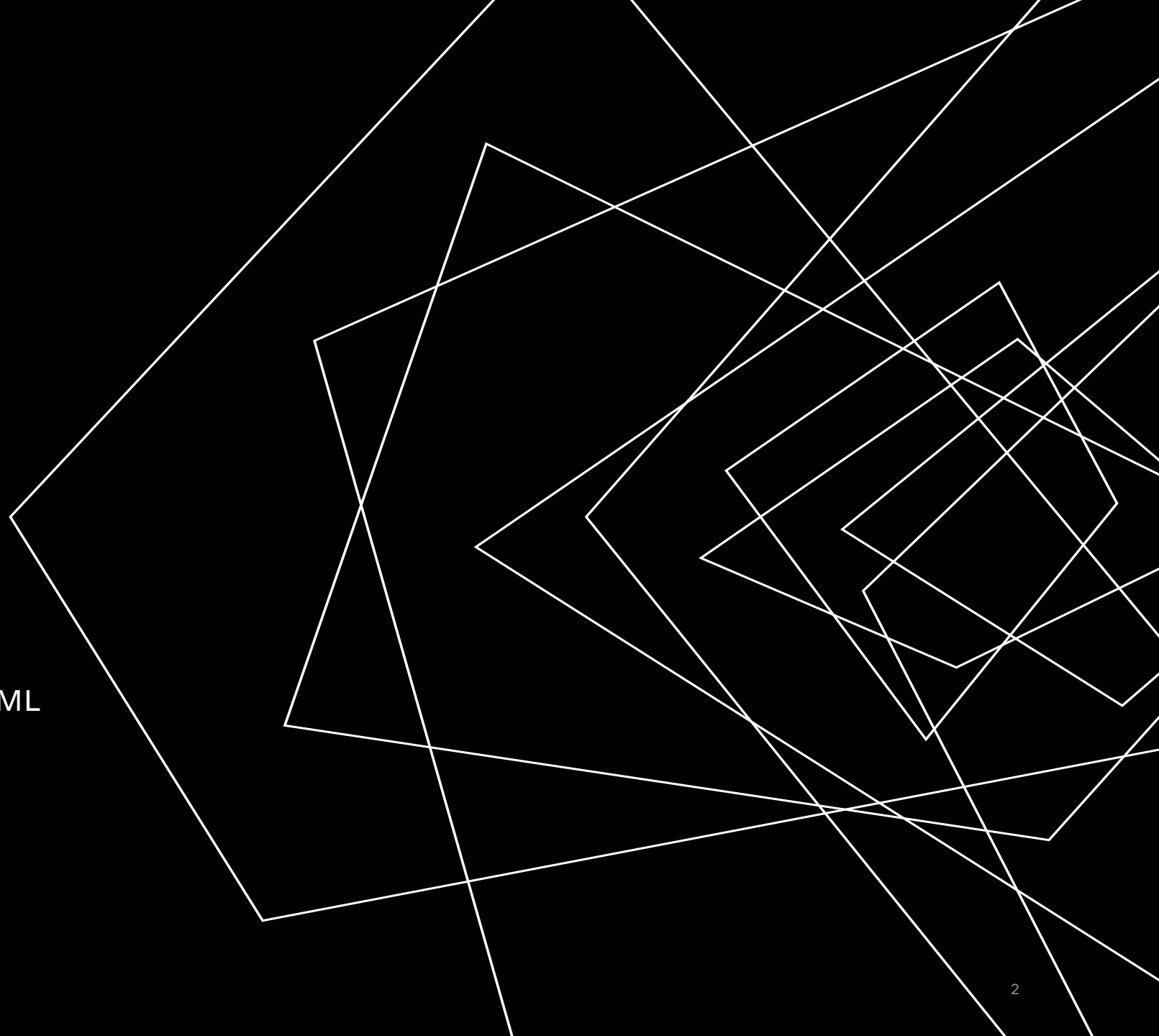
IMAGE PROCESSING IN THE FREQUENCY DOMAIN

BY
SAIFUL BARI IFTU [SDQ]
LECTURER, DEPT. OF CSE, BRAC UNIVERSITY

CONTENTS



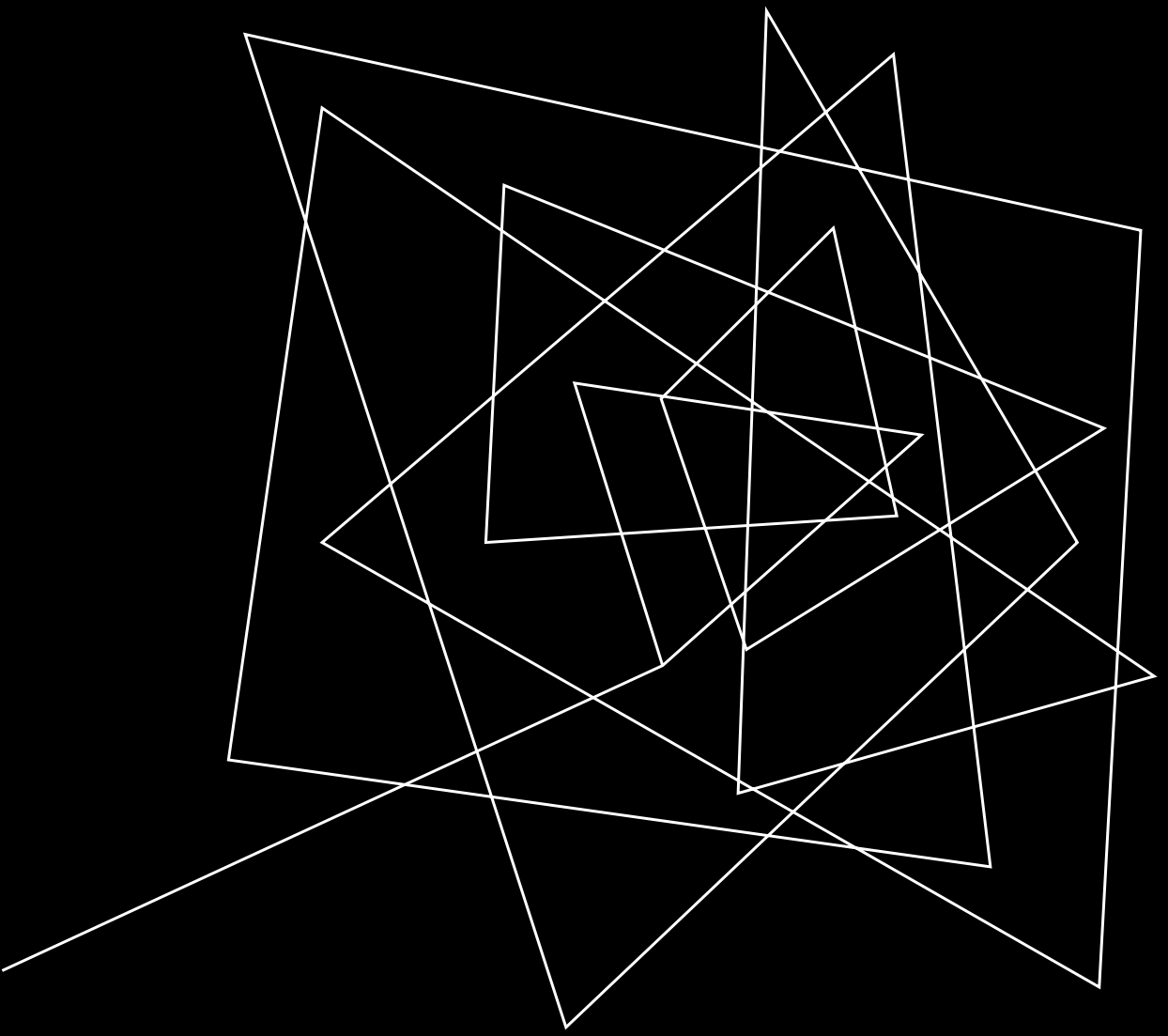


IMAGE PROCESSING IN THE FREQUENCY DOMAIN

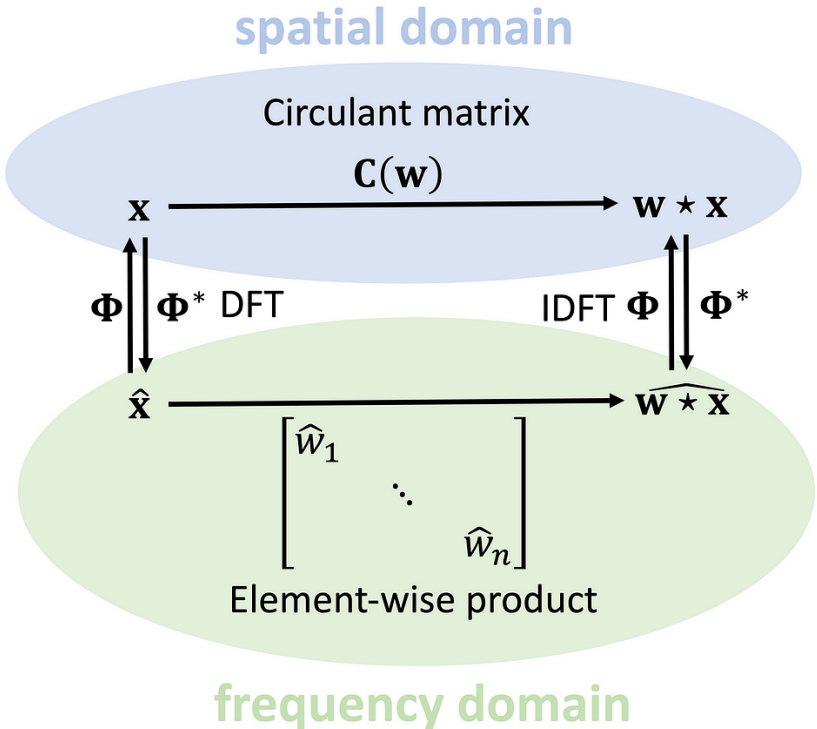
WHY FREQUENCY DOMAIN?

As we saw, It is possible to process images in the spatial domain without taking into account their frequency components. Still, there are several reasons to process images after transforming them to the frequency domain:

- **Efficient Convolution and Filtering:** Convolution in the spatial domain is computationally expensive for large filters, while in the frequency domain, it becomes simply **element-wise multiplication**, making operations like blurring and sharpening faster and more efficient.
- **Isolating Frequency Components:** The frequency domain provides a global view of an image's content, allowing easy isolation of low frequencies (smooth regions) or high frequencies (edges and noise).
- **Advanced Image Processing Capabilities:** Techniques like compression (e.g., JPEG) and noise suppression rely on targeting specific frequency bands. Also, the magnitude spectrum remains unaffected by various variations like scaling and rotation, which helps in pattern recognition.

CONVOLUTION vs MULTIPLICATION

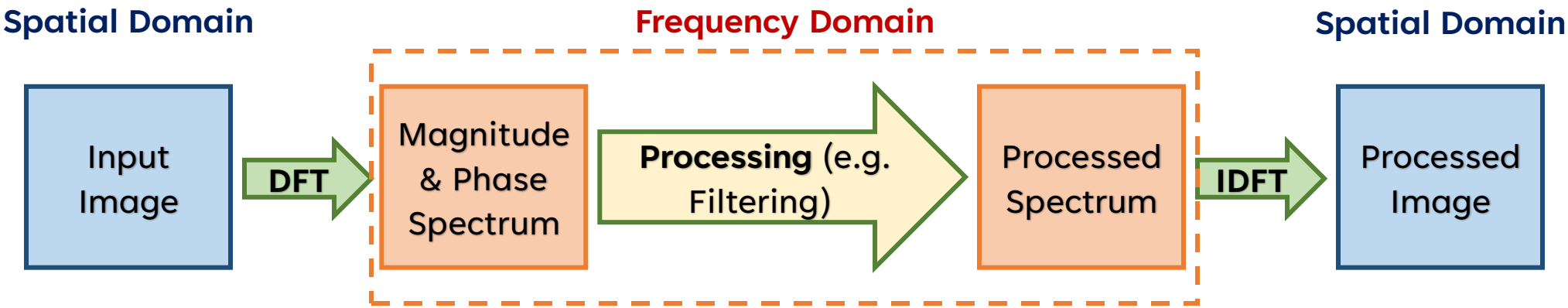
Convolution in the spatial domain involves sliding a filter over an image to blend pixel values. However, mathematically, convolution is complex for large filters due to overlapping calculations. The Convolution Theorem states that **convolution in one domain corresponds to simple element-wise multiplication in the other domain**.



