# Manual de usuario Interfaces EMG

Roberto Cáceres, Estudiante
**Contacto:** +502 5411-6850
**Correo:** cac17163@uvg.edu.gt

November 24, 2021

## Contents

**1 Introducción** **1**
1.1 Sobre las interfaces . . . . . 1
1.2 Sobre Dispositivo BITalino . . . . . 2

**2 Dispositivo Bitalino** **2**
2.1 BITalino Toolbox . . . . . 2
2.2 Conexión con Bitalino . . . . . 6
2.3 Ejemplos . . . . . 8
2.4 Consideraciones . . . . . 9

**3 Interfaces EMG** **10**
3.1 Interfaz recolectora de datos . . . . . 10
3.2 Interfaz clasificadora de datos . . . . . 14

# 1 Introducción

Se ha creado este documento para realizar un resumen para estudiantes e investigadores que quieran hacer uso de las interfaces de recolección y clasificación de señales electromiografícas (EMG). En este manual de usuario se explicará al interesado los requerimientos e implicaciones que conlleva el uso de la interfaz. Desde el uso del dispositivo Bitalino hasta los recursos necesarios para utilizar este con el software MATLAB. Este Manual de usuario no solo pretende guiar al interesado en los recursos creados en este trabajo de graduación, también pretende guiar al interesado en las experiencias e inconvenientes encontrados con cada uno de las etapas de este.

## 1.1 Sobre las interfaces

Se cuenta con dos interfaces, una de las interfaces cuenta con cumplir con la necesidad de recolectar señales EMG, que pueden ser utilizadas para investigaciones posteriores o como base de datos basada en diferentes experimentos. La segunda interfaz es la interfaz clasificadora de gestos, donde con entrenamientos previos y utilizando inteligencia artificial (IA) se mandan comandos al robot R17 de la Universidad del Valle de Guatemala.

### 1.2 Sobre Dispositivo BITalino

Este es un dispositivo muy poderoso basado en arduino. Básicamente es una caja de herramientas y sensores fisiológicos para el desarrollo de software y hardware. Este incluye comunicación bluetooth. El software integrado con el que cuenta el microcontrolador está basado en bajo el lenguaje PhP y Python.

## 2 Dispositivo Bitalino

En la universidad del Valle de Guatemala, se cuentan con 10 dispositivos bitalinos, cada uno de estos cuentan con conexión Bluetooth. Cada dispositivo cuenta con un *MAC Adress* diferente. El *MAC Adress* es un identificador único de 48 bits para identificar la totalidad del dispositivo de red. Datasheet Bitalino.

**MCU** - Es la unidad de procesamiento de datos, ella recibe de los puertos, organiza y envía datos a la unidad BT.

**PWR** - Es la unidad del bitalino que se encarga de administrar la energía al dispositivo completo. Es el que tiene el control de encendido y apagado.

**BT** - Esta unidad se encarga de recibir los datos del MCU y los transmite mediante el protocolo de este a otra unidad bluetooth.

**EMG** - Este sensor es capaz de detectar el cambio en el estado de un músculo gracias a la variación en el potencial eléctrico.

**EDA** - Este sensor mide el cambio en la conductividad de la piel.

**ECG** - Sensor que mide las pulsaciones cardiacas por medio del cambio en el potencial eléctrico.

**EEG** - Sensor que mide las señales en el cerebro por medio del cambio en el potencial eléctrico.

### 2.1 BITalino Toolbox

*BITalino Toolbox* es básicamente un firmware del software MATLAB para el dispositivo Bitalino. Este permite al usuario conectarse a los dispositvos bitalinos desde MATLAB, así como adquirir y analizar señales. Fue creado con la versión 2017a del software MATLAB y es compatible con cualquier versión después de esta.

Al tener una versión de MATLAB compatible con el *toolbox* se procede a instalar el *toolbox* desde el software. El primer paso es acceder a la ventana *Get Add-Ons* en MATLAB, luego en la ventana de buscada se coloca el nombre del *toolbox* y se procede a instalar este. Más adelante se hará una introducción al *toolbox* haciendo una breve descripción de las funciones más importantes de este. BITalino Toolbox.

