Signal Project

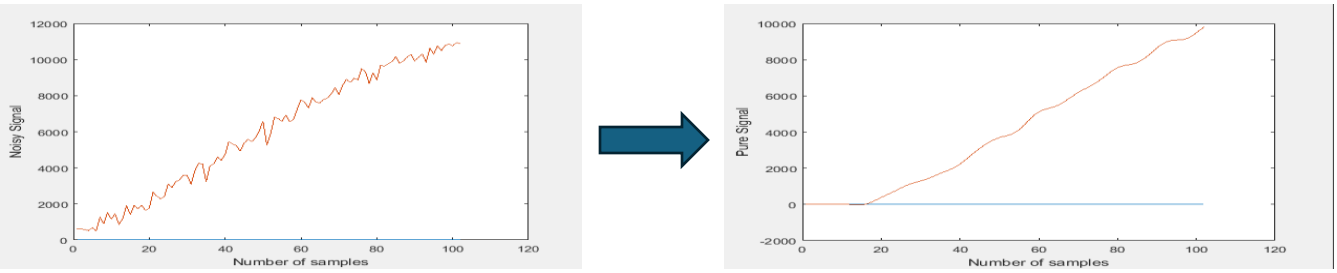
To : Dr Doaa Gamal

Prepared by:

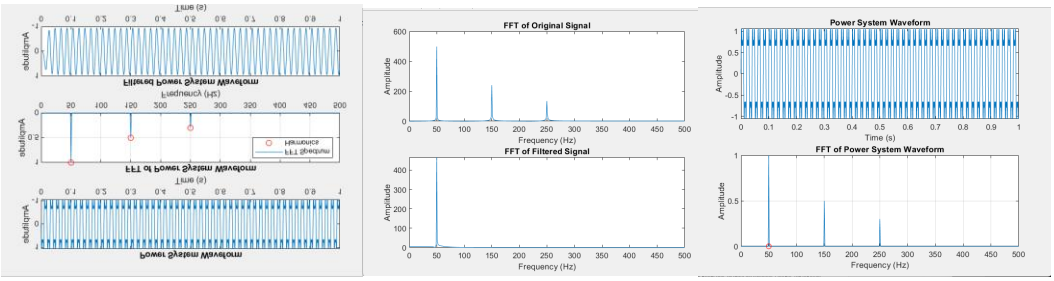
(1)	اميره سيد محمد علي	(لايحه جديده)
(2)	نادر جمال احمد محمد	(لايحه قديمه)
(3)	ندي سعيد احمد الدقي	(لايحه جديده)
(4)	ايه نبيل علي محمد	(لايحه جديده)
(5)	وجدي ممدوح صبري	(لايحه قديمه)

Projects We Have Made Using MATLAB (9 Projects):

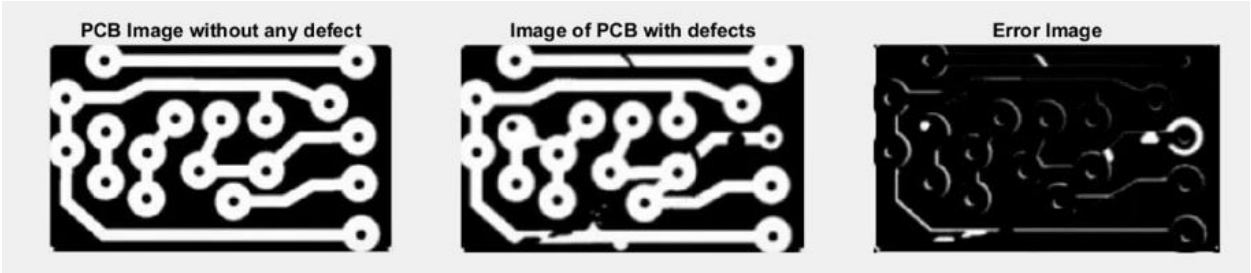
1)Noise Reduction of electrical power ouput of wind turbine:



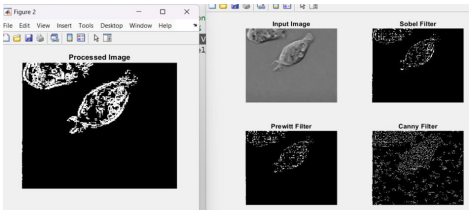
2)Power System Harmonic Analysis and remove Harmonics



3) Detect Defects in Products

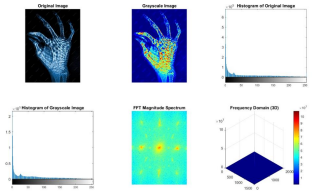


4)Detecting Cell

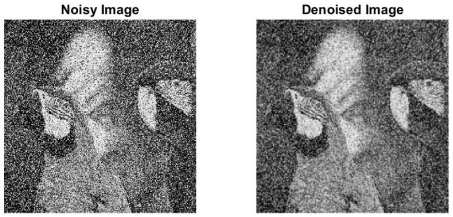


5) Person Detection With Printing Accuracy

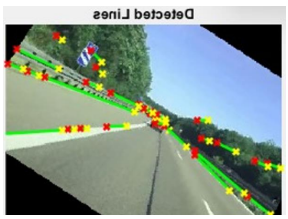
6)x-ray-imageFilter



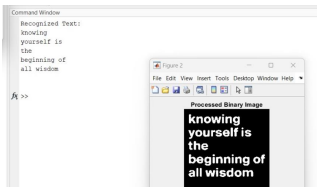
7)Denoising Image



8)Line Detection



9)Read Text From Image Using OCR



- **You Can access all codes of these Projects in this Repository:**

https://github.com/AmiraSayedMohamed/Learn_Matlab

- **We also have made A playlist explaining each Project and the code step by step here it's the link :**

https://youtube.com/playlist?list=PLH_uDwiVYu9zTKv9ByzZ05PmSx2Gw07OR&si=k721Mb3yM3RF938e

---

1) Noise Reduction of electrical power ouput of wind turbine:

Introduction:

Wind energy is considered one of the most important and inexpensive renewable energy. Wind turbines collect the kinetic energy of the air and convert it to electricty to help power the grid using mechanical power to spin a generator and creating electric power.

However, this electricity transported to the grid or the load could be accompanied with some sort of noise signal. This noise is existed due to several reasons. **Anemometer** which is a device that measures the wind speed and notifies the wind turbine controller when it is so windy and if it is OK to start the turbine again if it stopped. This device could be joined with some noise as it's considered to be a measuring device **(measurement noise).**

Almost all measurement devices have problmes with data processing because of noise which is mixed into measured signal. Very frequently measured quantity is lost inside the noise and filtering is inseparable activity during the measuring procedd

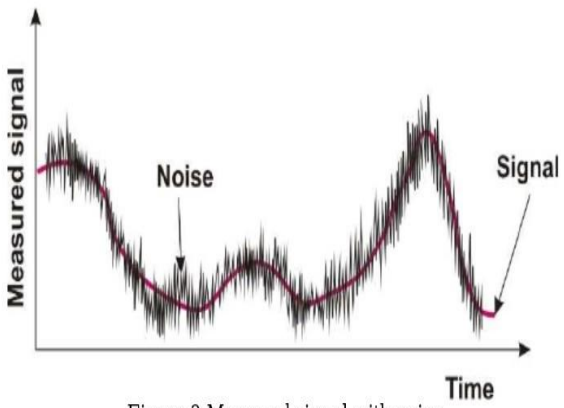


Figure 2 Measured signal with noise

Also, cables which are used for the transmission and distribution electrc power could be affected by noise from sending to receiving end. This noise comes from neighboring cables in the same route which can have **nonuniform electric field** which can create noise on the conductor of the cable and the signal inside the cable.

Noise can also be existed due to multiple wind turbines that don't operate at a **synchronized speed**.

One of the key digital signal processing (**DSP**) applications is noise reduction. In this application, a digital FIR filter removes noise in the signal that is contaminated by noise existing in the broad frequency range.

Avantages of noise reduction in wind turbines Systems:

1. **Higher efficiency**: in measuring and defining the relation beween Upstream wind velocity and mechanical output power.
2. **More accuracy** : high precision and perfomance in the generation and transmission operation of electricity from machine system.
3. **Increased clarity:** and remove the unwanted part of the signal.

The primary goal of this project is to:

4. Creating a simple and practical **wind turbine model** for .
5. **Analysize** this model and detect the unwanted noise affected the velocity and power measured relation .
6. using **DSP, Fast Fourier Transform (FFT), frequency spectrum analysis and matlab filtering algorithms** on the obtained data.
7. **Noise reduction** using suitable mechanisms of filtering and detect the filter properties and type required for this operation.

