarrow$$

$$c_{m,n} = \tilde{u}_{m,n}$$

for all $m = -\lfloor \frac{M}{2} \rfloor, ..., -\lfloor \frac{M}{2} \rfloor + M - 1$ *and* $n = -\lfloor \frac{N}{2} \rfloor, ..., -\lfloor \frac{N}{2} \rfloor + N - 1$.

Proof. [⇒] This a direct consequence of property (1.7). □

In other words, interpreting each $u_{k,l}$ for $k = 0, ..., M-1$ and $l = 0, ..., N-1$ as sample values on the grid $G := \left(\frac{ka}{M}, \frac{lb}{N}\right)_{\substack{0\le k\le M-1\\ 0\le l\le N-1}}$, the trigonometric polynomial

$$P(x,y) = \sum_{m=-\lfloor \frac{M}{2} \rfloor}^{\left(-\lfloor \frac{M}{2} \rfloor + M - 1\right)} \sum_{n=-\lfloor \frac{N}{2} \rfloor}^{\left(-\lfloor \frac{N}{2} \rfloor + N - 1\right)} \tilde{u}_{m,n}\exp\left(\frac{2i\pi mx}{a}\right)\exp\left(\frac{2i\pi ny}{b}\right)$$

with

$$(\tilde{u}_{m,n})_{m,n} = \mathrm{DFT}\left((u_{k,l})_{k,l}\right)$$

is the only trigonometric polynomial of degrees $\lfloor \frac{M}{2} \rfloor$ and $\lfloor \frac{N}{2} \rfloor$ and MN frequencies which interpolates the samples $(u_{k,l})_{k,l}$ on the grid G.

With an appropriate parameterization of the $\mathbb{R}^2$ plan, we can set for convenience $a = M$ and $b = N$. Expressing then distances in pixels and setting the MN frequencies of our interpolating polynomial to be in

$$\left(\frac{2\pi m}{M}, \frac{2\pi n}{N}\right)_{\substack{-\lfloor \frac{M}{2} \rfloor \le m \le -\lfloor \frac{M}{2} \rfloor + M - 1\\ -\lfloor \frac{N}{2} \rfloor \le n \le -\lfloor \frac{N}{2} \rfloor + N - 1}}$$

that is to say

$$Sp(P) \subset [-\pi, \pi]$$

Note : We should underline here that this interpretation of DFT coefficients as Fourier Coefficients of a trigonometric polynomial (which is by nature a band limited function) rely heavily on the DFT convention exposed in **??**. Indeeed if we consider coefficients $\hat{u}_{m,n}$ with $m = 0, .., M-1$ and $n = 0, ..., N-1$ there is no more monotone mapping between this indexing and frequencies. Althought it gives the very same set of coefficients, it forbids the interpretation of DFT coeffcients as Fourier coefficients.

To summarize, the DFT interpolate of a digital image introduced here is a continuous band limited representation of the image. We underlined the link between this continuous function and the DFT coefficients vector of the digital image samples. In the following, we will demonstrate that this continuous representation offers an way to apply exactly and simply the *Gaussian Smoothing operator* on images.

1.3 Gaussian Smoothing Operator

Gaussian Convolution of an image

We are interested here in the *Gaussian smoothing operator* on bidimensional continuous signals, which is the convolution operator with a normalized gaussian function : for $u : (x, y) \in \mathbb{R}^2 \mapsto u(x, y) \in \mathbb{R}$

$$G_\sigma u(x, y) := G_\sigma * u(x, y) = \int_{x' \in \mathbb{R}} \int_{y' \in \mathbb{R}} G_\sigma(x', y') u(x - x', y - y') dx' dy' \quad (1.8)$$

The Gaussian operator is parameterized by the standard deviation σ.

$$G_\sigma(x, y) = \frac{1}{2\pi\sigma^2} \exp\left(-\frac{x^2 + y^2}{2\sigma^2}\right)$$

Scale space representation of an image

Gaussian smoothing is central in *scale-space theory*. Gaussian smoothing allows to build a scale invariant representation of the signal by simulating all blurs (i.e all zoom outs of the signal). This multiscale representation of the signal is called *scale space*.

Definition 4. *The* scale space representation *of a continuous image*

$$u : (x, y) \mapsto u(x, y)$$

is the family of images derived from u by Gaussian smoothing

$$(G_\sigma u)_{\sigma > 0}$$

referred in the following, as the 3-d function v

$$\begin{array}{rll} v : & \mathbb{R}^2 \times \mathbb{R}_+ & \to \mathbb{C} \\ & (x, y, \sigma) & \mapsto G_\sigma * u(x, y) \end{array}$$

This is the approach conducted for instance in the SIFT procedure. After building this multiscale representation of the image, extrema of an analysis operator (typically 3-D extrema of Laplacian) are extracted as they constitutes key points of an image.