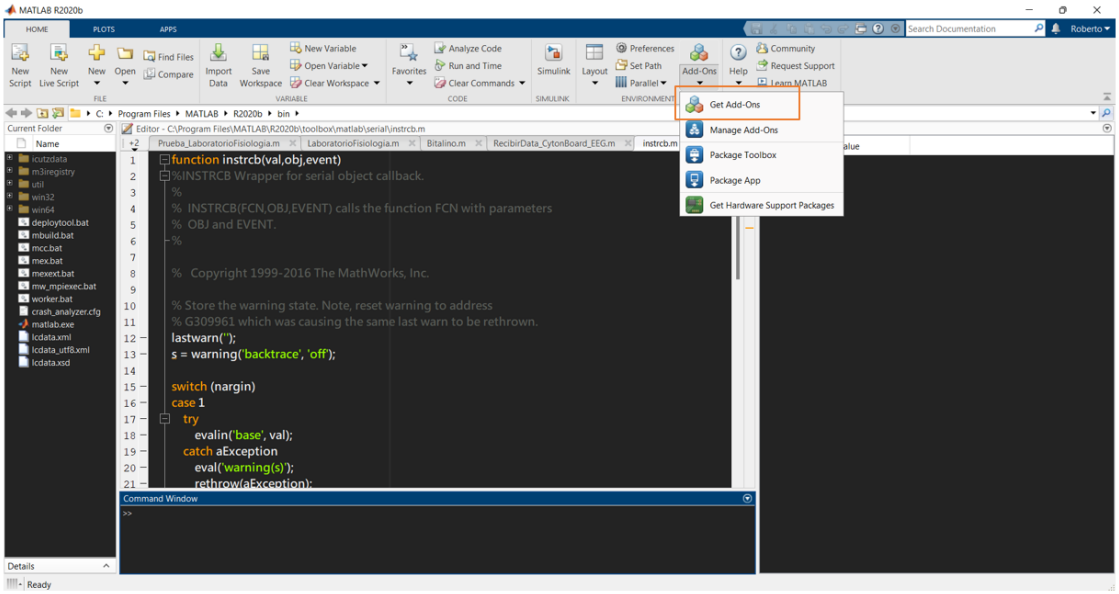


Figure 1: Seleccionar Add-Ons

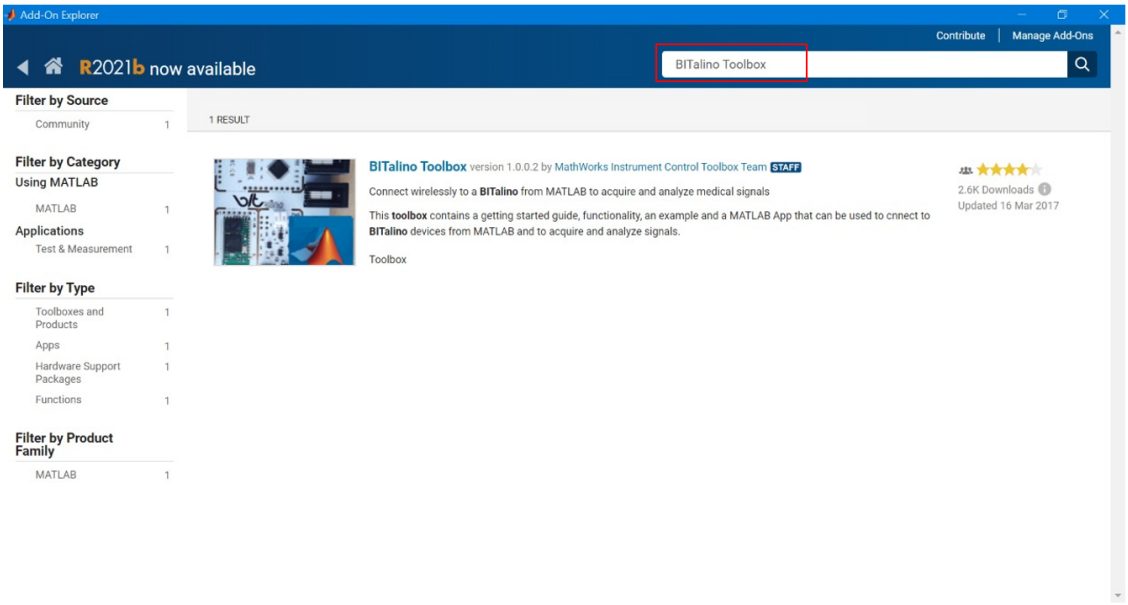


Figure 2: Buscar por el nombre del Toolbox

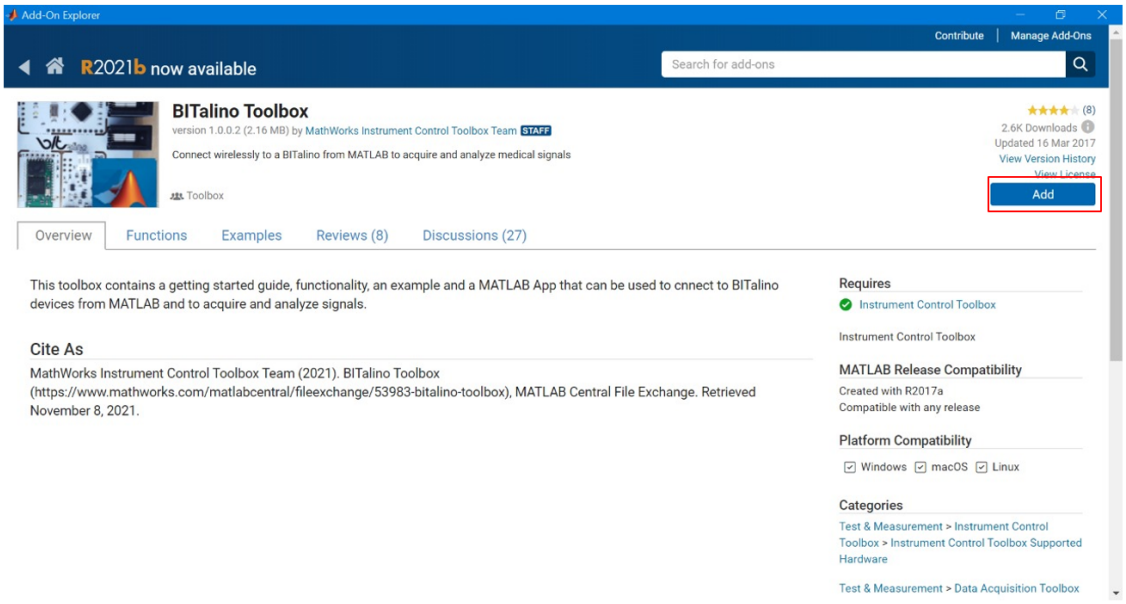


Figure 3: Presionar el boton Add e instalar el toolbox

**BITalino Constructor** - Esta función construye un objeto bitalino con argumentos de entrada opcionales. Este argumento de entrada es el nombre con el que se identifica el bitalino que se esté utilizando. Por defecto el firmware establece una frecuencia de muestreo de 100Hz pero que podemos cambiar como en el siguiente ejemplo.

```
Bit = Bitalino('BITalino-00-97');
Bit.AvailableSamples %Numero de muestras disponibles en el bitalino
Bit.SampleRate = 1000; %Establece esta frecuencia de muestreo
delete(Bit)%Elimina el objeto Bitalino creado
%Frecuencias de muestreo disponible:
% 1000hz sample rate
% 100Hz sample rate
%  10Hz sample rate
%   1Hz sample rate
```

**Configurar BITalino para transmitir datos a MATLAB** - Con esta función se inicia la adquisición de datos del Bitalino bloqueando la ejecución de comandos de MATLAB. Cualquier dato existente en el búfer del objeto Bitalino se borra. Esta función es necesaria para que el dispositivo bitalino empiece a transmitir la data en segundo plano y no interrumpa la ejecución de matlab

```
startBackground(Bit)
```

Es necesario cerciorarse que en efecto el dispositivo está en modo streaming, esto se puede notar observando que el led del dispositivo empieza a palpitar rápidamente.

**Configurar BITalino para detener la transmisión de datos en MATLAB** - Detiene la adquisición de datos del Bitalino y establece el dispositivo en estado inactivo. Cualquier dato en el búfer no se borra.

```
stopBackground(Bit)
```

**Leer datos adquiridos del BITalino** - Con esta función se leen los datos del búfer interno del dispositivo bitalino. El búfer se redimensiona para eliminar los datos existentes del búfer una vez que el usuario los vuelva a leer.

```
Data = read(Bit) %entrega los valores en el bufer
```

**Leer datos instantáneos del Bitalino** - Con está función se leen los valores actuales de los sensores del dispositivo bitalino en el búfer interno, el búfer no cambia de tamaño para eliminar los datos existentes del búfer cuando está adquiriendo estos datos en segundo plano. Tiene un argumento de entrada, que es la cantidad de datos que se quieren leer del búfer.

```
DataActual = readCurrentValues(Bit,1) %entrega el valor actual en el bufer
```

## 2.2 Conexión con Bitalino

Al tener instalado *toolbox* se procede a emparejar el dispositivo bitalino con la computadora que por obvias razones tiene que contar con conexión bluetooth. Al encontrar el dispositivo se deben considerar dos cosas importantes, se solicitará un pin para el emparejamiento el cual es ′1234′. Es importante que se tenga claro que este emparejamiento solo se realizará una vez, las siguientes conexiones se realizarán por medio del software MATLAB. El nombre con el que aparece el dispositivo bitalino en el primer emparejamiento es el nombre que se utilizará en MATLAB para la conexión.

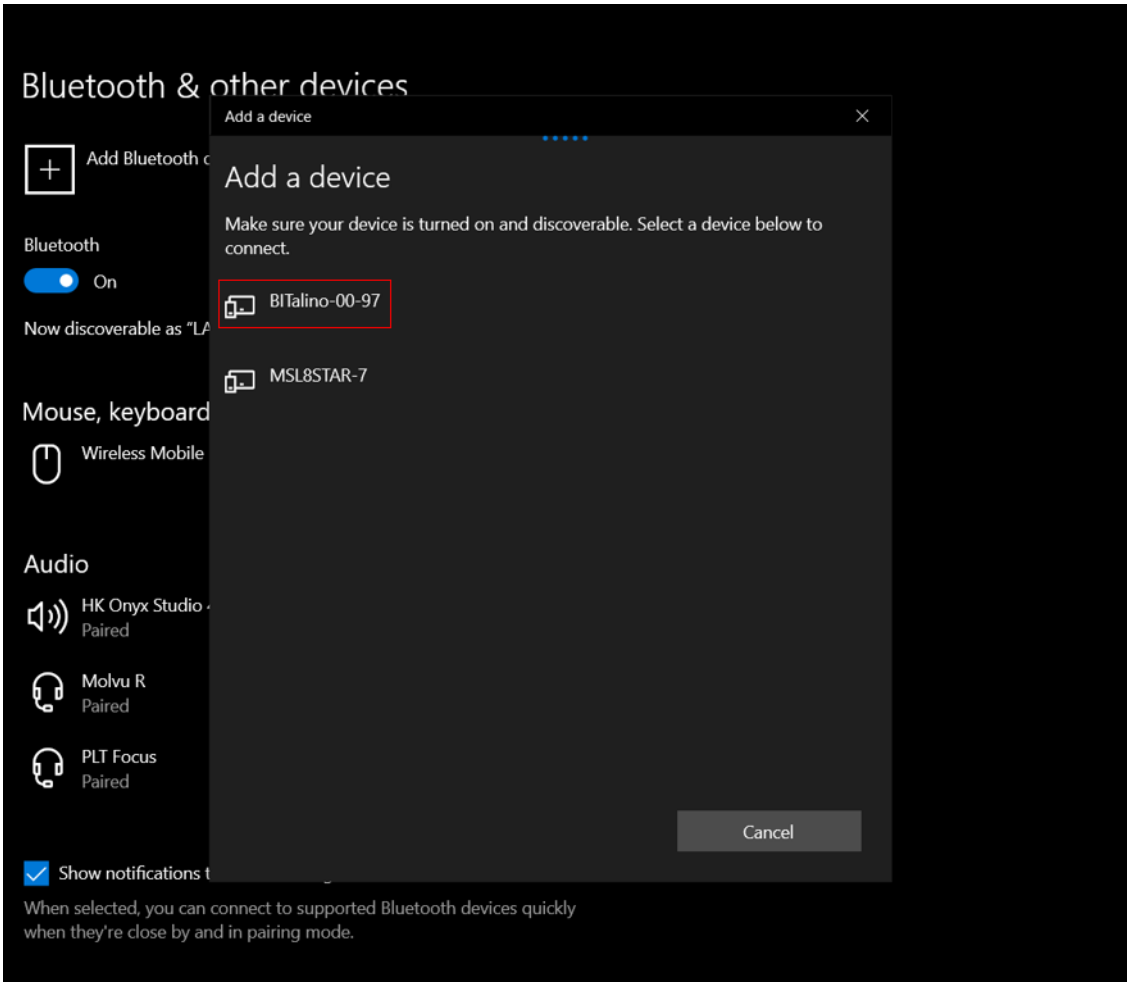


Figure 4: Se encuentra el dispositivo bitalino desde el menú de bluetooth de Windows

