Lab Sheet

IA 3018 – Data Acquisition Systems

Department of Instrumentation and Automation Technology
University of Colombo

Practical 02

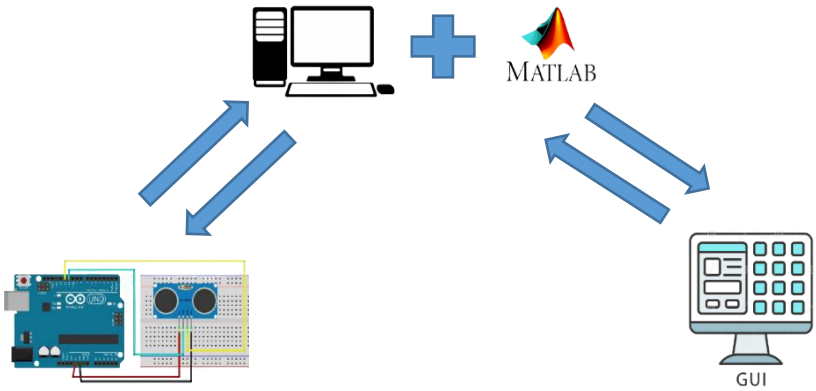

Figure 01:

<u>Part 01</u> - Ultrasonic Sensor Interfacing with Arduino and MATLAB.

1. Add the Arduino support package and Ultrasonic libraries to MATLAB.
2. Connect the Arduino board to computer and record the COM port and board name.(Nano, UNO, Mega, Due)
3. Check the Arduino support package are working in MATLAB using above recorded data in part 2.
4. Check the Ultrasonic library are working (If not add the ultrasonic library to MATLAB) using below command.

```
>> a=arduino('COM7','UNO','Libraries','Ultrasonic')
Updating server code on board Uno (COM7). This may take a few minutes.

a =
```

5. Create the circuit as figure 02.

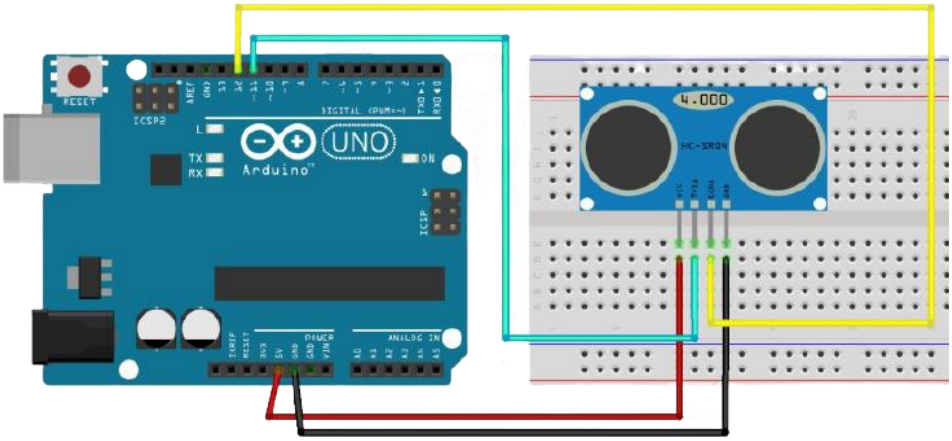

Figure 02: Circuit diagram

6. Write a Matlab code for read the distance value using ultrasonic sensor.

```
>> sensor = ultrasonic(a,'D8','D9')

sensor =

  Ultrasonic with properties:

    TriggerPin: 'D8'
       EchoPin: 'D9'

>> readDistance(sensor)

ans =
```

Figure 03: Example code

<u>Part 02</u> – Read distance values using Ultrasonic sensor and Arduino IDE

1. Write an Arduino code for read distance values and monitor the voltage value. (Appendix 01)
2. Upload the Arduino code to Arduino UNO board and check the code.
3. Open the Serial Monitor and check the output voltage values as figure 04.

