

# Introduction to Computer Vision

## Lab 2: Frequency Domain Filtering & Hybrid Images

ESIN, UIR — Spring 2026

Ilias TOUGUI

## Contents

<b>1</b>	<b>Introduction</b>	<b>2</b>
<b>2</b>	<b>The Convolution Theorem</b>	<b>3</b>
2.1	2D Convolution: A Reminder . . . . .	3
2.2	The 2D Fourier Transform . . . . .	3
2.3	The Convolution Theorem . . . . .	3
<b>3</b>	<b>Frequency Domain Filtering</b>	<b>4</b>
3.1	Reading the Magnitude Spectrum . . . . .	4
3.2	Ideal Filters . . . . .	4
3.2.1	Ideal Low-Pass Filter (ILPF) . . . . .	4
3.2.2	Ideal High-Pass Filter (IHPF) . . . . .	5
3.2.3	The Gibbs Phenomenon . . . . .	5
3.3	Gaussian Filters . . . . .	5
3.3.1	Gaussian Low-Pass Filter . . . . .	6
3.3.2	Gaussian High-Pass Filter . . . . .	6
3.3.3	The Uncertainty Principle . . . . .	6
<b>4</b>	<b>Hybrid Images</b>	<b>7</b>
4.1	Concept . . . . .	7
4.2	Why it Works: Human Visual Perception . . . . .	7
4.3	Requirements for a Good Hybrid Image . . . . .	7
4.4	Parameter Selection . . . . .	8
<b>5</b>	<b>Lab Instructions</b>	<b>9</b>
5.1	.ipynb Notebook . . . . .	9
5.2	Deliverables . . . . .	9
5.3	How to Prepare Your Submission . . . . .	9

# 1 Introduction

In the previous labs, we studied 1D convolution and computed it by hand. In this lab, we move to **2D images** and ask a fundamental question:

*Is there a faster way to compute convolution than sliding a kernel over every pixel?*

The answer is **yes** — and it comes from a remarkable mathematical result called the **Convolution Theorem**. This theorem connects two seemingly different operations: convolution in the *spatial domain* and multiplication in the *frequency domain*.

By the end of this lab, you will be able to:

- State and apply the 2D Convolution Theorem.
- Explain the computational advantage of FFT-based filtering.
- Design and apply ideal and Gaussian low-pass and high-pass filters.
- Explain the Gibbs Phenomenon and why Gaussian filters avoid it.
- Create hybrid images by combining frequency components of two images.

## 2 The Convolution Theorem

### 2.1 2D Convolution: A Reminder

Given an image  $f(x, y)$  and a filter kernel  $h(x, y)$ , their **2D convolution** is:

$$(f * h)(x, y) = \sum_m \sum_n f(m, n) h(x - m, y - n) \quad (1)$$

For an image of size  $N \times N$  and a kernel of size  $K \times K$ , computing equation (1) directly requires  $\mathcal{O}(N^2 K^2)$  multiplications. For a  $1000 \times 1000$  image and a  $101 \times 101$  kernel, that is over **10 billion operations**.

### 2.2 The 2D Fourier Transform

The **2D Discrete Fourier Transform (DFT)** of an image  $f(x, y)$  of size  $M \times N$  is:

$$F(u, v) = \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} f(x, y) e^{-j2\pi(\frac{ux}{M} + \frac{vy}{N})} \quad (2)$$

where  $u$  and  $v$  are the **spatial frequencies** in the horizontal and vertical directions respectively. The inverse transform recovers  $f(x, y)$ :

$$f(x, y) = \frac{1}{MN} \sum_{u=0}^{M-1} \sum_{v=0}^{N-1} F(u, v) e^{j2\pi(\frac{ux}{M} + \frac{vy}{N})} \quad (3)$$

The **Fast Fourier Transform (FFT)** computes equation (2) in only  $\mathcal{O}(N^2 \log N)$  operations — a dramatic improvement over the naive  $\mathcal{O}(N^4)$  direct computation.

### 2.3 The Convolution Theorem

#### Convolution Theorem (2D)

Let  $f(x, y)$  and  $h(x, y)$  be two 2D signals with Fourier transforms  $F(u, v)$  and  $H(u, v)$  respectively. Then:

$$f(x, y) * h(x, y) \longleftrightarrow F(u, v) \cdot H(u, v)$$

That is, **convolution in the spatial domain is equivalent to pointwise multiplication in the frequency domain**.

This means that instead of sliding the kernel over every pixel, we can:

1. Compute  $F = \mathcal{F}\{f\}$  and  $H = \mathcal{F}\{h\}$  (two FFTs)
2. Multiply pointwise:  $G = F \cdot H$  ( $\mathcal{O}(N^2)$ )
3. Compute  $g = \mathcal{F}^{-1}\{G\}$  (one inverse FFT)

The total cost is  $\mathcal{O}(N^2 \log N)$ , **independent of the kernel size  $K$** .