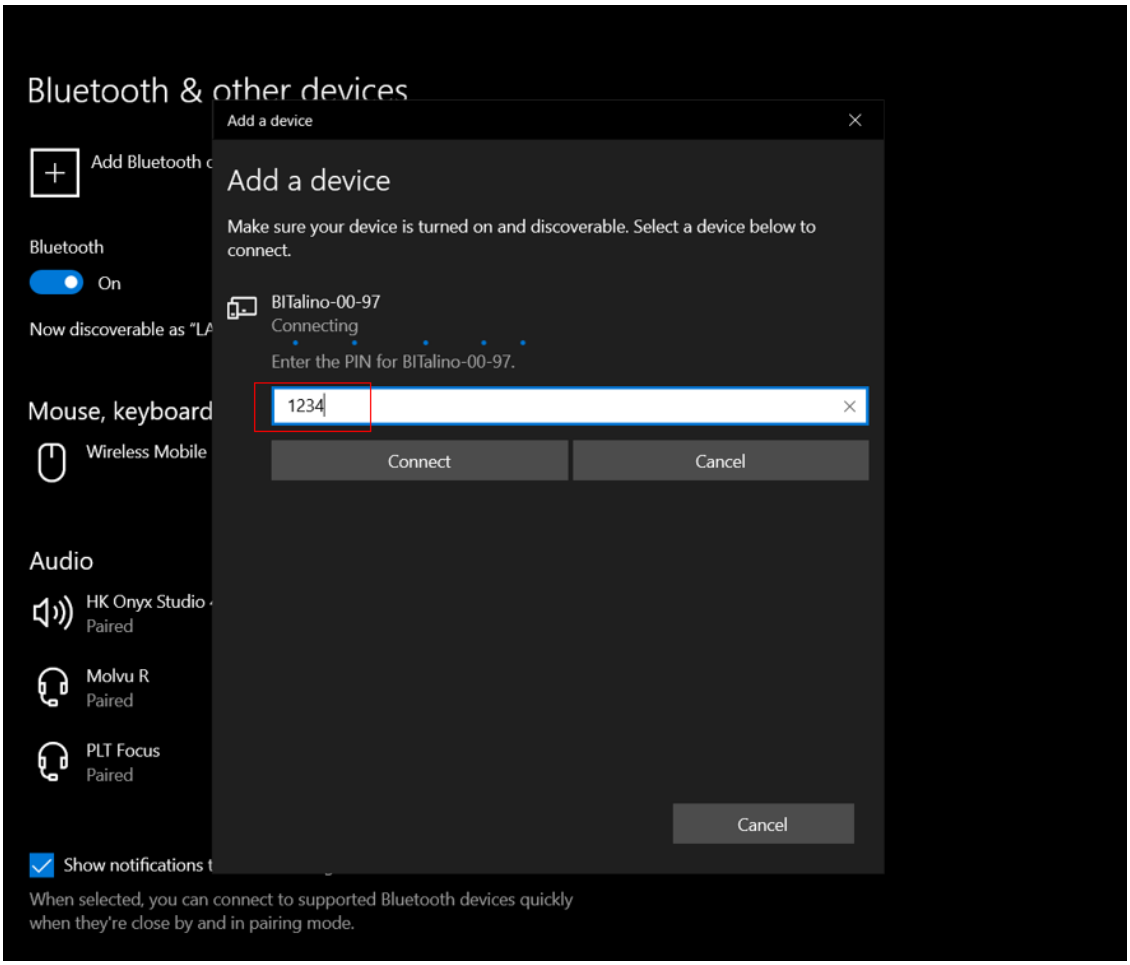


Figure 5: Seleccionar Add-Ons

## 2.3 Ejemplos

**Recolección de Data dispostivo Bitalino** - Este código permitirá la conexión con el dispositivo bitalino y permitirá la recolección de data dependiendo de la cantidad de tiempo que el usuario quiera grabar. La data de cada ventana se almacena en una celda que puede ser exportada y guardada como un .csv para ser utilizada luego en lo que se necesite.

```
% Roberto Caceres 17163
%codigo para recolectar Datos y visualizar en tiempo real.
% Configuracion Bitalino.
Bit = Bitalino('BITalino-00-97'); %Creamos la conexion con el dispositivo
    bitalino y esa conexion la volvemos un objeto.
% Definicion de variables para el ciclo.
TiempoVentana = 5;      %Tiempo que se quiere visualizar en el ploteo de datos
     (segundos)/Tambien se les llaman Epocas
fs=1000; %La frecuencia de muestreo es la cantidad de muestras que obtenemos
    por segundo por parte del bitalino
DataEMG = zeros(5000,1);
EMG=zeros(5000,1);
EMG_raw=0;
data=0;
data_n=zeros(4800,1)
Index=1;
i=1;
Offset = zeros(1,1);
ctrl =1;
DataEnBruto = cell(0,0);
time=0;
ctrl2=1;
DataAnterior=0;
% Grafica
Epocas = TiempoVentana *1000;
L = Epocas; %longitud del vector epocas
t=linspace(0, Epocas/1000, L);
tiempos = zeros(Epocas,1);
h1= plot( t, tiempos);
ylim([-0.5 0.5]);
title('ChEMG')
ylabel('V')
xlabel('tiempo (s)')
% Se preparan los filtros
F_pb = filtro_pasa_banda(1000,20,450);    %Disenar filtro pasa banda
F_notch = filtro_rechaza_banda(1000);     %Disenar filtro rechaza banda
% Empezar a recolectar data en  streaming
Bit.SampleRate = 1000;
startBackground(Bit)
%% Comienzo del loop
%esta funcion prepara al bitalino para empezar funcionar en modo streaming
%y no dejara de enviar hasta que se haga un b.stopBackground mientras
%tanto se empieza a recibir lo datos de acuerdo a como se lleno el bufer.
while true

    while (time < TiempoVentana)      %la variable m determina el tiempoque
        nosotros asignaramos para llenar el buffer
        pause(1)   % se le da una pausa de 1 segundo por cada vez que pase,
            para que se logre llenar el buffer en el tiempo que se requiere
```

```
45          time = time + 1 %variable que aumenta cada 1 segundo.
46      end
47       data = read(Bit);% tomamos los ultimos valores en el buffer.
48        time = 0; %se reinicia el contador para la siguiente vez que se llene el
              buffer
49      %n=n+1%contador para ingresar los datos a la tabla.
50      EMG_raw = data(:,6); %se toma solo la columna de la senal EMG
51      EMG = EMG_raw*(3.3/1024);
52
53      data_f = filter(F_pb,EMG); % a este vector tenemos que filtrarlo con el
            filtro pasa banda
54      data_n = filter(F_notch, data_f);     %Aplicar filtro notch para
            corrientes de 60 Hz, que es la banda de frecuencia a la que trabaja la
             energia electrica en Guatemala
55      Epocas = size(data_n,1) ; %se toma el tamano del buffer obtenido
56      L = Epocas; %longitud del vector epocas
57      t=linspace(0, Epocas/1000, L);
58      h1 = plot( t, data_n) %se plotea esa muestra
59      drawnow limitrate % esta funcion es para poder plotear dentro del while
60      if ctrl2 >1
61      DataEnBruto(:,ctrl2)= num2cell(data_n(1:4800)); %En esta celda guardamos
            cada corrida o cada iteracion
62      fprintf('Done - %d \n',ctrl2);
63      end
64      ctrl2 =ctrl2+1;
65      %En matlab es muy importante el orden de la linea de codigo porque
66      %segun el orden este ejecuta el comando.
67  end
68  %% Parar la adquisicion de data
69  stopBackground(Bit)%para el  modo streaming
70  %%
71  delete(Bit)%elimina el objeto creado
```

## 2.4 Consideraciones

Al empezar a utilizar el dispositivo bitalino, se siguió una curva de aprendizaje para poder desarrollar las interfaces que se presentan a continuación. Como se mencionó con anterioridad uno de los objetivos que trata de cubrir este manual de usuario es cubrir el tiempo de esa curva de aprendizaje y ayudar al interesado a usar directamente el dispositivo bitalino en lo que necesite.