Scale space axiomatic - Gaussian smoothing properties

The reason for using the Gaussian operator instead of the many other operators have been extensively studied over two decades. This constitutes the field of *Scale-Space Theory*. Here, we will briefly list natural requirements (or axioms) published in many celebrated papers. A more detailed survey of *scale-space* axiomatics can be found in [REF weickert].

Let's note in the following Op_σ the *scale-space* operator applied on images. Le's demote $u(\mathbf{x})$ a continuous image defined for every $\mathbf{x} = (x, y) \in \mathbb{R}^2$. The output image being noted $Op_\sigma(u(\mathbf{x}))$.

- **Linearity with respect to multiplications:** For any real λ,

$$Op_\sigma(\lambda u(\mathbf{x})) = \lambda Op_\sigma(u(\mathbf{x}))$$

  .

- **Translation invariance:** If $T_\mathbf{y} u(\mathbf{x}) := u(\mathbf{x} - \mathbf{y})$ denotes the translation operator, we have

$$Op_\sigma T_\mathbf{y} u(\mathbf{x}) = T_\mathbf{y} Op_\sigma(u(\mathbf{x})$$

- **Scale invariance:** If $H_\lambda u(\mathbf{x}) := u(\lambda \mathbf{x})$ denotes an expansion of u by a factor λ^{-1}, we have

$$Op_\sigma \left[H_\lambda u(\mathbf{x})\right] = H_\lambda \left[Op_{\sigma'} u(\mathbf{x})\right]$$

  with $\sigma' = \lambda\sigma$

- **Rotation invariance:** If $R_\theta u(\mathbf{x}) := u(R_\theta \mathbf{x})$ is the image rotation of an angle $-\theta$, we have.

$$Op_\sigma \left[R_\theta u(\mathbf{x})\right] = R_\theta \left[Op_\sigma u(\mathbf{x})\right]$$

- **Semi group property:** Considering 2 smoothing scales σ_1 and σ_2 we have the relation:

$$Op_{\sigma_2} \left[Op_{\sigma_1} u(\mathbf{x})\right] = Op_{\sqrt{\sigma_1^2 + \sigma_2^2}} u(\mathbf{x})$$

The combination of these natural requirement makes the Gaussian smoothing the only valid local smoothing operator on images.

$$Op_\sigma u(\mathbf{x}) = G_\sigma * u(\mathbf{x})$$

with scale parameter σ beeing the standard deviation of the Gaussian kernel.

Furthermore, the Gaussian operator is the fundamental solution of the heat equation. More precisely:

Proposition 4. *The Gaussian convolution $G_\sigma * u_0(\mathbf{x})$ of image $u_0(\mathbf{x})$ is the unique solution to the diffusion equation with initial value u_0*

$$\begin{cases} \dfrac{\partial v}{\partial \sigma}(x,y,\sigma) = 2\sigma\Delta(x,y,\sigma), (\sigma > 0) \\ v(x,y,0) = u_0(x,y) \end{cases} \tag{1.9}$$

This link between Gaussian smoothing and the Laplacian operator is used extensively in image processing. Indeed the laplacian differential operator can be approximated by a *Difference of Gaussian* as in SIFT scale space or in Burt and Adelson's edge detector.

1.4 ALGORITHM PRINCIPLE : Exact Smoothing of the DFT Interpolation

Considering a digital image of size M and N

$$(u_{k,l})_{\substack{0\leq k\leq M-1 \\ 0\leq l\leq N-1}}$$

and its DFT interpolate

$$u(x,y) = \sum_{m=-\lfloor \frac{M}{2} \rfloor}^{(-\lfloor \frac{M}{2} \rfloor + M - 1)} \sum_{n=-\lfloor \frac{N}{2} \rfloor}^{(-\lfloor \frac{N}{2} \rfloor + N - 1)} \tilde{u}_{m,n} \exp\left(\frac{2i\pi m x}{M}\right) \exp\left(\frac{2i\pi n y}{N}\right)$$

(with $(\tilde{u}_{m,n})_{m,n} = \mathrm{DFT}\left((u_{k,l})_{k,l}\right)$.

The Exact Gaussian Smoothing algorithm rely on a very simple property:

Proposition 5. *Considering a pure wave of frequence ξ*

$$\begin{array}{rccc} g_\xi : & \mathbb{R}^N & \to \mathbb{R} \\ & x & \mapsto \exp(ix\xi) \end{array}$$

and a convolution kernel $f : \mathbb{R}^N \to \mathbb{R}$ we have

$$\forall x \in \mathbb{R}^N, f * g_\xi(x) = \hat{f}(\xi) g_\xi(x)$$

Proof.

$$\begin{aligned} f * g_\xi(x) &= \int_{x' \in \mathbb{R}^N} f(x') g_\xi(x - x') dx' \\ &= \int_{x' \in \mathbb{R}^N} f(x') \exp(-i\xi\dot{x}') dx' g_\xi(x) \\ &= \hat{f}(\xi) g_\xi(x) \end{aligned}$$

□

As a direct consequence, the Gaussian convolution of a trigonometric polynomial is still a trigonometric polynomial, with same frequencies and weighted Fourier coefficients.