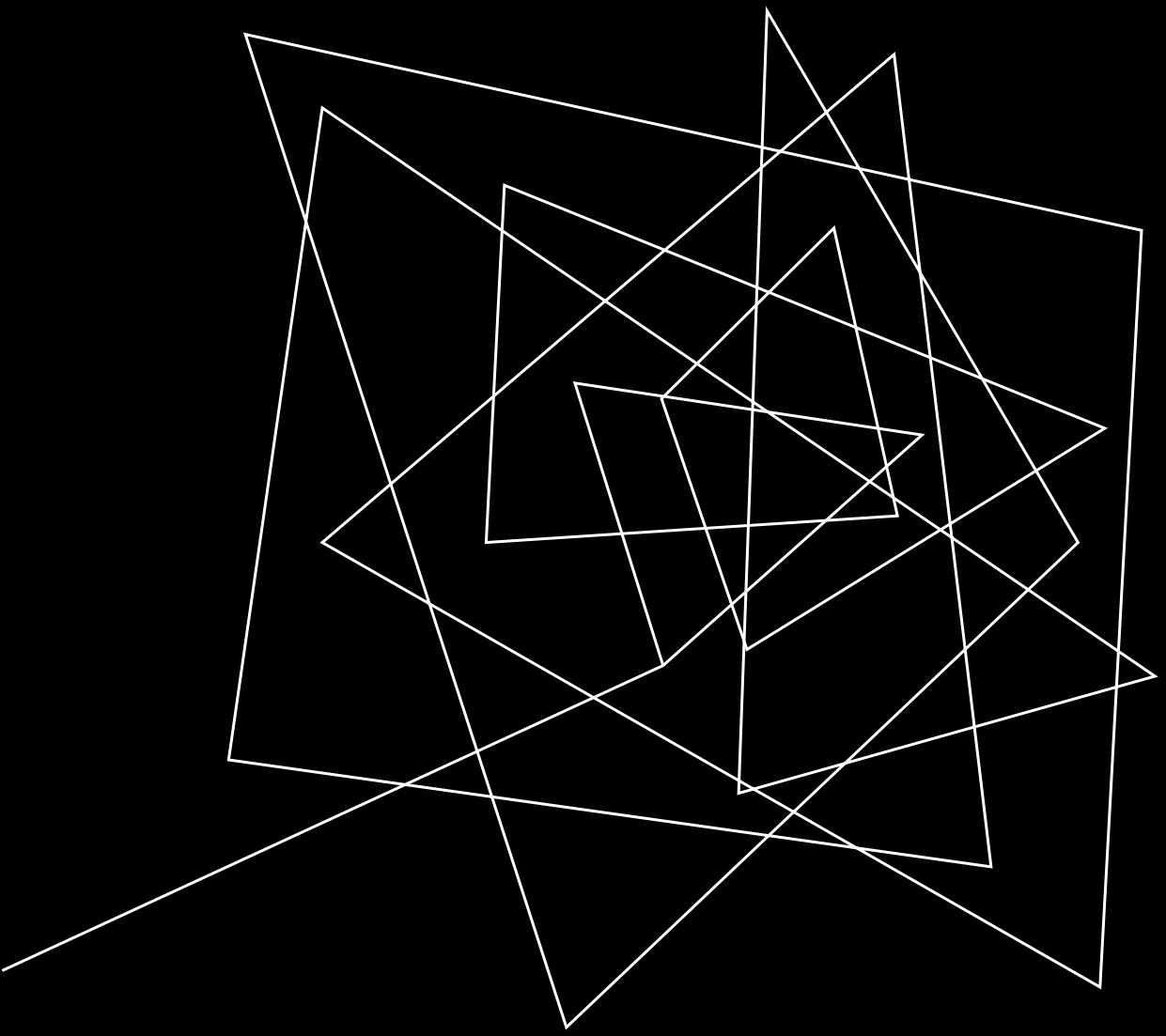
Source: https://towardsdatascience.com/deriving-convolution-from-first-principles-4ff124888028

IMAGE PROCESSING WORKFLOW

Unlike the spatial domain, the frequency-domain image processing workflow starts by using the Fourier Transform or similar methods to **transform** the image into its frequency-domain representation. In this transformed space, image **processing** operations like filtering, noise reduction, or enhancement take place. Once the processing is done, the **Inverse Fourier Transform** brings the image back into the interpretable spatial domain. The final output is a **processed image** that's been improved or cleaned up using the advantages of frequency-domain processing.



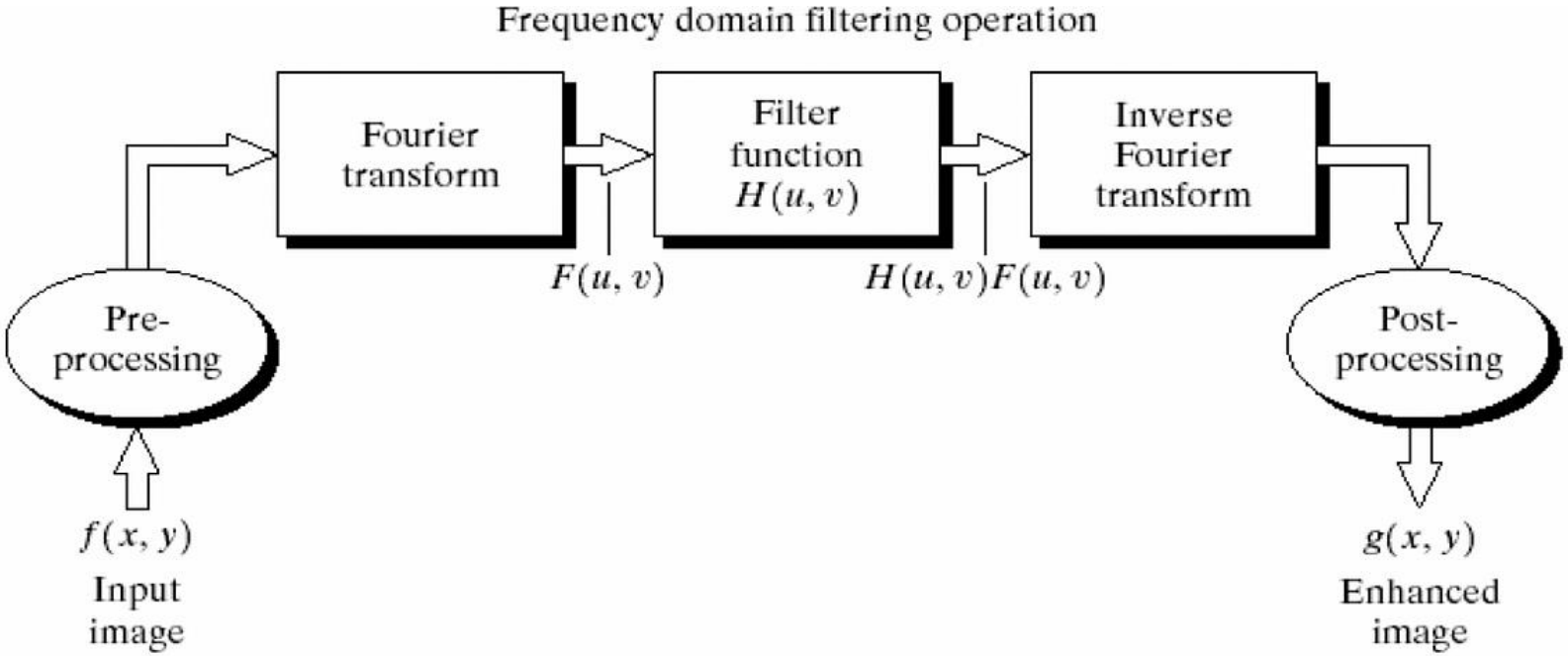
Workflow of Image Processing in the Frequency Domain



FILTERING IN THE FREQUENCY DOMAIN

FILTERING IN FREQUENCY DOMAIN

Filtering in the frequency domain involves manipulating the frequency components of an image to achieve specific effects like smoothing, sharpening, or noise removal. This process is different from filtering in the spatial in the sense that the **filter is applied directly to the frequency representation** of the image, often as a **multiplication operation** with a frequency-domain mask, instead of the more complex **convolution operation** we saw in the spatial domain. After filtering, the Inverse Fourier Transform is used to convert the modified frequency data back to the spatial domain.



TYPES OF FILTERS

There are several types of filters in the frequency domain. However, the following are most commonly used and referred to:

- **Low-Pass Filter**: Retains low frequencies while suppressing high frequencies, useful for smoothing or noise reduction.
- **High-Pass Filter**: Retains high frequencies while suppressing low frequencies, used for enhancing edges or fine details.
- **Band-Pass Filter**: Allows a specific range of frequencies to pass through while blocking others, helpful for extracting features or removing specific patterns.
- **Band-Stop Filter (Notch Filter)**: Removes specific frequency bands, often used to eliminate periodic noise.

The Filtering Operation for images in the frequency domain is simply multiplying the filter with the input signal:

$$\text{Filtered Spectrum, } G(u,v) = F(u,v) \bullet H(u,v)$$

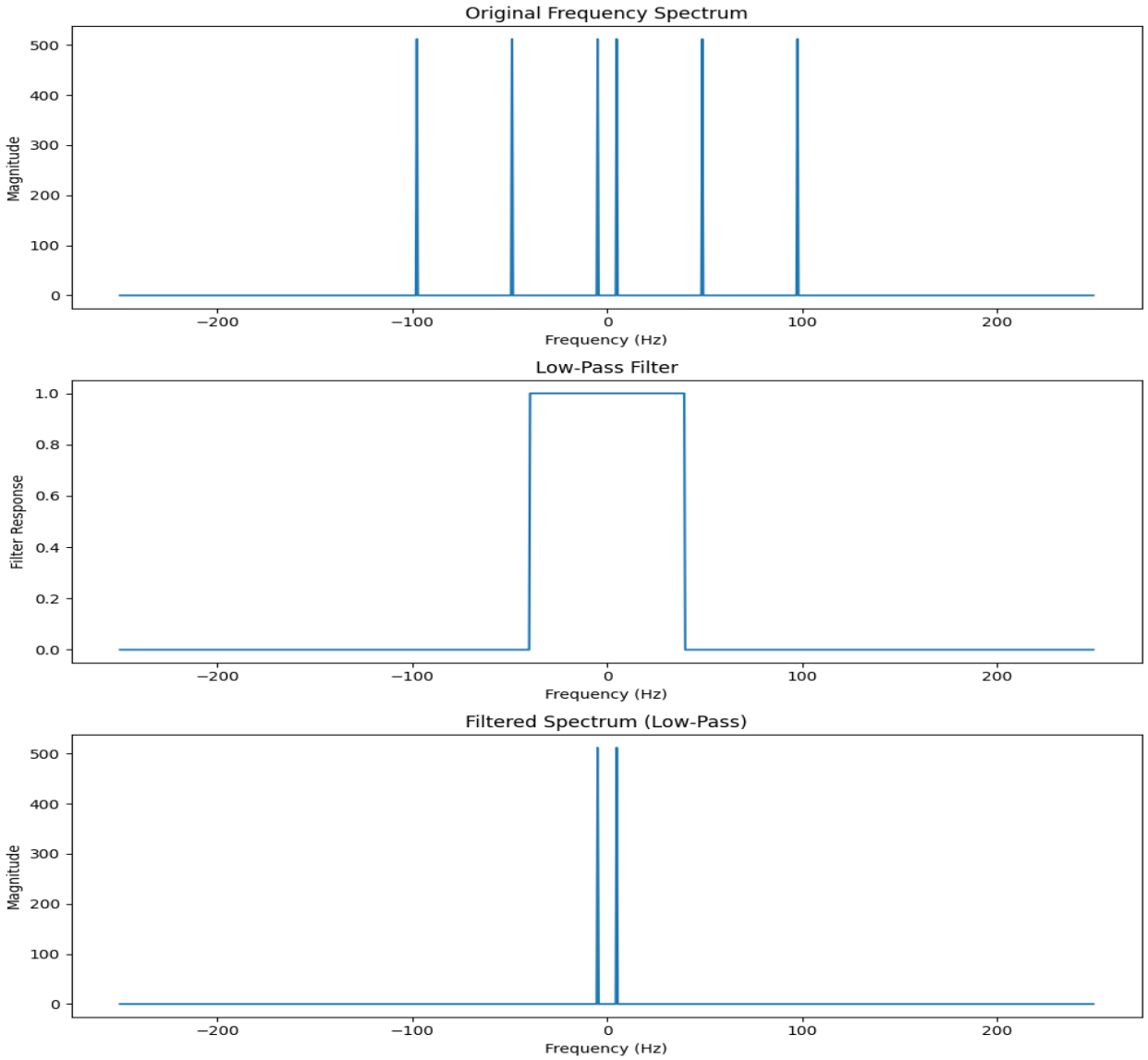
Where, $F(u,v)$ is the frequency domain representation of the original signal and $H(u,v)$ is the frequency domain filter.

FILTERING IN 1D: LOW PASS

Observe how a **low pass** filter works in 1D.

The filter is designed with a cutoff frequency (**40** in this case), which defines the threshold between the frequencies it passes and those it blocks. Frequencies below this cutoff are retained, while those above are suppressed.

The filter **multiplies the Fourier-transformed signal by a filter mask**. For frequencies below the cutoff, the mask values are **1** (full pass), and for frequencies above, the mask values are **0** (completely blocked) or gradually decrease (if a smoother transition is needed). After multiplication with 0, the high frequency components also become 0 and get canceled.