Por eso es importante mencionar las complicaciones que se tuvieron en esta curva de aprendizaje que se siguió. Al tratar de recibir la data en tiempo real y plotear los datos de acuerdo a la frecuencia de muestreo a la que estaba el dispositivo, el usuario se encuentra con problemas de reloj directamente de la computadora. La función que se utiliza para leer un dato en tiempo real, solo lee el último valor que aparece en el buffer del bitalino, eso implica que el código siempre se leerá más rápido que lo que llega el próximo dato, esto obliga al usuario realizar códigos que no permiten el análisis en tiempo real, sino que en un post-procesamiento de la reciente ventana.

Otro problema encontrado es que los comandos que se envían a través de MATLAB al dispositivo bitalino por momentos no los recibe bien, por lo que a veces existen problemas de conexión y se tiene que reiniciar el dispositivo o cuando se le pide entrar en modo *streaming* no lo hace, y se tiene que enviar los comandos varias veces hasta que entre al modo *streaming.*

# 3 Interfaces EMG

Para poder utilizar está interfaz es necesario contar con el software MATLAB desde la versión 2017a que es para la que se diseñó el *BITalino Toolbox*. Así mismo es necesario tener instalado los *toolbox*: *Signal Processing Toolbox*, que es utilizado para poder diseñar los filtros para procesar la señal y *EMG Feature Extraction Toolbox* que es un paquete que cuenta con 40 tipos de características para señales EMG con los que se pueden hacer los entrenamientos. Este último no es necesario si se implementarán otras características y se decide hacer uso de otro tipo de análisis. Todos los archivos que se deben de tener en el directorio donde se correrá la interfaz se especifican en el repositorio de GitHub.

## 3.1 Interfaz recolectora de datos

Está interfaz sirve para recolectar las señales en bruto y las características de estas en dos archivos separados, para esto se tienen que seleccionar las características a extraer y la clase a grabar.

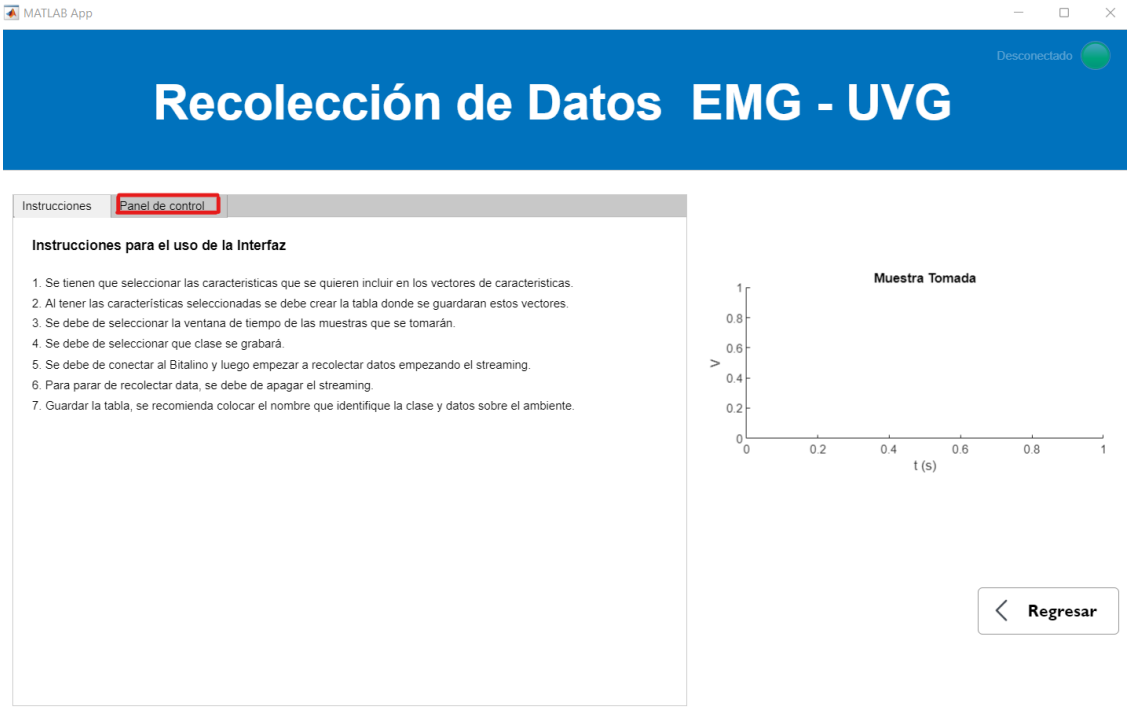


Figure 6: Instrucciones y Panel de control

La interfaz ya cuenta con instrucciones para su correcto uso. El primer paso es pasar a la ventana panel de control.

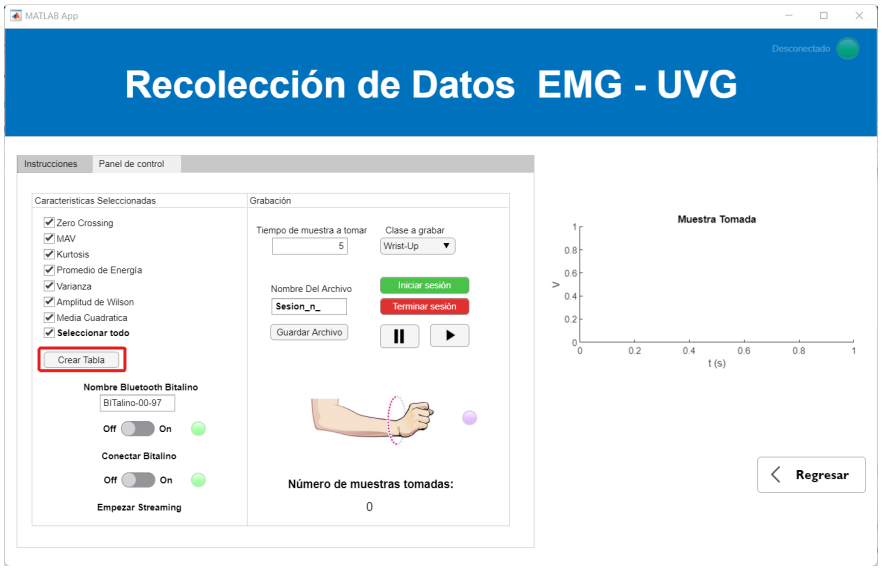


Figure 7: Selección de características

Se deben seleccionar las características que se quieren extraer de las señales. Al seleccionar las características se debe presionar el botón crear tabla. Esto prepara la tabla donde serán almacenadas las muestras.

