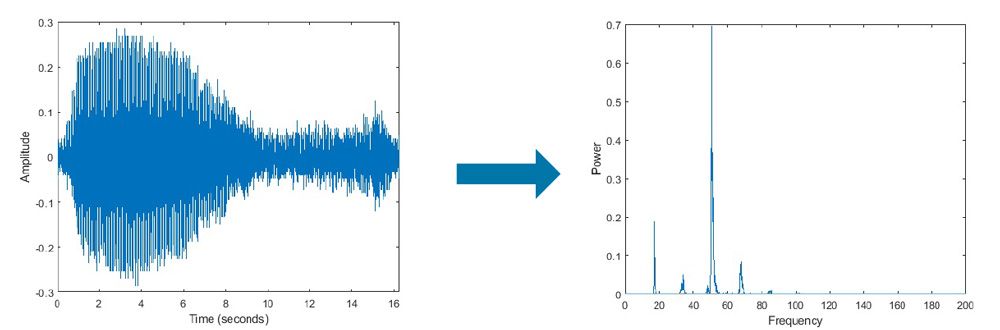
**Image Processing Using Fast Fourier Transform**

**What Is a Fast Fourier Transform (FFT)?**

A fast Fourier transform (FFT) is a highly optimized implementation of the discrete Fourier transform (DFT), which convert discrete signals from the time domain to the frequency domain. FFT computations provide information about the frequency content, phase, and other properties of the signal.



[Fast Fourier Transform (FFT) - MATLAB & Simulink (mathworks.com)](https://in.mathworks.com/discovery/fft.html)

Blue whale audio signal decomposed into its frequency components using FFT.

Popular FFT algorithms include the Cooley-Tukey algorithm, prime factor FFT algorithm, and Rader’s FFT algorithm. The most commonly used FFT algorithm is the Cooley-Tukey algorithm, which reduces a large DFT into smaller DFTs to increase computation speed and reduce complexity. FFT has applications in many fields.

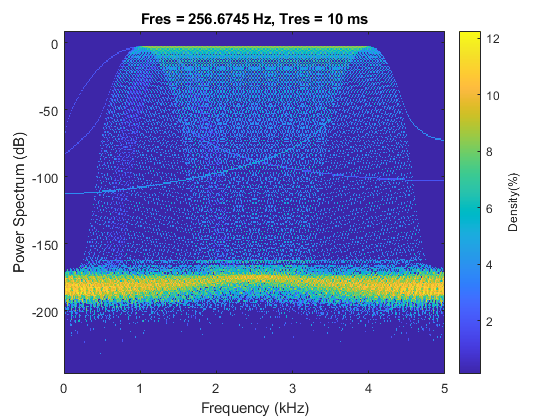
**FFT Applications**

In signal processing, FFT forms the basis of frequency domain analysis (spectral analysis) and is used for signal filtering, spectral estimation, data compression, and other applications. Variations of the FFT such as the short-time Fourier transform also allow for simultaneous analysis in time and frequency domains. These techniques can be used for a variety of signals such as audio and speech, radar, communication, and other sensor data signals. FFT is also sometimes used as an intermediate step for more complex signal processing techniques.

In image processing, FFT is used for filtering and image compression. FFT is also used in physics and mathematics to solve partial differential equations (PDEs).