In particular, convolving the gaussian kernel with the image DFT interpolate is equivalent to weighting the digital image's DFT coefficients. Formally

$$\begin{aligned} u(x,y) &= \sum_{m=-\lfloor \frac{M}{2} \rfloor}^{(-\lfloor \frac{M}{2} \rfloor + M - 1)} \sum_{n=-\lfloor \frac{N}{2} \rfloor}^{(-\lfloor \frac{N}{2} \rfloor + N - 1)} \tilde{u}_{m,n} \exp\left(2i\pi \left(\frac{mx}{M} + \frac{ny}{N}\right)\right) \\ v(x,y,\sigma) &= \sum_{m=-\lfloor \frac{M}{2} \rfloor}^{(-\lfloor \frac{M}{2} \rfloor + M - 1)} \sum_{n=-\lfloor \frac{N}{2} \rfloor}^{(-\lfloor \frac{N}{2} \rfloor + N - 1)} \left[\hat{G}_\sigma(\frac{2\pi m}{M}, \frac{2\pi n}{N}) \tilde{u}_{m,n}\right] \exp\left(2i\pi \left(\frac{mx}{M} + \frac{ny}{N}\right)\right) \end{aligned}$$

where each weighting factor $\hat{G}_\sigma(\frac{2\pi m}{M}, \frac{2\pi n}{N})$ is the Continuous Fourier Tranform of the Gaussian Function evaluated on the frequency $(\frac{2\pi m}{M}, \frac{2\pi n}{N})$,

$$\hat{G}_\sigma(\frac{2\pi m}{M}, \frac{2\pi n}{N}) = \exp\left(-\frac{\sigma^2}{2}\left(\left(\frac{2\pi m}{M}\right)^2 + \left(\frac{2\pi n}{N}\right)^2\right)\right)$$

.

Note 1: It is easy to check that those new DFT coefficients correspond to a real valued signal. Indeed, this weighting of the DFT coefficients doesn't affect any of its conjugate symmetry properties. Therefore the inverse Discrete Fourier transform of those coeffcients leads indeed to a real valued signal.

Note 2: Extension of the signal: DFT interpolate is defined on the entire plan $\mathbb{R}^2$. It extends the image to the entire plan via periodization of the signal. Another way to extend the signal to the entire plan is to symmetrize the signal before periodization. This technique will be discussed in 1.5

ALGORITHM IMPLEMENTATION : Exact Gaussian Smoothing of DFT interpolate

- **Smoothing Parameter σ, standard deviation of the Gaussian smoothing kernel.**

- **Input monochromatic image of height M and widht N:**

$$U = [u_{k,l}]_{\substack{0 \leq k \leq M-1 \\ 0 \leq l \leq N-1}}$$

- **Discrete Fourier Transform of the input image ($M \times N$ complex coefficients):**

$$\hat{U} = DFT(U) = [\hat{u}_{m,n}]_{\substack{-\lfloor \frac{M}{2} \rfloor \leq m \leq -\lfloor \frac{M}{2} \rfloor + M - 1 \\ -\lfloor \frac{N}{2} \rfloor \leq n \leq -\lfloor \frac{N}{2} \rfloor + N - 1}}$$

- **Samples of the Continuous Fourier Transform of the Gaussian kernel G_σ:**

$$\hat{G} = \left[\exp\left(-\frac{\sigma^2}{2} \left(\left(\frac{2\pi m}{M} \right)^2 + \left(\frac{2\pi n}{N} \right)^2 \right) \right) \right]_{\substack{-\lfloor \frac{M}{2} \rfloor \leq m \leq -\lfloor \frac{M}{2} \rfloor + M - 1 \\ -\lfloor \frac{N}{2} \rfloor \leq n \leq -\lfloor \frac{N}{2} \rfloor + N - 1}}$$

- **Point-Wise multiplication in the Fourier domain:**

$$\hat{V} = \hat{G}.\hat{U} = \left[\hat{u}_{m,n} . \exp\left(-\frac{\sigma^2}{2} \left(\left(\frac{2\pi m}{M} \right)^2 + \left(\frac{2\pi n}{N} \right)^2 \right) \right) \right]_{\substack{-\lfloor \frac{M}{2} \rfloor \leq m \leq -\lfloor \frac{M}{2} \rfloor + M - 1 \\ -\lfloor \frac{N}{2} \rfloor \leq n \leq -\lfloor \frac{N}{2} \rfloor + N - 1}}$$

- **Inverse Discrete Fourier Transform to get the $M \times N$ real samples of the output smoothed image:**

$$V = IDFT(\hat{V}) = [v_{k,l}]_{\substack{0 \leq k \leq M-1 \\ 0 \leq l \leq N-1}}$$

Implementation note 1: FFTW library computes the $M \times N$ coefficients $\hat{u}_{m,n}$ with $m = 0, ..., M-1$ and $n = 0, ..., N-1$. Therefore a conversion is required to get the set of DFT coefficients computed with $m = -\lfloor \frac{M}{2} \rfloor, ..., -\lfloor \frac{M}{2} \rfloor + M - 1$, $n = -\lfloor \frac{N}{2} \rfloor, ..., -\lfloor \frac{N}{2} \rfloor + N - 1$ and intepretable as Fourier coefficients.

