ESCUELA COLOMBIANA DE INGENIERA JULIO GARAVITO
LABORATORIO DE CIRCUITOS ELECTRICOS AC
MANUAL DE INSTRUCCIONES

Integrantes:

- Carlos Valencia.
- Felipe Ramírez.
- Sergio Sotelo.
- Andrés Rodríguez.

INTRODUCCION:

En la Escuela Colombiana de Ingeniería Julio Garavito, desde el curso de circuitos eléctricos de corriente alterna CEAC, se ha desarrollado un proyecto cuyo objetivo es analizar una palabra de su agrado y posteriormente, permitir que un sistema de reconocimiento de voz filtre dicha palabra y realice una acción predeterminada por medio de un Hardware sencillo.

Este manual de instrucciones ha sido elaborado con la intención de guiar paso a paso a quien desee reproducir el proyecto mencionado. De ante mano, debemos tener acceso a el Software MATLAB en donde se elaborará el proyecto en su mayoría, un microcontrolador Arduino UNO para implementación de Hardware y componentes electrónicos mencionados en el proceso de elaboración.

Se ha dividido el proyecto en () partes para una mejor comprensión y aplicación; además, se le otorgarán direcciones URL que podrán explicar más a fondo algunos pasos mencionados a continuación.

1. CREACIÓN DE LA INTERFAZ GRÁFICA.

Para este procedimiento se utilizó la herramienta del Software MATLAB que permite diseñar una interfaz gráfica GUI de manera sencilla y rápida, como se muestra a continuación:

En el inicio predeterminado de MATLAB escriba la instrucción ***guide***, posteriormente se desplegará una ventana como la que puede ver en la figura 1.

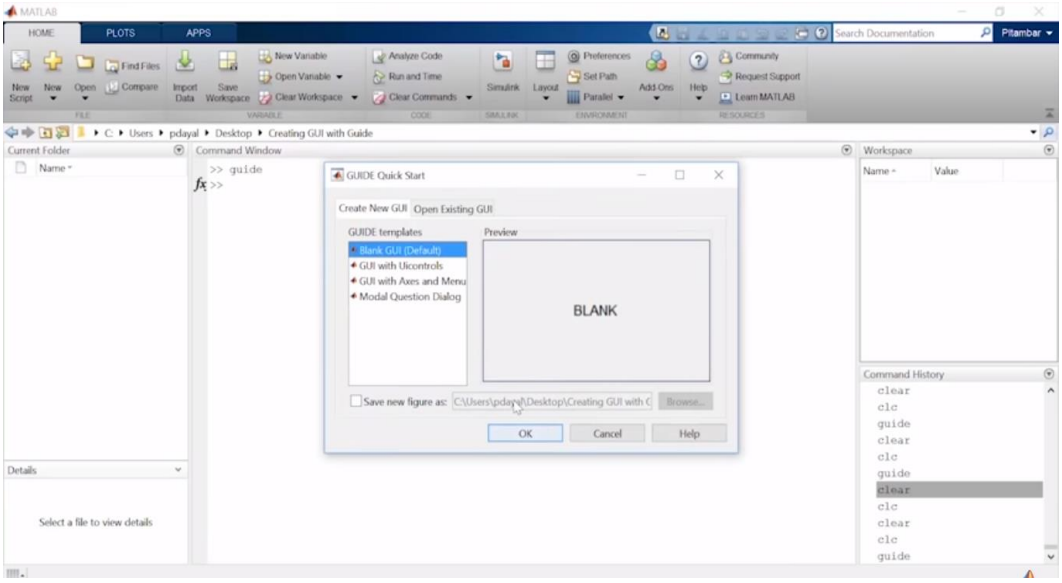

Figura 1. Ventana de selección de GUI.

En esta ventana puede escoger entre una GUI prediseñada y funciones ***Callback*** diseñadas para facilidad del diseñador, o una interfaz en blanco que cuenta con un panel en el que se añaden los ***Widgets*** que necesitemos para lograr una óptima comunicación entre el usuario y la herramienta a diseñar, se muestra en la Figura 2.

