Real-Time Control Systems CPE533
Course Project

This project is going to focus on the main concepts of real-time system programming (RTOS), as well as basic concepts of sensor calibration and filtering. We will base this project on ARM CMSIS RTOS (RTX) Ver 1.0 and use Keil µVision 5.27+. For the sensory input, you will be provided with actual traces from real sensors since we don't have access to actual boards.

You can start doing this project at once without threads as separate functions called in main(). You don't need to worry about timing or sampling rates at this stage, just the processing. Once you learn how to do periodic RTOS threads and synchronization, you just need to copy your codes into these threads.

Project Details

In the project folder, you are provided with two sensor logs, and six accelerometer calibration logs. All logs are generated from an actual triaxial (3D) accelerometer (LSM6DSL) and triaxial (3D) Magnetometer (LIS3MDL). The accelerometer was configured to measure acceleration in the range ±1g (±1000mg) as our project assumes no linear acceleration present (board not mounted on a moving object, e.g. car). The magnetometer was configured to measure the magnetic field in the range ±4Gauss (±4000mGa). Therefore, the data points in the log files provided are within these ranges. Both sensors were configured to provide data at an ODR (Output Data Rate) of 104Hz. This means each 104 samples in any file were collected over a period of one second. However, to simplify the project timing, assume that we collected 100 samples in each second, therefore we will read a new sample every 10ms.

Part 1 (MATLAB)

You first step will be to use the six accelerometer calibration logs to derive the calibration matrix. These logs have been collected using the procedure we described in class where we position each axis to coincide with or in opposing direction the *g* vector. Ideally, we would get values around ±1g (1000mg) and 0g on the other axes. You are required to perform the calibration as an offline step in MATLAB using the Least Squares Method (LSM). **The deliverable of this step is a calibration matrix (scalars and offsets).** No calibration is required for the magnetometer sensor (no soft iron or hard iron calibration).

The data in the main two logs (that of the accelerometer and the magnetometer) must be manually imported into arrays in header files and included in the project. Use as many data points in your array as you can given the constraints of your system memory and operational requirements. You have to justify why you have used all or some of these data points.

Part 2 (Embedded C)

In the next step, you are required to design a CMSIS RTOS-based system. Each sensor will have a thread to read the sensor data (data acquisition) from the array one by one at a rate of 10ms. The sensor must be read at the specified ODR. The accelerometer data acquisition thread will also apply data calibration on the read data at the same time (multiply the data by the calibration matrix and add the offset). Remember each thread will read three values for each sensor, one for each dimension. All threads will share data using shared global variables. Do not use mail boxes. Once you reach the end of the arrays, restart from the beginning as we are simulating a continuous system.

Part 3 (Embedded C)

Two other threads will filter out the data of the accelerometer and the magnetometer. **These threads must only start when new data is read.** You are required to use a simple 1K Kalman filter as described in class.

You must use a separate filter for each axis and for each sensor. You must carry out few experiments to determine the proper values of the uncertainty (noise covariance) variables: Q, P, and R. You must do this for each axis of the accelerometer data; and for each axis of the magnetometer sensor. Since both sensors provide data of different nature and properties, the filter characteristics (i.e. Q, P, R) might not necessarily be the same.

Hint: To conduct your analysis, print the filtered output in Keil *printf* window, copy/paste it into a file or excel sheet, plot the filtered output against the original data in Excel or MATLAB, compare the two signals in terms of smoothing effect, distortion, … etc. Good values of Q, P, and R must smooth the signal and never distort it.

Part 4 (Embedded C)

Finally, once each data point is filtered from all sensors for all dimensions, it is time to make sense of the data. We are only interested in the pitch and roll angles. The first step is to derive the roll and pitch angles from the accelerometer data. You can use Equation 1 and Equation 2:

$$\text{Pitch} = \alpha = \arctan\left(\frac{A_{x1}}{\sqrt{(A_{y1})^2 + (A_{z1})^2}}\right) \quad (1)$$

$$\text{Roll} = \beta = \arctan\left(\frac{A_{y1}}{\sqrt{(A_{x1})^2 + (A_{z1})^2}}\right) \quad (2)$$

As for the magnetometer, we are interested in finding the relative *heading* of the board. The heading is computed by:

$$\text{heading} = \arctan(A_y, A_x) \times 180 / \pi; \quad (3)$$

However, use atan2() function instead of atan() for this part.

- For some operations, you may use the CMSIS DSP library (optional). However, you need to include the "math.h" in your project to do the mathematical operations required.
- Use normal single-precision floating point numbers.
- Use the base CMSIS-RTOS project provided to quickly jump into design, programming, and testing.
- When submitting your project, please do delete all files inside the "Objects" and "Listings" folders before compressing your project. You can also do so from Project→ Clean Targets. If you don't. your email attachment might fail.
- **Do not submit single files. I must be able to click on the project and open it directly in Keil.**
- No report is required for this project.

Software Tools

Keil MDK v5. 29 (Click to go to installation page)

The ready project templates found on the course webpage

Good Luck and Enjoy 😊