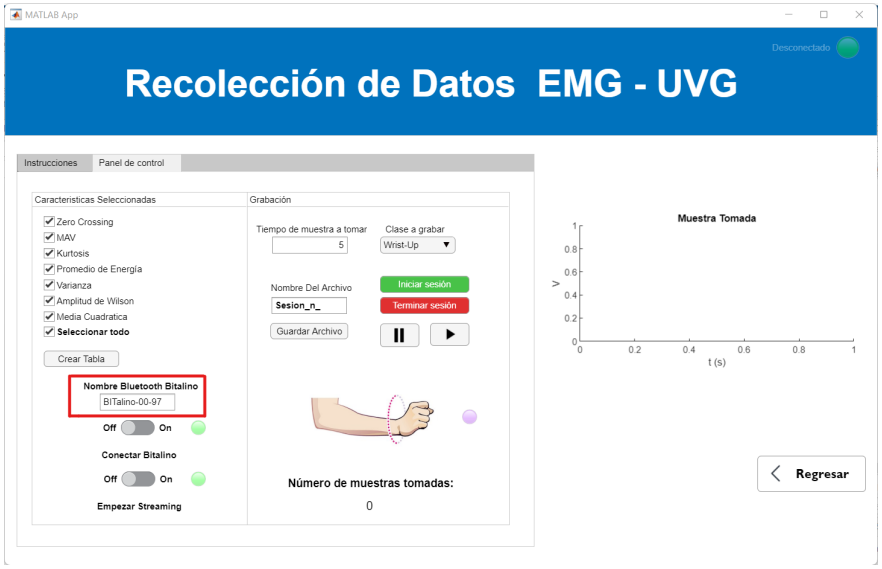


Figure 8: Colocar el nombre del dispositivo a conectar.

Cuando se realizó el apareamiento con él dispositivo bitalino, se observó un nombre con el que se realizó la conexión bluetooth. Es el mismo nombre que se tiene colocar en **Nombre Bluetooth Bitalino**.

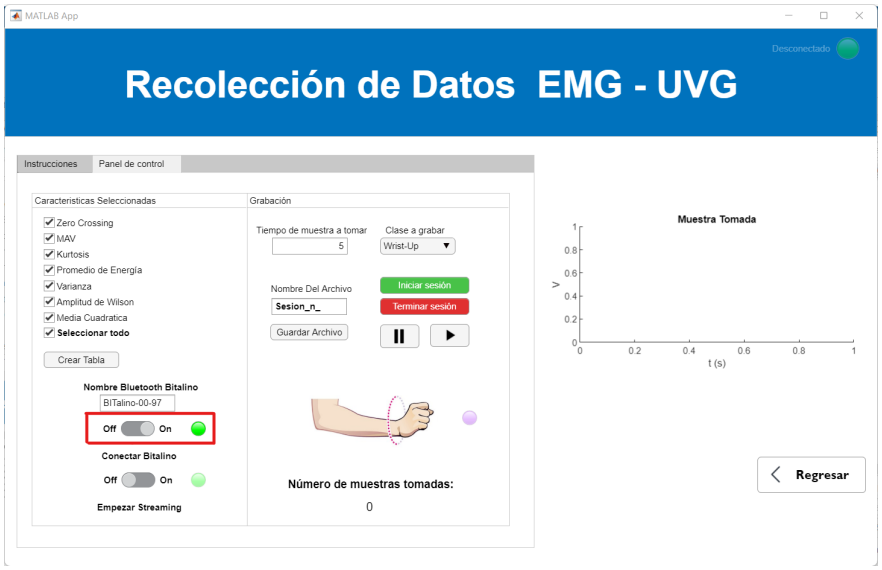


Figure 9: Conexión al dispositivo bitalino

Si la luz verde no se enciende, esto significa que la conexión no fue realizada, es recomendable apagar el dispositivo bitalino, presionar *off* en la interfaz y volver a probar la conexión.

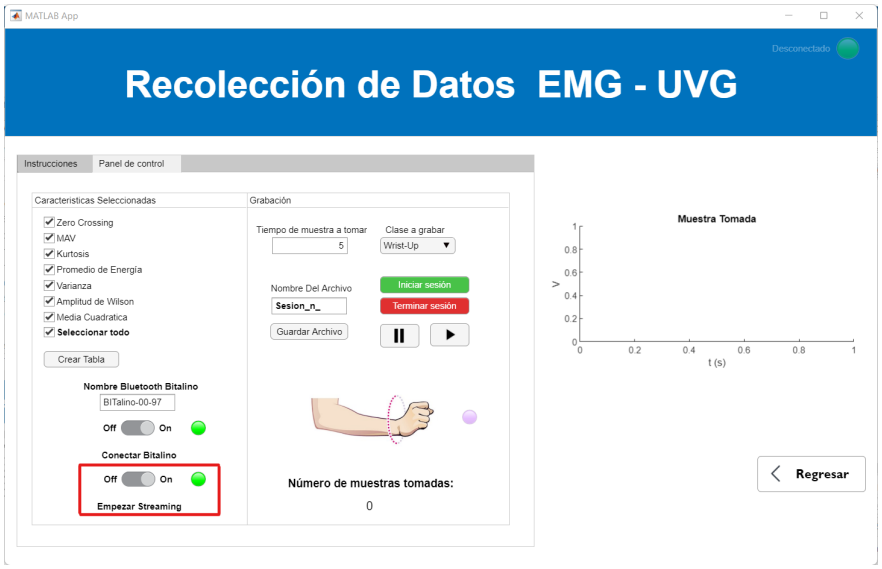


Figure 10: Bitalino modo *streaming*

Luego de realizar la conexión con el dispositivo bitalino, se tiene que empezar a recibir la data en modo *streaming* en segundo plano, para eso se presiona el botón **Empezar streaming**, se encenderá la luz, pero esto no significará que todo está listo. El dispositivo estará listo hasta que el led que está en la placa del bitalino empiece a parpadear, de lo contrario no se puede empezar el proceso de recolección. Se tiene que apagar y encender el botón **Empezar streaming**, hasta que está luz empiece a parpadear y no este constante.

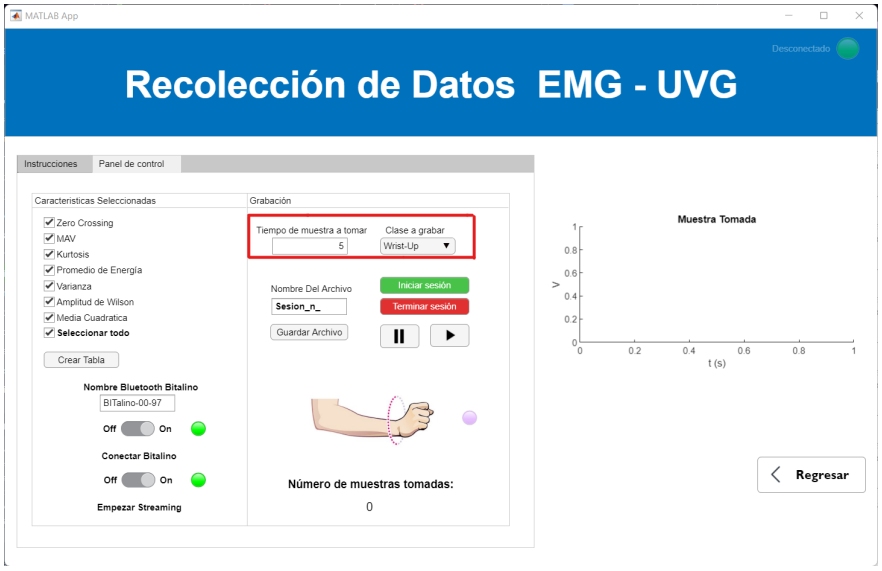


Figure 11: Selección duración ventana y clase a grabar

El siguiente paso es indicar a la interfaz de cuanto es la ventana de tiempo de grabación y la clase que se va a grabar.