Implementation note 2: FFTW library takes advantage of the nature of the input signal. The bi-dimensional signal being real, the conjugate symmetry of its DFT coefficients allows to store and compute only $M \times (\lfloor \frac{N}{2} \rfloor + 1)$ complex coefficients. (see FFTW online documentation `http://www.fftw.org/fftw3_doc/Multi_002dDimensional-DFTs-of-Real-Data.html#Multi_002dDimensional-DFTs-of-Real-Data`).

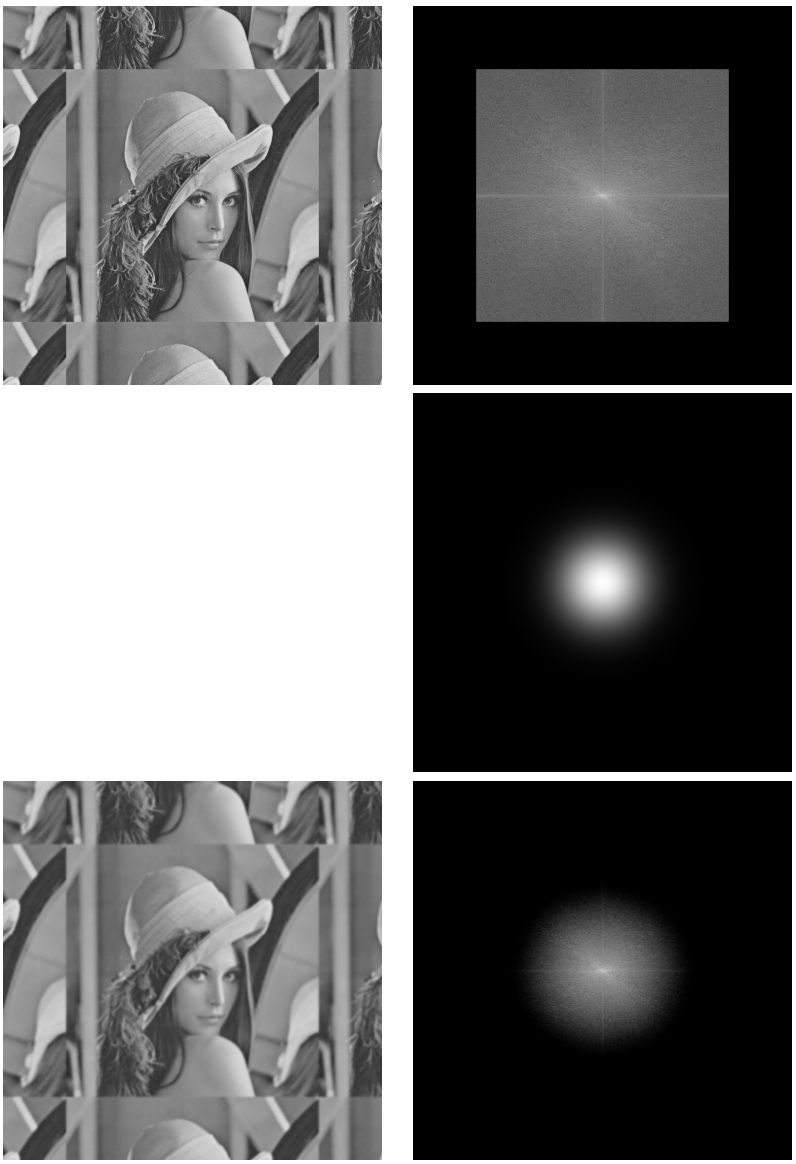

Figure 1.1: Exact Gaussian Smoothing Algorithm. The DFT interpolate considered here is a trigonometric polynomial, it follows then that the image is implicitly extended to $\mathbb{Z}^2$ via periodization and that its representation in Fourier is a finite sum of $M \times N$ Dirac masses. (note also that borders in the figure with Fourier kernel correspond to frequencies $-\pi$ and π, **note** I feel that it's important to mention this to explain why the kernel figure is not isotropic in the case of a non squared image)

1.5 Exact Smoothing on the DFT Interpolation of a symmetrized discrete signal

Whenever we manipulate the DFT interpolate of a digital image, we implicitly extend it to the whole $\mathbb{Z}^2$ plane via periodization, eventually producing strong discontinuities at borders. Consequently when applying the exact Gaussian smoothing operator, discontinuities at borders spread, each border "diffusing" itself on the opposite side of the image.