Steps and techiques Used are :

1. Firstly, we create a smiple **model** for the analysis operation of **wind turbine** and the operation generating electric power accordin to Energy Conversion Basics equations of wind energy which are:

$$\mathbf{Pm = 1/2\ \rho.A.V^3.C_p}$$

Where :
ρ = air density, 1.225 (kg / m3) .
A =area swept by the rotor blades, (m2).
V = velocity of the air, m/s.
Cp = the power coefficient of the rotor or rotor efficiency.

$$\boldsymbol{\lambda = \omega.R\ /V}$$

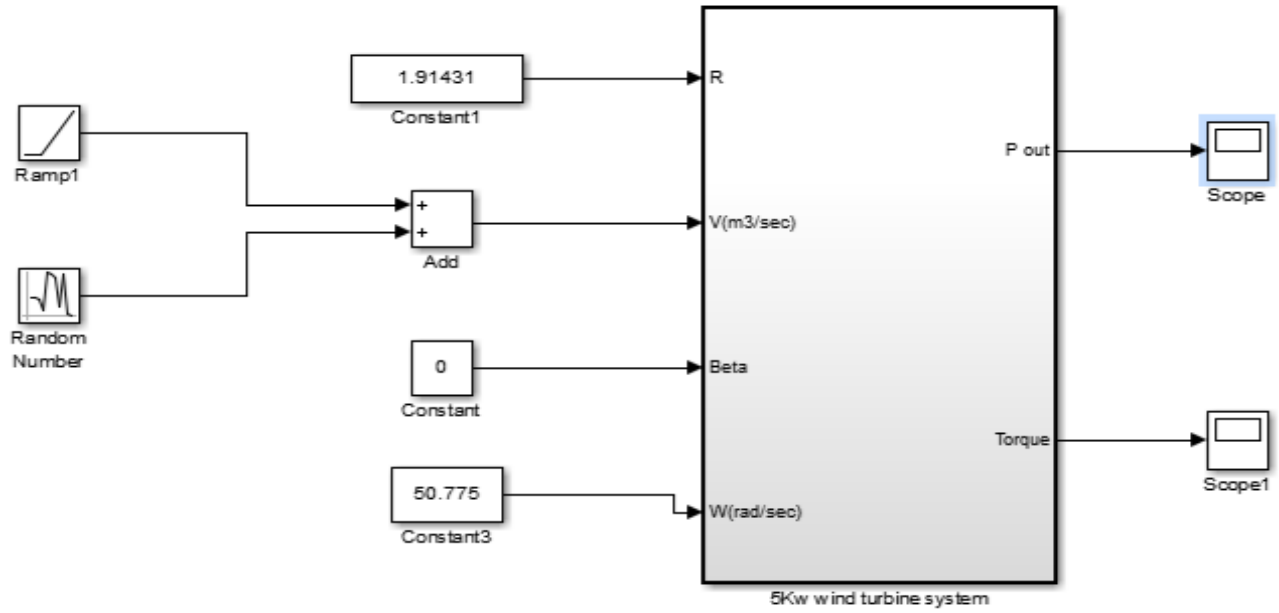
where:
λ: Tip speed ratio.
ω: angular speed.
R: rotor radius.

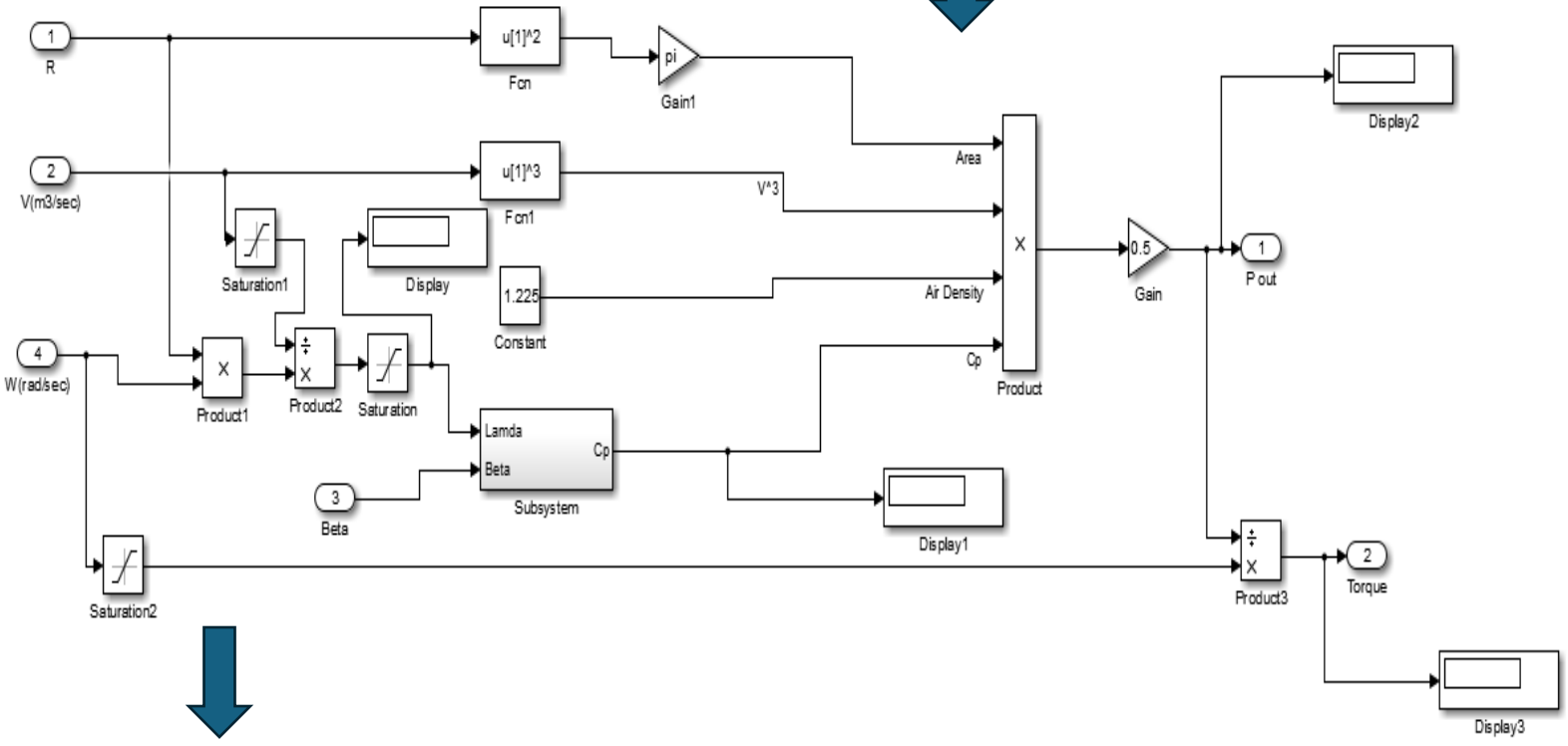
$$\mathbf{Cp(\lambda,\beta)=}C1(C2/\ \lambda i - C3^*\ \beta - C4^*\ \beta\hat{}x - C5)^*\exp(-C6/\ \lambda i).$$

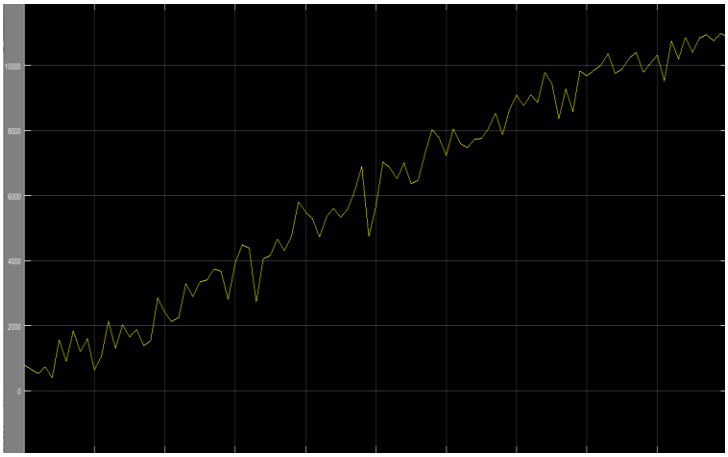
1/ λi = (1/ λ+0.08 β) – (0.035/1+ β^3).