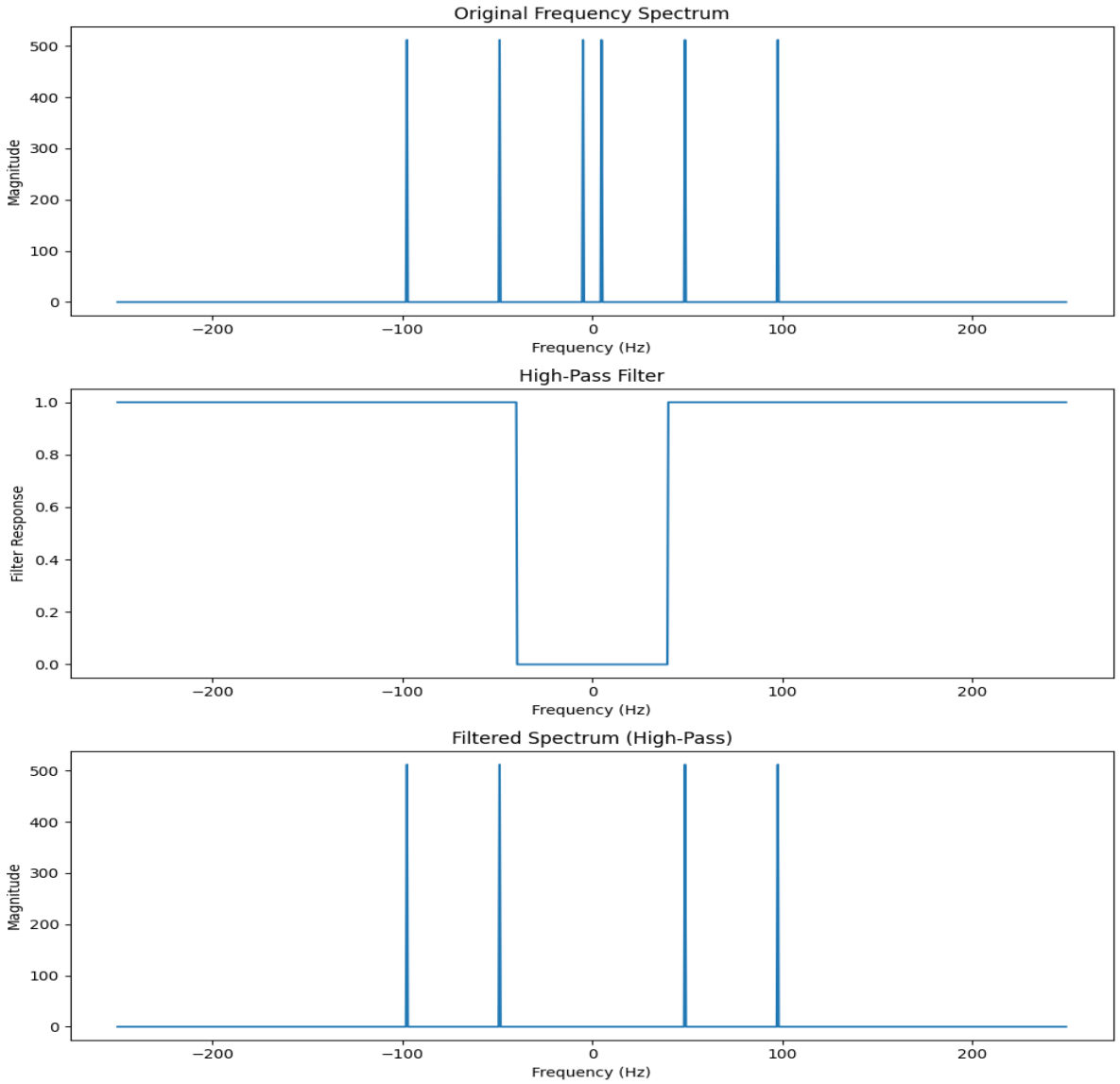


FILTERING IN 1D: HIGH PASS

Observe how a **high pass** filter works in 1D.

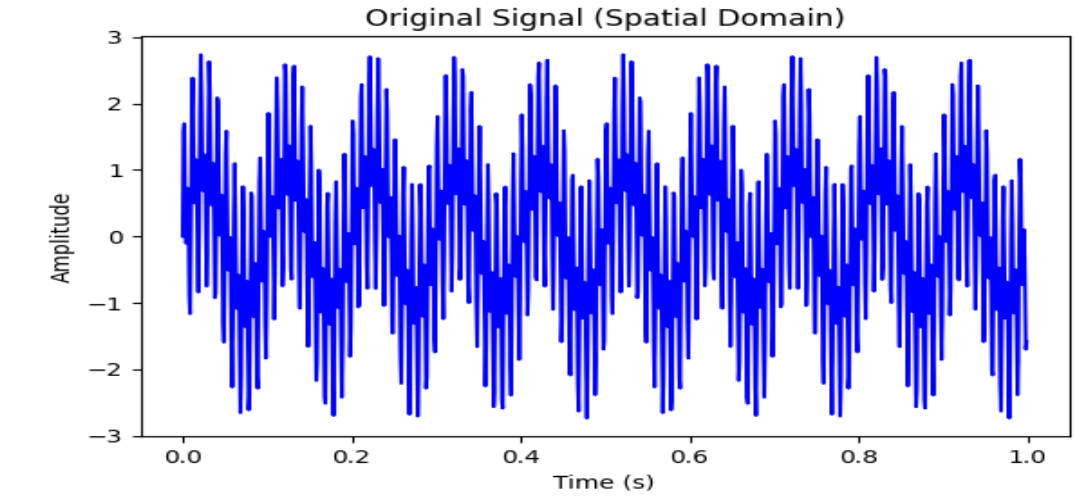
The filter is designed with a cutoff frequency (**40** in this case), which defines the threshold between the frequencies it passes and those it blocks. Frequencies below this cutoff are suppressed, while those above are retained.

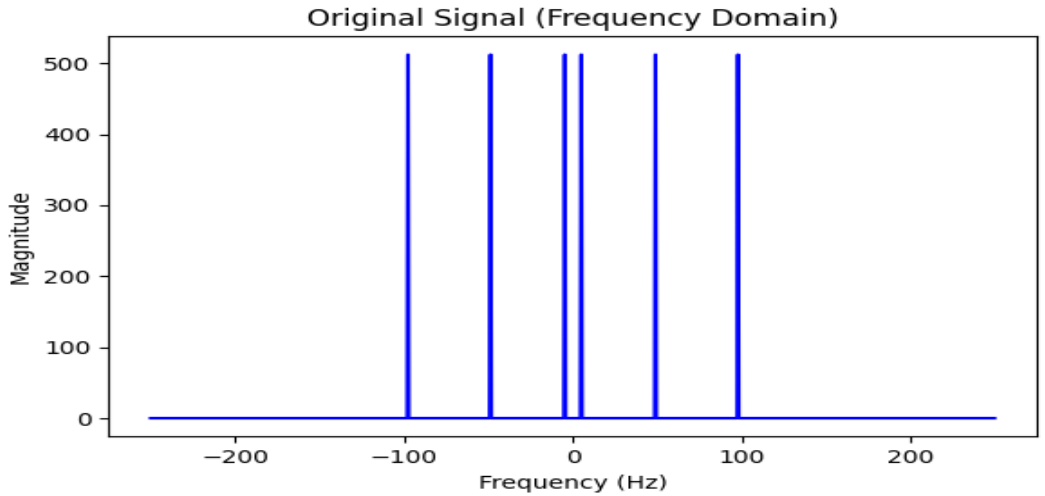
The filter too **multiplies the Fourier-transformed signal by a filter mask**. For frequencies above the cutoff, the mask values are **1** (full pass), and for frequencies below, the mask values are **0** (completely blocked) or gradually decrease (if a smoother transition is needed). After multiplication with 0, the low frequency components also become 0 and get cancelled.

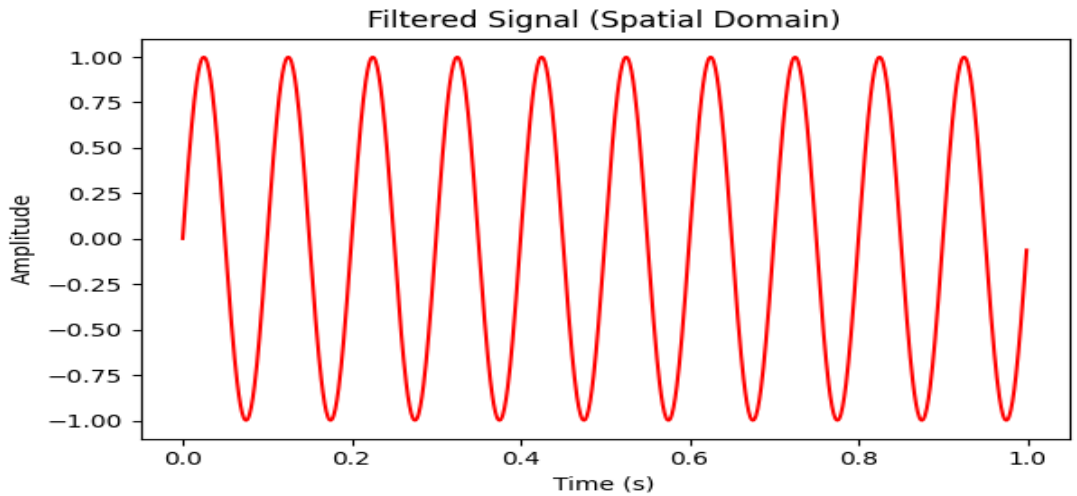


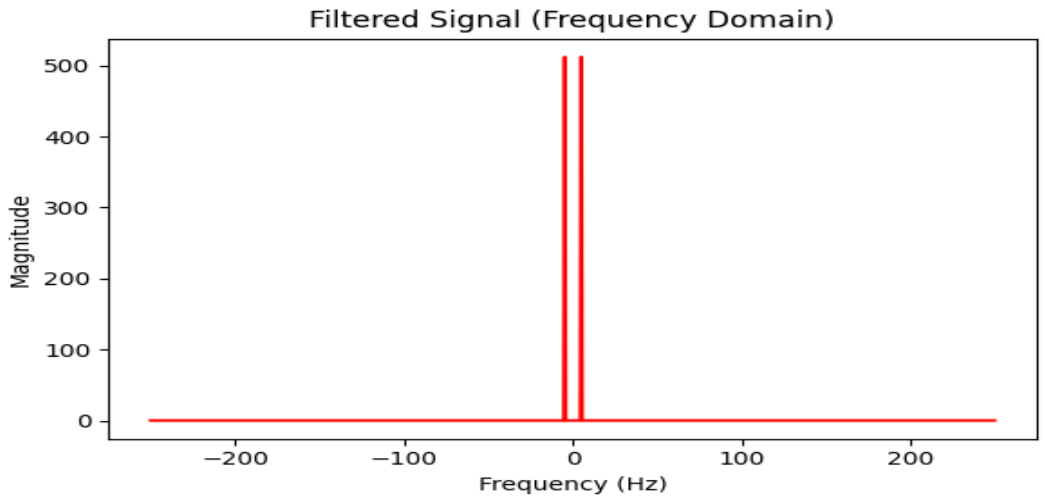
FILTERING IN 1D: EFFECT IN SPATIAL DOMAIN

Observe how the 1D signal has been smoothened by eliminating rapid variations after applying the **low pass** filter **in the frequency domain**.



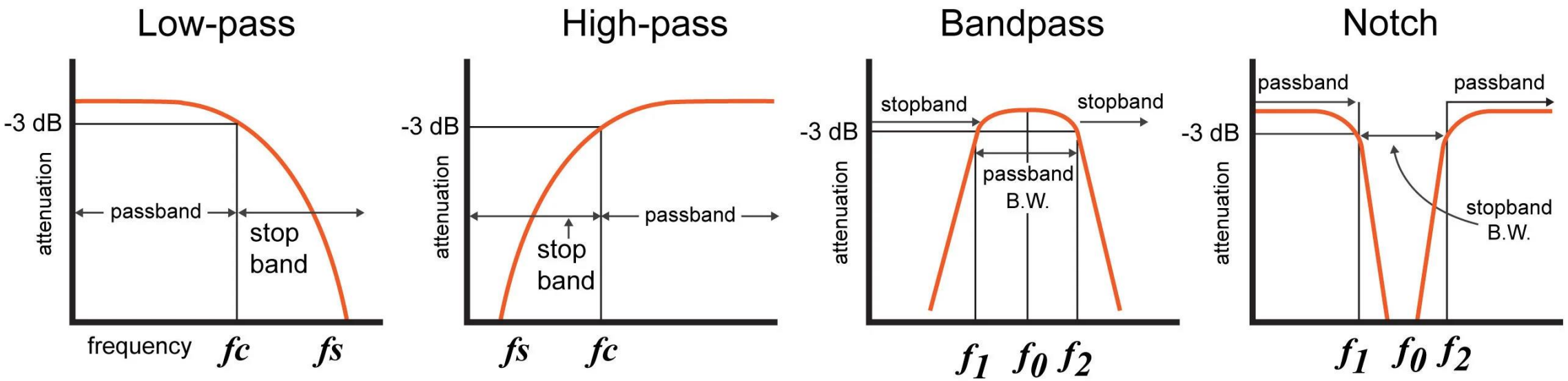






DIFFERENT TYPES OF FILTERS IN 1D

Take a look at different types of 1D filters below. The negative frequencies have been ignored as they convey the same information as the positive frequencies.



Source: https://www.allaboutcircuits.com/technical-articles/an-introduction-to-filters/

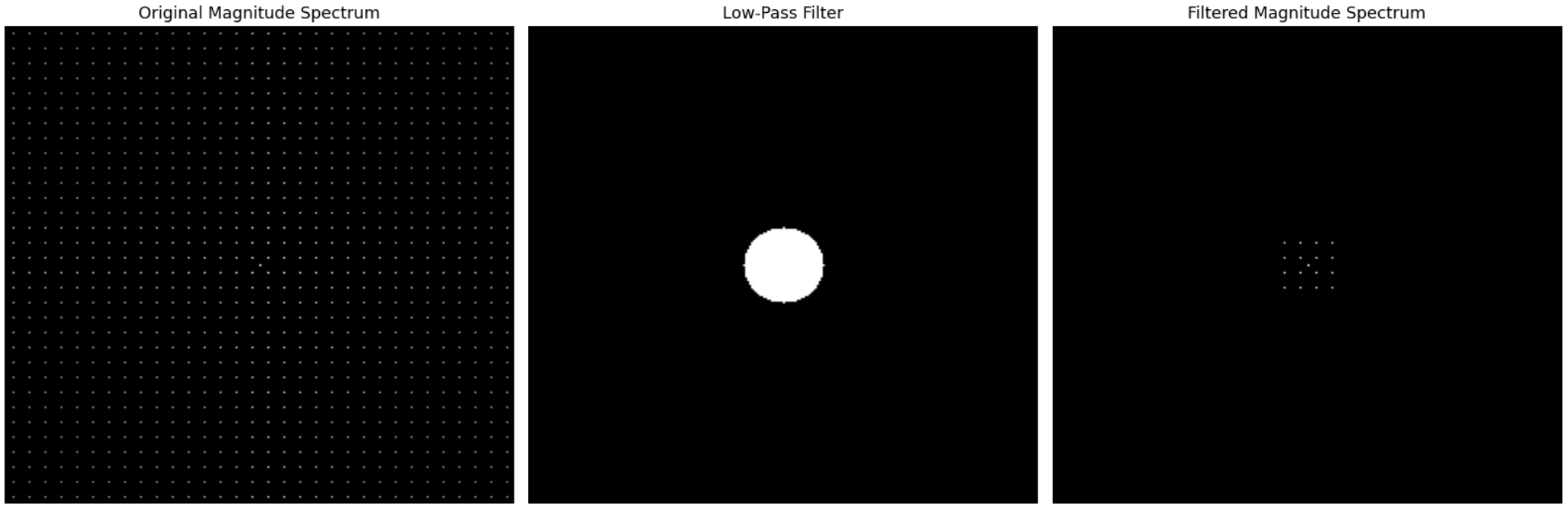
Frequency Domain Filters in 1D

FILTERING IN 2D IMAGES: LOW PASS

Observe how a **low-pass** filter works in 2D images.

