Functioning Serial Monitor for Arduino

Table of Contents

The following program receives serial data from an Arduino microprocessor and displays the incoming numerical data as a live plot of value over time.

In this specific case, the program is designed to forces measured by a load cell connected to a Sparkfun HX711 Load Cell Amplifier. The Arduino must be programmed to output only the numerical value of force, at intervals of at least 50 milliseconds.

Clear the workspace and close all figures

```
clear
close all;
```

Create serial object for Arduino

```
s = serial('COM3','BaudRate',9600); % change the COM Port number as
 needed
```

Connect the serial port to Arduino

```
s.InputBufferSize = 16;
try
    fopen(s);
catch err
    fclose(instrfind);
    error('Make sure you select the correct COM Port where the Arduino
 is connected.');
end
```

```
Error using FunctioningSerialMonitor (line 20)
Make sure you select the correct COM Port where the Arduino is
 connected.
```

Create a figure window to monitor the live data

The code below uses a while loop to constantly update the animatedline by attempting to interpret the serial port values. When a value is received, the numerical value is added to the animatedline, the figure is redrawn and a copy of the data points of the entire animated line is updated in the xdat and ydat variables for saving later on.

```
Tmax = inf; % Total time for data collection (s)
figure
h = animatedline;
ax = gca;
ax.YGrid = 'on';
ax.YLim = [-10 1500];
xlabel ('Time (s)'), ylabel('Force in centiNewtons (10^-2 N)'),
stop = false;
startTime = datetime('now');
xdat=[];
ydat=[];
while ~stop
    A=[];
    while isempty(A)
        A=[str2double(fscanf(s))]; % Read Distance Value from Serial
    end
    t =  datetime('now') - startTime; % Get current time
    addpoints(h,datenum(t),A) % Add points to animation
    xdat=[xdat seconds(t)];
    ydat=[ydat A];
    ax.XLim = datenum([t-seconds(15) t]); % Update axes
    datetick('x','keeplimits')
    drawnow
end
```

Data Save

Enter these commands into the command console after pausing and stopping the program to save the data. **Do not stop the program with CTRL+C. This will wipe the data from the memory. Instead, use pause and stop in the editor panel to stop the program.**

```
data=[xdat;ydat];
fileID = fopen(""+datestr(now,'dd_mmm_yy HHMM')+"HRS.txt",'w');
fprintf(fileID,'%f %f\n',data);
fclose(fileID);
```

Published with MATLAB® R2019a