*Signal Processing*

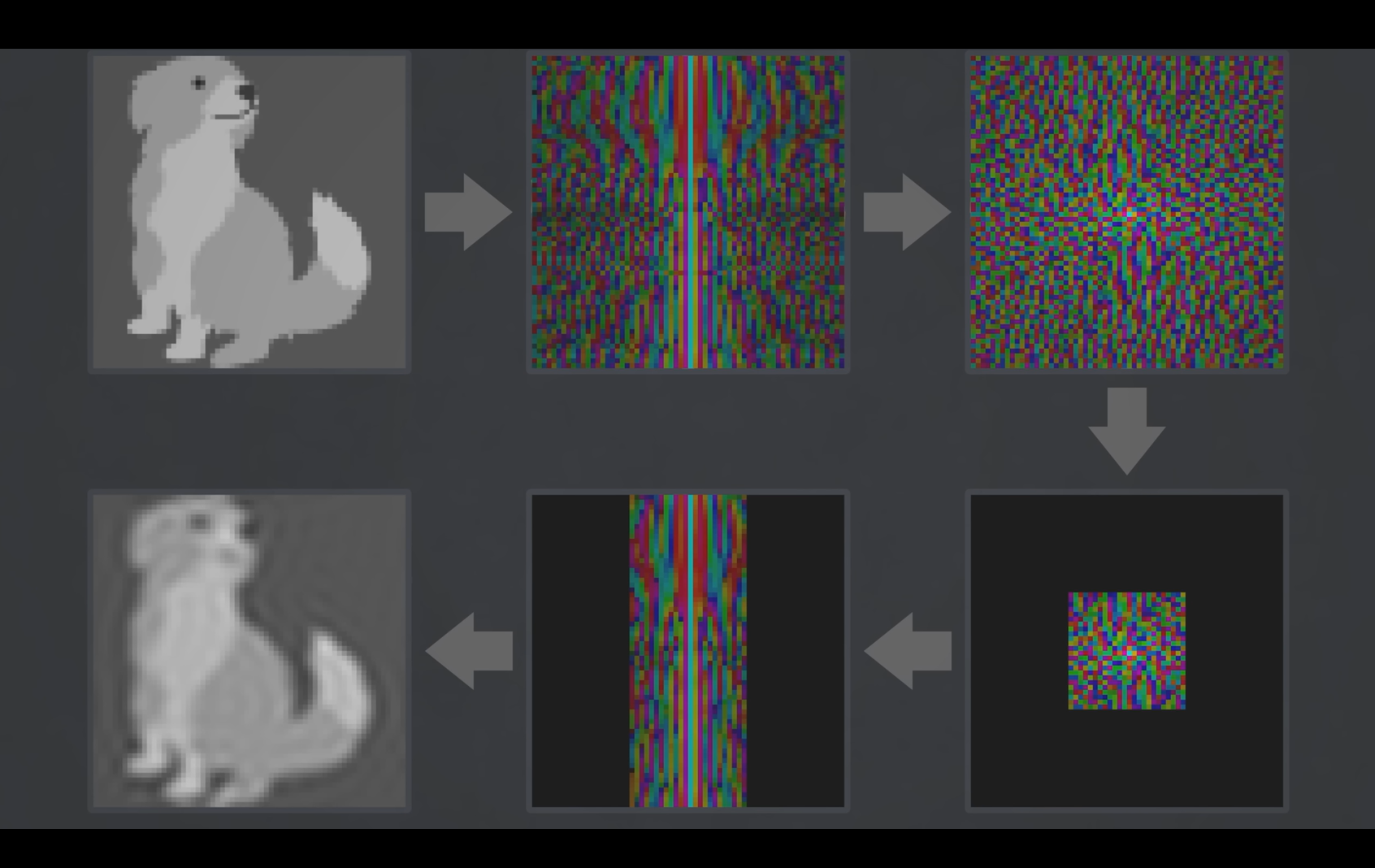
* FFT-Based Time Frequency Analysis
* Transform Time-Frequency Signals into Frequency-Domain Analysis
* Audio Processing
* Dynamic Range Compression Using Overlap-Add Reconstruction
* Pitch Shifting and Time Dilation
* Radar and Communications
* Frequency Agility in Radar, Communications, and EW Systems
* Image Processing
* Video Focus Assessment



[Fast Fourier Transform (FFT) - MATLAB & Simulink (mathworks.com)](https://in.mathworks.com/discovery/fft.html)

Persistence spectrum, a type of time-frequency view, that can be used for spectral analysis of signals.

**Image compression using FFT**



https://www.youtube.com/watch?v=nmgFG7PUHfo&t=1430s&pp=ygUcZm91cmllciB0cmFuc2Zvcm0gdmVyaXRhc2l1bQ%3D%3D

**Code Explanation**

Initialization of Variables: The code initializes several empty arrays to store input images, grayscale images, compressed images, compression qualities, output formats, and output file names.

**Selecting Images**: The code prompts the user to select multiple image files (JPEG, PNG, BMP) using the uigetfile function. If the user cancels the selection, it displays a message and exits.

**File Name Handling**: To ensure consistency, it converts the selected file names into a cell array format, even if only one file is selected.

Output Directory Specification: You can specify the output directory where the compressed images will be saved by setting the outputDirectory variable to the desired path.

Image Processing Loop: The code enters a loop to process each selected image one by one. For each image:

* It reads the image file using imread and converts it to grayscale using rgb2gray.
* It performs a Discrete Fourier Transform (DFT) on the grayscale image to obtain a compressed image in the frequency domain.
* The user is prompted to enter a compression quality (a value between 0 and 100). If no value is entered or an invalid value is provided, it defaults to a quality of 95.
* The user is presented with a choice of output formats (PNG, JPEG, BMP) for the compressed image.
* It generates a unique output file name for the compressed image based on the original file name.
* The compressed image is saved to the specified output directory. For JPEG images, the compression quality is set, while for other formats, it is not specified.
* It calculates and displays the file sizes of the original and compressed images in kilobytes (KB).

Displaying Images and Details: After processing all selected images, the code displays each original image, grayscale image, compressed image magnitude, and the corresponding compressed image in separate figures. It also shows the original file size and compressed file size for each image.

Convert Back Buttons: It creates buttons for each image, labeled "Convert Back," which allows the user to perform an Inverse Discrete Fourier Transform (IDFT) to convert the compressed image back to its original form and display it.

Compression Ratio Calculation: It calculates the compression ratios for each image based on the specified compression qualities (assuming quality is a percentage).

Saving Compression Details: The user is prompted to save the compression details, including file names, compression ratios, output formats, and output file names, as a MAT file.

Convert Back Function: This function is called when the "Convert Back" button is clicked. It performs the IDFT on the compressed image to display the uncompressed image.