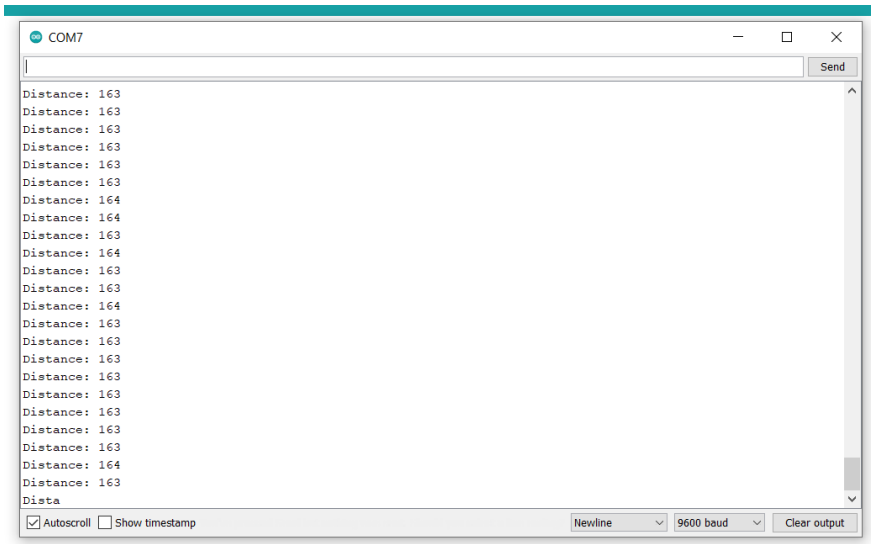

Figure 04: Arduino serial monitor

<u>Part 03</u> – Create a GUI for displaying obstacle distance value on Matlab.

1. Open GUI window in Matlab.
2. Select blank GUI and save as “Obstacle_Distance”.
3. Add topic for the GUI using “Text” function and change the appearance.
4. Add another two “Text” function as figure 05.

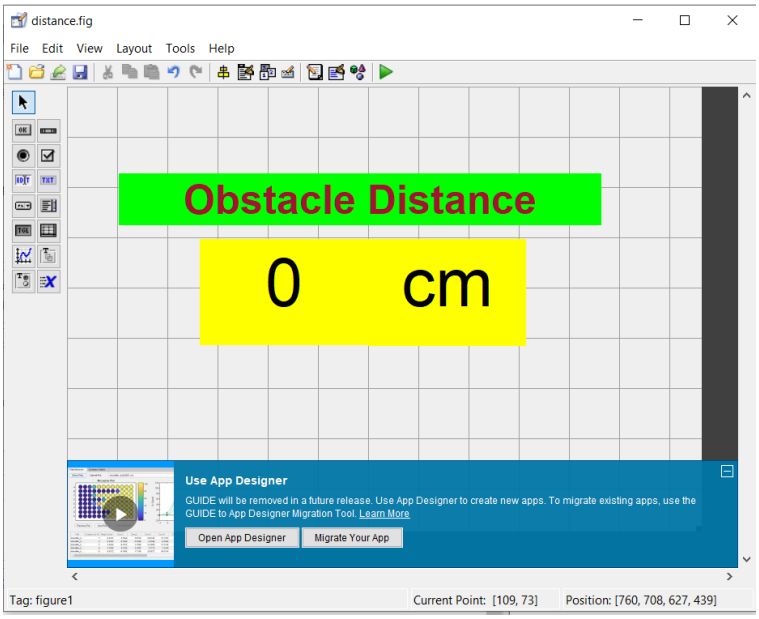

Figure 05: GUI editor

5. Record the Baud rate of Arduino COM port using Device Manger setting in computer.
6. Edit the GUI script using bellow (Figure 06 and Figure 07) commands.

