ELA001

Examiner:

October 29, 2024

Course Information

- **Course code:** ELA001
- **Subject:** Electronics
- **Credits:** 7.5
- **Main field of study:** Electronics, depth G2F

Tasks for UdeA

The following tasks will be completed by University of Antioquia (UdeA):

1. **Hardware Interface for Sensors:**

   - Design and implement the hardware interface for collecting sensor data.
     The following are some example API calls the hardware interface should have:

```
// VL53L1 Time-of-Flight Sensor
VL53L1_Init();                              // Initialize the sensor
VL53L1_GetDistance();                       // Get distance in mm
VL53L1_GetRangeStatus();                    // Get range status
VL53L1_SetRangingMode(uint8_t mode);        // Set ranging mode
VL53L1_SetInterruptThresholds(uint16_t low, uint16_t high); // Set interrupt thresholds

// VL6180 Proximity and Ambient Light Sensor
VL6180_Init();                              // Initialize the sensor
VL6180_GetProximity();                      // Get proximity value
VL6180_GetRange();                          // Get range distance
VL6180_SetInterruptConfig(uint8_t config);  // Configure interrupt settings

// APDS9960 Gesture and Color Sensor
APDS9960_Init();                            // Initialize the sensor
APDS9960_EnableProximity();                 // Enable proximity sensing
APDS9960_GetProximity();                    // Get proximity value
APDS9960_GetColor();                        // Get RGBC color data
APDS9960_SetProximityInterruptThreshold(uint8_t low, uint8_t high); // Set proximity interrupt
APDS9960_ClearInterrupt();                  // Clear interrupts
```

   - **BNO055 Orientation Sensor:**
     - Implement the interface for the BNO055 sensor to retrieve orientation, acceleration, and other motion-related data.
       Example API calls for the BNO055 sensor interface:

```
// BNO055 Sensor Initialization
BNO055_Init();                                // Initialize the BNO055 sensor

// Sensor Configuration
BNO055_SetOperationMode(uint8_t mode);        // Set the operation mode (e.g., IMU, NDOF)

// Sensor Data Retrieval
```

```
BNO055_GetQuaternionAngles(float *yaw, float *pitch, float *roll); // Get
    orientation data (Euler angles)
BNO055_GetAcceleration(float *x, float *y, float *z);         // Get linear
    acceleration
BNO055_GetGyro(float *gx, float *gy, float *gz);              // Get
    gyroscope data
BNO055_GetMagnetometer(float *mx, float *my, float *mz);      // Get
    magnetometer data
```

- **AS5600 Magnetic Encoder:**
  - Implement the interface for the AS5600 magnetic encoder to retrieve angular position. Example API calls for the AS5600 sensor interface:

```
// AS5600 Sensor Initialization
AS5600_Init();                              // Initialize the AS5600 sensor

// Get Raw Angle
AS5600_GetRawAngle();                       // Get the raw angle from the
    sensor

// Get Adjusted Angle
AS5600_GetAngle();                          // Get the adjusted angle (
    taking zero-position into account)

// Set Zero Position
AS5600_SetZeroPosition();                   // Set the zero position for
    angle measurements
```

2. **Hardware Interface for Motors:**

- Develop the hardware interface to control the motors for a three-wheeled omni-directional drive system.
- Integrate the interface with the motor drivers.
  Example API calls for the motor control interface:

```
// Motor Initialization
Motor_Init();                             // Initialize the motor controller

// Motor Control
Motor_SetSpeed(uint8_t wheel_id, int speed); // Set the speed of a specific
    wheel
Motor_GetSpeed(uint8_t wheel_id);            // Get the current speed of the
    wheel
Motor_SetDirection(uint8_t wheel_id, int dir); // Set the direction (e.g.,
    forward, reverse)
Motor_GetDirection(uint8_t wheel_id);        // Get the current direction of the
     wheel

// Encoder Readings
Motor_ReadEncoder(uint8_t wheel_id);         // Read the encoder value for a
    specific wheel

// Stop Motor
Motor_Stop(uint8_t wheel_id);                // Stop the motor
```

It is not necessary to implement all of these functions and you can change them as you see fit, but we want an interface for common use of each sensor (i.e. sample data, retrieve range values etc...).

Embedded System Setup

Our system will run on an embedded platform using micro-ROS alongside FreeRTOS for real-time task management. micro-ROS will be used for communication and coordination between the different components, while FreeRTOS will handle task scheduling and prioritization.

Function Documentation Requirement

Each function implemented in the project must be properly documented using doxygen. The documentation should include the following:

- A clear description of what the function does.
- The parameters it accepts (data type and description).
- The return values (if applicable) and what they represent.
- Any special conditions or limitations for using the function.
- Example usage, if necessary, to demonstrate how the function is called and utilized in the code.