A popular trick to avoid this edge effect is to switch to a less unatural boundary condition by first symmetrizing the $M \times N$ original image to a $2M \times 2N$ and then taking its DFT interpolate.

AlGORITHM: Exact Gaussian Smoothing on DFT interpolate with mirror symmetrization of the signal

- **Smoothing Parameter σ, standard deviation of the Gaussian smoothing kernel.**
- **Input monochromatic image of height M and widht N:**

$$U = [u_{k,l}]_{\substack{0 \le k \le M-1 \\ 0 \le l \le N-1}}$$

- **Mirror symmetrization resulting in an image of size $2M \times 2N$:**

$$U^M = \left[u^M_{k,l}\right]_{\substack{0 \le k \le 2M-1 \\ 0 \le l \le 2N-1}}$$

where $u^M_{k,l} = u_{k',l'}$ with

$$k' = \begin{cases} k & \textbf{if} \quad 0 \le k \le M-1 \\ 2M-1-k & \textbf{if} \quad M \le k \le 2M-1 \end{cases}$$

and

$$l' = \begin{cases} l & \textbf{if} \quad 0 \le l \le N-1 \\ 2N-1-l & \textbf{if} \quad N \le l \le 2N-1 \end{cases}$$

- **Discrete Fourier Transform of the symmetrized input image ($2M \times 2N$ complex coefficients):**

$$\hat{U}^M = DFT(U^M) = \left[\hat{u}^M_{m,n}\right]_{\substack{-M \le m \le M-1 \\ -N \le n \le N-1}}$$

- **Samples of the Continuous Fourier Transform of the Gaussian kernel G_σ:**

$$\hat{G} = \left[\exp\left(-\frac{\sigma^2}{2}\left(\left(\frac{2\pi m}{M}\right)^2 + \left(\frac{2\pi n}{N}\right)^2\right)\right)\right]_{\substack{-M \le m \le M-1 \\ -N \le n \le N-1}}$$

- **Point-Wise multiplication in the Fourier domain:**

$$\hat{V}^M = \hat{G}.\hat{U}^M = \left[\hat{u}^M_{m,n}.\exp\left(-\frac{\sigma^2}{2}\left(\left(\frac{2\pi m}{M}\right)^2 + \left(\frac{2\pi n}{N}\right)^2\right)\right)\right]_{\substack{-M\leq m\leq M-1\\ -N\leq n\leq N-1}}$$

- **Inverse Discrete Fourier Transform to get the $M\times N$ real samples of the mirrored smoothed image:**

$$V^M = IDFT(\hat{V}^M) = \left[v^M_{k,l}\right]_{\substack{0\leq k\leq 2M-1\\ 0\leq l\leq 2N-1}}$$

- **Extraction of the up-left quarter of the mirrored image:**

$$V = [v_{k,l}]_{\substack{0\leq k\leq M-1\\ 0\leq l\leq N-1}}$$

with $v_{k,l} = v^M_{k,l}$ for $k = 0, ..., M-1$ and $l = 0, ..., N-1$.

1.6 Alternative implementations - Discrete Convolution

The Continuous Gaussian convolution is usually approximated by a discrete convolution: Instead of considering a continuous representation of the image, we convolve the discrete signal

$$(u_{k,l})_{\substack{0\leq k\leq M-1\\ 0\leq l\leq N-1}}$$

by a sampled and truncated Gaussian kernel

$$(G_\sigma(k,l))_{(k,l)\in\mathbb{Z}^2}$$

That is to say that a pixel at a distance of more than $K\sigma$ is considered zero.

$$(G_\sigma * u)_{k,l} := \sum_{k'=-\lfloor K\sigma\rfloor}^{\lfloor K\sigma\rfloor}\sum_{l'=-\lfloor K\sigma\rfloor}^{\lfloor K\sigma\rfloor} G_\sigma(k-k', l-l')u_{k',l'} \tag{1.10}$$

Usually truncation of the Gaussian kernel is performed at a distance of 3σ (4σ in SIFT implementation) values of the Gaussian kernel are there small enough to constitute a good approximation.

Yet the sum of discrete values of the Gaussian kernel is slightly different to 1. This can lead to an important loss in energy while convolving the image. Therefore the truncated Gaussian kernel is usually normalized by the sum of all its non-zero samples.

Another discrete convolution implementation have been proposed in [Lindeberg]. It is the fundamental solution of a discrete analogue of the diffusion equation. The non-creation of structure in the discrete *scale-space* representation is the central requirement that motivates this approach.

1.7 DEMO - Checking the Semi-Group Property

`http://dev.ipol.im/~reyotero/ipol_demo/rorm_the_heat_equation2/`.

