**Digital Signal Processing Lab**

# Lab EEL-325

Lab Journal: 11(CEP)



Name: OMAR

Class: BCE-6

Enrollment No: 01-132182-024

**Lab # 11**

**Audio Compressor using Matlab**

## Objective

The objective of this task is to make a GUI for Audio Compressor i.e. the GUI will take an input signal and compress it and display it on output. The input signal will be shown in first axes and output on 2nd.

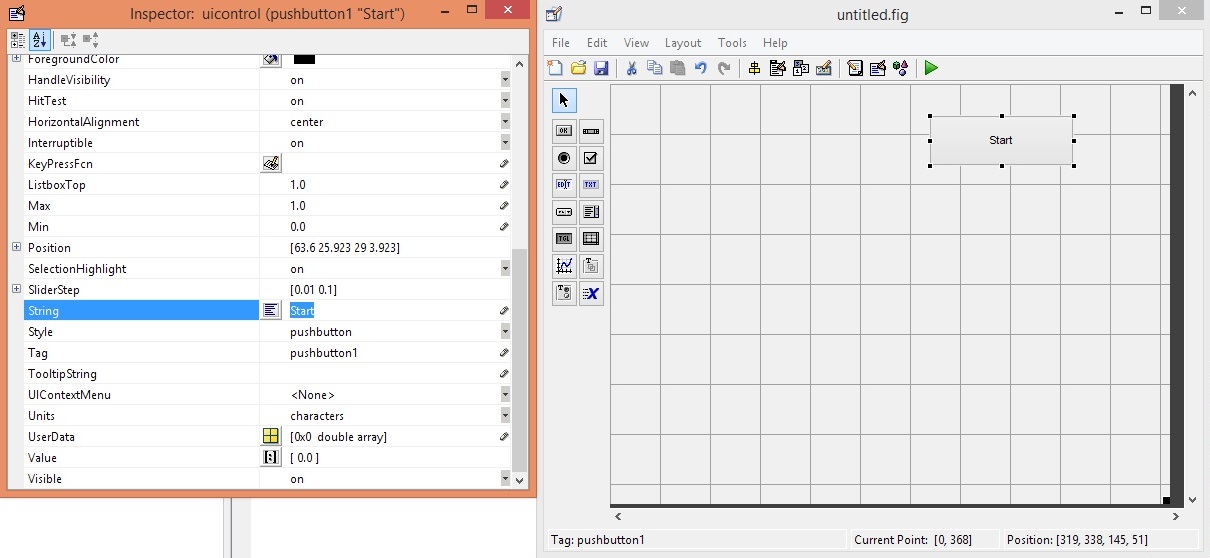
## Introduction

**GUI(Graphical User Interface):**

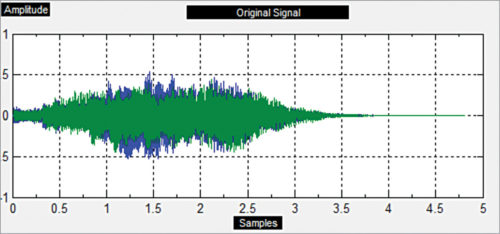
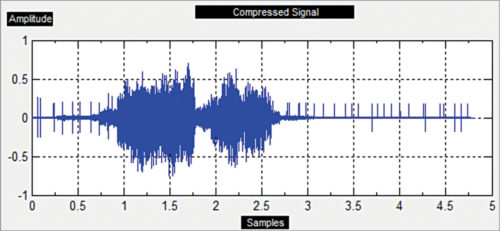
Graphical user interfaces (GUIs), also known as apps, provide point-and-click control of your software applications, eliminating the need for others to learn a language or type commands in order to run the application. You can share apps both for use within MATLAB and also as standalone desktop or web apps.

You can choose from the following three ways to create an app in MATLAB:

* **Convert a script into a simple app:** Choose this option when you want to share a script with students or colleagues and allow them to modify variables using interactive controls.
* **Create an app interactively**: Choose this option when you want to create a more sophisticated app using a drag-and-drop environment to build the user interface.
* **Create an app programmatically:** Choose this option when you want to create an app’s user interface by writing the code yourself.