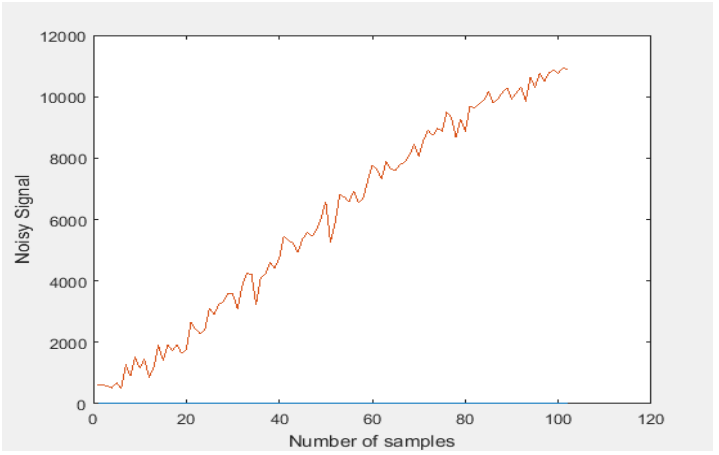
Where :
B: Blade pitch angle (deg).
C1 to C6 : are constants.

For Power= 5kw :
λ =8.1, Cp(λ,β)=0.41034 , Vw=12 (m/s), β=0, R=1.91431
C1=0.5, C2=116, C3=0.4, C4=0, C5=5, C6=21,
λi =11.304, W=50.775



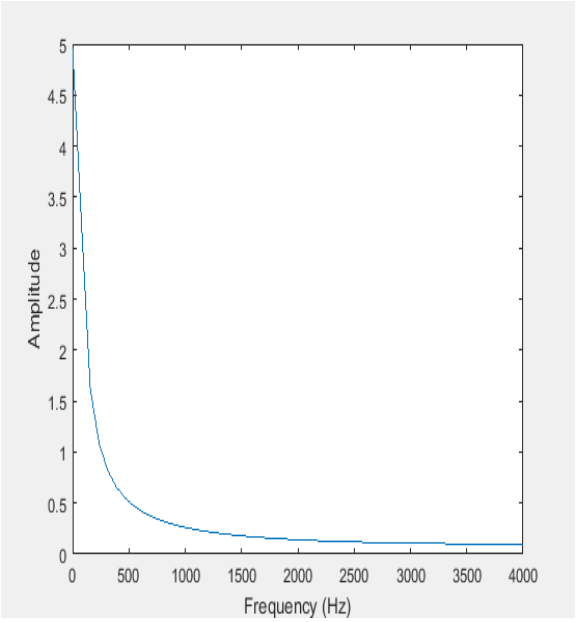






2.using Fast Fourier Transform (FFT) in matlab: we need to understand the **spectral** component of the signal and where our signal lies and what are the noisy component which we want to remove. So by knowing the **frequency range** our signal liesthen we can directly **deign the filter a**nd this is down by **FFT function.**

We also need to convert the frequency response of the signal into single-sided spectrum, in the one-sided amplitude spectrum, the negative-indexed frequency components are added back to the corresponding positive-indexed frequency components; thus each amplitude value other than the DC term is doubled. It represents the frequency components up to the folding frequency.



3. **Designig digital FIR filter**: it removes noise in the signal that is contaminated by noise existing in the broad frequency range. We can see that noise is broadband, existing from 1000 Hz to the folding frequency of 4,000 Hz.

Assuming that the desired signal has a frequency range of only 0 to 800 Hz, we can filter noise from 800 Hz and beyond. A lowpass filter would complete such a task. **Passband** frequency range: 0 Hz to 800 Hz and **Stopband** frequency range: 1 kHz to 4 kHz.

lowpass filtering will remove the noise ranging from 1,000 to 4,000 Hz, and hence the signal-to-noise power ratio will be improved.

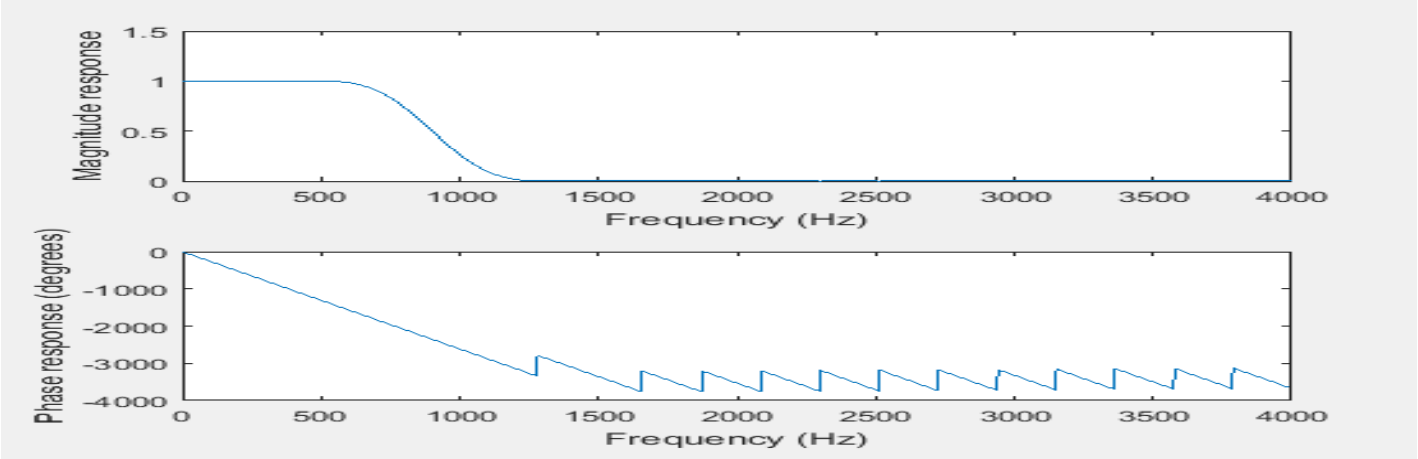
Calculations:

- Filter type = lowpass FIR filter.
- Passband frequency range = 0–800 Hz.
- Stopband frequency range = 1,000–4,000 Hz
- Window type = Rectangular window.

- cutoff frequency = (1000+800)/2 = 900 Hz.
- Delta_f = (1000 – 800)/8000 = 0.025.
- N = 0.9/Delta_f = 36, Choose odd number N = 37:

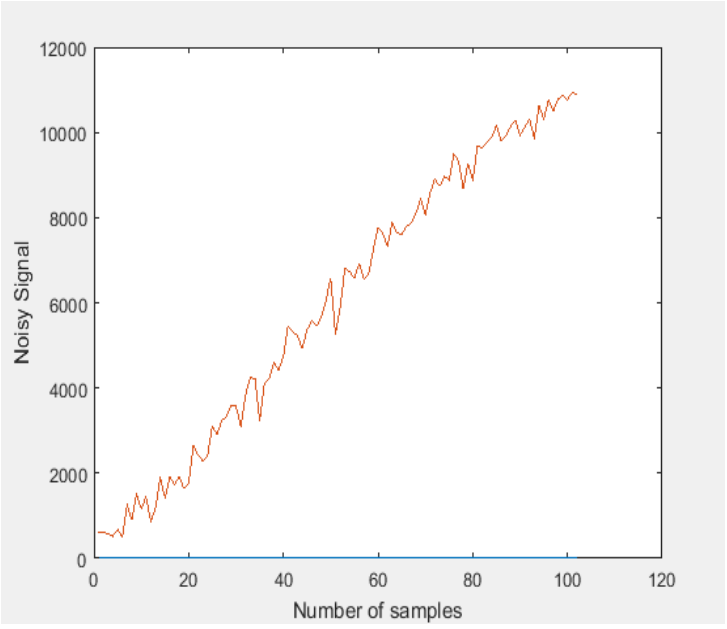
Frequency response of the designed low pass filter:

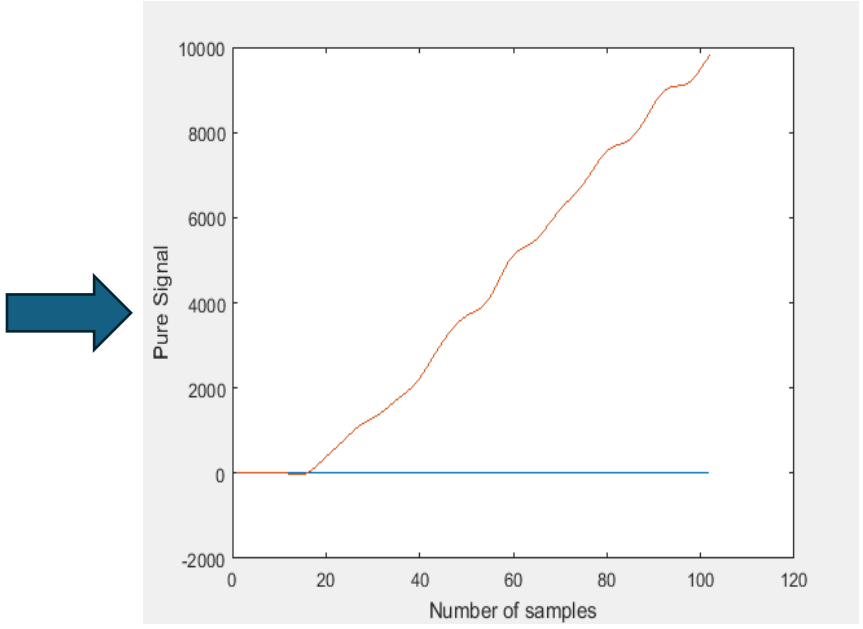
- Using **Freqz** function : Evaluate Frequency response of digital filter.
- And **Fir1**: FIR filter design of N'th order lowpass FIR filter.