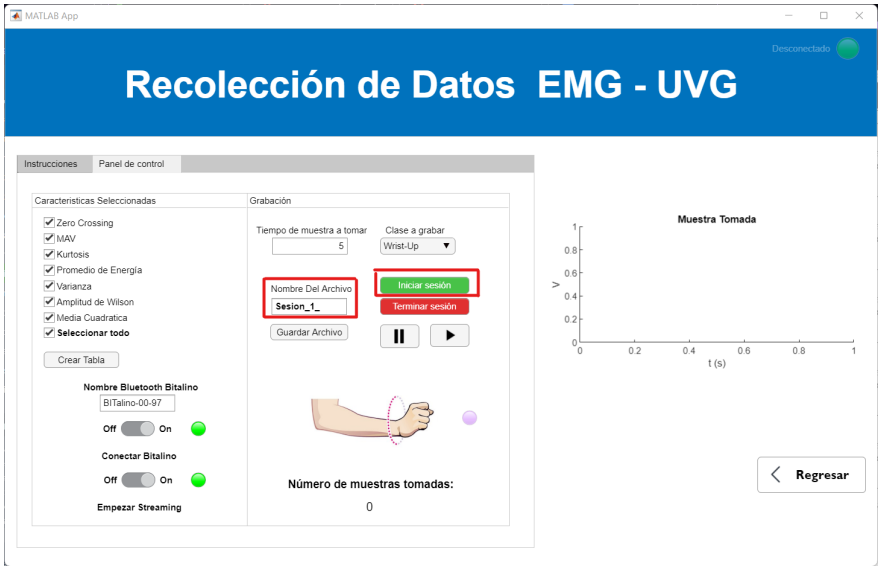


Figure 12: Colocar el número de sesión a grabar y empezar sesión

Se debe de colocar el número de sesión a grabar, no tiene que colocarse el nombre de la clase, ya que al guardarse automáticamente coloca el nombre de la clase seleccionada.

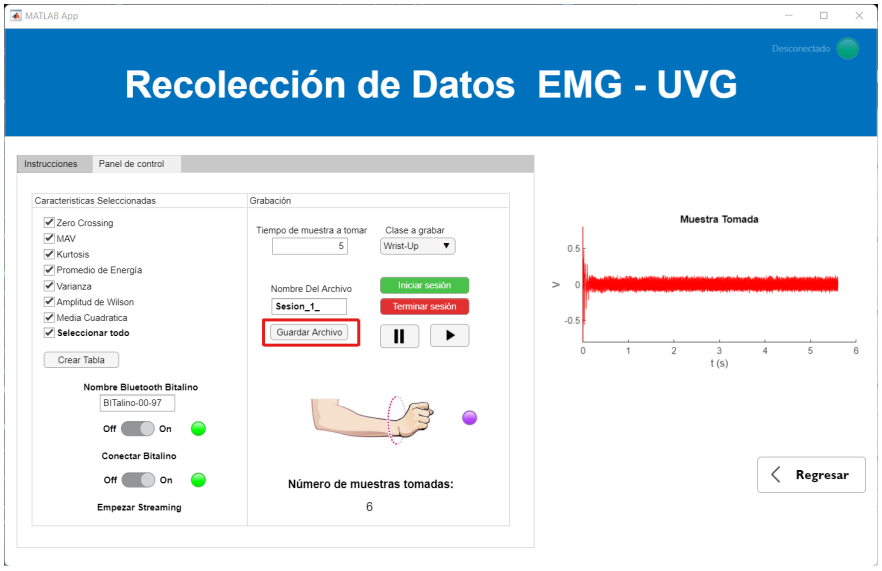


Figure 13: Guardar la sesión

Al llegar al número de muestras que se necesiten, se debe de pausar la interfaz. No se debe de presionar el botón **Terminar sesión** ya que esto borraría toda la data recolectada. Luego de pausar la interfaz se debe de presionar el botón **Guardar archivo**. Los dos .xls se guardarán en el directorio donde tenga guardada su interfaz.

## 3.2 Interfaz clasificadora de datos

Es necesario saber que está interfaz debe ser modificada si el vector de características que entrara en el modelo clasificador no cumple con las mismas dimensiones que el modelo base de entrenamiento que obtiene al clasificar con la data obtenida en la interfaz recolectora. La conexión con el robot R17, se explica de una manera superficial, dentro de este manual, para más información se debe de dirigir al manual de usuario escrito por José David Pellecer para el uso del Robot R17. Se recomienda hacer pruebas con la interfaz clasificadora y probar el modelo entrenado, antes de utilizarse con el robot R17, para evitar cualquier tipo de incidente.

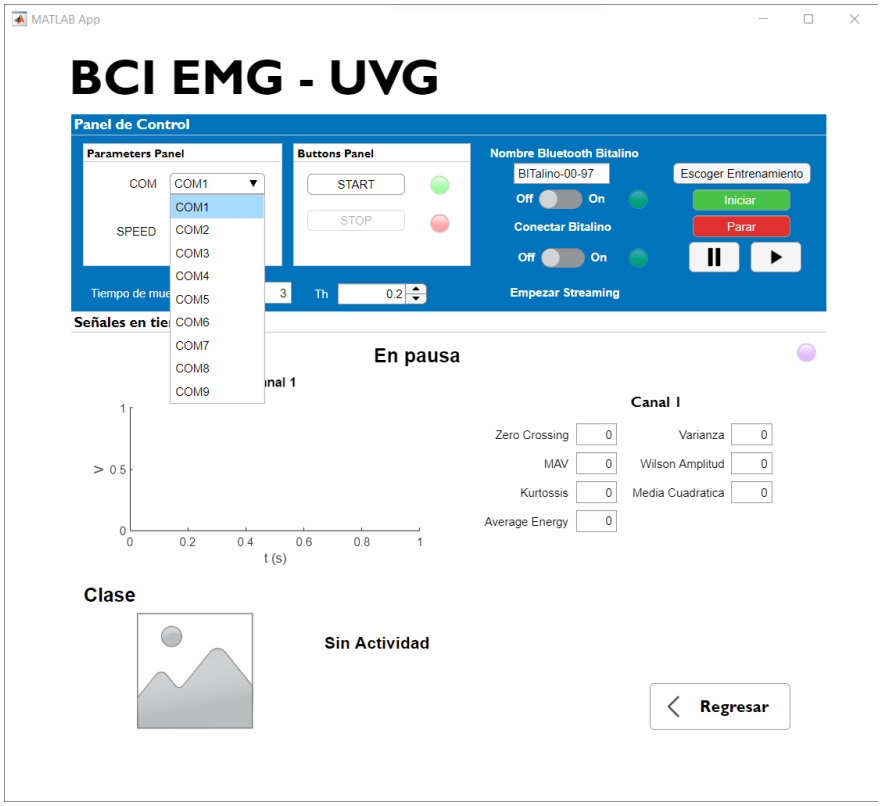


Figure 14: Se debe de escoger el COM al que pertenece la conexión con el robot R17.

Se debe seleccionar el COM que se creó al conectar el robot R17 a la computadora, si ninguno de los COM que aparecen como opción le sirve, puede agregar el que necesite desde *Designer App* de MATLAB. Ël parametro *speed* no se debe de modificar a menos que sea requerido, si se requiere saber más sobre este parámetro, de nuevo es necesario ir al manual de usuario del robot R17.

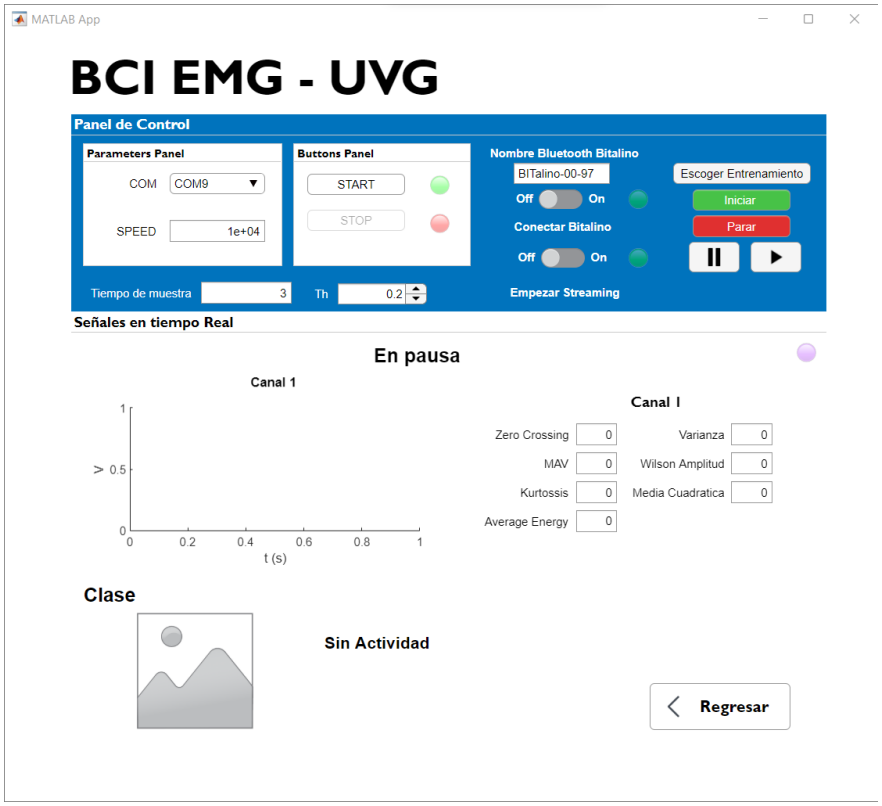


Figure 15: Presionar el botón start para conectarse con el Robot R17.