In this demo, the reader can check if the exact implementation of the Gaussian smoothing operator verify the semi-group property. Let's remind that the semi-group property is required for *scale-space* representation of the image. It states that for two smoothing parameter σ_1 and σ_2 we have

$$G_{\sigma_1} G_{\sigma_2} = G_{\sqrt{\sigma_1^2+\sigma_1^2}} \tag{1.11}$$

The demo applies iteratively the Gaussian smoothing operator with scale parameter σ_1 and σ_2 to an image and compare it to the equivalent ''direct' convolution with scale parameter $\sqrt{\sigma_1^2 + \sigma_2^2}$. The exact implementation of the Gaussian smoothing is compared to the discrete convolution with extra parameter k defining the number of samples considered in the discrete kernel.

It appears that the two ways lead to the exact same result (up to numerical noise) in accordance with the Semi-Group Property.

The same experiment conducted this time with discrete approximation of the Gaussian convolution shows that this important axiomatic property doesn't hold with the classic approximated implementation.

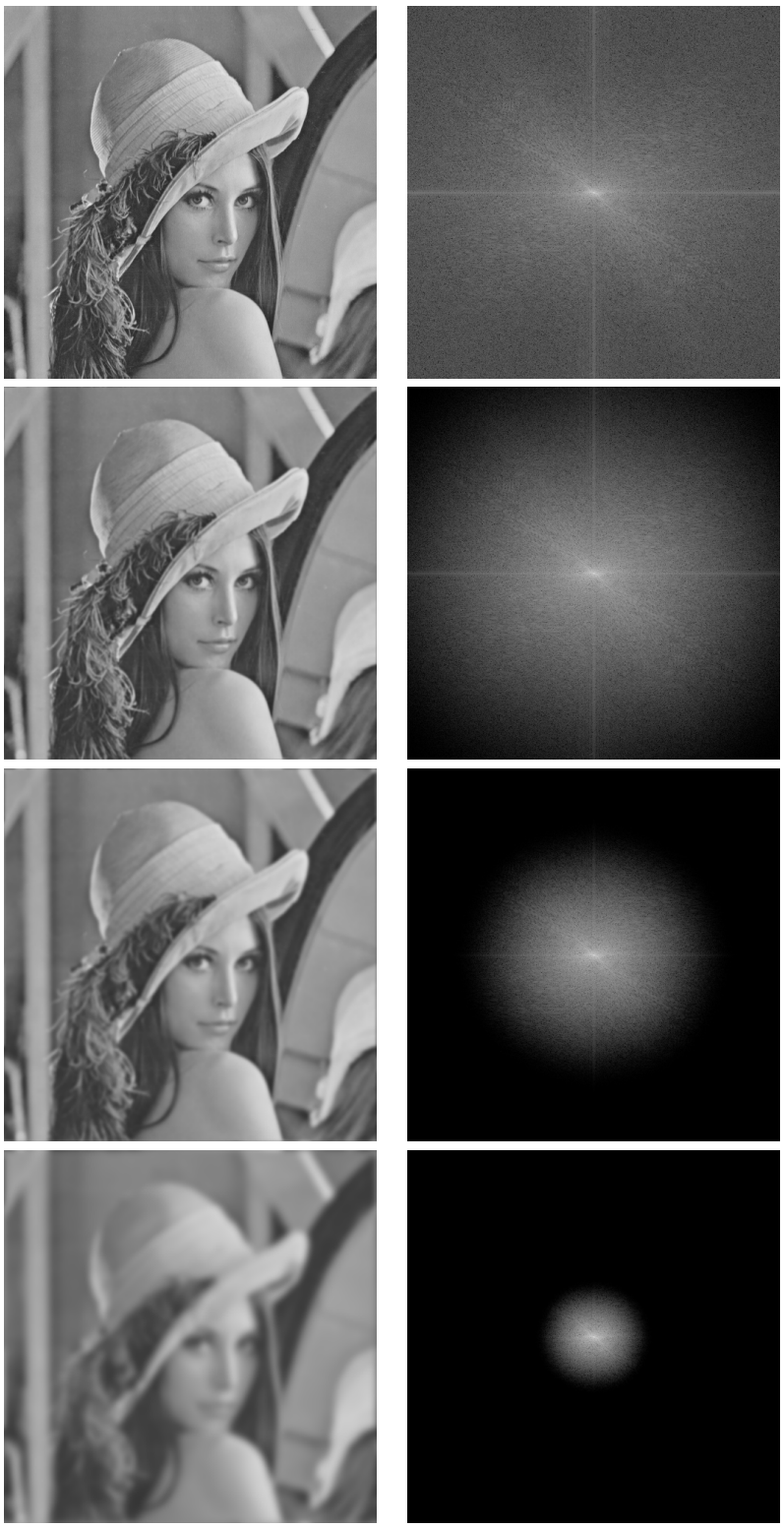

Figure 1.2: Initial image lena with successive smoothing, $\sigma \in \{1.0, 2.0, 5.0\}$ along the corresponding attenuation of high frequencies in the Fourier domain. With important smoothing we notice easily the border discontinuities produced by simple periodization