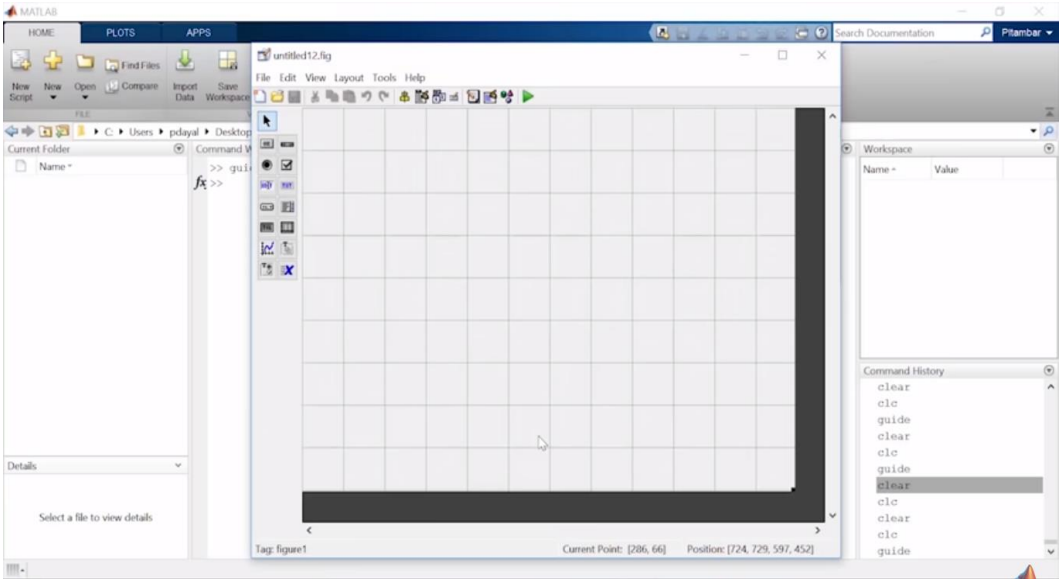

Figura 2. Interfaz de diseño de la GUI.

Al momento de seleccionar algún ***Widget*** se le aconseja entrar a la pestaña de la interfaz de diseño llamada ***View*** y posteriormente seleccionar la opción ***Property Inspector*** para nombrar cada Widget con su respectiva funcionalidad, se desplegará una ventana como la de la Figura 3, esto le permitirá al momento de describir las funciones ***Callback*** de la GUI identificar cada ***Widget*** y no tener problemas de identación o confusión con otros Widgets. Además, en esta ventana también podemos cambiar el texto que aparece sobre un botón o texto, con la opción ***String***.