Al presionar el botón start, es bueno observar la consola principal de MATLAB, para poder observar si la conexión fue realizada con éxito.

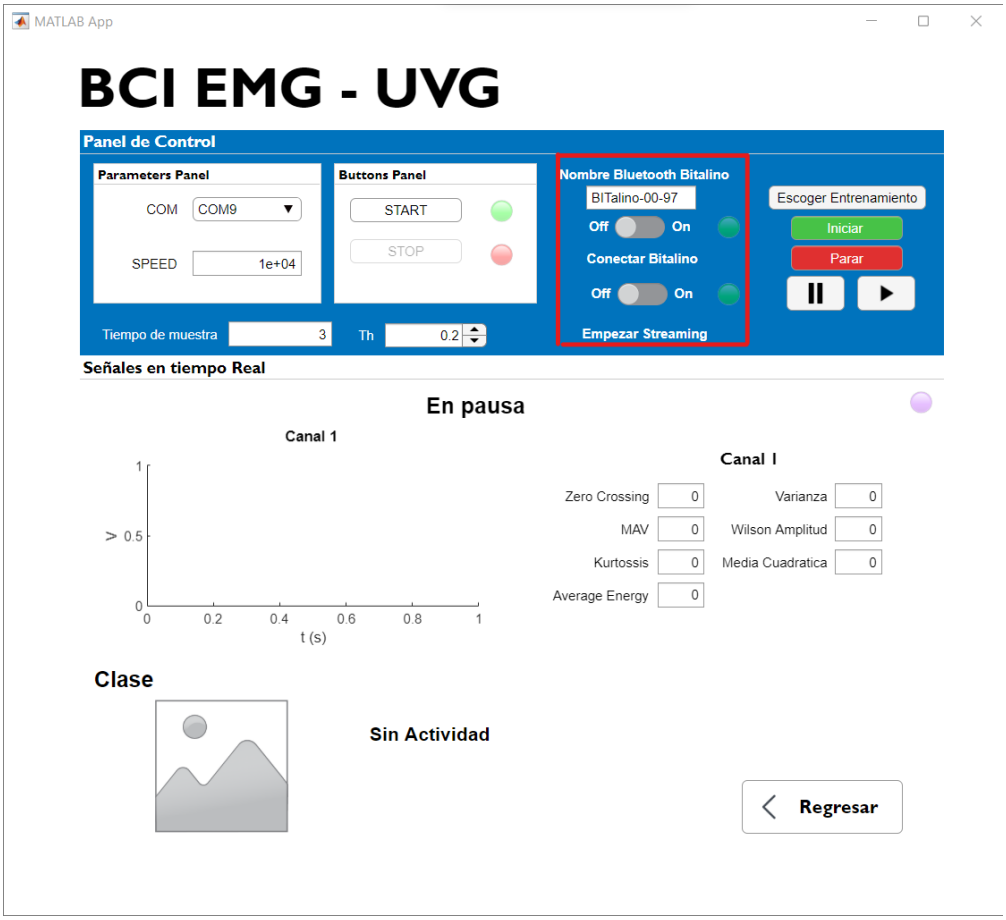


Figure 16: Se conecta con el dispositivo bitalino

La conexión con el dispositivo bitalino sigue siendo de la misma manera que con la interfaz recolectora. Con las mismas particularidades que se presentan al momento de conectarse al dispositivo.

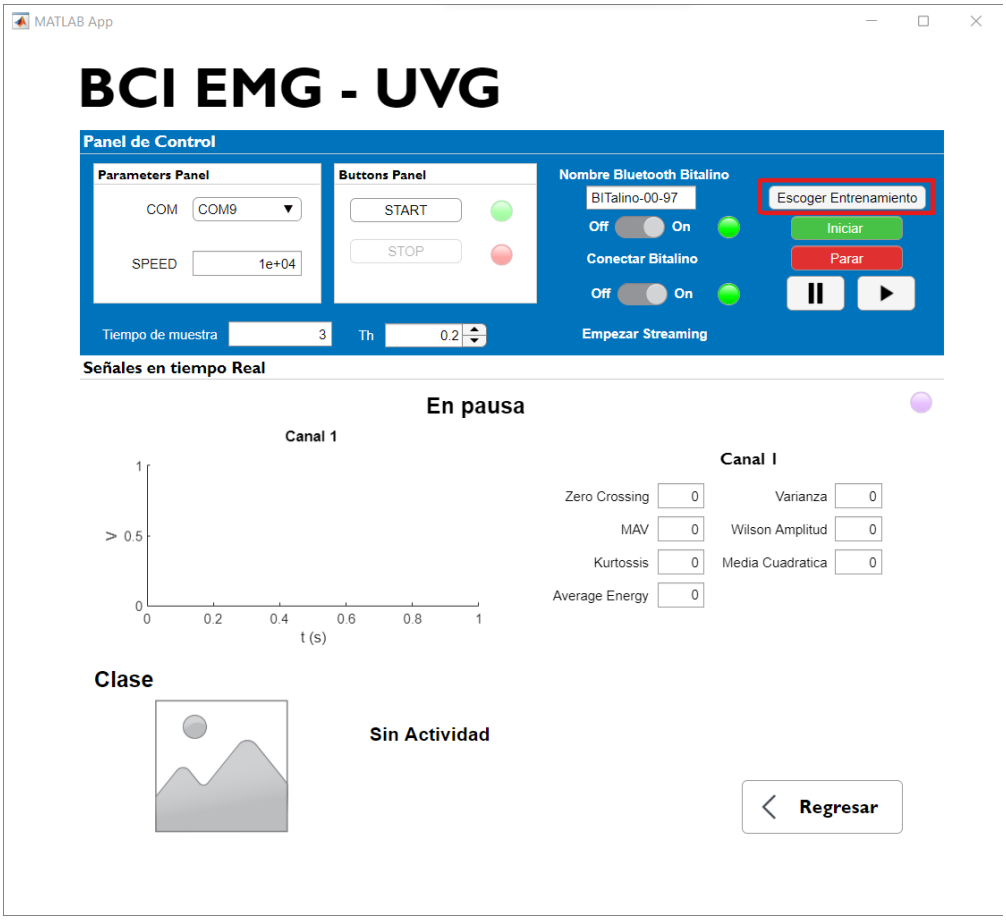


Figure 17: Escoger entrenamiento

Al presionar el botón **Escoger Entrenamiento** se desplegará una ventana que nos permite explorar nuestros archivos.