### Computational Comparison

Method	Complexity	Depends on kernel size?
Spatial convolution	$\mathcal{O}(N^2 K^2)$	Yes
FFT-based convolution	$\mathcal{O}(N^2 \log N)$	No

For large kernels ( $K \gg 1$ ), FFT convolution is significantly faster.

## 3 Frequency Domain Filtering

### 3.1 Reading the Magnitude Spectrum

The magnitude spectrum  $|F(u, v)|$  tells us which frequencies are present in an image and how strong they are. After shifting the zero-frequency DC component to the center, we can interpret it as follows:

- **Center** (low frequencies): slow spatial variations — smooth regions, overall shapes, background.
- **Outer edges** (high frequencies): rapid spatial variations — sharp edges, fine textures, noise.

Most of the energy in natural images is concentrated in the low frequencies (the center of the spectrum), which is why a heavily blurred image still looks recognizable.

### 3.2 Ideal Filters

#### 3.2.1 Ideal Low-Pass Filter (ILPF)

An Ideal Low-Pass Filter keeps all frequencies within a circle of radius  $r_c$  (the *cutoff frequency*) and completely zeroes out everything outside:

$$H_{LP}(u, v) = \begin{cases} 1 & \text{if } \sqrt{u^2 + v^2} \leq r_c \\ 0 & \text{otherwise} \end{cases} \quad (4)$$

**Effect:** Removing high frequencies blurs the image — only the smooth, slowly-varying parts survive.

### 3.2.2 Ideal High-Pass Filter (IHPF)

An Ideal High-Pass Filter is the complement of the ILPF:

$$H_{\text{HP}}(u, v) = 1 - H_{\text{LP}}(u, v) = \begin{cases} 0 & \text{if } \sqrt{u^2 + v^2} \leq r_c \\ 1 & \text{otherwise} \end{cases} \quad (5)$$

**Effect:** Removing low frequencies leaves only sharp edges and fine texture details.

### 3.2.3 The Gibbs Phenomenon

#### Ringing Artifacts

The ideal circular mask creates a **sharp, abrupt cut** in the frequency domain. This hard edge is mathematically equivalent to a sinc-like function in the spatial domain — one that oscillates and never fully dies out. When convolved with the image, these oscillations appear as visible **ripple rings around sharp edges**, known as the **Gibbs Phenomenon**.

Think of it like hitting a table sharply: the surface vibrates before settling. A Gaussian filter is a gentle press — no sudden shock, no ringing.

Filter	Frequency domain	Spatial domain
Ideal LP/HP	Sharp circular edge	Oscillating sinc → <b>ringing</b>
Gaussian LP/HP	Smooth Gaussian taper	Gaussian → <b>no ringing</b>

**The key takeaway:** any filter with a *sharp, abrupt boundary* in the frequency domain will produce ringing in the spatial domain. The Gaussian avoids this entirely because it has *no sharp boundary* — it fades out gradually in both domains.

## 3.3 Gaussian Filters

The solution to ringing is to use a filter that **gradually tapers** to zero instead of cutting off abruptly. The most natural choice is the **Gaussian filter**.

### 3.3.1 Gaussian Low-Pass Filter

A 2D Gaussian kernel with standard deviation  $\sigma$  is:

$$h_\sigma(x, y) = \frac{1}{2\pi\sigma^2} \exp\left(-\frac{x^2 + y^2}{2\sigma^2}\right) \quad (6)$$

Its Fourier Transform is also a Gaussian:

$$H_\sigma(u, v) = \exp\left(-\frac{2\pi^2\sigma^2(u^2 + v^2)}{N^2}\right) \quad (7)$$

#### The Gaussian is Self-Dual

The Fourier Transform of a Gaussian is a Gaussian. This unique property means the filter is **smooth in both domains simultaneously** — no sharp transitions, no ringing artifacts.

### 3.3.2 Gaussian High-Pass Filter

A Gaussian high-pass filter is defined as:

$$\text{HP}(f) = f - \text{LP}_\sigma(f) \quad (8)$$

That is, we subtract the blurred version from the original. This naturally leaves only the high-frequency components (edges and textures).

### 3.3.3 The Uncertainty Principle

There is a fundamental trade-off between spatial and frequency resolution:

#### Heisenberg Uncertainty Principle (Signal Processing)

A signal cannot be simultaneously compact in both the spatial and frequency domains.  
Formally:

$$\Delta x \cdot \Delta u \geq \frac{1}{4\pi}$$

For Gaussian filters this means:

- **Large  $\sigma$ :** wide kernel in space  $\rightarrow$  narrow Gaussian in frequency  $\rightarrow$  **more blur** (fewer frequencies pass)
- **Small  $\sigma$ :** narrow kernel in space  $\rightarrow$  wide Gaussian in frequency  $\rightarrow$  **less blur** (more frequencies pass)

## 4 Hybrid Images

### 4.1 Concept

Hybrid images were introduced by Oliva, Torralba, and Schyns in their **SIGGRAPH 2006** paper “*Hybrid Images*”. The key observation is that human visual perception behaves like a low-pass filter at large viewing distances: fine details (high frequencies) are lost, and only the overall shape (low frequencies) remains visible.