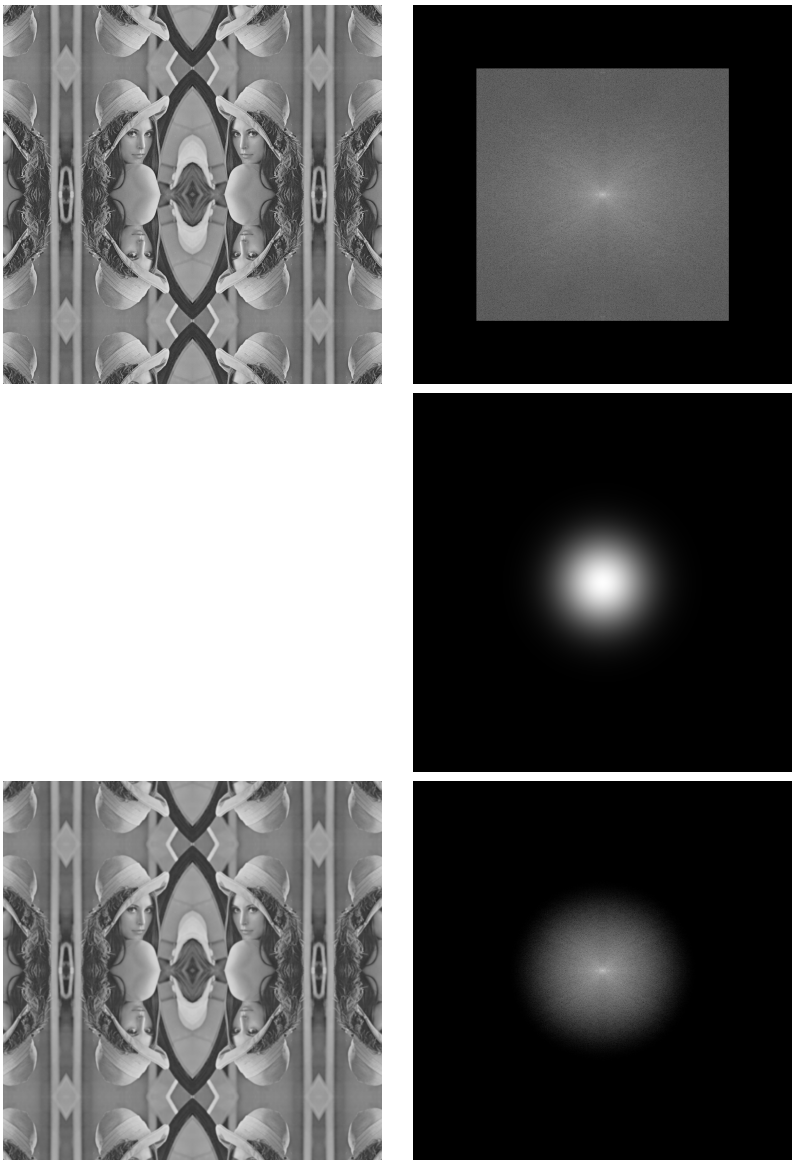

Figure 1.3: Exact Gaussian Smoothing Algorithm with symmetrization. Again the DCT interpolate considered here is a trigonometric polynomial, it follows then that the image is implicitly extended to $\mathbb{Z}^2$ via symmetrization and periodization and that its representation in Fourier is a sum of $2M \times 2N$ Dirac masses.