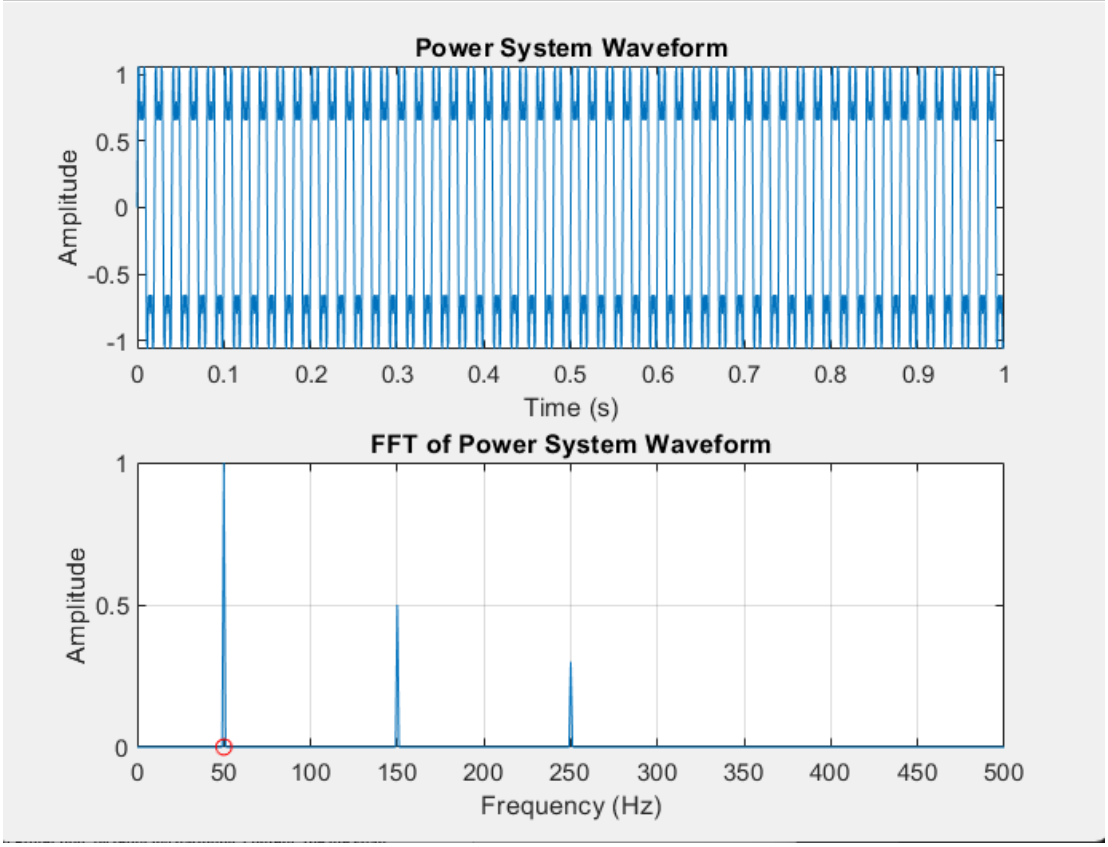
Performing digital filtering:

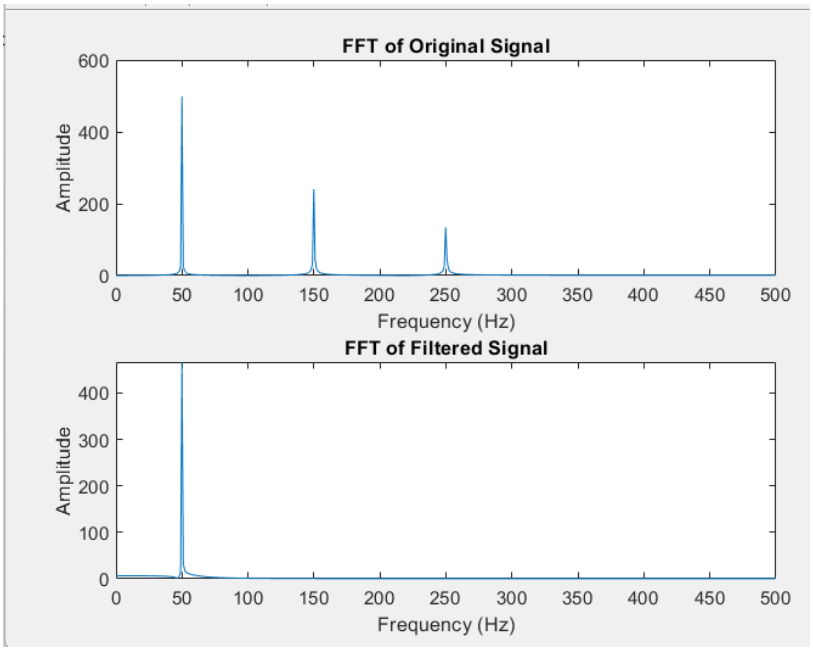
- Using **Filter** function: Y = **filter(B,A,X)** filters the data in vector X with the filter described by vectors A and B to create the filtered data Y.
- Notice that since we use the higher-order FIR filter, the signal experiences a **linear phase delay** of **(2M+1)** = 37 so it's equal to **18 samples**, as is expected.
- As we see below, a huge part of noise has been reduced from the signal to become almost pure and noiseless signal.





2)Power System Harmonic Analysis and Removal: Project Report





Introduction

In power systems, harmonics are unwanted frequencies that occur as integer multiples of the fundamental frequency (typically 50 or 60 Hz). These harmonics can cause equipment

malfunction, overheating, and energy inefficiency. The objective of this project is to analyze and remove harmonics from a power system waveform using **Fast Fourier Transform (FFT)** and **Filtering Techniques**.

This report discusses the techniques and steps used in the project and provides an explanation of the code implemented to detect and remove harmonics in the system. Additionally, we will analyze the results through graphs and visualizations.

Objective and Benefits

The primary goal of this project is to:

1. **Analyze power system waveforms** containing fundamental and harmonic frequencies.
2. **Detect harmonics** in the waveform using FFT.
3. **Remove unwanted harmonics** through a low-pass Butterworth filter.

Benefits of Harmonic Removal in Power Systems:

- **Improved Power Quality**: Reducing harmonics helps in improving the overall quality of the power supply.
- **Protection of Equipment**: Harmonics can cause significant damage to electrical equipment. By filtering them, we reduce the risks of overheating and failure of electrical components.
- **Efficiency Enhancement**: By removing unnecessary harmonic frequencies, the system becomes more efficient in transmitting power.

Techniques and Steps Used in the Code

1. Signal Generation

The power system waveform is generated by adding a fundamental frequency (50 Hz) and its harmonics at 150 Hz (second harmonic) and 250 Hz (third harmonic). This waveform represents a typical power supply with distortions caused by harmonics.

2. Harmonic Detection Using FFT

The **Fast Fourier Transform (FFT)** is applied to convert the time-domain signal into the frequency domain. FFT allows us to identify the frequency components present in the signal, including the fundamental and harmonic frequencies.

- The fft function is used to compute the FFT.
- The magnitude spectrum is plotted, highlighting the peaks at the harmonic frequencies (fundamental, second harmonic, and third harmonic).