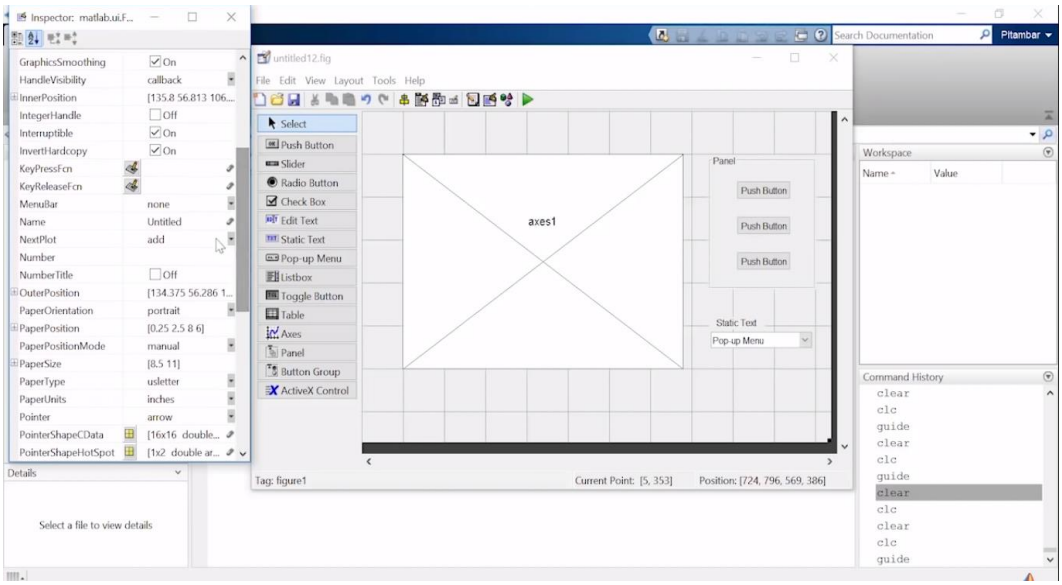

Figura 3. Ventana de inspector de propiedades

Para esta GUI se decidió utilizar planos cartesianos, botones y una caja de texto de salida donde se puede dar una respuesta a nuestro usuario en cada acción que realice, usted puede añadir más Widgets que considere necesarios para esta aplicación, el resultado lo puede observar en la Figura 4.

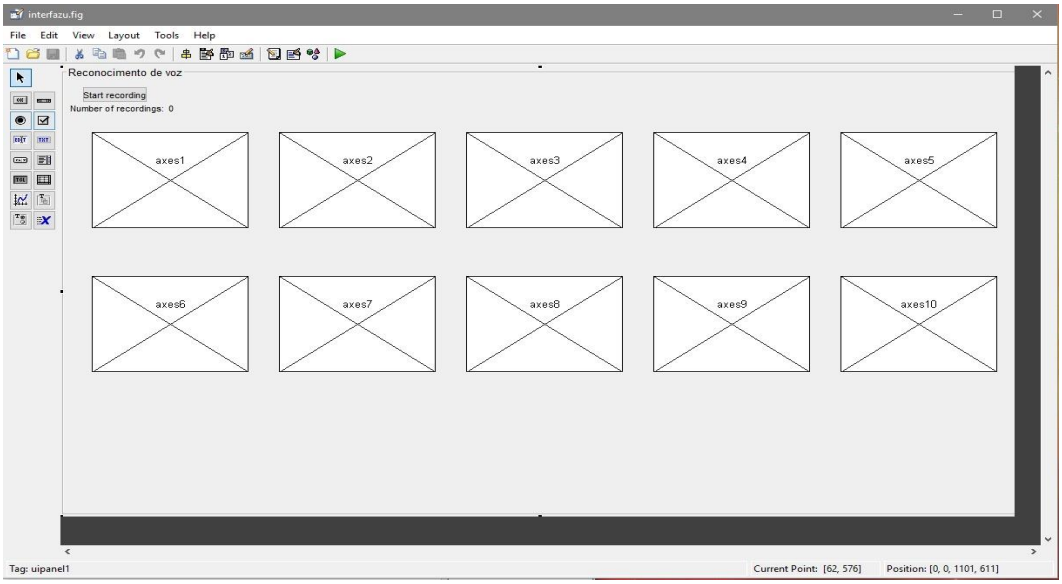

Figura 4. GUI diseñada para el funcionamiento del proyecto.

En el momento en el que usted considere haya terminado de diseñar su GUI proceda a presionar el botón de ***play*** que se encuentra en la parte superior de la interfaz de diseño, se desplegará una ventana en la que podrá guardar la interfaz en la dirección de si computador que desee, procure guardar todos los archivos de este proyecto en una misma carpeta para no crear conflictos al momento de compilar.