**Audio Compressor:**

* Audio frequencies range from 20Hz to 20kHz but these frequencies are not heard in the same way. Frequencies below 20Hz and above 20kHz are very difficult to hear, while those not much more than 20Hz, or not much less than 20kHz, cannot be heard by most people. We often need to process these audio signals for various applications. MATLAB is one of the best signal analysis and signal processing tools.
* Audio compression is a very good example of speech and signal processing. We use the Internet for various purposes including entertainment. Audio is common in all entertainment applications. If an audio file size is large, it takes more space to store.
* Audio/video compression frees up space substantially, which can then be utilised for other purposes. This article describes some important audio compression techniques.
* An audio signal sample is taken and analysed using MATLAB for frequency and amplitude. Haar and Daubenches algorithms are applied on the speech signal and the audio is compressed. Audio sizes before and after compression are compared.
* Audio frequencies range from 20Hz to 20kHz but these frequencies are not heard in the same way. Frequencies below 20Hz and above 20kHz are very difficult to hear, while those not much more than 20Hz, or not much less than 20kHz, cannot be heard by most people. We often need to process these audio signals for various applications. MATLAB is one of the best signal analysis and signal processing tools.
* Audio compression is a very good example of speech and signal processing. We use the Internet for various purposes including entertainment. Audio is common in all entertainment applications. If an audio file size is large, it takes more space to store.
* Audio/video compression frees up space substantially, which can then be utilised for other purposes. This article describes some important audio compression techniques.
* An audio signal sample is taken and analysed using MATLAB for frequency and amplitude. Haar and Daubenches algorithms are applied on the speech signal and the audio is compressed. Audio sizes before and after compression are compared.
* 
* 

**Task 1:**

* To make a GUI.
* To take input as an audio signal.
* To calculate its size.
* Compress the audio signal.
* Display it on axes
* Frequency of the input signal is also change by GUI.
* First axes will show the input signal
* Second axes will show the compressed signal

**Using Commands:**

**Source CODE:**

function varargout = untitled(varargin)

% UNTITLED MATLAB code for untitled.fig

% UNTITLED, by itself, creates a new UNTITLED or raises the existing

% singleton\*.

%

% H = UNTITLED returns the handle to a new UNTITLED or the handle to

% the existing singleton\*.

%

% UNTITLED('CALLBACK',hObject,eventData,handles,...) calls the local

% function named CALLBACK in UNTITLED.M with the given input arguments.

%

% UNTITLED('Property','Value',...) creates a new UNTITLED or raises the

% existing singleton\*. Starting from the left, property value pairs are

% applied to the GUI before untitled\_OpeningFcn gets called. An

% unrecognized property name or invalid value makes property application

% stop. All inputs are passed to untitled\_OpeningFcn via varargin.

%

% \*See GUI Options on GUIDE's Tools menu. Choose "GUI allows only one

% instance to run (singleton)".

%

% See also: GUIDE, GUIDATA, GUIHANDLES

% Edit the above text to modify the response to help untitled

% Last Modified by GUIDE v2.5 16-Jun-2021 13:58:07

% Begin initialization code - DO NOT EDIT

gui\_Singleton = 1;

gui\_State = struct('gui\_Name', mfilename, ...

'gui\_Singleton', gui\_Singleton, ...

'gui\_OpeningFcn', @untitled\_OpeningFcn, ...

'gui\_OutputFcn', @untitled\_OutputFcn, ...

'gui\_LayoutFcn', [] , ...

'gui\_Callback', []);

if nargin && ischar(varargin{1})

gui\_State.gui\_Callback = str2func(varargin{1});

end

if nargout

[varargout{1:nargout}] = gui\_mainfcn(gui\_State, varargin{:});

else

gui\_mainfcn(gui\_State, varargin{:});

end

% End initialization code - DO NOT EDIT

% --- Executes just before untitled is made visible.

function untitled\_OpeningFcn(hObject, eventdata, handles, varargin)

% This function has no output args, see OutputFcn.

% hObject handle to figure

% eventdata reserved - to be defined in a future version of MATLAB

% handles structure with handles and user data (see GUIDATA)

% varargin command line arguments to untitled (see VARARGIN)

% Choose default command line output for untitled

handles.output = hObject;

% Update handles structure

guidata(hObject, handles);

% UIWAIT makes untitled wait for user response (see UIRESUME)

% uiwait(handles.figure1);

% --- Outputs from this function are returned to the command line.

function varargout = untitled\_OutputFcn(hObject, eventdata, handles)

% varargout cell array for returning output args (see VARARGOUT);

% hObject handle to figure

% eventdata reserved - to be defined in a future version of MATLAB

% handles structure with handles and user data (see GUIDATA)

% Get default command line output from handles structure

varargout{1} = handles.output;

% --- Executes on button press in pushbutton1.

function pushbutton1\_Callback(hObject, eventdata, handles)

[x,fs]=audioread('sample.wav');

N=length(x);

vlcplayer=audioplayer(x,fs);

vlcplayer.play;

t=fft(x,N);