The filter is also designed with a cutoff frequency, which defines the threshold between the frequencies it passes and those it blocks. Frequencies below this cutoff are retained, while those above are suppressed.

For frequencies below the cutoff, the mask values are **1** (full pass), and for frequencies above, the values are **0** (completely blocked) or gradually decrease (if a smoother transition is needed). After multiplication with 0, the high-frequency components also become 0 and get canceled.

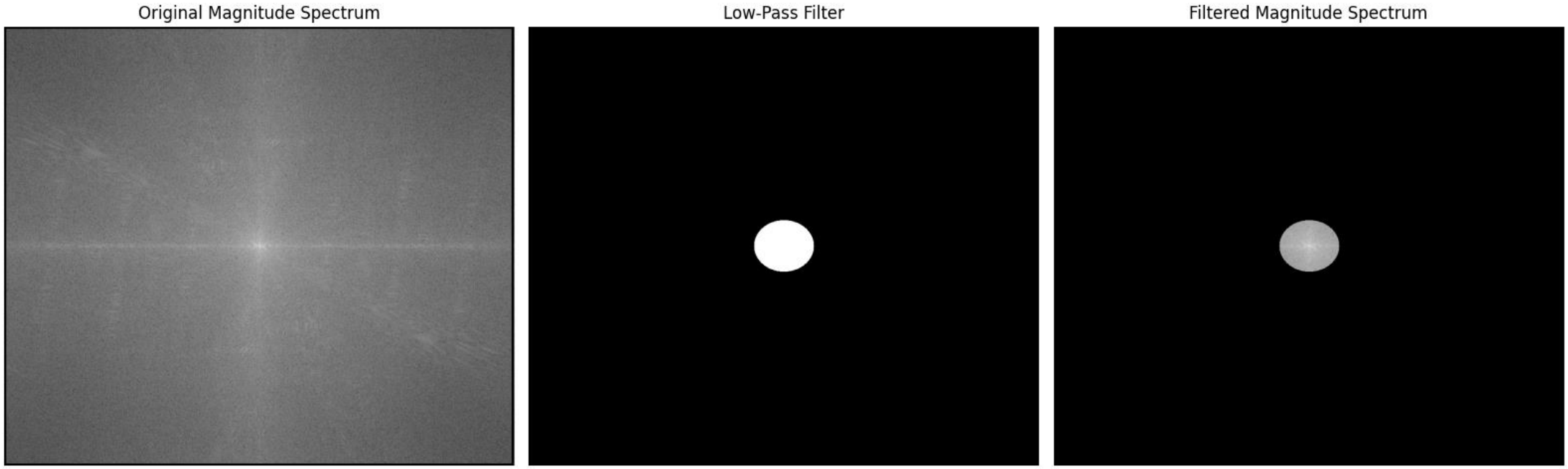


FILTERING IN 2D IMAGES: LOW PASS

Observe how a **low-pass** filter works in 2D images.

The filter is also designed with a cutoff frequency, which defines the threshold between the frequencies it passes and those it blocks. Frequencies below this cutoff are retained, while those above are suppressed.

For frequencies below the cutoff, the mask values are **1** (full pass), and for frequencies above, the values are **0** (completely blocked) or gradually decrease (if a smoother transition is needed). After multiplication with 0, the high-frequency components also become 0 and get canceled.

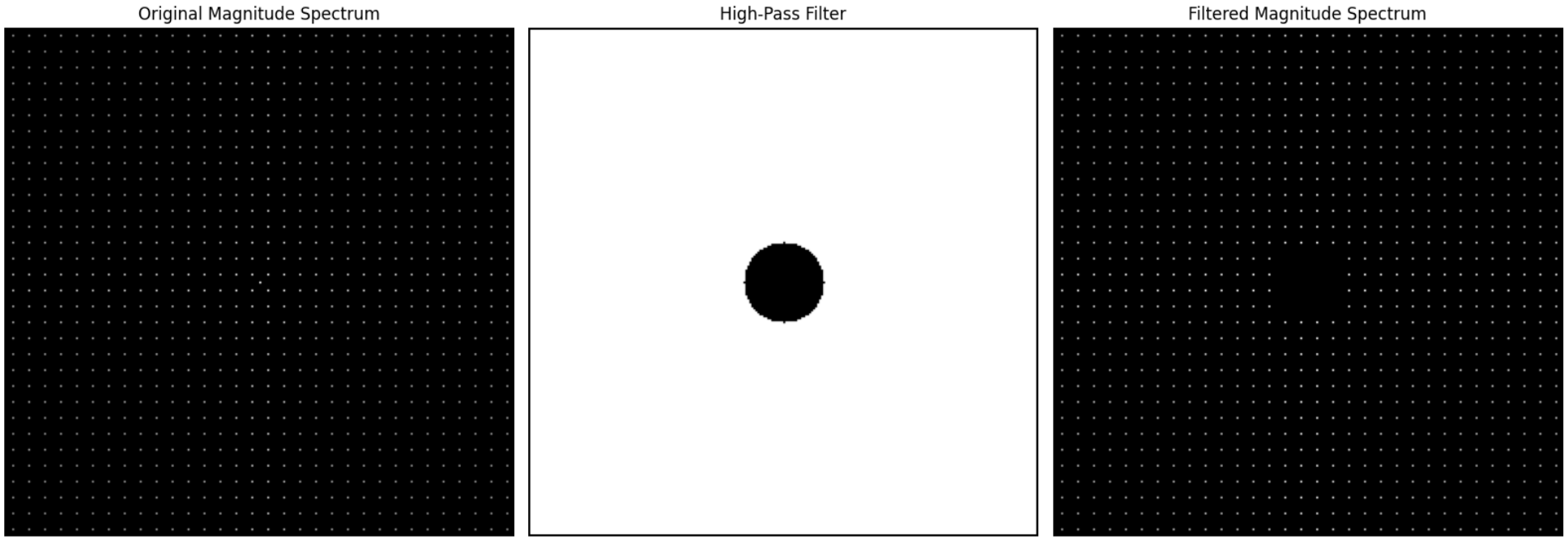


FILTERING IN 2D IMAGES: HIGH PASS

Observe how a **high-pass** filter works in 2D images.

The filter is once again designed with a cutoff frequency, which defines the threshold between the frequencies it passes and those it blocks. Frequencies above this cutoff are retained, while those below are suppressed.

For frequencies above the cutoff, the mask values are **1** (full pass), and for frequencies below, the values are **0** (completely blocked) or gradually decrease (if a smoother transition is needed). After multiplication with 0, the low-frequency components also become 0 and get canceled.

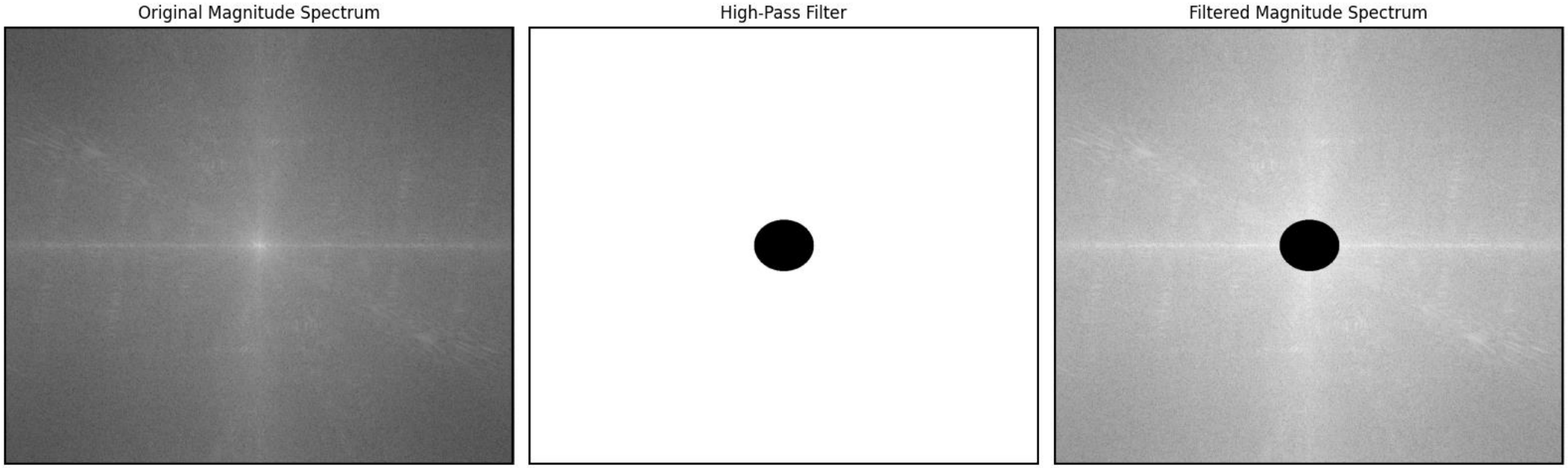


FILTERING IN 2D IMAGES: HIGH PASS

Observe how a **high-pass** filter works in 2D images.

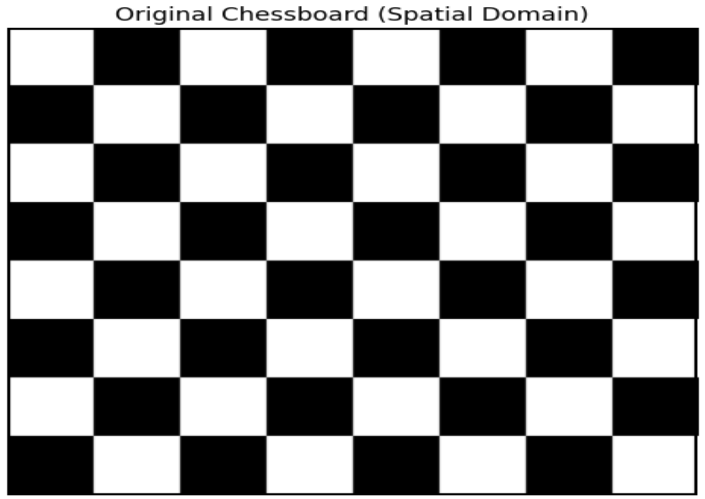
The filter is once again designed with a cutoff frequency, which defines the threshold between the frequencies it passes and those it blocks. Frequencies above this cutoff are retained, while those below are suppressed.

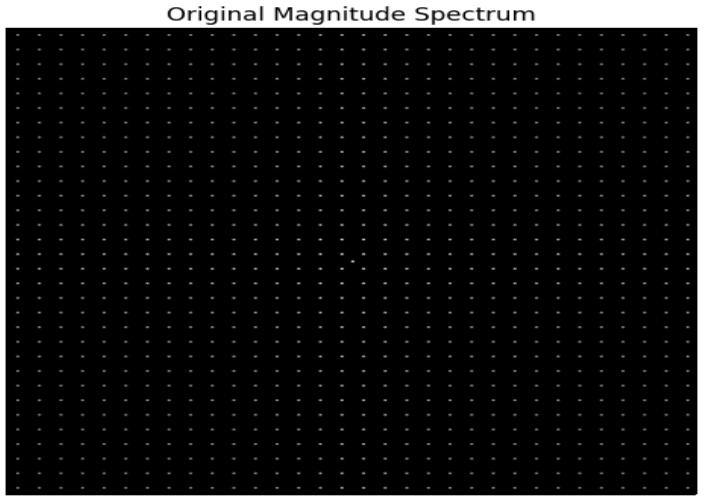
For frequencies above the cutoff, the mask values are **1** (full pass), and for frequencies below, the values are **0** (completely blocked) or gradually decrease (if a smoother transition is needed). After multiplication with 0, the low-frequency components also become 0 and get canceled.