Cuando guarde su GUI podrá apreciar el aspecto visual de su proyecto, en el caso en el que no esté conforme con el resultado podrá seguir editando su interfaz hasta que logre el aspecto que más le llame la atención. Ahora procederemos a implementar las funciones ***Callback*** que se han mencionado anteriormente, estas funciones están destinadas a implementar el funcionamiento de cada ***Widget*** a nivel de código de programación, MATLAB trabaja con un lenguaje de programación muy parecido al leguaje C o C++, por lo que sí está familiarizado con este lenguaje no será problema acostumbrarse a este Software. En la Figura 5 podrá apreciar el aspecto físico al que se afrentará para poder llevar a cabo el siguiente paso del desarrollo del proyecto que se desea realizar.

```
% --- Executes on button press in pushbutton1.
function pushbutton1_Callback(hObject, eventdata, handles)
anrecordismade
datosBase
% 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 GUIDAT00A)
set(handles.text3,'String',nr);
if nr == 1
    set(handles.axes1,'Color','white','XColor','black','YColor','black');
    set(handles.axes1,'XLim',[-2 2],'YLim',[-1 1]);
    %funcion que graba
    %funcion del diagrama de bode
    t = linspace(-2,2,1000);
    y = 0.8*sin(2*pi*1*t);
    z = 0.8*cos(2*pi*1*t);
    axes(handles.axes1)
    t = y0_freq1;
    y = abs(y0_fft);
    plot(t(1:371),y(1:371));
    title('Record 1')
    grid on
end
if nr == 2

    set(handles.axes2,'Color','white','XColor','black','YColor','black'); %
    set(handles.axes2,'XLim',[-2 2],'YLim',[-1 1]);
    t = linspace(-2,2,1000);
    y = 0.8*cos(2*pi*1*t);
```

Figura 5. Interfaz de implementación de funciones Callback.

Por último, en esta parte, se le facilita un URL en el que encontrará un video de YouTube que le ayudará con las dudas que hayan quedado para el diseñar de la GUI del primer paso del proyecto.

URL: https://www.youtube.com/watch?v=Ta1uhGEJFBE

2. IMPLEMENTACION DE FUNCIONES *CALLBACK*.

Para esta parte del proyecto confirme que MATLAB abrió automáticamente un archivo ***.m*** en el que se va a encontrar el código descrito a continuación.

Para el algoritmo aplicado, la idea general fue tomar un banco de muestras de sonido de la palabra a reconocer, extraer sus características generales y buscar dichas características en un sonido de entrada para clasificarlo como una palabra lo suficientemente similar a la palabra de base generada en el banco de muestras, todo esto, basándose en técnicas de procesamiento digital de señales, pero fundamentalmente la aplicación de filtros digitales.

En este algoritmo se inicia leyendo las muestras de sonido, a una frecuencia de muestreo de 44100 Hz, esto con el fin de obtener sus Transformadas de Fourier, y obtener las señales en el dominio de la frecuencia que servirán de base comparativa.

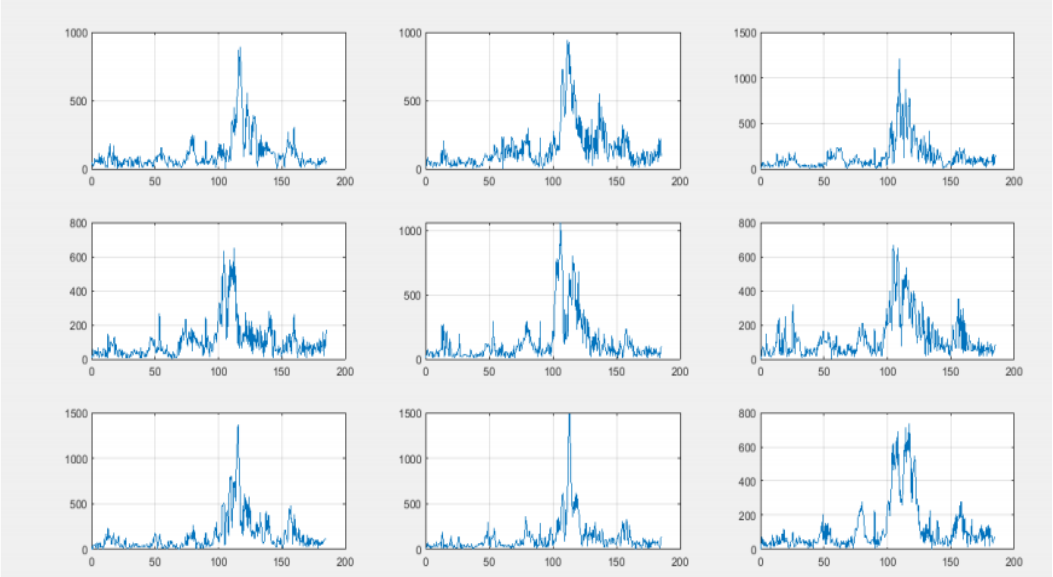

Figura 6. FFT señales base.