Editor - E:\Matlab\Example\distance.m

distance.m

```
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% varargin   unrecognized PropertyName/PropertyValue pairs from the
%            command line (see VARARGIN)

% Choose default command line output for distance
handles.output = hObject;
% Delete any opened ports in MATLAB
delete(instrfind)
% Create a Serial Object
handles.ser = serial('COM7', 'BaudRate',9600,'Terminator','LF',...
    'Timeout',10);
% Associate Serial Event, whenever Terminal Character is recived
handles.ser.BytesAvailableFcn = {@SerialEvent, hObject};
% Open Serial Port
fopen(handles.ser);
% Update handles structure
guidata(hObject, handles);
```

Figure 06: Code part 01

Editor - E:\Matlab\Example\distance.m

distance.m

```
varargout{1} = handles.output;

function SerialEvent(sObject, eventdata, hGui)
% get the updated handle
handles = guidata(hGui);
% get data from serial port
tmp_c = fscanf(sObject);
set(handles.textDistance, 'String', tmp_c)
% Updates handle structure
guidata(hGui, handles)

% --- Executes when user attempts to close figure1.
function figure1_CloseRequestFcn(hObject, eventdata, handles)
% hObject    handle to figure1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
fclose(handles.ser);
delete(handles.ser);
% Hint: delete(hObject) closes the figure
```

Figure 07: Code part 02

7. Run the scrip and check the output.

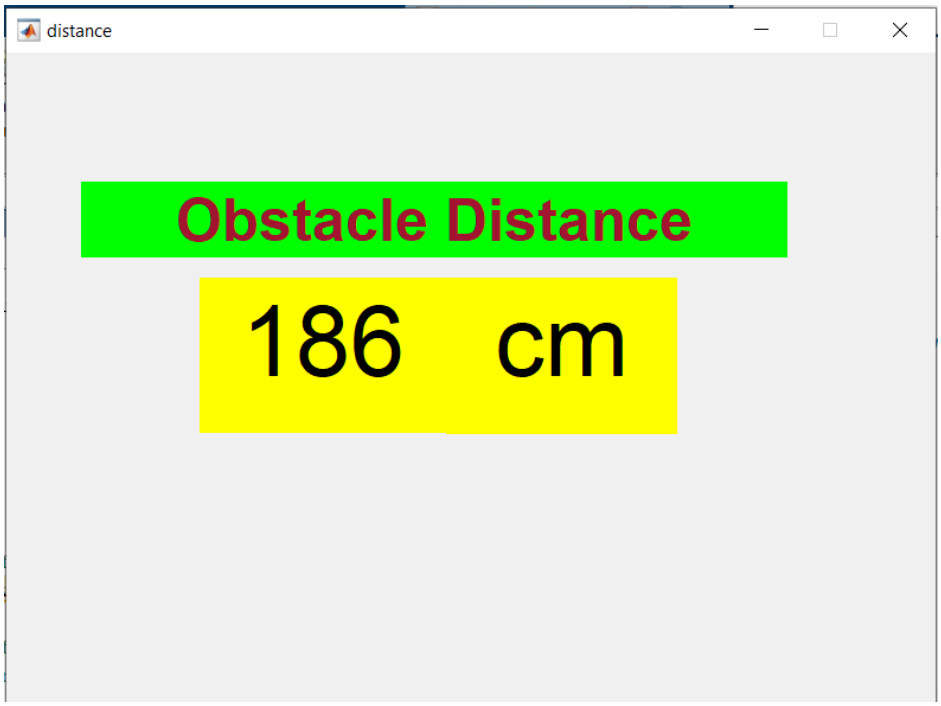

Figure 08: GUI output window

Appendix 01

Arduino code for ultrasonic sensor

ultrasonic2 | Arduino 1.8.19

File Edit Sketch Tools Help

ultrasonic2

```
const int trigPin = 9;
const int echoPin = 10;
// defines variables
long duration;
int distance;
void setup() {
pinMode(trigPin, OUTPUT); // Sets the trigPin as an Output
pinMode(echoPin, INPUT); // Sets the echoPin as an Input
Serial.begin(9600); // Starts the serial communication
}
void loop() {
// Clears the trigPin
digitalWrite(trigPin, LOW);
delayMicroseconds(2);
// Sets the trigPin on HIGH state for 10 micro seconds
digitalWrite(trigPin, HIGH);
delayMicroseconds(10);
digitalWrite(trigPin, LOW);
// Reads the echoPin, returns the sound wave travel time in microseconds
duration = pulseIn(echoPin, HIGH);
// Calculating the distance
distance= duration*0.034/2;
// Prints the distance on the Serial Monitor
Serial.print("Distance: ");
Serial.println(distance);
}
```