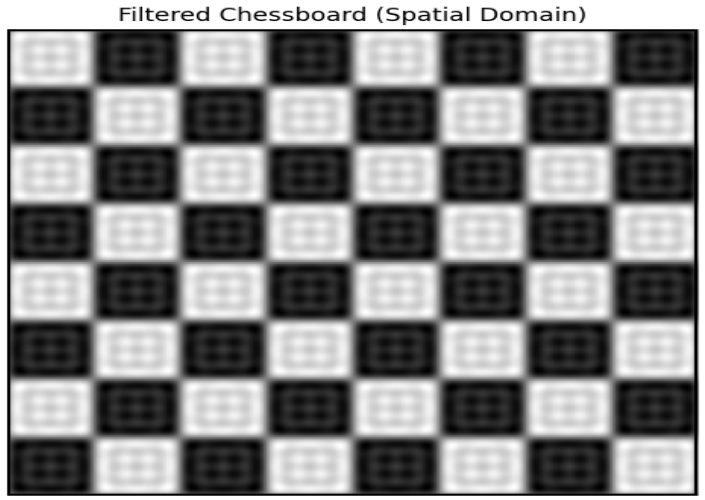


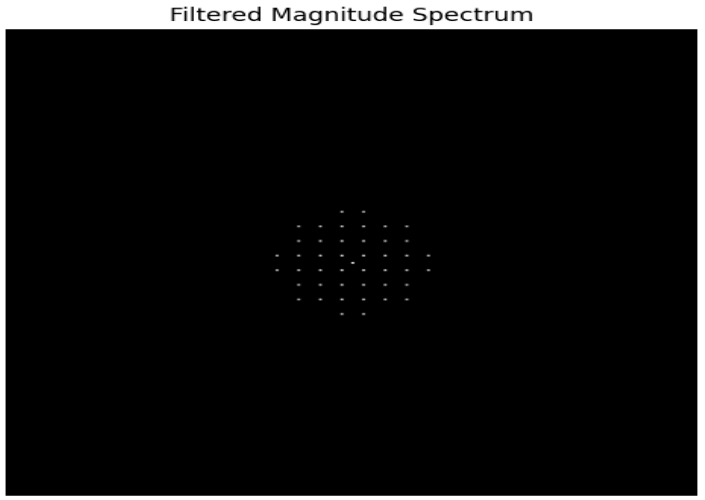
FILTERING IN 2D: LOW PASS

After applying the low pass filter in the frequency domain, observe how the 2D signal has been blurred by smoothing out sharp edges.



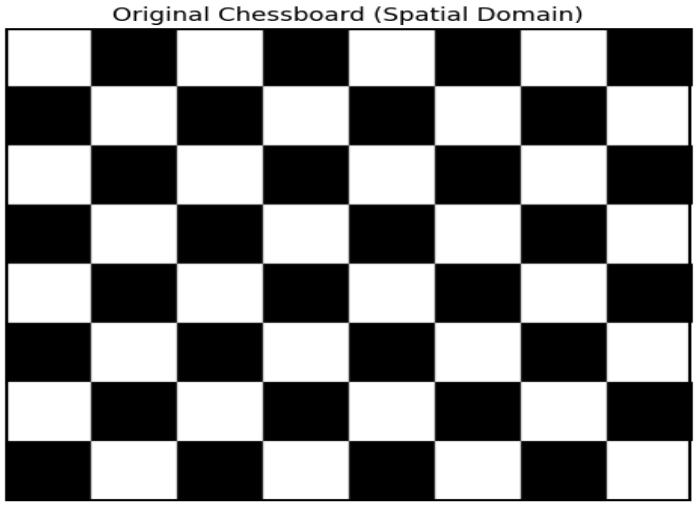


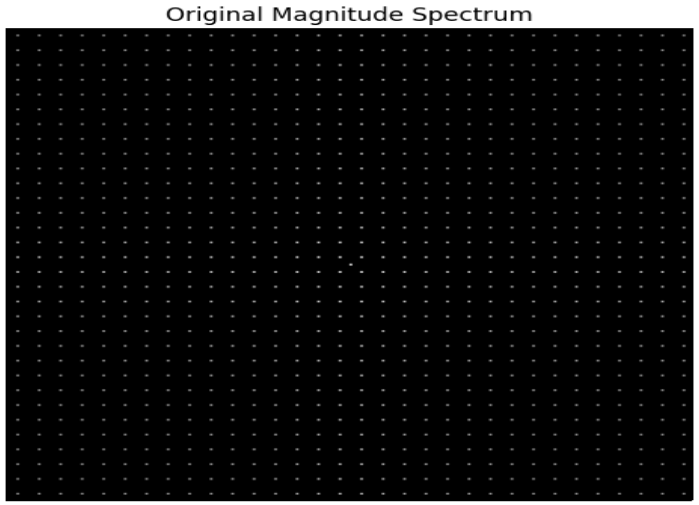


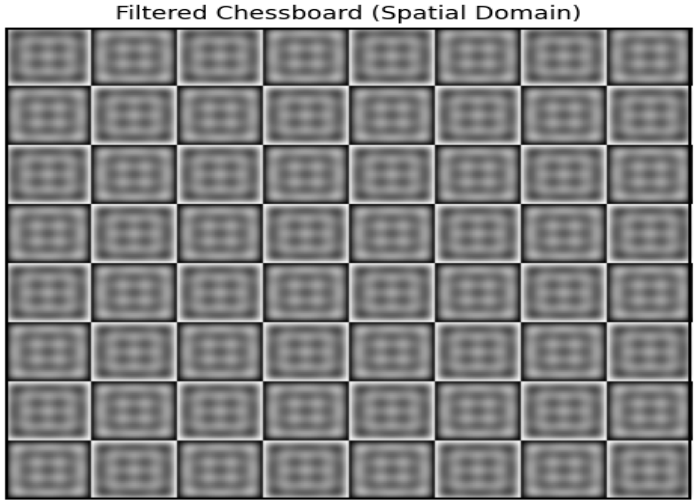


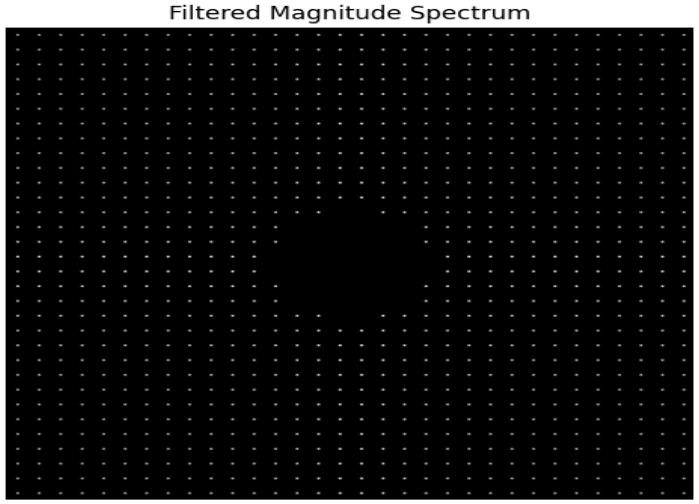
FILTERING IN 2D: HIGH PASS

After applying the high pass filter in the frequency domain, observe how the 2D signal has been sharpened by enhancing sharp edges and canceling out the smoother regions.



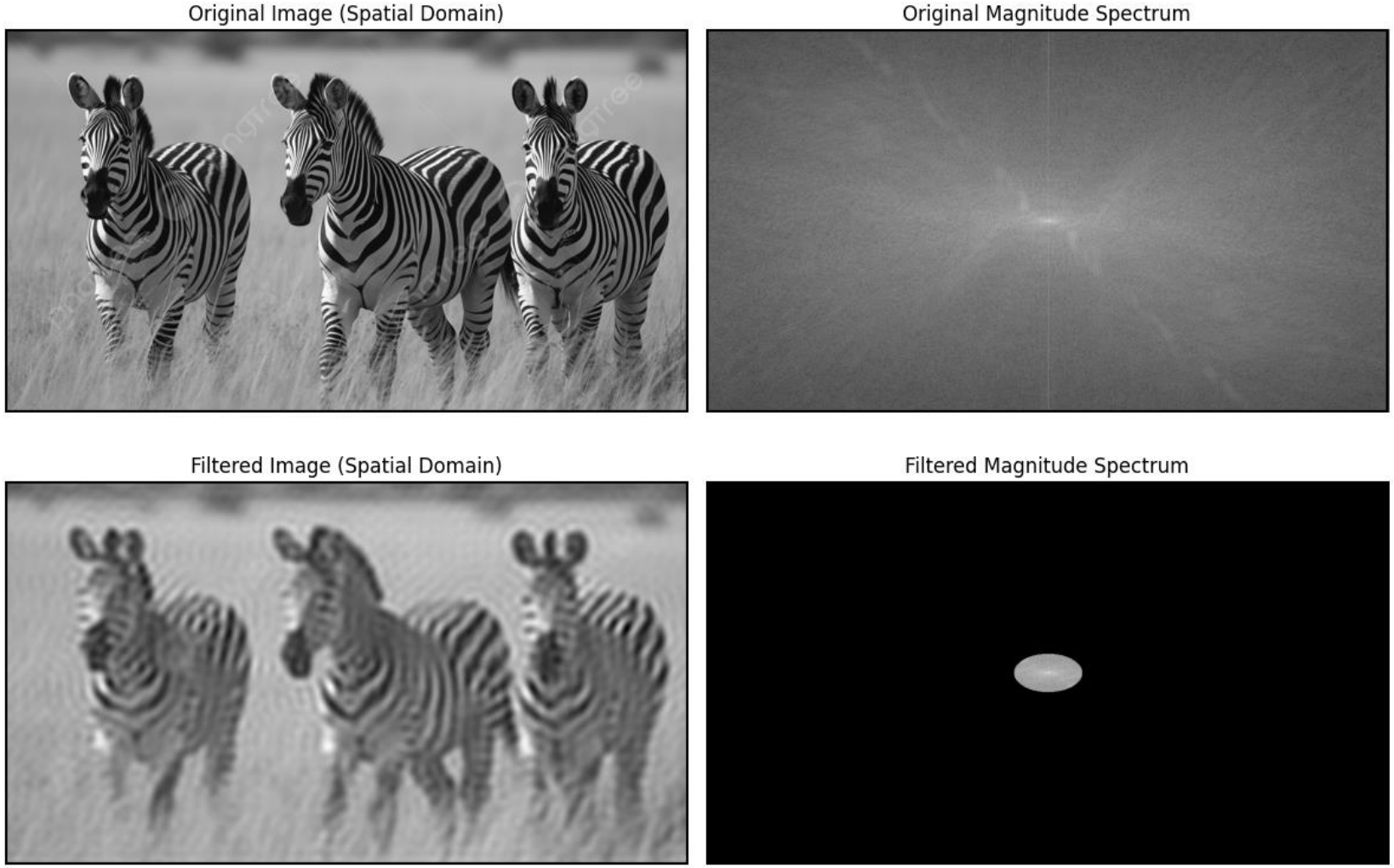






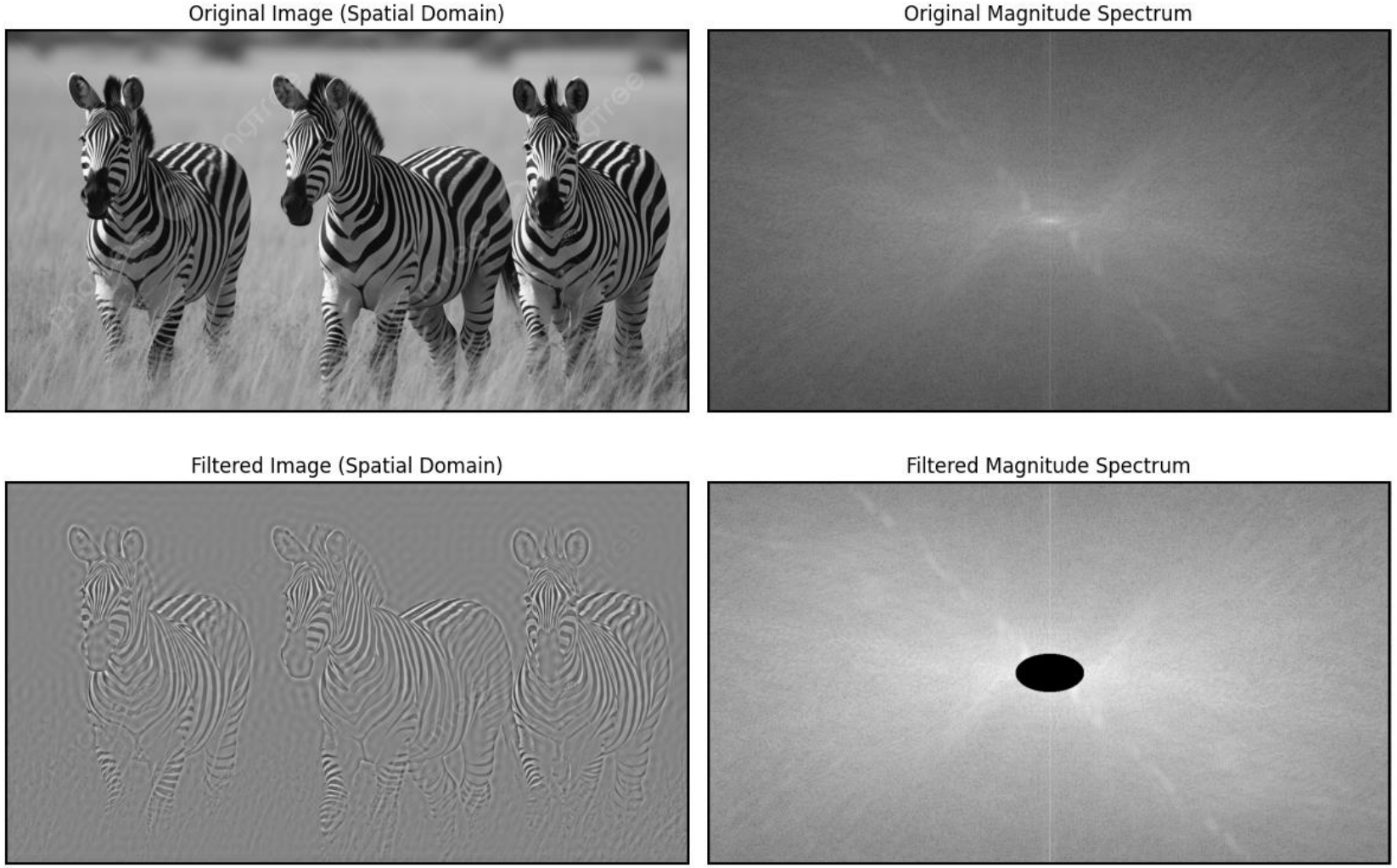
FILTERING IN 2D: LOW PASS

After applying the low pass filter in the frequency domain, observe how the 2D signal has been blurred by smoothing out sharp edges.