Posteriormente se recibe una señal de entrada, cuya Transformada de Fourier también es calculada y se le aplica un filtro pasa banda que delimite las frecuencias entre 60 y 200 Hz.

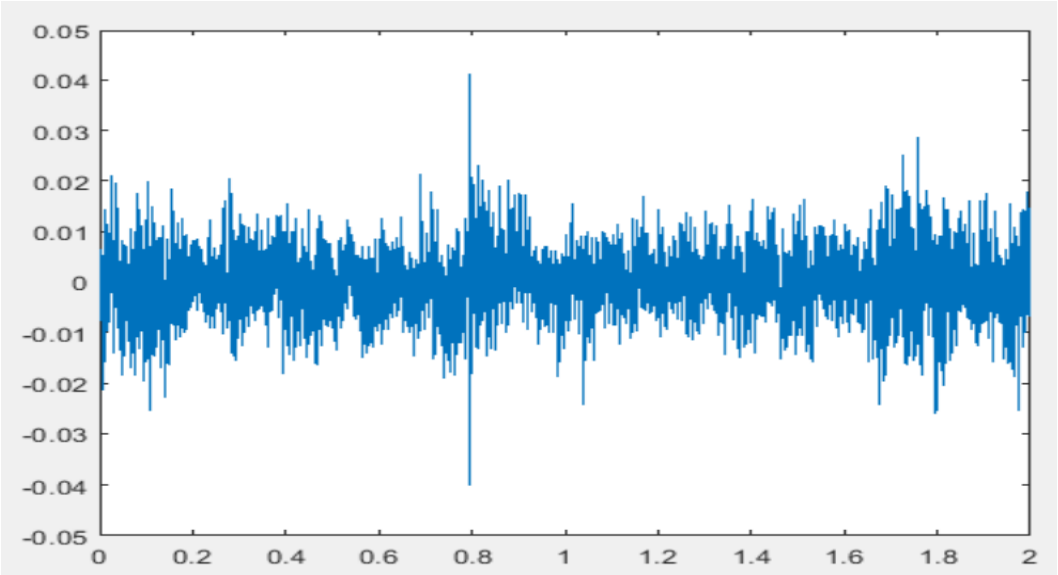

Figura 7. Señal de entrada correspondiente a ruido ambiental únicamente.

La característica de mayor relevancia fue un pico hallado en la banda de frecuencias previamente descrita, por lo cual en el algoritmo se busca hallar el valor del menor pico en la Transformada de Fourier de las señales base y posteriormente, en la señal de entrada filtrada buscar un pico similar a este. De ser encontrado, se clasifica la señal de entrada como la palabra a reconocer.