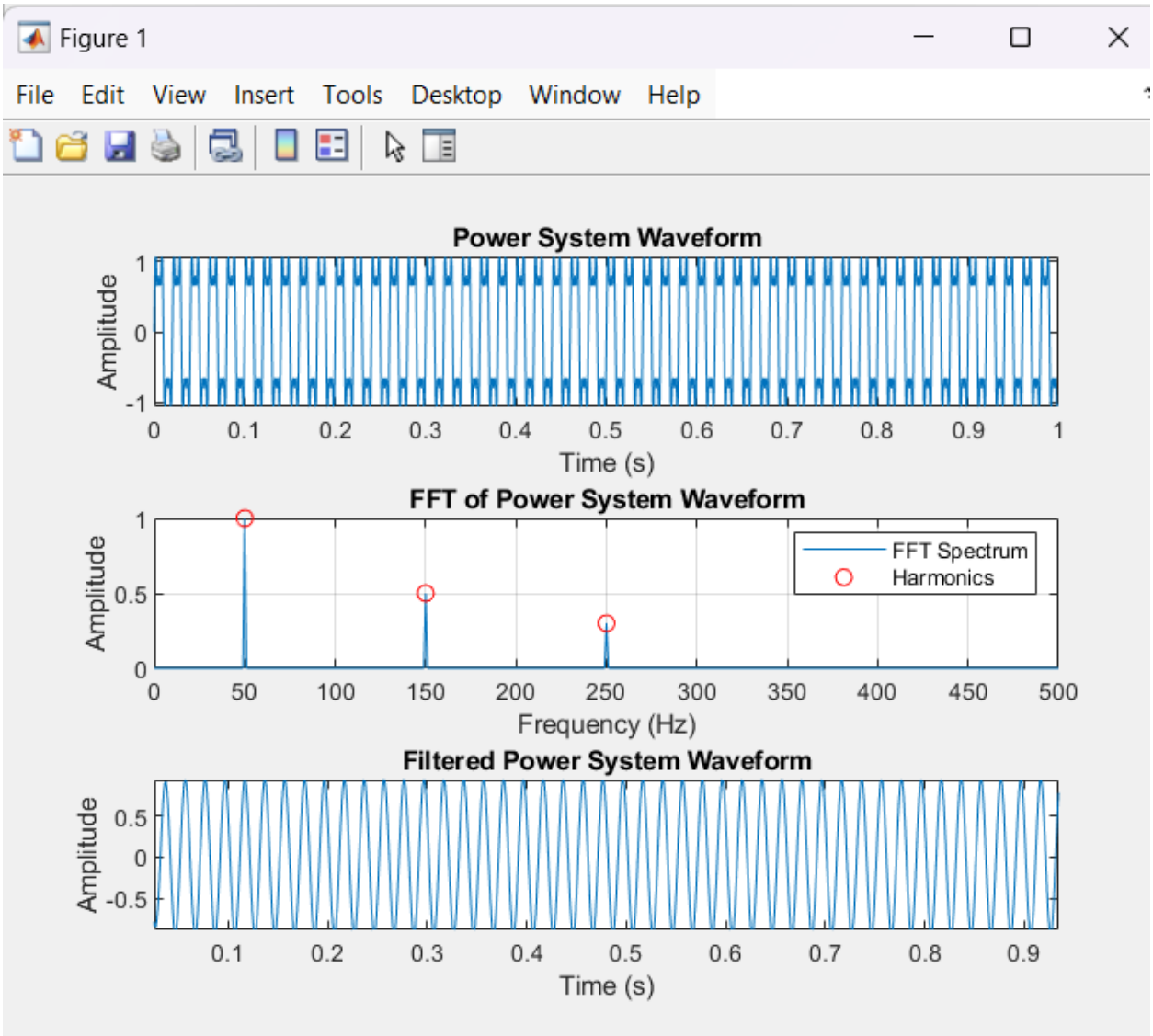
3. Harmonic Filtering Using a Low-pass Filter

A **Butterworth low-pass filter** is applied to remove frequencies above a specified cutoff frequency (60 Hz). This filter attenuates high-frequency harmonics while preserving the fundamental frequency of the signal.

- A 6th-order Butterworth filter is designed using the butter function.
- The filter's frequency response is applied to the waveform to obtain a filtered signal with reduced harmonics.

Graph Explanations

Figure 1: Time-Domain Signal and FFT of Power System Waveform

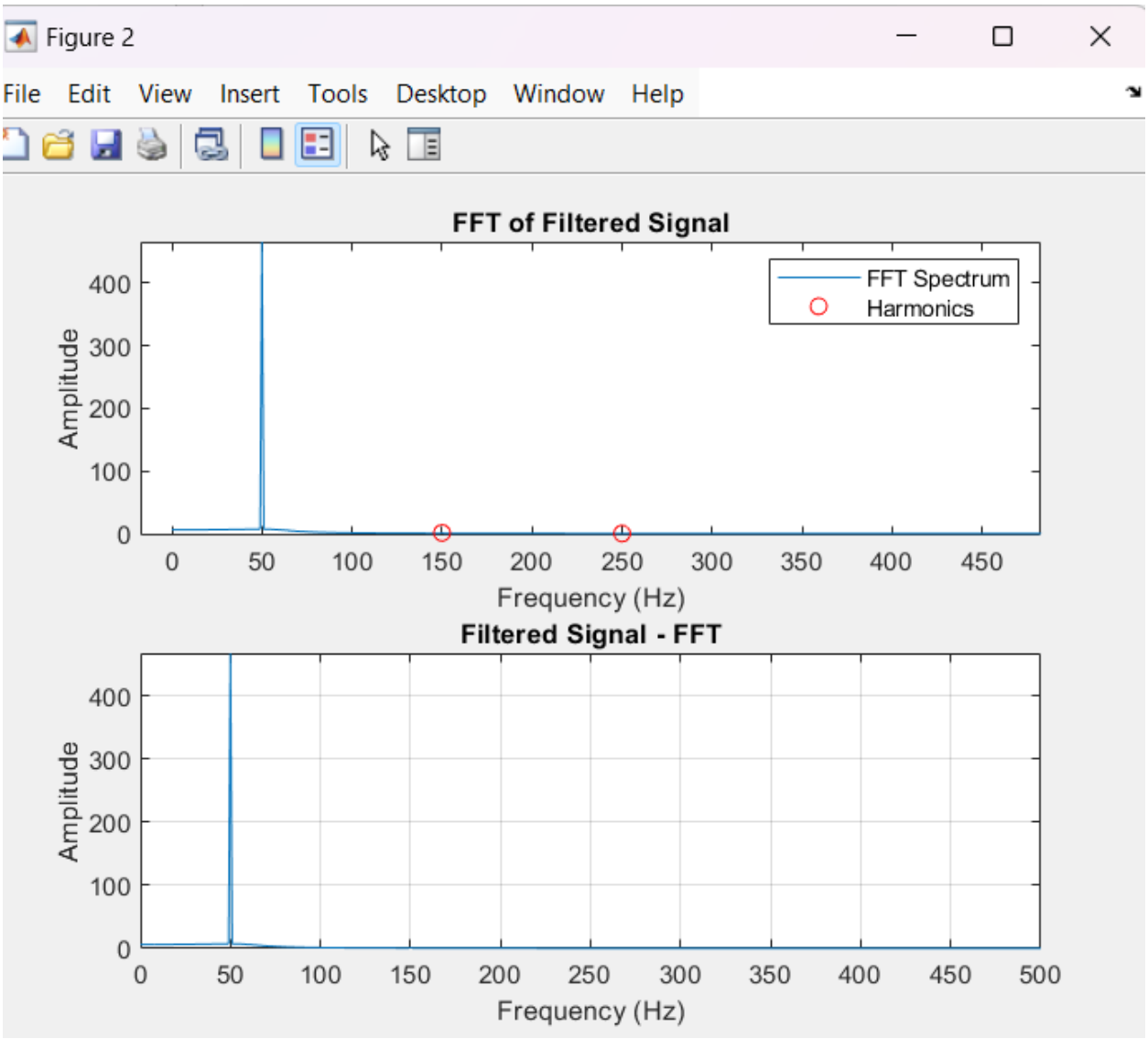


- **Time-Domain Signal** (Top graph):
    - The first graph shows the **original power system waveform** which contains the fundamental frequency (50 Hz) along with the second (150 Hz) and third (250 Hz) harmonics.

- The waveform is a sum of sinusoidal signals with different amplitudes and frequencies.

- **FFT of Power System Waveform** (Middle graph):
  - The second graph shows the **frequency domain representation** of the original signal.
  - The peaks at 50 Hz, 150 Hz, and 250 Hz represent the **fundamental frequency** and **harmonics**.
  - These frequencies are clearly visible, indicating the presence of harmonic distortions in the power system.