X=fftshift(t);

f=-fs/2:fs/N:(fs/2-fs/N);

%figure(1)

axes(handles.axes1);

plot(f,abs(X));

title('original audio signal');

% hObject handle to pushbutton1 (see GCBO)

% eventdata reserved - to be defined in a future version of MATLAB

% handles structure with handles and user data (see GUIDATA)

% --- Executes on button press in pushbutton2.

function pushbutton2\_Callback(hObject, eventdata, handles)

[x,fs]=audioread('sample3.wav');

N=length(x);

%vlcplayer=audioplayer(x,fs);

%vlcplayer.play;

t=fft(x,N);

X=fftshift(t);

f=-fs/2:fs/N:(fs/2-fs/N);

%figure(1)

Xr=zeros(1,N);

Xr((N/4)+1:(3\*N/4))= X((N/4)+1:(3\*N/4)); %%FORMULA

axes(handles.axes2);

plot(f/2,abs(Xr));

%xr= real(ifft(fftshift(Xr))); %%reconstruction

%audiowrite('50% compressed.wav',xr,fs);

%title('50% compressed audio')

%xlabel('freq(hq)');

%ylabel('magnitude');

% hObject handle to pushbutton2 (see GCBO)

% eventdata reserved - to be defined in a future version of MATLAB

% handles structure with handles and user data (see GUIDATA)

**Source Code for GUI:**

function varargout = untitled(varargin)

% UNTITLED MATLAB code for untitled.fig

% UNTITLED, by itself, creates a new UNTITLED or raises the existing

% singleton\*.

%

% H = UNTITLED returns the handle to a new UNTITLED or the handle to

% the existing singleton\*.

%

% UNTITLED('CALLBACK',hObject,eventData,handles,...) calls the local

% function named CALLBACK in UNTITLED.M with the given input arguments.

%

% UNTITLED('Property','Value',...) creates a new UNTITLED or raises the

% existing singleton\*. Starting from the left, property value pairs are

% applied to the GUI before untitled\_OpeningFcn gets called. An

% unrecognized property name or invalid value makes property application

% stop. All inputs are passed to untitled\_OpeningFcn via varargin.

%

% \*See GUI Options on GUIDE's Tools menu. Choose "GUI allows only one

% instance to run (singleton)".

%

% See also: GUIDE, GUIDATA, GUIHANDLES

% Edit the above text to modify the response to help untitled

% Last Modified by GUIDE v2.5 27-Jun-2021 11:02:00

% Begin initialization code - DO NOT EDIT

gui\_Singleton = 1;

gui\_State = struct('gui\_Name', mfilename, ...

'gui\_Singleton', gui\_Singleton, ...

'gui\_OpeningFcn', @untitled\_OpeningFcn, ...

'gui\_OutputFcn', @untitled\_OutputFcn, ...

'gui\_LayoutFcn', [] , ...

'gui\_Callback', []);

if nargin && ischar(varargin{1})

gui\_State.gui\_Callback = str2func(varargin{1});

end

if nargout

[varargout{1:nargout}] = gui\_mainfcn(gui\_State, varargin{:});

else

gui\_mainfcn(gui\_State, varargin{:});

end

% End initialization code - DO NOT EDIT

% --- Executes just before untitled is made visible.

function untitled\_OpeningFcn(hObject, eventdata, handles, varargin)

% This function has no output args, see OutputFcn.

% hObject handle to figure

% eventdata reserved - to be defined in a future version of MATLAB

% handles structure with handles and user data (see GUIDATA)

% varargin command line arguments to untitled (see VARARGIN)

% Choose default command line output for untitled

handles.output = hObject;

% Update handles structure

guidata(hObject, handles);

% UIWAIT makes untitled wait for user response (see UIRESUME)

% uiwait(handles.figure1);

% --- Outputs from this function are returned to the command line.

function varargout = untitled\_OutputFcn(hObject, eventdata, handles)

% varargout cell array for returning output args (see VARARGOUT);

% hObject handle to figure

% eventdata reserved - to be defined in a future version of MATLAB

% handles structure with handles and user data (see GUIDATA)

% Get default command line output from handles structure

varargout{1} = handles.output;

% --- Executes on button press in pushbutton1.

function pushbutton1\_Callback(hObject, eventdata, handles)

[x,fs]=audioread('sample3.wav');

N=length(x);

vlcplayer=audioplayer(x,fs);

vlcplayer.play;

t=fft(x,N);

X=fftshift(t);

f=-fs/2:fs/N:(fs/2-fs/N);

%figure(1)

axes(handles.axes1);

plot(f,abs(X));

title('original audio signal');