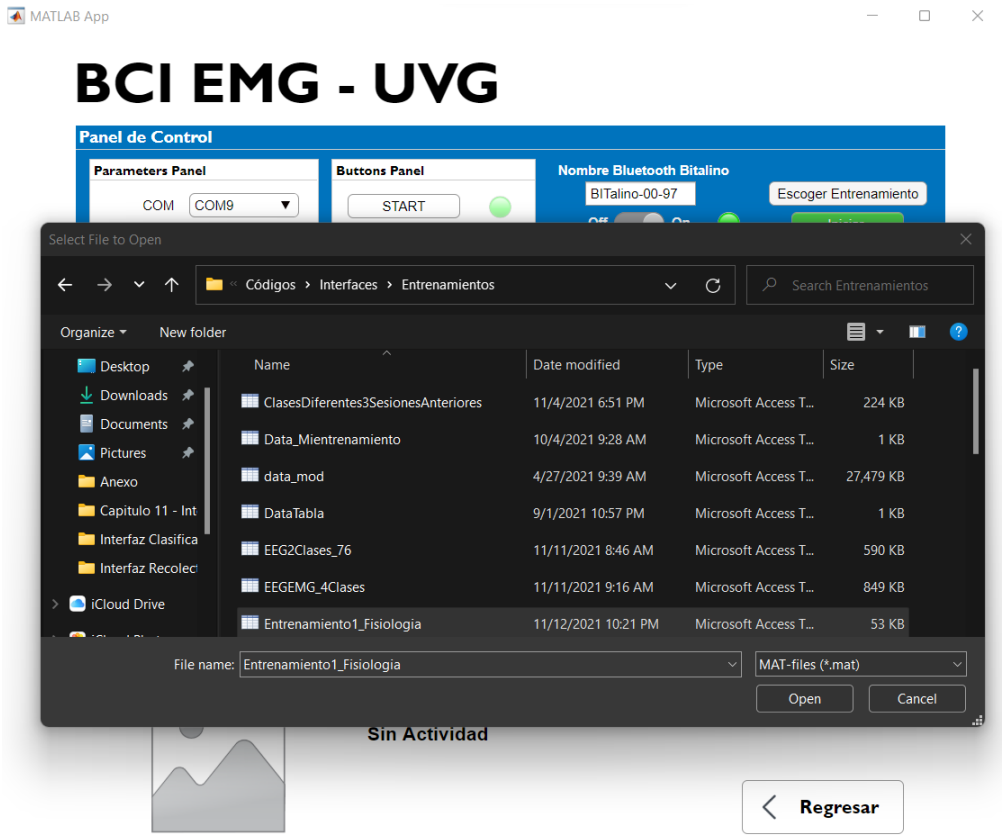


Figure 18: Escoger entrenamiento formato .mat

Es obligatorio usar un entrenamiento que cumpla con ser un formato .mat para que el software MATLAB pueda reconocerlo.

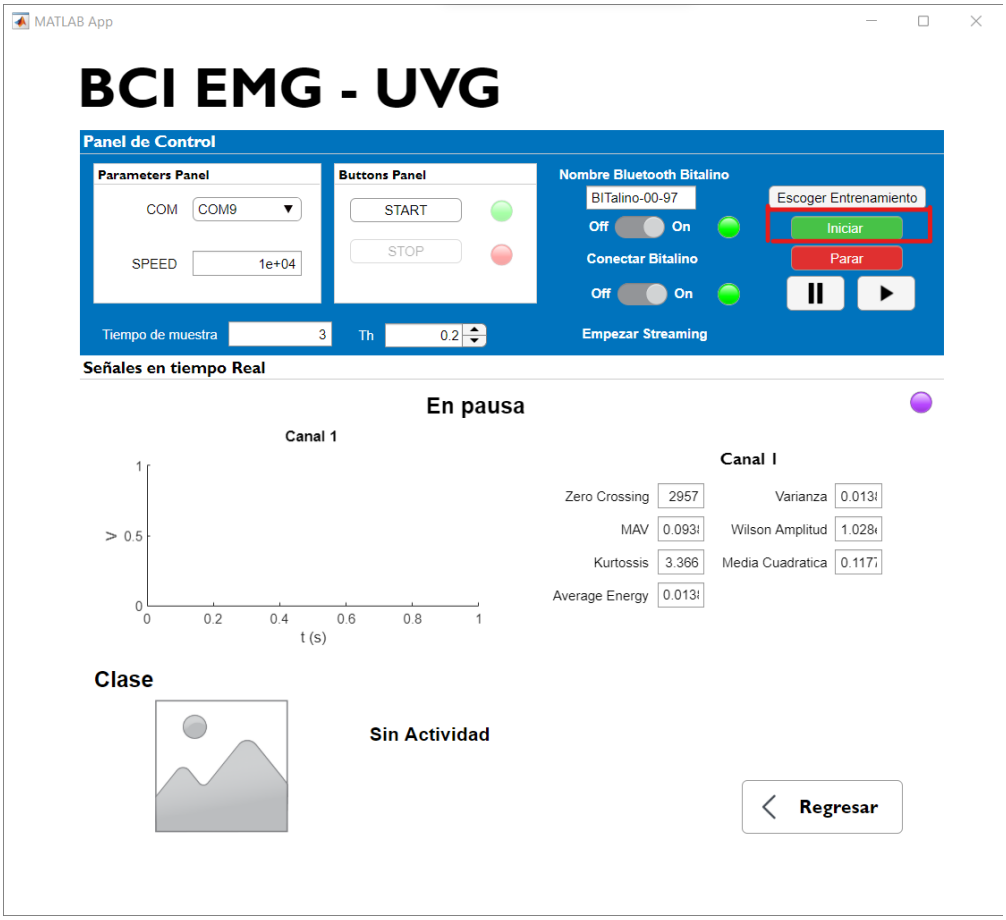


Figure 19: Iniciar Clasificación de señales

Al tener seleccionada la ventana de tiempo y el th que se requiere se puede presionar el botón **Iniciar** que empieza con la sesión de clasificación.

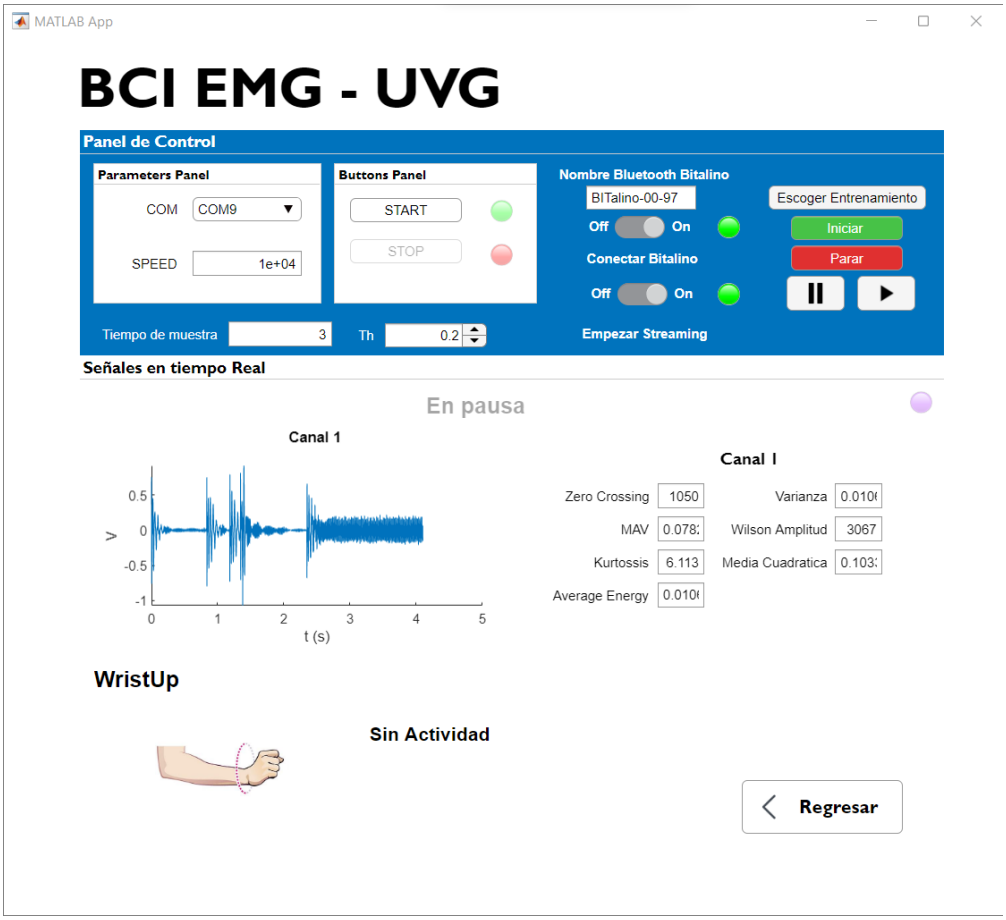


Figure 20: Ejemplo Clasificación

Como se puede observar en las últimas dos FIGURAS 19 y 20, en la primera ya está recibiendo señales, pero no clasifica porque no detecta actividad eso gracias el th asignado. En la segunda detecto actividad y clasifica esa señal. Se pueden utilizar los botones de pausa y reanudación por si se necesita cambiar de posición los electrodos o bien dejar de recolectar. si se presiona el botón **Terminar** sé borra el entrenamiento y se reinician variables internas.