Figure 2: Filtered Signal and FFT of Filtered Signal



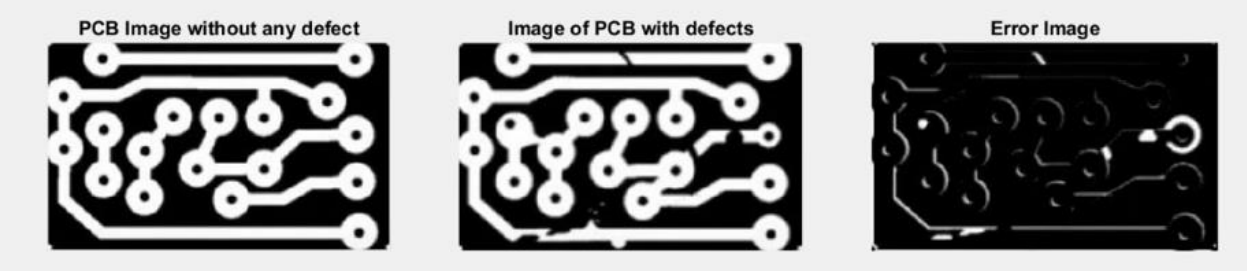
- **Filtered Power System Waveform** (Bottom graph):
  - This graph shows the **filtered waveform** after applying the low-pass filter.
  - As expected, the waveform now appears smoother, and the high-frequency harmonics have been attenuated. The fundamental frequency is preserved, but the higher-order harmonic frequencies are significantly reduced.

- **FFT of Filtered Signal** (Bottom graph on the second figure):
    - The FFT of the filtered signal is shown to compare with the original FFT.
    - The peaks corresponding to harmonics (150 Hz, 250 Hz) are no longer visible, indicating that the filter has successfully removed the unwanted harmonics.

---

3) Detect Defects in Products

(PCB Defect Detection Using Image Processing)



1. Introduction

The detection of defects in printed circuit boards (PCBs) is a crucial task in the electronics manufacturing process, as it ensures the quality and functionality of the final product. This project focuses on developing a simple image processing method for detecting defects on PCBs by comparing a reference image (a PCB without defects) with an image of the PCB that may contain defects. The key steps involve loading the images, resizing them to match the dimensions, and calculating the error (or difference) between the reference and defective images.

2. Objective

The main objective of this project is to compare two PCB images—one without defects and one with potential defects. By doing so, the differences between the two images are visualized, helping to detect discrepancies that indicate defects. This process uses basic image manipulation techniques such as image subtraction to highlight areas of difference between the two images.

3. Methodology

The methodology followed in this project is as follows:

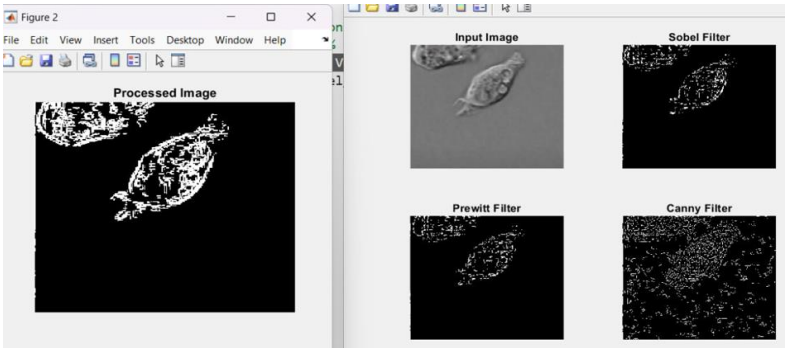
1. Image Loading: The first step involves loading two PCB images—one representing a PCB without defects and the other representing a defective PCB. These images are read using the imread function in MATLAB.
2. Image Resizing: The image with defects is resized to match the dimensions of the reference image using the imresize function. This ensures that the images are of the same size and alignment before performing the comparison.

3. Image Display: A figure is created with three subplots for visual comparison. The three images displayed are:
    - The reference PCB image without defects.
    - The defective PCB image.
    - The error image (difference between the reference and defective images).
4. Error Calculation: The difference between the reference and defective images is calculated by subtracting one image from the other. The result is displayed as the "error image," where discrepancies (or defects) between the two PCB images will be clearly visible.
5. Visualization: The imshow function is used to display the images. Additionally, subplot spacing is adjusted for better visualization using the Position property of the subplot axes.

---

4) Detecting Cell

(Edge Detection and Morphological Processing of an Image 😊) :



Introduction

Edge detection is a critical aspect of image processing used to identify boundaries of objects within an image. The goal of this project is to demonstrate the application of various edge detection filters, namely Sobel, Prewitt, and Canny, on an input image. Furthermore, morphological operations like dilation are applied to enhance the results. The project involves processing an image of a cell and comparing the results from different edge detection techniques.

Objective

- Objective: To apply and compare different edge detection methods (Sobel, Prewitt, and Canny) on an image and to enhance the results through morphological processing using dilation.

- Image Used: An image of a cell, referred to as "cell.jpg," is used as the input for edge detection.

Methodology

The project follows these main steps:

1. Image Preprocessing:
    - The input image is loaded and displayed.
    - The image is converted to grayscale, as edge detection algorithms generally work on single-channel images.
2. Edge Detection:
    - Three different edge detection filters are applied to the grayscale image:
        - Sobel Filter: Detects edges by calculating the gradient of the image intensity at each pixel.
        - Prewitt Filter: Similar to Sobel, but uses a different kernel for detecting horizontal and vertical edges.
        - Canny Filter: A multi-stage algorithm that provides better edge detection through noise reduction and edge tracing by hysteresis.
3. Morphological Processing (Dilation):
    - After applying the Sobel filter, dilation is applied using horizontal and vertical structuring elements to enhance the edges.
4. Results:
    - The results of edge detection and morphological processing are displayed for analysis.

Techniques Used

1. Grayscale Conversion:

- The input image is first converted from RGB to grayscale using MATLAB's rgb2gray() function, as edge detection techniques are most effective on single-channel images.

2. Edge Detection Filters:

- Sobel Filter: The Sobel operator is used to compute the gradient magnitude at each pixel in the image. It highlights edges by looking for areas of rapid intensity change in both horizontal and vertical directions.
- Prewitt Filter: Similar to the Sobel operator, but it uses a different kernel. It is used for detecting edges based on a similar principle, but it is more sensitive to diagonal edges compared to the Sobel filter.
- Canny Filter: The Canny edge detector is a multi-step algorithm that provides more accurate edge detection by:
    - Smoothing the image with a Gaussian filter.

- Finding intensity gradients.
- Applying non-maximum suppression to thin edges.
- Using hysteresis to track edge continuity and remove false edges.

3. Morphological Processing (Dilation):

- Dilation is applied to the edges obtained from the Sobel filter to enhance and thicken the edges. Dilation is performed using horizontal and vertical structuring elements created with the strel() function.

---

5) Person Detection With Printing Accuracy

Introduction

People detection in images is a crucial task in computer vision that has a wide range of applications, such as security surveillance, autonomous vehicles, and human-computer interaction. This project demonstrates the use of MATLAB's built-in functions and the **PeopleDetector** object from the Computer Vision Toolbox to detect people in an image. The algorithm applies a detector to identify the presence of people in the input image, draws bounding boxes around detected individuals, and displays the detection scores.

Objective

The primary objective of this project is to detect people within an image using MATLAB's **PeopleDetector** object and visualize the results. The goal is to:

- Load and display an image.
- Apply the people detection algorithm to identify individuals in the image.
- Draw bounding boxes around detected people and display detection scores.
- Handle cases where no people are detected in the image.

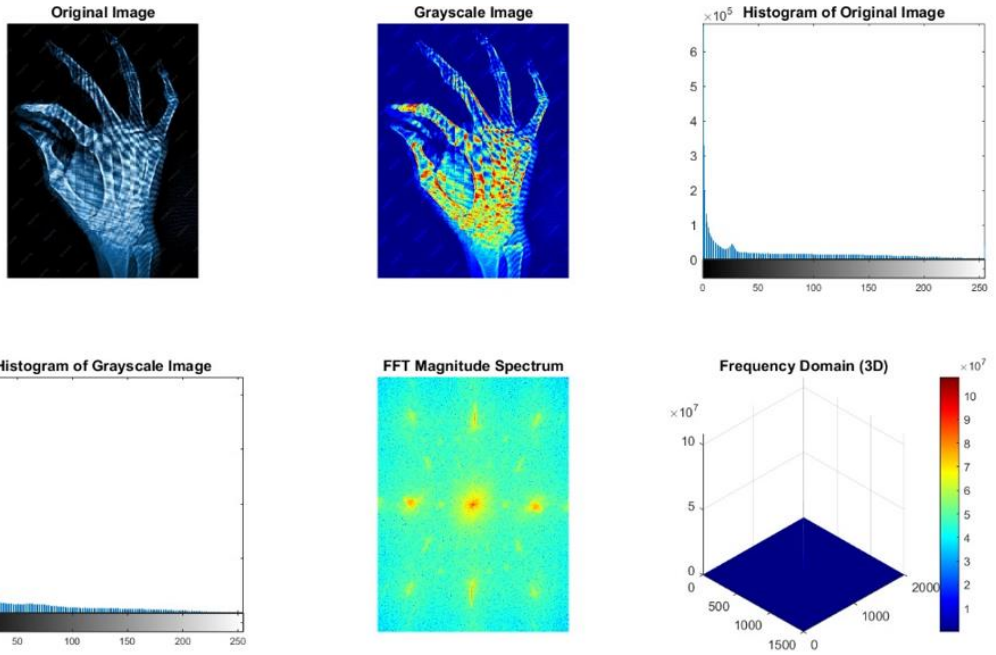
Methodology

1. **Image Loading and Display**:
    - The project starts by loading the image using MATLAB's imread() function. The image is then displayed using imshow().
2. **People Detection**:

- The vision.PeopleDetector object is used to detect people in the input image. This detector is based on machine learning models that are capable of identifying human figures based on features such as shape, size, and texture.
- The function peopleDetector(x) returns bounding boxes (bboxes) around detected people along with a score indicating the confidence of the detection.