% hObject handle to pushbutton1 (see GCBO)

% eventdata reserved - to be defined in a future version of MATLAB

% handles structure with handles and user data (see GUIDATA)

% --- Executes on button press in pushbutton2.

function pushbutton2\_Callback(hObject, eventdata, handles)

[x,fs]=audioread('sample3.wav');

N=length(x);

%vlcplayer=audioplayer(x,fs);

%vlcplayer.play;

t=fft(x,N);

X=fftshift(t);

f=-fs/2:fs/N:(fs/2-fs/N);

%figure(1)

Xr=zeros(1,N);

Xr((N/4)+1:(3\*N/4))= X((N/4)+1:(3\*N/4)); %%FORMULA

axes(handles.axes2);

plot(f/2,abs(Xr));

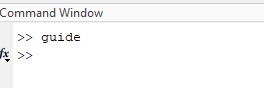
% hObject handle to pushbutton2 (see GCBO)

% eventdata reserved - to be defined in a future version of MATLAB

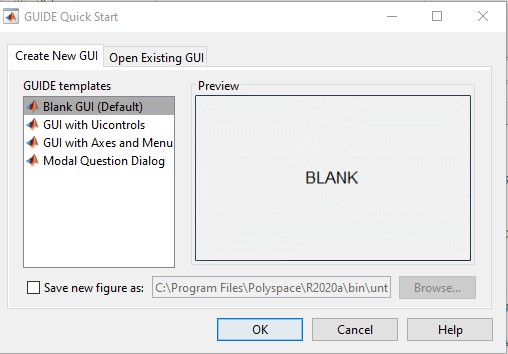
% handles structure with handles and user data (see GUIDATA)