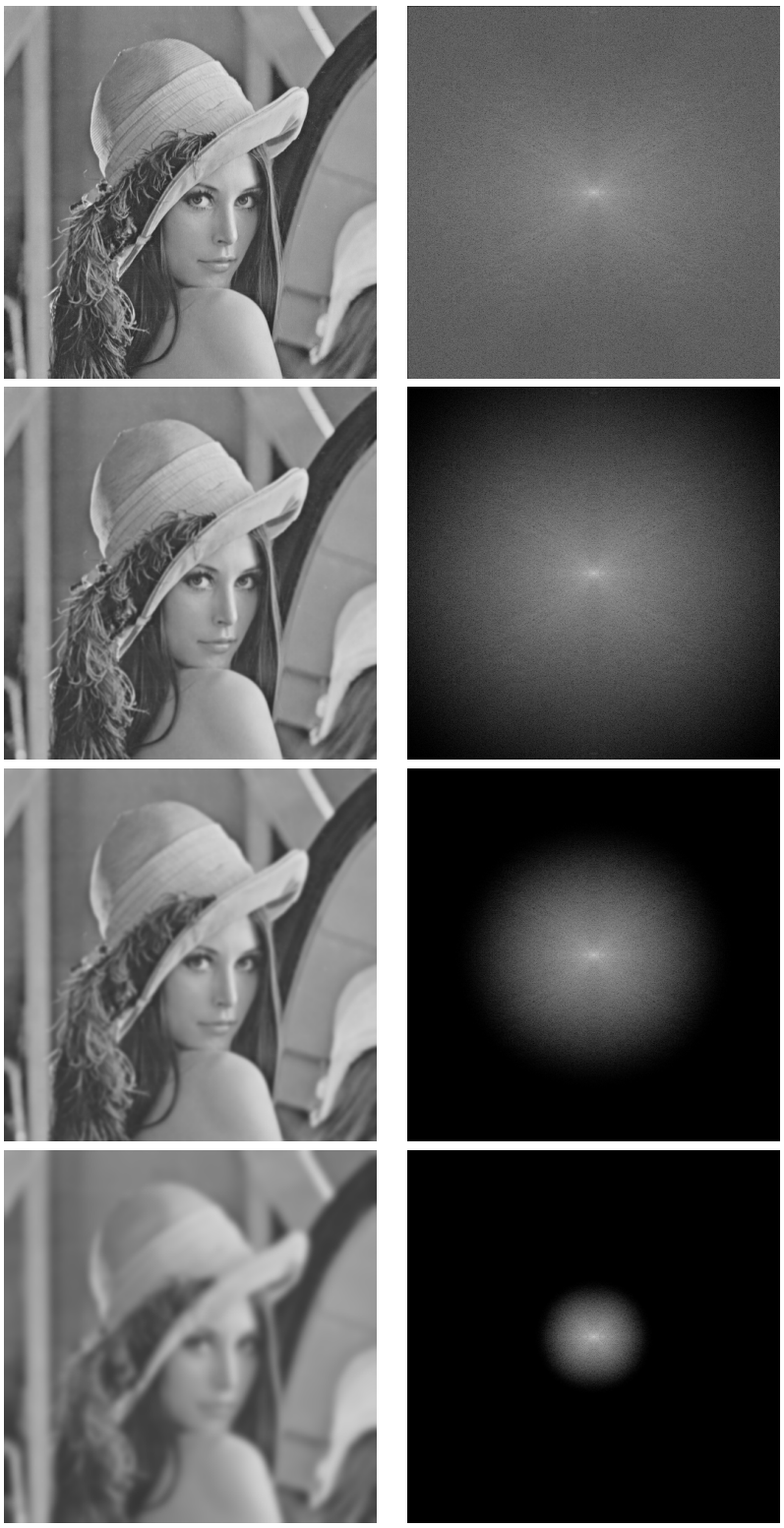

Figure 1.4: Initial image lena with successive smoothing, $\sigma \in \{1.0, 2.0, 5.0\}$ along the corresponding attenuation of high frequencies on the Discrete Cosine Transform. Observe the absence of periodization artefact on each border

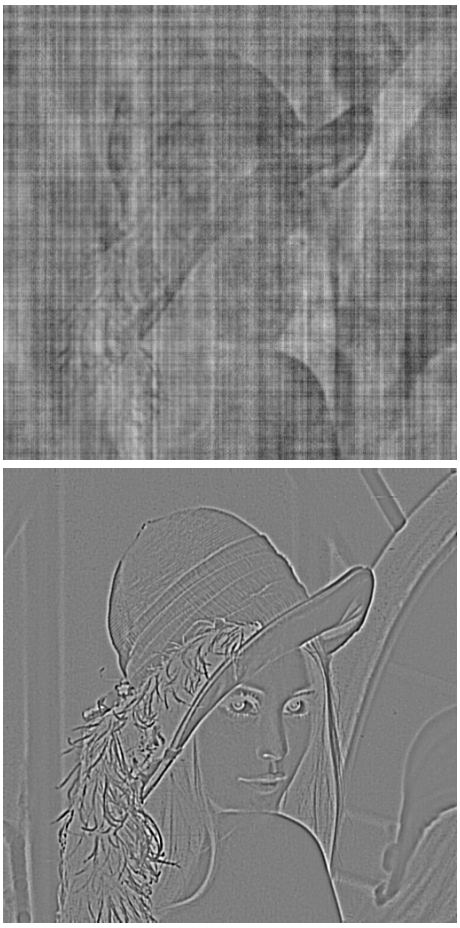

Figure 1.5: Difference between direct and iterated smoothings. The mean and standard deviation of each difference image are set respectively to 128 and 20 for better visualization. Parameters are $\sigma = 1.7$, $N = 10$, $width = 4$ (as in SIFT), the values on the difference image are below the double precision: with an image dynamic of $0 - 255$ RMSE(exact)$= 9, 0.10^{-14}$ while RMSE(discrete)$= 1.33$.