3. **Bounding Box Annotation**:
   - If people are detected in the image (i.e., if the sum of the bounding box coordinates is not zero), the bounding boxes are drawn on the image using the insertObjectAnnotation() function.
   - The resulting image with annotated bounding boxes is displayed, showing the detected people and the detection scores.
4. **Handling No Detection**:
   - If no people are detected, the original image is displayed without any annotations, and the title "No People Detected" is shown.

6)x-ray-imageFilter



. **Introduction**

X-ray imaging is a vital tool in medical diagnostics and industrial applications. Enhancing X-ray images using computational techniques can help improve visualization and analysis. This project involves applying filtering and transformation techniques to an X-ray image, analyzing its characteristics in the spatial and frequency domains, and visualizing the results.

The implemented approach uses MATLAB to:

1. Read and preprocess the X-ray image.
2. Convert the image to grayscale for simplicity.
3. Apply Fast Fourier Transform (FFT) to examine frequency-domain characteristics.
4. Visualize the image and its transformations through various plots.

Objectives

1. Enhance and analyze an X-ray image using frequency-domain techniques.
2. Generate meaningful visualizations for both the spatial and frequency domains.
3. Explore the relationship between the image's pixel intensity and its frequency components.

Methodology

1. Reading and Preprocessing the Image

- The original image is loaded using the imread function.
- The image is converted to grayscale using rgb2gray, as color information is not critical for X-ray analysis.

2. Frequency Domain Analysis

- The 2D Fast Fourier Transform (FFT) is applied to the grayscale image using fft2. This transformation converts the spatial domain representation into the frequency domain.
- The zero-frequency component is shifted to the center for better visualization using fftshift.
- The magnitude spectrum is computed in log scale (log(1 + abs(fftShifted))) to enhance the dynamic range of frequency components for visualization.

3. Visualization

The project employs six subplots to visualize:

1. The original X-ray image.
2. The grayscale version of the image.
3. The histogram of pixel intensities in the original image.
4. The histogram of pixel intensities in the grayscale image.
5. The FFT magnitude spectrum.
6. A 3D representation of the frequency domain.

Results

1. Original Image and Grayscale Conversion

- The original X-ray image was successfully read and displayed.
- Conversion to grayscale reduced complexity while preserving important intensity information.

2. Frequency Domain Analysis

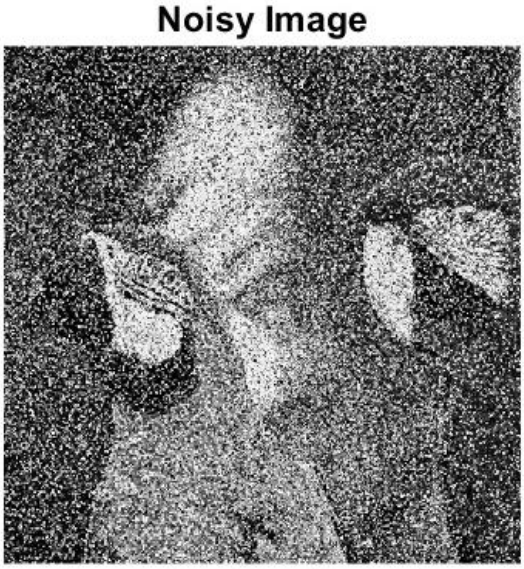
- The FFT magnitude spectrum revealed the frequency components present in the image. High-intensity areas in the spectrum correspond to dominant frequencies.
- A 3D surface plot of the frequency domain provided a more intuitive visualization of frequency amplitude variations.

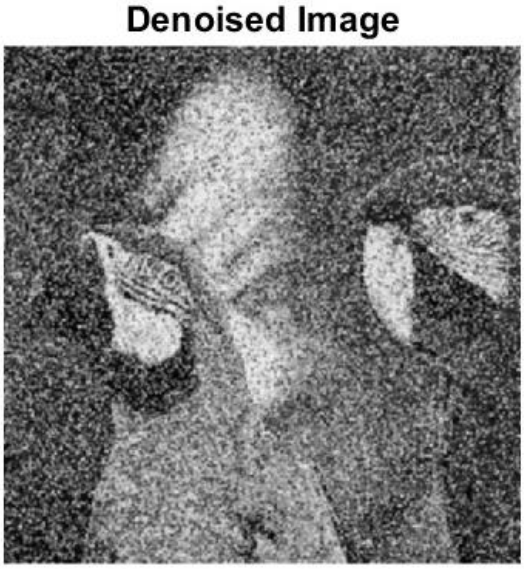
3. Histogram Analysis

- The histograms of the original and grayscale images highlighted the distribution of pixel intensities, aiding in understanding the image's contrast and dynamic range.

7)Denoising Image

Using Gaussian Filter





Introduction

Image denoising is a critical process in digital image processing, aimed at improving the quality of images that are corrupted by noise. Noise can be introduced during image capture due to factors like low light, sensor errors, or compression artifacts. The purpose of this project is to demonstrate the application of a Gaussian filter to remove noise from an image, thereby enhancing its quality and making it more suitable for further analysis or processing.

In this project, we use a Gaussian filter to perform the denoising operation on an image that has been artificially corrupted by noise. The project showcases how a standard image processing technique can be applied to reduce noise while preserving important details in the image.

Objective

The goal of this project is to:

- Read and display a noisy image.
- Apply a Gaussian filter for image denoising to reduce the noise.
- Compare the original noisy image with the denoised image.
- Adjust the layout for clear visualization of the before and after images.

Methodology

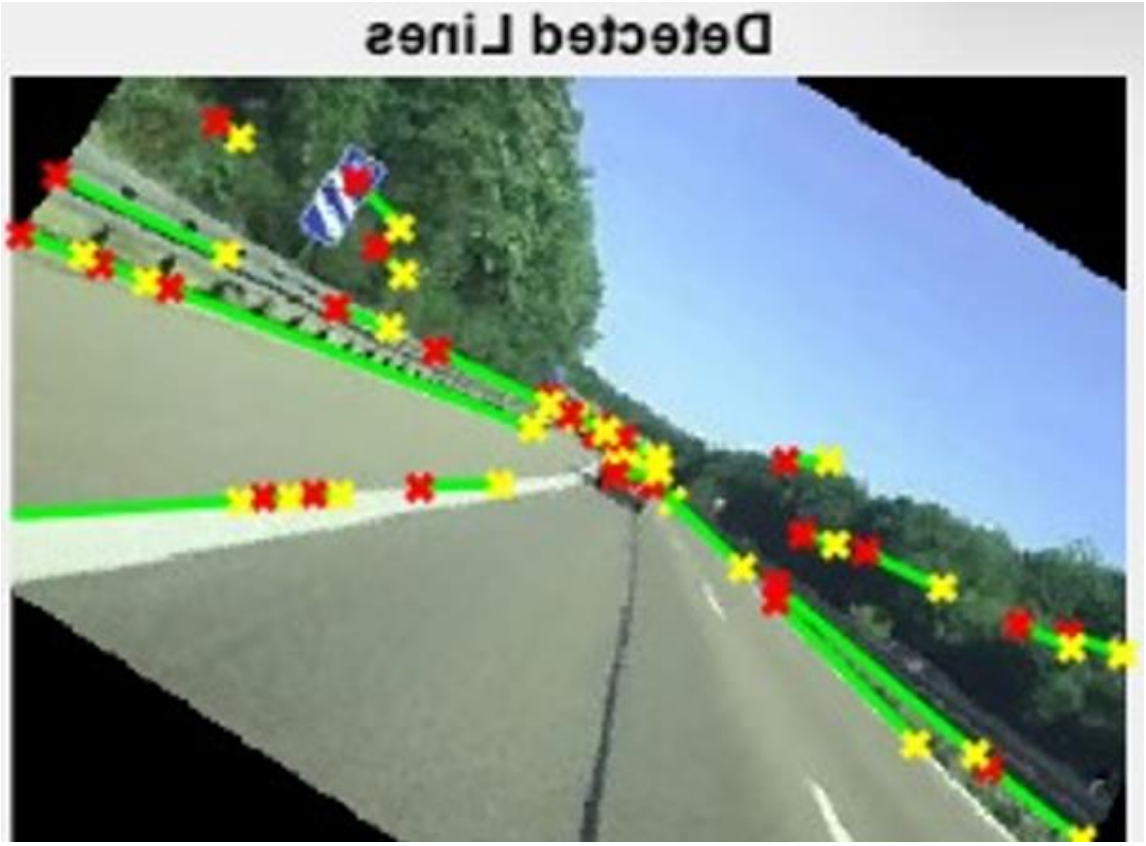
1. Reading the Noisy Image:
    - The first step involves loading the noisy image using the imread() function. If the image is a color image, it is converted into grayscale using rgb2gray() since the Gaussian filter is typically applied to grayscale images.
2. Gaussian Filter Application:
    - A Gaussian filter is applied to the image using the imgaussfilt() function in MATLAB. This filter uses a Gaussian function to smooth the image and