**Steps to create GUI:**

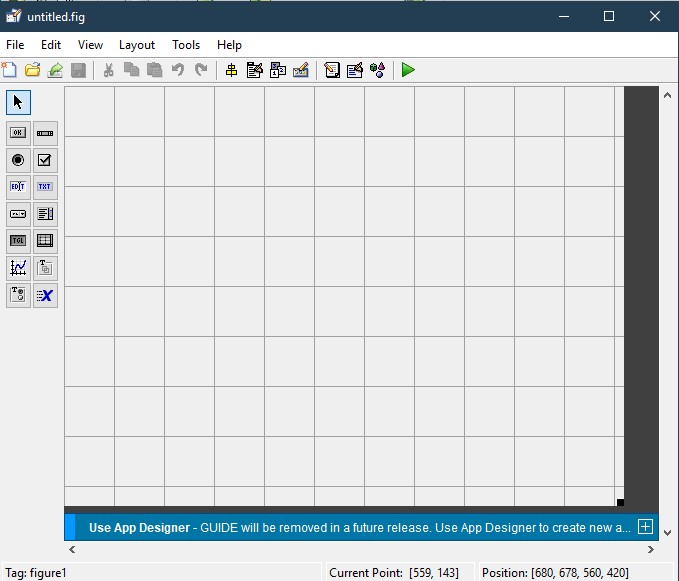
**Step 1:**



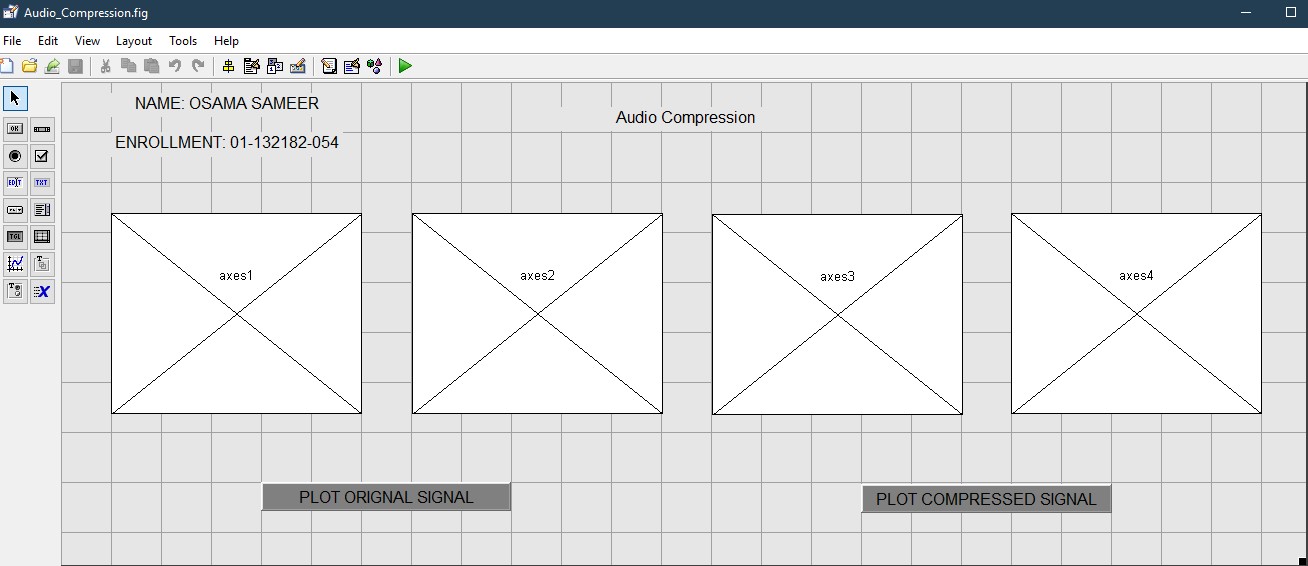
**Step 2:**



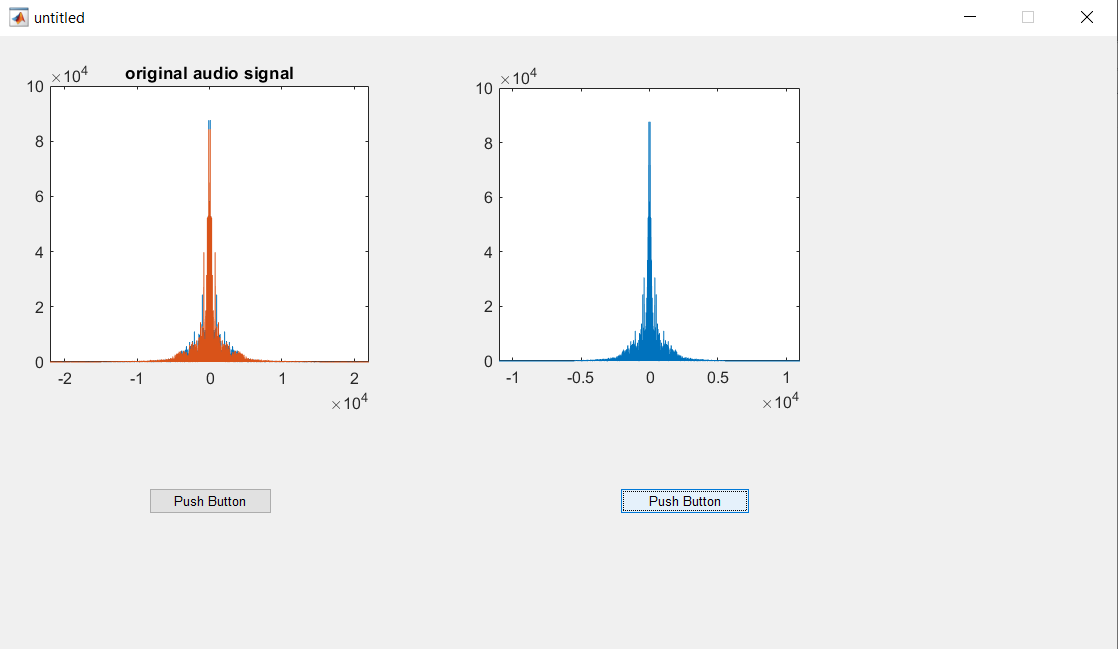
**Step 3:**



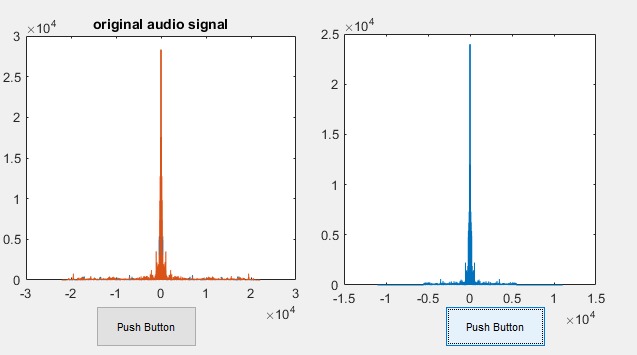
**Step 4:**



**OUTPUT:**



**Output by using a diferrent input audio signal(sample.wav):**

****

**Conclusion: -**

In this lab we learned about

* How to design a GUI in MATLAB by simply drag and drop.
* Hoe to read a signal using audio read command.
* How to compress a signal by taking its Discrete Fourier Transform and then dividing its frequency to half and then plotting the signal on a new axis.