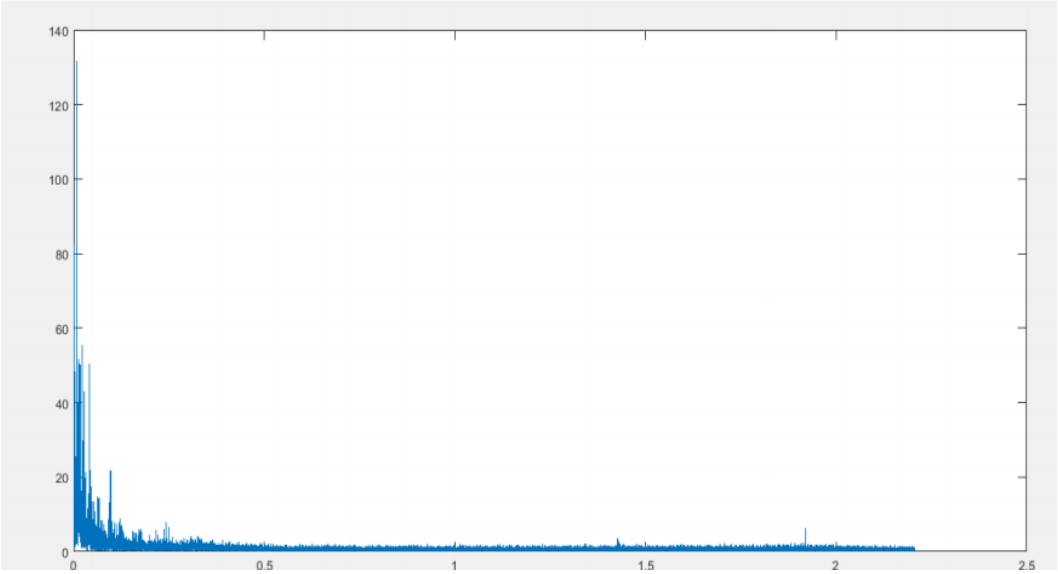

Figura 8. Transformada de Fourier de la señal de entrada.

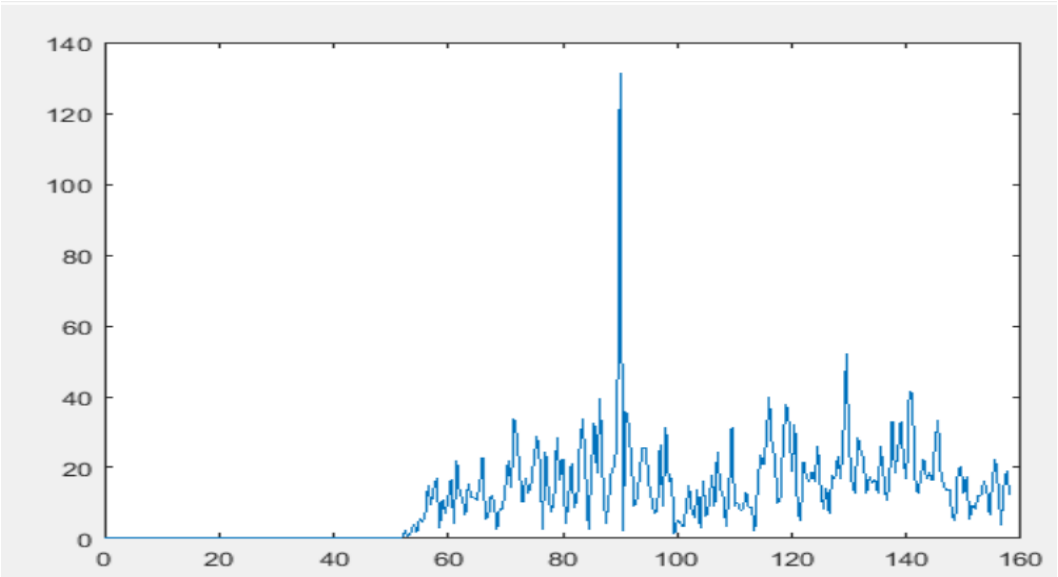

Figura 9. FFT de la señal de entrada filtrada por el pasa bandas.

Con esta base teórica, usted ya puede identificar en cada comando del código que es lo que se está haciendo e incluso mejorarlo. El siguiente link lo va a llevar a un repositorio de GitHub en el que se encuentra el proyecto anteriormente descrito, el archivo llamado interfazu.m es en el que se encuentra la implementación de las funciones ***Callback*** que se han mencionado en este documento, se va a realizar una explicación de lo que se hace aquí.

GitHub URL: https://github.com/carlosVal0/matlabVoiceRec

- global nr

Esta instrucción define una variable de tipo global que vamos a utilizar como un contador para desplegar los ejes uno por uno para ordenarlos como se ve en la Figura 4.