remove high-frequency noise. The parameter sigma, which represents the standard deviation of the Gaussian kernel, controls the amount of smoothing. Larger values of sigma result in stronger smoothing, which can be adjusted according to the level of noise.

3. Visualization:
    - The project uses MATLAB's subplot() function to display both the noisy image and the denoised image side by side for comparison. The figure is resized using set(gcf, 'Position', [100, 100, 800, 400]) for better visualization. The titles of the images are set with appropriate font size for clarity.
4. Layout Adjustment:
    - The figure window is resized and the background color is set to white using set(gcf, 'Color', 'w') to make the display cleaner and more visually appealing.
    - ---

8)Line Detection

Using Hough Transform:



Introduction

Line detection is a fundamental task in computer vision and image processing, which plays a crucial role in various applications such as road lane detection, object recognition, and image segmentation. In this project, the Hough Transform is applied to detect straight lines within an image. Specifically, the task involves rotating an image, converting it to grayscale, detecting edges, and using the Hough Transform to identify lines in the image. The detected lines are then visualized on the image with highlighted endpoints to provide a clear understanding of the line detection process.

Objective

The objective of this project is to:

1. Apply the Hough Transform on a rotated image.
2. Detect edges in the image using the Canny edge detection method.
3. Use the Hough Transform to detect lines in the image.
4. Visualize and annotate the detected lines on the image.
5. Display the total number of detected lines.

Methodology

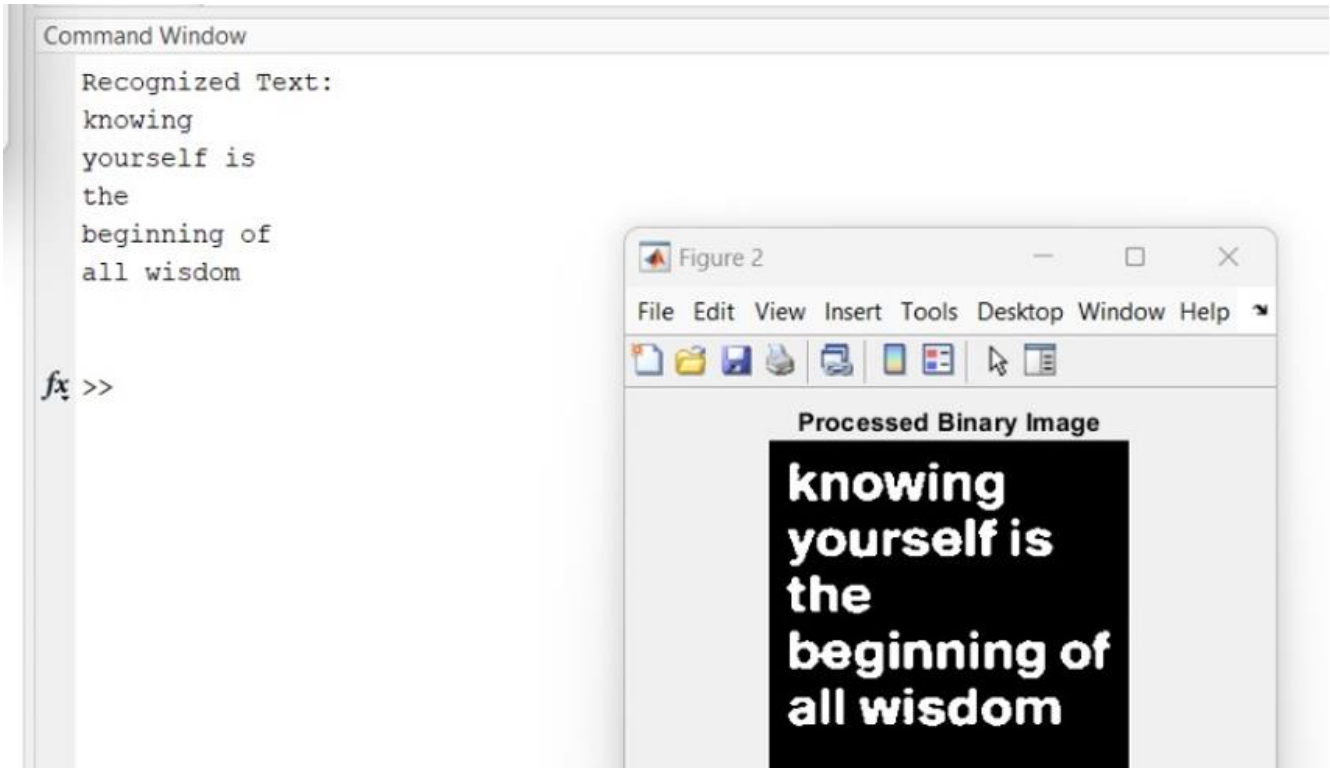
The project follows these main steps:

1. Image Loading and Rotation:
    - The image is first loaded using the imread() function. In this case, the image is a street scene, which is then rotated by 33 degrees using the imrotate() function with the 'crop' option to ensure the image remains within the same size.

2. Grayscale Conversion and Edge Detection:
    - The rotated image is converted into a grayscale image using the rgb2gray() function. This step is necessary because edge detection techniques typically work on single-channel images (grayscale images). The Canny edge detection method is then applied to detect edges in the grayscale image. This method is widely used because it provides good edge localization and reduces noise effectively.
3. Hough Transform:
    - The Hough Transform is applied using the hough() function. This mathematical transformation is used to detect straight lines in an image by mapping each edge pixel into a parameter space (polar coordinate system). The result of the Hough Transform is an accumulator matrix, where each cell corresponds to a potential line in the image.
4. Peak Detection in Hough Space:
    - The houghpeaks() function is used to detect peaks in the Hough Transform's accumulator matrix. These peaks correspond to the most prominent lines in the image. A threshold is applied to identify only the most significant peaks, and the top 5 peaks are selected.
5. Line Extraction and Visualization:
    - The houghlines() function is used to extract the parameters of the lines corresponding to the detected peaks. The function returns information about the endpoints of the detected lines. These lines are then plotted on the original rotated image, with the endpoints marked using different colors for visual clarity.
6. Display and Final Output:
    - The rotated image is displayed with the detected lines overlaid on it. The endpoints of each detected line are marked with yellow and red "x" markers to distinguish the starting and ending points. The total number of detected lines is displayed in the command window.

---

9) Text Recognition Using OCR:



Objective

The goal of this project is to process an image by converting it to grayscale, binarizing it, applying noise reduction, and then performing Optical Character Recognition (OCR) to extract and display the recognized text.

Methodology

1. Image Loading: The image is first loaded using imread() and displayed with the title 'Original Image'.
2. Grayscale Conversion: The color image is converted to grayscale using rgb2gray() for easier processing.
3. Binarization: The grayscale image is binarized using the imbinarize() function with a threshold of 0.5 to separate the text from the background.
4. Noise Reduction: A median filter (medfilt2()) is applied to the binary image to remove noise and improve text clarity.
5. OCR: The ocr() function is used to perform Optical Character Recognition (OCR) on the processed binary image, recognizing any text present.
6. Display Text: The recognized text is displayed in the command window.

😊 😊

Thank You