FILTERING IN 2D: HIGH PASS

After applying the high pass filter in the frequency domain, observe how the 2D signal has been sharpened by enhancing sharp edges and canceling out the smoother regions.

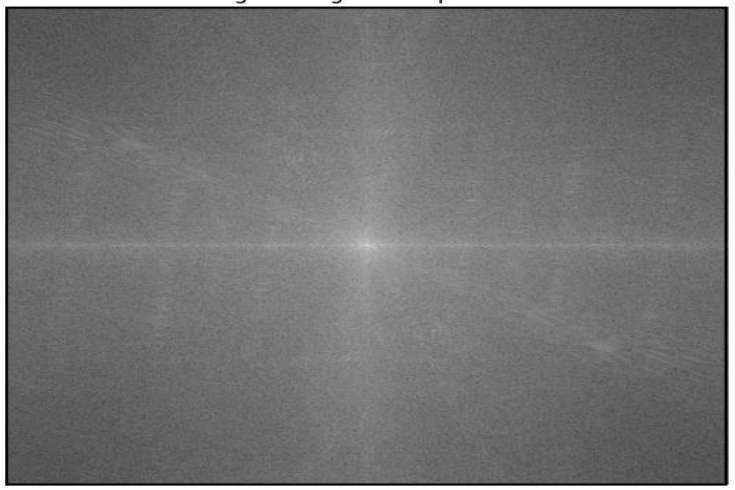


FILTERING IN 2D: LOW PASS

After applying the low pass filter in the frequency domain, observe how the 2D signal has been blurred by smoothing out sharp edges.

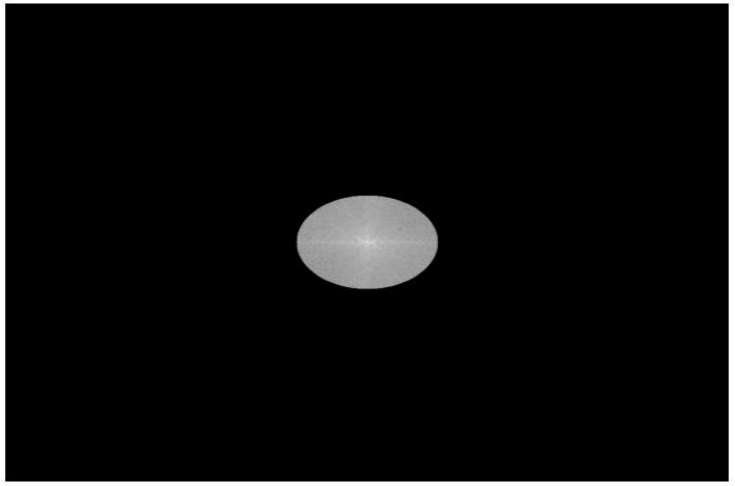
Original Image (Spatial Domain)

Original Magnitude Spectrum



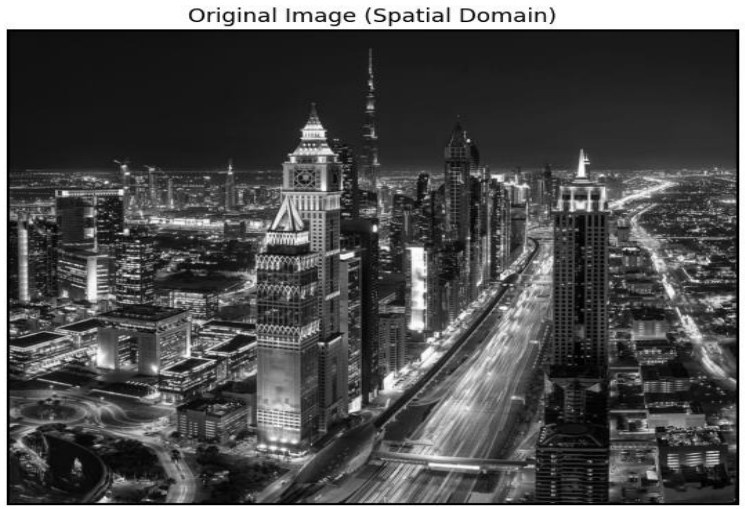
Filtered Image (Spatial Domain)

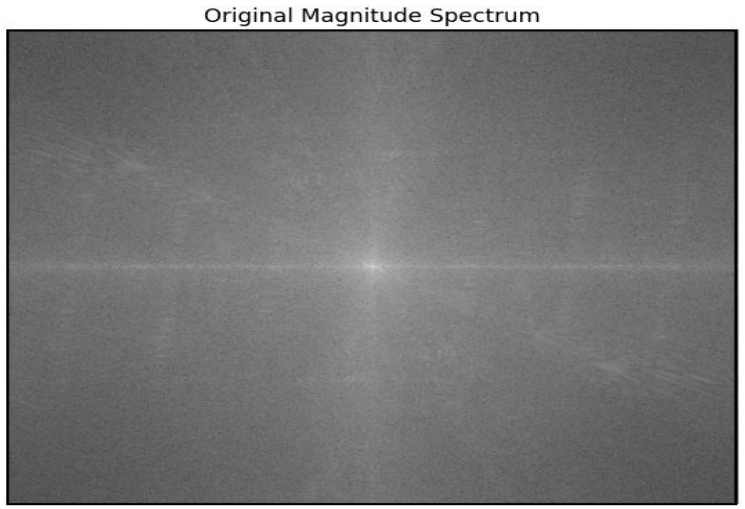
Filtered Magnitude Spectrum

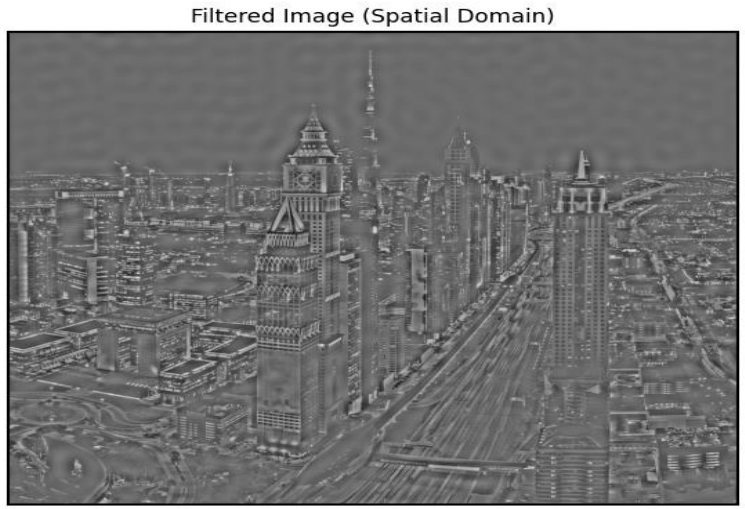


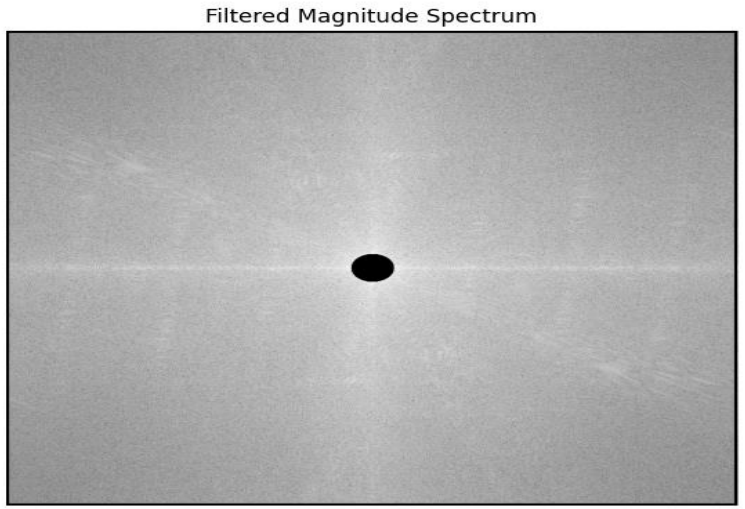
FILTERING IN 2D: HIGH PASS

After applying the high pass filter in the frequency domain, observe how the 2D signal has been sharpened by enhancing sharp edges and canceling out the smoother regions.







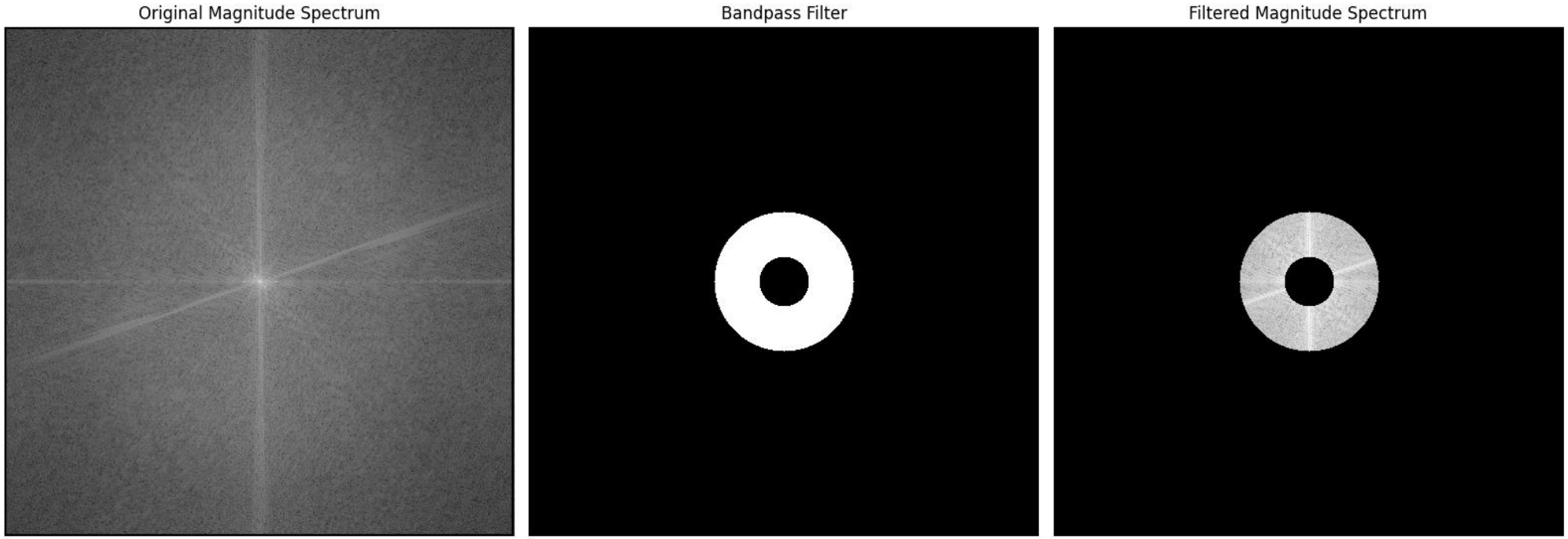


FILTERING IN 2D IMAGES: BAND PASS

Observe how a **band-pass** filter works in 2D images.

The filter is designed with two cutoff frequencies, which define the thresholds between the frequencies it passes and those it blocks. Frequencies above the higher cutoff and below the lower cutoff are suppressed, while those between these two cutoffs are retained.

For frequencies between the cutoffs, the mask values are **1** (full pass), and for frequencies above and below both, the values are **0** (completely blocked) or gradually decrease (if a smoother transition is needed).

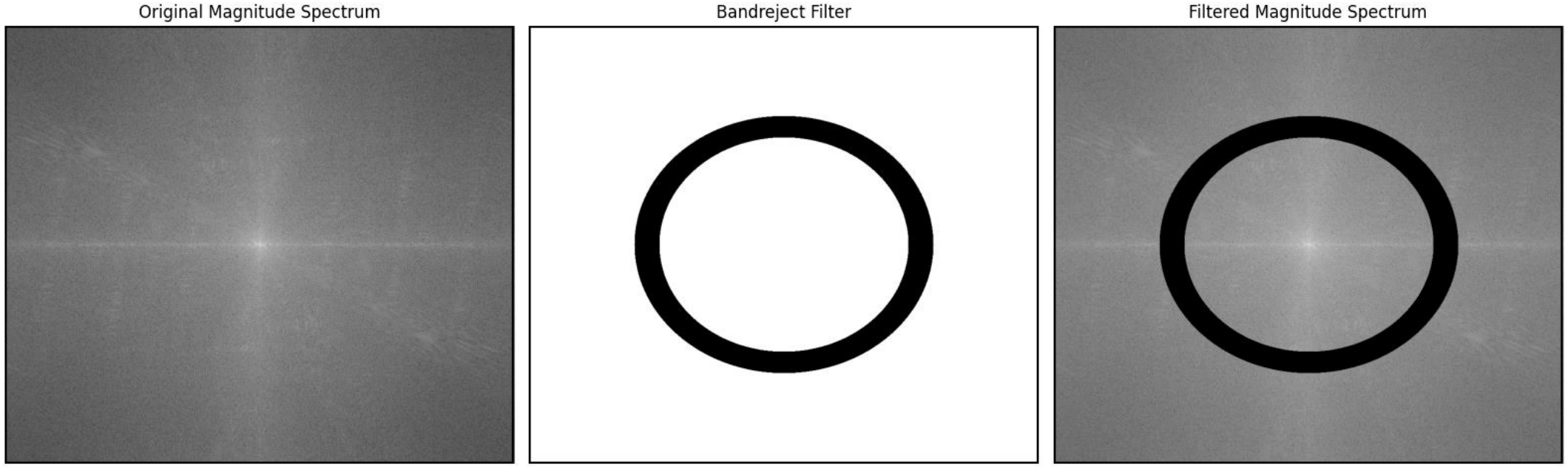


FILTERING IN 2D IMAGES: BAND STOP (NOTCH)

Observe how a **band-stop (Notch)** filter works in 2D images.

