Assignment 1: Developing Sensor Class

Intent: Skills in utilising, classes, functions, pointers and macros will be examined in developing sensor class on a provided mechatronics sensor specification.

Individual Task

Weight: 5%

Task: Write a program in C++ using object oriented paradigms that implements a given Mechatronics senor in line with a specification. The solution will need to provide appropriate access to sensor attributes (encapsulates the parameters) and enable access to underlying sensor data and attributes.

Rationale: In a Mechatronics System we would use a class to represent a real-world object such as a Sensor. In large projects to facilitate testing it is common to develop a mock (fake) sensor class that behaves the same as the real sensor. This allows for some system testing without the presence of all hardware. Your task is to create such a mock sensor class that encapsulates attributes of a specific sensor. Students are allocated one sensor (Laser or 3 Axis Accelerometer), check UTSOnline for allocation.

Details of the Marking Criteria are available in the Subject Outline.

Due: Sunday 1st April 2018 23:59.

Specifics

Create a Sensor Class that allows:

1. Initialising all the required variables to enable connecting to the sensor
2. Obtaining all the hardware specific fixed parameters of the sensor
3. Setting configurable parameters of the sensor
4. Inform if values are sane, use default values otherwise
5. Obtaining sample sensor data
    a. Read of data is blocking so the sensor works in poll mode - sensor provides data at specific sampling rate (ie at rate of 30Hz, data should only appear every 1/30 seconds and will block otherwise).
    b. Data should be generated as random values within the specific sensor range/resolution

Create a Main that

1. Initialises the sensor
2. Queries the fixed sensor parameters
3. Sets sensor parameters as specified by the user
4. After 1-3 is executed continuously queries and displays data from the sensor with the sample sequence number until the program terminates

Assessment

Criteria	Weight (%)	Description / Evidence
Encapsulation of all sensor attributes and data with appropriate access methods.	50	Appropriate selection of member access (variables and methods). Differentiate access to functions required by the class-user and those used to implement the inner workings of the class should be private.
Proper code execution	25	User input of parameters works as expected, data values reported as per sensor description (sample number, values and timing).
Modularity of software	25	Appropriate use class declarations, definitions, default values and naming convention allowing for reuse of Sensor class. Sensor class and Main interface in ways allowing use of class in others contexts (ie. instead of user main, a robot with no console interface).

Sensor 1 : Laser Range Finder

Specifications

Model	UTM-XXL
Baud	38400 or 115200
Port	USB (typically /dev/ttyACMX) …where X=0,1,2
Field of View	270 degrees
Angular Resolution	1.0deg or 5.0deg
Scanning Time	50ms/scan or 25ms/scan
Max Distance	8.0m
Min Distance	0.1m

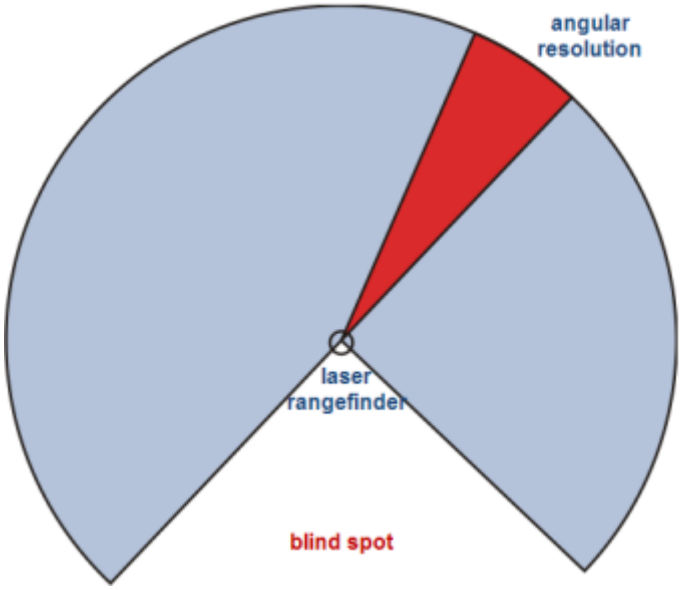

Figure 1 – Relating Angular Resolution and Field of View (the image has Field of View 270deg). Image from http://www.robotreviews.com/blog/eigenlance/robots-and-laser-rangefinders-part-2, last viewed 30 March 2016

Sensor 2 : 3 Axis Accelerometer Sensor

Specifications

Model	HWE-XXL
Baud	19200 or 38400
Port	USB (typically /dev/ttyUSBX) …where X=0,1,2
Sampling Time	10Hz
Max Acceleration	$10m/s^2$, $20m/s^2$, $50m/s^2$
Sampling Resolution	2^{10}

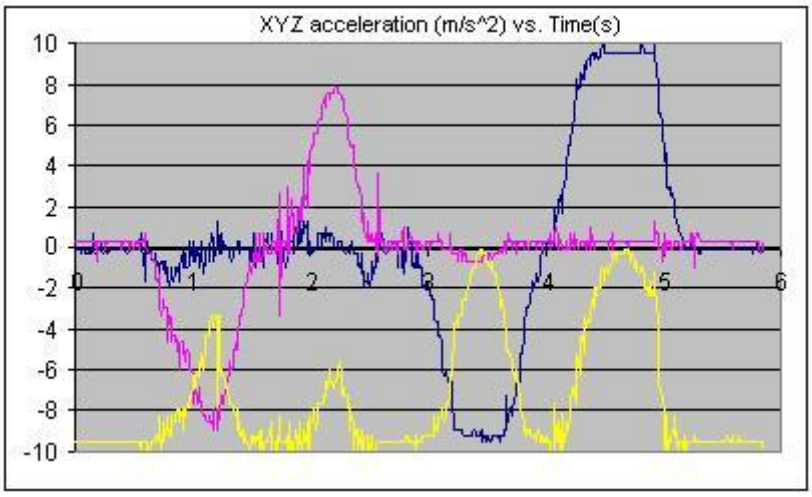

Figure 2 - Sample 3 axis accelerometer data with $10m^2/s$ limit, image from http://profmason.com/wp-content/uploads/2007/05/acceleromter.JPG, last viewed 30 March 2016