- set(handles.axes1,'Color','none','XColor','none','YColor','none');

En esta serie de instrucciones que va a encontrar en el documento se le está indicando a la GUI que vacíe el contenido de los ejes, simplemente para mejorar la estética del programa

- if nr == 2

En esta serie de instrucciones de la forma if nr == Num, se configura la posición y tamaño de cada grafica que va a aparecer en la interfaz, teniendo como resultado una serie de graficas que concluyen con una opción para grabar una palabra y compararla con un banco de palabras filtradas que se pre guardaron en este programa.

-

3. IMPLEMENTACIÓN DE HARDWARE EN ARDUINO UNO.

Para esta parte usted puede decidir en qué plataforma puede realizar la implementación, puede ser en TINKERCAD.com o en algún simulador del microprocesador Arduino UNO, en este caso, se realiza la implementación en físico, complementándolo con una instrucción que viene desde MATLAB que detallamos en la Figura 10.

Tenga en cuenta que al conectar MATLAB al Arduino UNO u otra versión, las instrucciones que MATLAB envía al Arduino no se conservan en memoria, pues son enviadas en tiempo real, lo que puede provocar que según el computador en el cuál se ejecute el programa el rendimiento pueda decaer por momentos debido al tiempo que tarda MATLAB en establecer los protocolos de comunicación necesarios para transferir datos al Arduino.

En cuanto a las conexiones hechas en el Arduino, el circuito es relativamente simple. El programa está hecho para conectar un Servo-motor en el pin D3, debido a que nos provee una señal digital PWM, lo que permite controlar el Servo-motor con mayor precisión. Mientras que conectado al Pin D8, se conecta un LED del color de su preferencia. No olvide asegurarse que la conexión entre el Arduino y el Computador fue llevada a cabo correctamente antes de ejecutar.

```
if nr == 0
    a = arduino;
    configurePin(a,"D8","DigitalOutput");
    set(handles.text4,'String','Recording');
    [inputVoice, t2, inputVoiceFFT, inputVoice_freq] =  grabacionVoz();
    set(handles.text4,'String','Recording finished');
    set(handles.axes11,'Color','white','XColor','black','YColor','black');
    set(handles.axes11,'XLim',[-2 2],'YLim',[-1 1]);
    axes(handles.axes11)
    t = inputVoice_freq;
    y = abs(inputVoiceFFT);
    plot(t(1:371),y(1:371));
    grid on
    title('Your record')
    s = servo(a,"D3");
    [prs, prsFFT, prsFFT_freq] = inputVoiceFilter(inputVoice);
    Rec = reconocer(prsFFT,minimumSpike);
    set(handles.axes12,'Color','white','XColor','black','YColor','black');
    set(handles.axes12,'XLim',[-2 2],'YLim',[-1 1]);
    axes(handles.axes12);
    plot(prsFFT_freq(1:371),abs(prsFFT(1:371)));
    grid on
    title('your record with filter');
    set(handles.text5,'String',Rec);
    if Rec == "abrir"
        writeDigitalPin(a,"D8",1);
        pause(2);
        writePosition(s,1);
    end
    writePosition(s,0);
end
```

Figura 10. Conexión de la interfaz en MATLAB con un microcontrolador Arduino UNO.

- a= arduino

Es una instrucción que MATLAB reconoce con una librería especial para poder realizar el procedimiento que se describe en la figura 10. Esta inicializa literalmente las funciones que se van a desarrollar con el arduino.

- configurePin(a,"D8","DigitalOutput");

En esta instrucción le estamos diciendo al programa donde va a ubicarse la salida de nuestro programa en la arquitectura del Arduino UNO.

El objetivo de este manual es explicar de una manera sencilla los pasos para poder implementar un proyecto como este, sin embargo, el contenido de los otros archivos que se encuentran en el repositorio tienen una base teórica más amplia de lo que se busca explicar aquí, por lo que se recomienda descargar y comprobar el funcionamiento del proyecto solo con las instrucciones que se han dado en este manual.