The filter is also designed with two cutoff frequencies, which define the thresholds between the frequencies it passes and those it blocks. Frequencies above the higher cutoff and below the lower cutoff are retained, while those between these two cutoffs are suppressed.

For frequencies above and below the cutoffs, the mask values are **1** (full pass), and for frequencies between them, the values are **0** (completely blocked) or gradually decrease (if a smoother transition is needed).

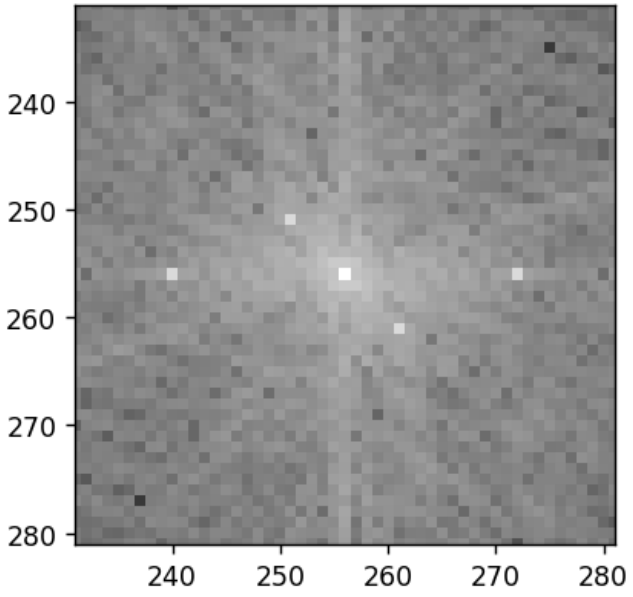


FILTERING IN 2D IMAGES: BAND STOP (NOTCH)

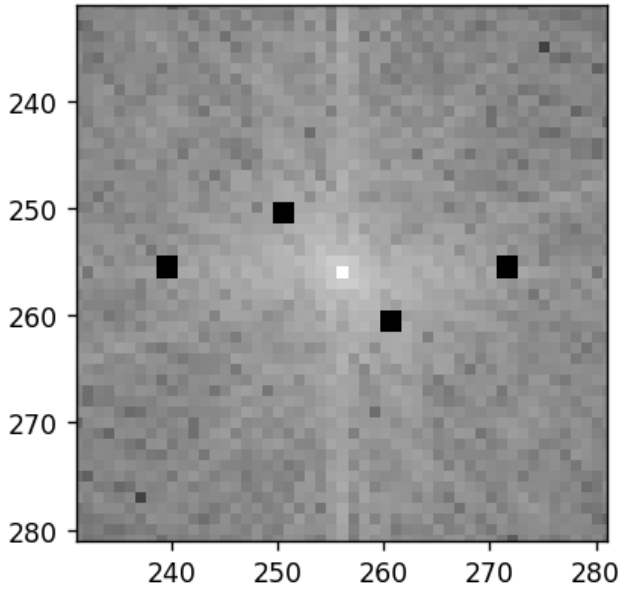
Observe how a **band-stop (Notch)** filter has been applied to eliminate noisy patterns containing specific frequencies to produce a clean image.

The DFT of the image on the right shows specific frequencies that represent the unwanted patterns in the horizontal and an angled direction. Then, filtering was applied to reject only those specific frequencies to get the filtered denoised image.

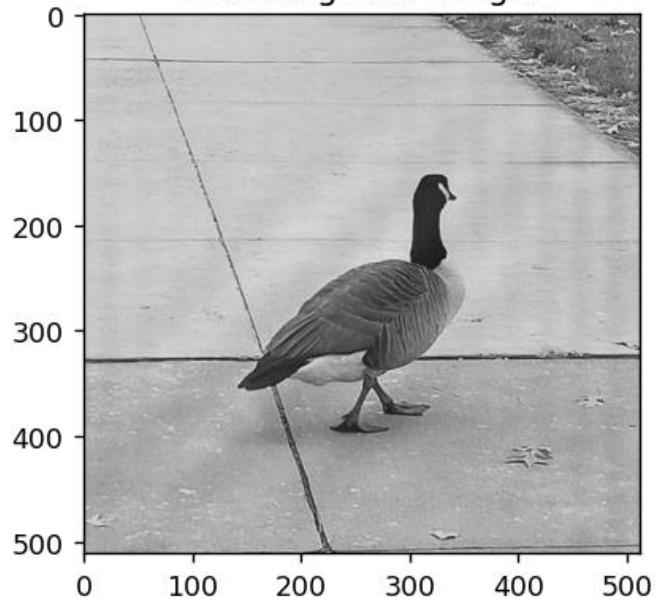
Noisy goose image FT spectrum (Zoomed in).



Filtered goose image FT spectrum (Zoomed in).



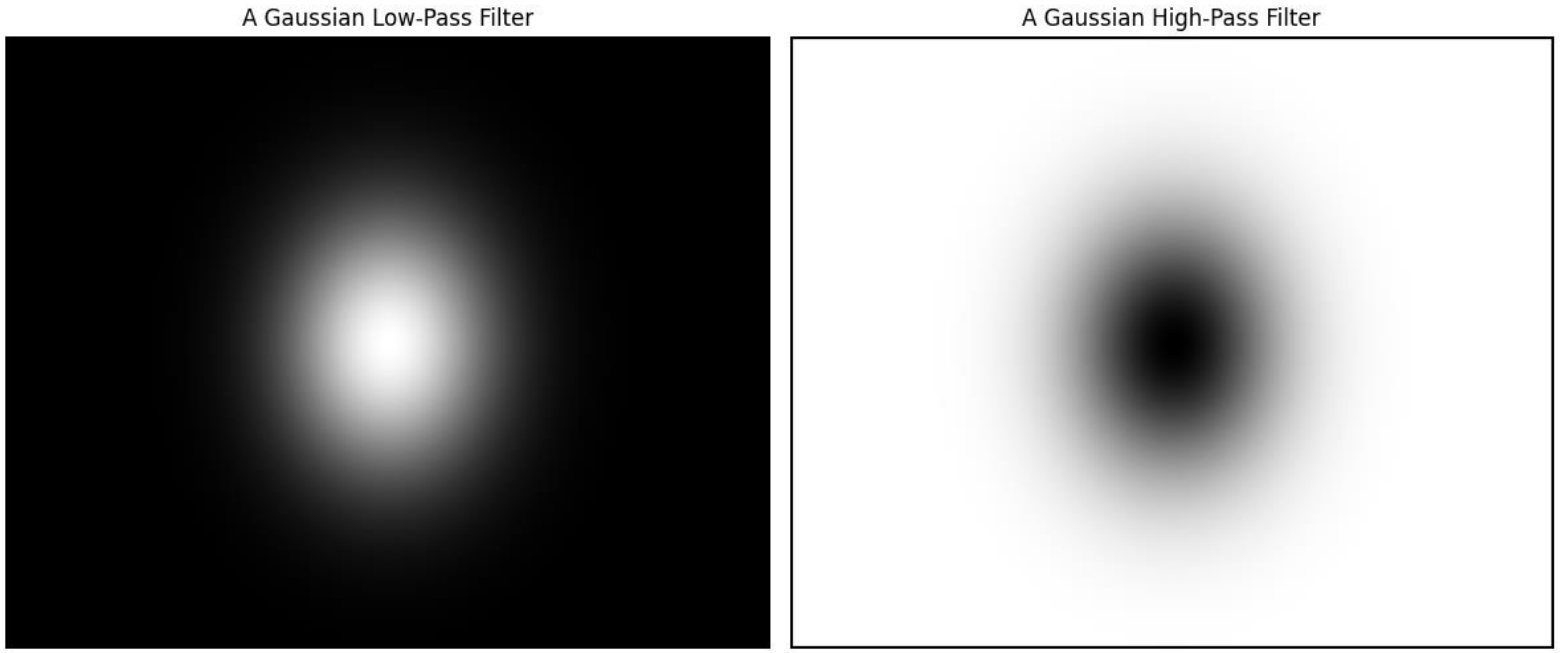
Filtered goose image.



GAUSSIAN FILTERS

Gaussian Filters are widely used in image processing and signal processing. The weights in a Gaussian filter follow a bell-shaped curve, which gives more importance to the center pixels and less to those farther away. This mimics natural processes, like how the human eye perceives intensity changes.

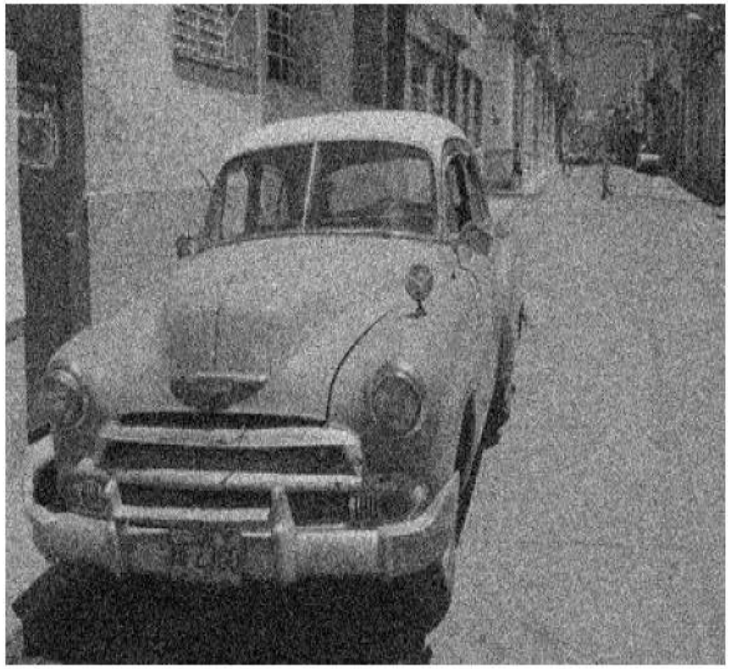
Since Gaussian filters emphasize nearby pixels more, they tend to preserve local structures and gradual intensity changes better. For example, slight gradients are less smoothed out. In the frequency domain, the Gaussian filter corresponds to another Gaussian, providing a smooth and predictable attenuation of high frequencies.



GAUSSIAN FILTERS: NOISE REDUCTION

Observe how **Gaussian Low Pass Filters** perform better at removing salt & pepper noise while preserving the picture quality compared to the **Standard Low Pass Filters** with similar cut-off frequencies.

Original Image



Standard Filtered Image (Low-Pass)

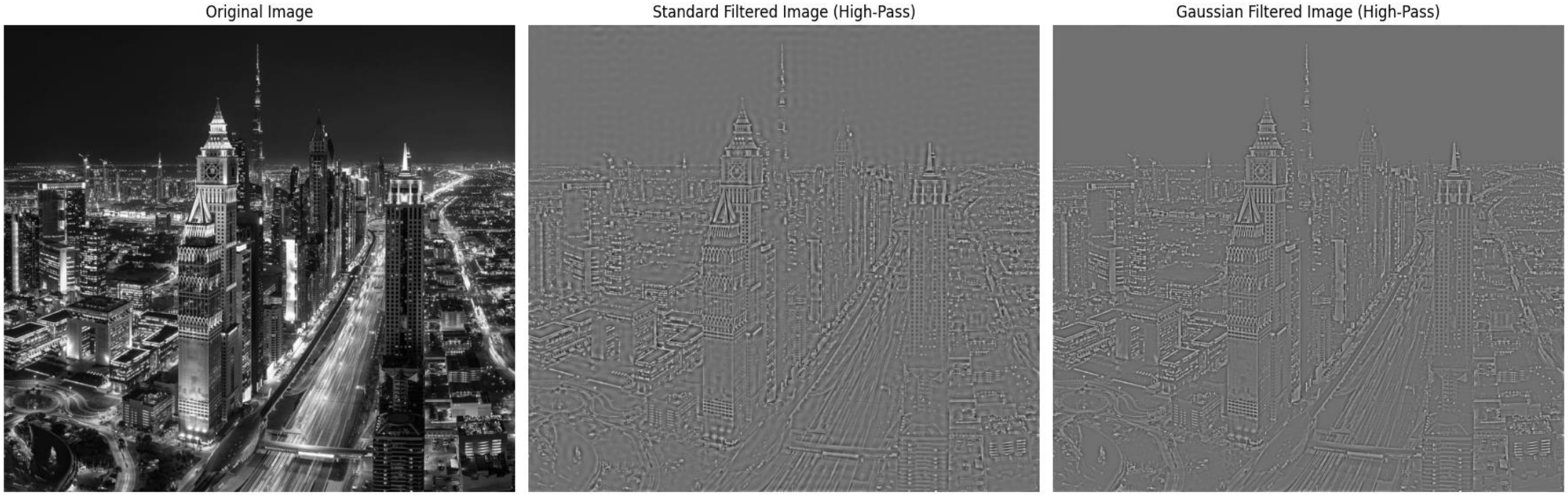
Gaussian Filtered Image (Low-Pass)

GAUSSIAN FILTERS: NOISE REDUCTION

Observe how **Gaussian Low Pass Filters** perform better at removing salt & pepper noise while preserving the picture quality compared to the **Standard Low Pass Filters** with similar cut-off frequencies.

GAUSSIAN FILTERS: EDGE DETECTION

Observe how a **Gaussian High Pass Filter** enhances the edges and reveals the textures/structures much more efficiently compared to a Standard High Pass Filter with a similar cut-off frequency.