#### Hybrid Image

A hybrid image  $H$  is formed by combining:

- The **low-pass** filtered version of image  $A$  (perceived from far)
- The **high-pass** filtered version of image  $B$  (perceived up close)

$$H = \text{LP}_{\sigma_1}(A) + \alpha \cdot \text{HP}_{\sigma_2}(B)$$

where  $\alpha \in [0.5, 1.5]$  controls the blending weight of the high-frequency component.

### 4.2 Why it Works: Human Visual Perception

When viewing a hybrid image:

- **Up close:** the visual system resolves fine details. The high-frequency component of  $B$  dominates — you see image  $B$ .
- **From far away:** the visual system acts as a low-pass filter, averaging out fine details. Only the low-frequency component of  $A$  survives — you see image  $A$ .

### 4.3 Requirements for a Good Hybrid Image

For the illusion to work well, the two images must satisfy:

1. **Spatial alignment:** Key features (eyes, nose, dominant shapes) must be roughly aligned between the two images. Misalignment breaks the perceptual grouping.
2. **Similar framing:** Both images should have the same general composition (e.g., both are close-up portraits, or both are full-body shots). A face and a landscape rarely combine well.
3. **Complementary frequency content:** Image  $A$  (low-pass) should be clearly recognizable even when heavily blurred. Image  $B$  (high-pass) should have strong, distinctive high-frequency content (edges, textures).

4. **Similar luminance:** The two images should have comparable brightness and contrast. If one image is much darker, it will be overpowered by the other. Histogram matching can correct this.

## 4.4 Parameter Selection

The quality of the hybrid depends critically on three parameters:

Parameter	Effect	Recommendation
$\sigma_{\text{low}}$ (far image blur)	Larger $\rightarrow$ more aggressive blur	$\sigma \in [5, 15]$
$\sigma_{\text{high}}$ (close image edges)	Smaller $\rightarrow$ sharper edges	$\sigma \in [2, 8]$
$\alpha$ (HP weight)	Higher $\rightarrow$ stronger close image	$\alpha \in [0.7, 1.2]$

### Quick Test for a Good Image Pair

Before combining two images, apply a strong blur ( $\sigma = 15$ ) to both:

- Can you still recognize image  $A$  after blurring?  $\rightarrow$  **Good low-freq source.**
- Does image  $B$  become unrecognizable after blurring?  $\rightarrow$  **Good high-freq source.**

If both conditions hold, the pair will produce a convincing hybrid.

## 5 Lab Instructions

### 5.1 .ipynb Notebook

Open the provided notebook (better on Colab or Kaggle). It is organized into three parts:

#### 1. Part 1 — Convolution Theorem in Practice

You will benchmark spatial convolution vs. FFT-based convolution using `scipy.signal` and observe the computational difference.

#### 2. Part 2 — Frequency Domain Filtering

You will use provided functions to apply ideal and Gaussian low-pass and high-pass filters to images, visualize the spectrum overlays, and compare the ringing artifacts.

#### 3. Part 3 — Hybrid Images

Complete the provided code to implement the hybrid images pipeline yourself, namely the (`hybrid_image_color` and `visualize_hybrid_color`) functions.

### 5.2 Deliverables

Submit a single merged PDF file on Connect UIR before the deadline, containing the following two documents **in this order**:

1. **Completed notebook** exported as a .pdf with all cells executed and outputs visible, including your implementation of hybrid images in color.
2. **Short report** (.docx exported as PDF, maximum 2 pages) containing:
  - Your name, Group, and Student ID
  - Answers to all report questions
  - A screenshot of your best hybrid image (your own image pair)

### 5.3 How to Prepare Your Submission

Follow these two steps in order:

#### 1. Export your notebook to PDF

Run all cells (`Runtime → Run all`), then download your notebook (`File → Download → Download .ipynb`) and convert it using:

<https://www.vertopal.com/en/convert/ipynb-to-pdf>

## 2. Merge both PDFs into one file

Go to the following website, upload your notebook PDF first and your report PDF second, then download the merged file:

[https://www.ilovepdf.com/merge\\_pdf](https://www.ilovepdf.com/merge_pdf)

### Submission Rules

- Submit **one single merged PDF** — not two separate files.
- Make sure all notebook cells have visible outputs before exporting.
- **Do not send your submission by email. (I won't correct it)**

## References

1. Oliva, A., Torralba, A., & Schyns, P.G. (2006). *Hybrid Images*. ACM SIGGRAPH 2006.
2. Gonzalez, R.C., & Woods, R.E. (2018). *Digital Image Processing* (4th ed.). Pearson. Chapters 4 (Frequency Domain) & 5 (Filtering).
3. Oppenheim, A.V., & Schafer, R.W. (2010). *Discrete-Time Signal Processing* (3rd ed.). Pearson.
4. NumPy Documentation: `numpy.fft.fft2`  
<https://numpy.org/doc/stable/reference/routines.fft.html>
5. SciPy Documentation: `scipy.signal.fftconvolve`  
<https://docs.scipy.org/doc/scipy/reference/generated/scipy.signal.fftconvolve.html>