**Problem Description:**

You can work in your groups for this project—but, individually working your own code is recommended. However, the reports uploaded must be individual. Your uploads will include your code and a YouTube link to your 2-minute project video, and a report of your experiments and a review of the given paper.

Your project goal is to create a Graphical User Interface (GUI) using MATLAB, Object-Oriented Programming and the Appdesigner system. The GUI should have a central area (one or more) where graphing is done and menu items, buttons and LEDs will be located as needed. The system should connect Matlab with an Arduino microcontroller with a sonar distance sensor hooked up to it and a servo to simulate a robot. The system should take distance data and time data from the computer (see: tic/toc) from a simulated beating heart. You will clean and smooth this data and show the distance vs. time plot of your heart data. You should also allow for continuous plotting of sensor data.

You will also broadcast this data on a ROS2 node to another MATLAB program. This subscriber node will read the data and display a simulated robot (you can use the Puma560 or other prebuilt robot or another robot you have built). This robot will move the end-effector back and forth to match the incoming data. You will also simulate the movement in hardware with a servo motor.

Pin6 Echo

A picture containing text, electronics

Description automatically generatedA picture containing electronics, projector

Description automatically generated

Pin7 Trig



Note: Power and ground are on the left (red, power) and (blue, ground)

In this example, the trigger is pin seven and, the echo is pin six.

The following features should be part of your system.

1. It should have a menu or button to initialize the Arduino and the distance sensor. Use the given example program, but make sure that the system can have the variables of echo and transmit pins, port etc., to be input as part of the GUI.
2. Create a simulated heart from an appropriately inflated balloon. You should simulate a heart beating at regular intervals applying and releasing pressure on the balloon. Use music or a metronome for consistency.
3. The data captured should be cleaned and smoothed using standard techniques (e.g. median filter) or your developed techniques.
4. To help you determine good positioning, create LEDs (on the breadboard and the GUI) that indicate when good data has been captured. If the heart is in motion, a green LED lights up; a red LED lights up if it is stopped (between beats). Make sure to use the appropriate resistors to ensure the LEDs don’t burn out.
5. You will also hook up a small servo motor to your Arduino. As the heart beats, your simulated robot will also move the same distance away from the ‘heart’. Do the math to figure out approximately how many degrees are needed for how much movement.
6. You should have a menu or GUI features that allow the data to be captured from actual hardware or a file you have saved from the GUI and can reload. This will allow you to debug code and get it working without the hardware hooked up. Do this first and it will make debugging much easier.
7. In the center of the GUI, create a plotting system of your design. Make sure to be able to plot the original data and the smoothed data.

**Experiments:**

* 2D Continuous plotting: Create a button that continuously plots your sensor values for a selectable amount of time.
* Collect data from a heartbeat of at least three different frequencies.
* Using the mouse and clicking on the peaks of your graph, determine the heart rate for each of the collections.
* How accurately can the hardware robot follow the beating heart?
* What is the timelag in transmission and final movement of the servo?

**Project Requirements:**

* Read and summarize the enclosed paper and explain how your project and this relates to the paper. What are the weaknesses of the paper and where are the technological gaps to make it into a real system? How might you conduct further research and development?
* A critical exercise in this project is organizing these codes in a structured manner (objected-oriented structured programming). **Your program codes must be well documented**. Reuse code whenever possible, create mini functions that can be used between methods (do not duplicate code that can be written as functions).
* Indicate in the commenting what portions of the code were done by which group member. Indicate also what % was done by which member.
* Create a 2-minute video of your project. Show the software running, the hardware setup, the data collection, and how your simulated robot arm worked. All members of the team should narrate some portion of the video.

Make sure your code is well organized, neat, clear. Make the code and video something you could show at an interview.

**Make sure the code does not contain cut and pastes from other student code or the internet. This will result in a zero for the project for both.**

**Programming Tips:**

**How to read an ultrasound sensor in MATLAB?**

Make sure to install the MATLAB Arduino support package—and the ultrasound library.

In order to load Arduino with the ultrasound library installed, you will have to issue a command like this: Note that ‘COM7’ should be replaced with your com and ‘D7’ ‘D6’ should be replaced with your pins on the ultrasound.

a = arduino('COM7','Uno','Libraries','Ultrasonic');

% app.Arduino is a private variable of your class, you can

% name it anything you want.

% connect to the Ultrasound sensor the class variable

% is UltraSound. Same as above.

us = a.ultrasonic('D7','D6');

To Read data from the Ultrasound sensor:

value =us.readDistance();

Here, create an array private variable that you read data into.

**How to move a servo?:**

Description:

Set the position of a standard servo motor shaft as a

ratio of the motor's min/max range, from 0 to 1

Example:

a = arduino();

s = servo(a, 'D9');

writePosition(s, 0.60);

**How to read time in Matlab?:**

Read up on the tic/toc features to get the time values.

tstart = tic;

… read sensor for instance

T = toc (tstart)… T will be the amount of time between the last tic.

**How to create a ROS2 node in MATLAB and how to create a subscriber and publisher?**

% The topic type is inferred (if topic /chatter exists)

chatPub = ros2publisher(node,"/chatter","std\_msgs/String");

chatSub = ros2subscriber(node,"/chatter",@testCallback);

The @testCallback is the function you would write to act on the incoming data from the publisher node. See class notes and examples.