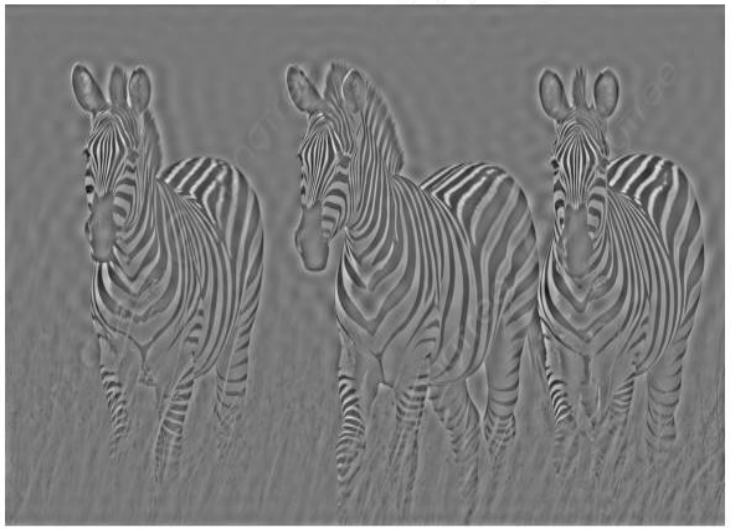


GAUSSIAN FILTERS: REVEALING TEXTURE

Observe how a **Gaussian High Pass Filter** enhances the edges and reveals the textures/structures much more efficiently compared to a Standard High Pass Filter with a similar cut-off frequency.

Original Image

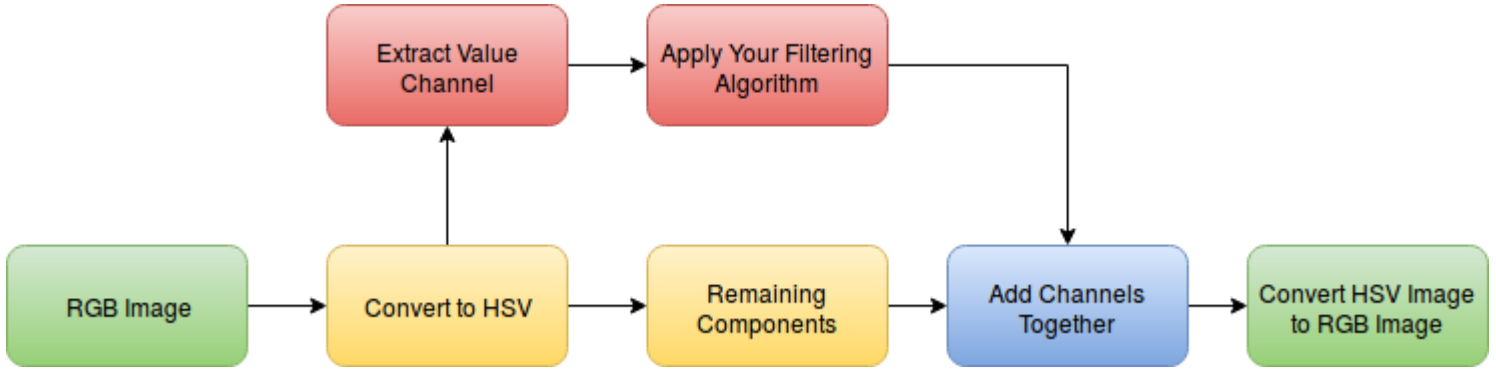
Standard Filtered Image (High-Pass)



Gaussian Filtered Image (High-Pass)

FILTERING IN COLOR IMAGES

Filtering in color images is typically not done by applying it to every channel of **RGB** color space. So, we need to move to an alternative color space like **HSV (Hue, Saturation, Value)** or **YCbCr (Luminance, Blue-difference, Red-difference)**, where the intensity (luminance) and color information (chrominance) are represented separately. **The filter is applied only to the luminance component**, such as the Y channel in YCbCr or the Value channel in HSV, while the chrominance components remain unchanged, preserving the color integrity. This ensures that only the brightness or structural details of the image are modified, leaving the colors unaffected. **After filtering, the modified luminance component is combined with the untouched chrominance components, and the image is converted back to RGB for display**. This approach avoids color distortions and is effective for enhancing details or reducing noise while maintaining accurate colors.

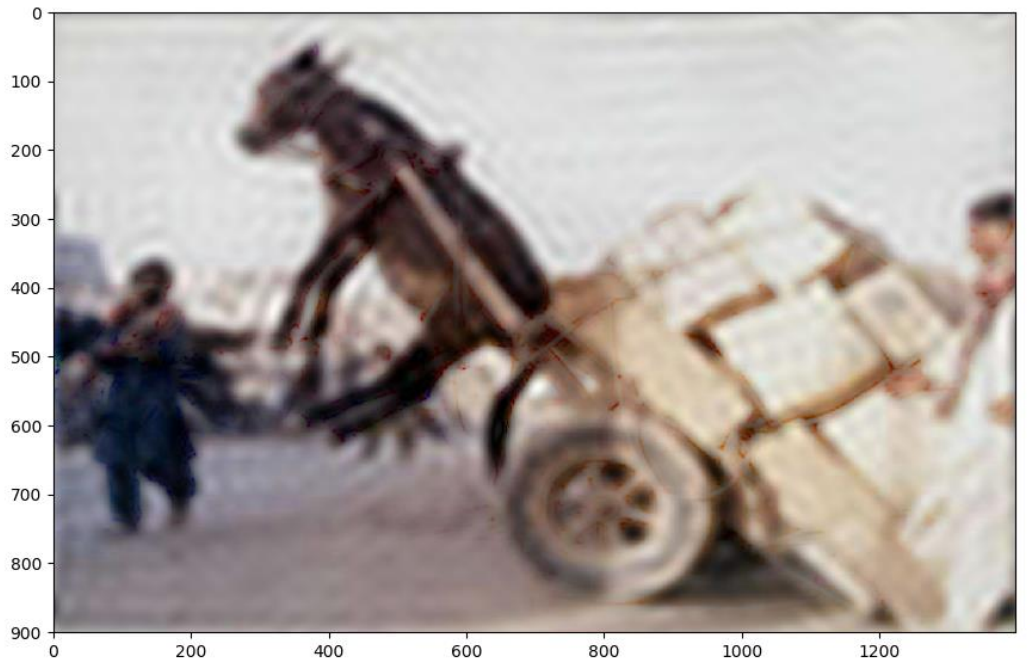


Filtering by Converting RGB Images to HSV Color Space

FILTERING IN COLOR IMAGES: LOW PASS

Observe the effect of applying low-pass filtering on the **Value** channel in the HSV color space and adding the filtered output to the untouched remaining channels (**Saturation** and **Hue**) to convert them back to RGB in the images below.

Original Image

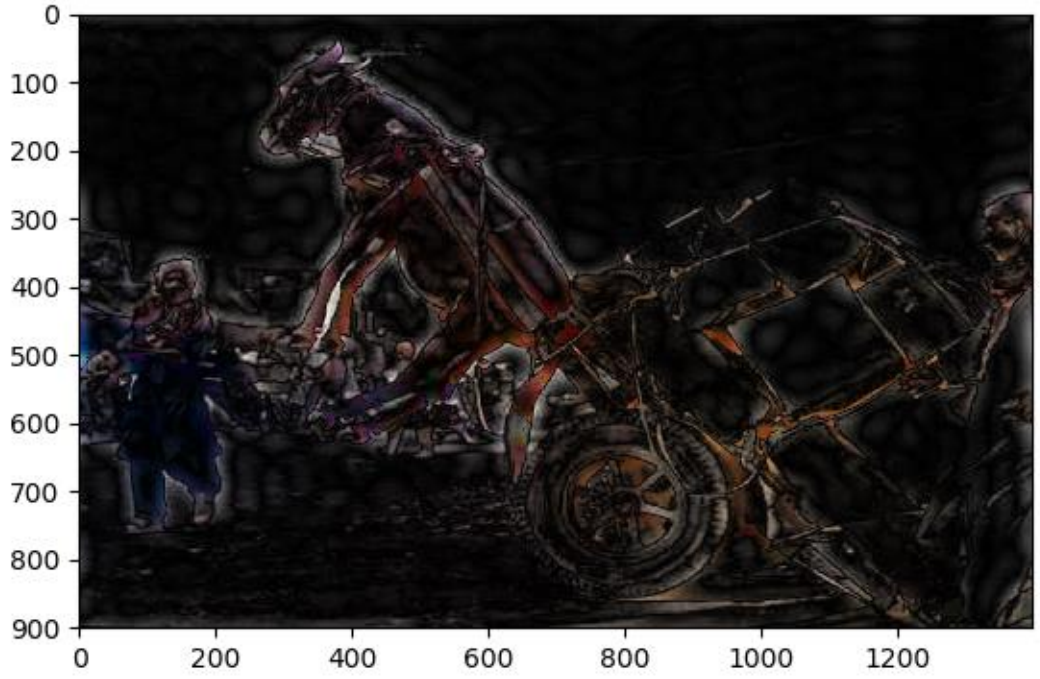


Filtered Image (Low-Pass)

FILTERING IN COLOR IMAGES: HIGH PASS

Observe the effect of applying high-pass filtering on the **Value** channel in the HSV color space and adding the filtered output to the untouched remaining channels (**Saturation** and **Hue**) to convert them back to RGB in the images below.

Original Image



Filtered Image (High-Pass)

HIGH BOOST FILTERING

High-boost Filtering in the frequency domain is an image processing technique used to enhance high-frequency components while preserving low-frequency content to a certain extent. It can amplify details such as edges, making the image appear sharper.

High-boost filtering is expressed as:

$$H_{hb}(u,v) = A \cdot H_{lp}(u,v) + (1 - H_{lp}(u,v))$$

Where:
$H_{hb}(u,v)$: High-boost filter
$H_{lp}(u,v)$: Low-pass filter
A : Boosting factor (A>1)

This can be decomposed into:

$$H_{hb}(u,v) = A \cdot H_{lp}(u,v) - (H_{lp}(u,v) - 1)$$

The boosting factor A determines how much emphasis is given to the original image (low frequencies).

HIGH BOOST FILTERING

Original Image

High-Boost Filtered (A=0.5)

HIGH BOOST FILTERING

Original Image

High-Boost Filtered (A=0.1)

HIGH BOOST FILTERING

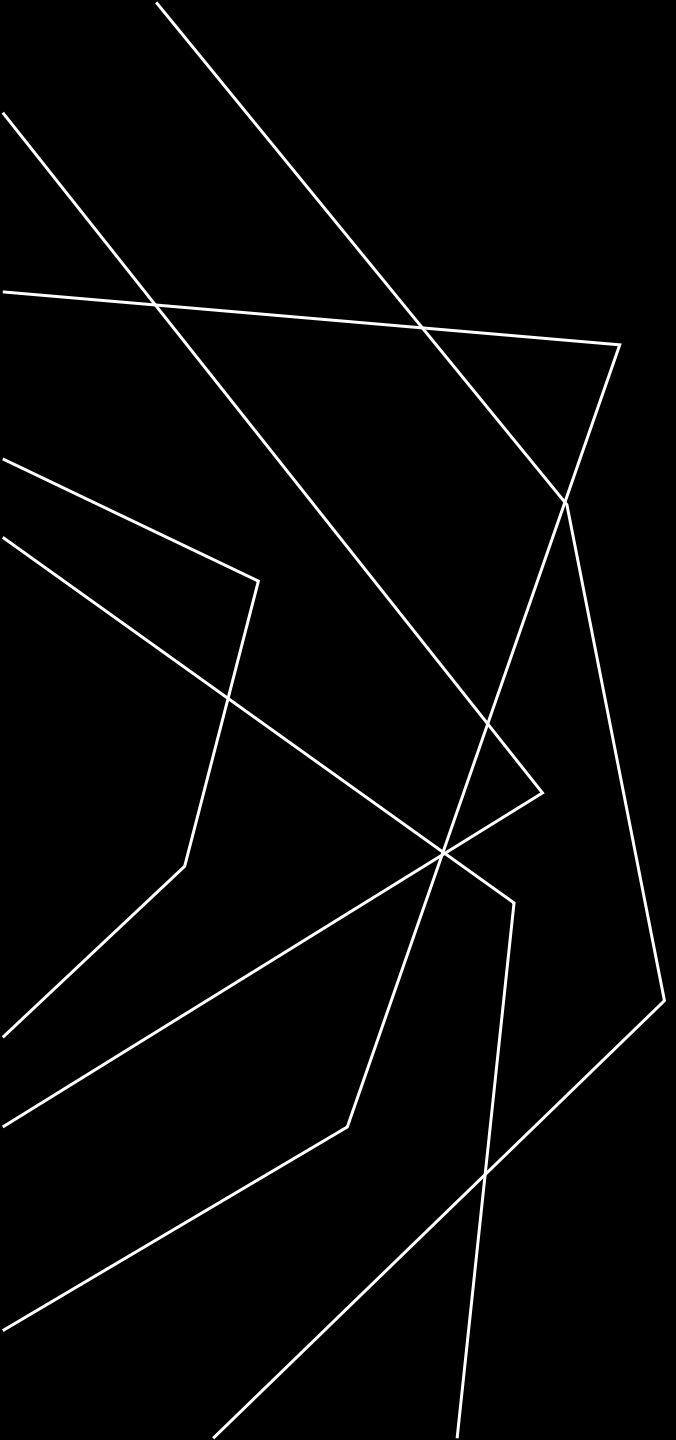
HIGH BOOST FILTERING

Original Image

High-Boost Filtered (A=-0.5)

REFERENCES

1. Image Processing Handbook – J. Russ, CRC Press, 2007.
2. Digital Image Processing – R. C. Gonzalez and R. E. Woods, Pearson, 2008.
3. OpenCV Documentation on Color Spaces: https://docs.opencv.org/2.4/modules/imgproc/doc/miscellaneous_transformations.html
4. https://sbme-tutorials.github.io/2021/cv/cv.html
5. https://longcreek.me/blog/2020/fourier-filtering
6. https://stackoverflow.com/questions/71907248/algorithm-for-periodic-noise-removal-in-images-using-band-reject-filters-notch



THANK YOU