HX711 - 로드 셀 기반 무게 측정 센싱 로직 구현 및 검증

- 3선식 로드셀(Strain Gauge) 4EA 연결, HX711 Amp. 칩을 통한 무게 측정 데이터 확인.
- Raw Data 및 실 측정 무게의 비교 검토를 통한 데이터 보정 로직 적용
  - 보정 기준 무게 2kg (2L 물)
  - 1차 방정식 형태의 로직 적용
    - (기울기 = 0.03011, 편차 = 696.0)

```
/*
   --------------------------------------------------------------------------------------
   HX711_ADC
   Arduino library for HX711 24-Bit Analog-to-Digital Converter for Weight Scales
   Olav Kallhovd sept2017
   --------------------------------------------------------------------------------------
*/

/*
   Settling time (number of samples) and data filtering can be adjusted in the config.h file
   For calibration and storing the calibration value in eeprom, see example file "Calibration.ino"

   The update() function checks for new data and starts the next conversion. In order to acheive maximum effective
   sample rate, update() should be called at least as often as the HX711 sample rate; >10Hz@10SPS, >80Hz@80SPS.
   If you have other time consuming code running (i.e. a graphical LCD), consider calling update() from an interrupt routine,
   see example file "Read_1x_load_cell_interrupt_driven.ino".

   This is an example sketch on how to use this library
*/

#include <HX711_ADC.h>
#if defined(ESP8266)|| defined(ESP32) || defined(AVR)
#include <EEPROM.h>
#endif

//pins:
const int HX711_dout = 6; //mcu > HX711 dout pin
const int HX711_sck = 7; //mcu > HX711 sck pin

//HX711 constructor:
HX711_ADC LoadCell(HX711_dout, HX711_sck);

const int calVal_calVal_eepromAdress = 0;
unsigned long t = 0;
```

```
35
36 void setup() {
37   Serial.begin(57600); delay(10);
38   Serial.println();
39   Serial.println("Starting...");
40
41   float calibrationValue; // calibration value
42   calibrationValue = 696.0; // uncomment this if you want to set this value in the sketch
43   // calibrationValue = 0.00;
44 #if defined(ESP8266) || defined(ESP32)
45   //EEPROM.begin(512); // uncomment this if you use ESP8266 and want to fetch this value
   from eeprom
46 #endif
47   //EEPROM.get(calVal_eepromAdress, calibrationValue); // uncomment this if you want to
   fetch this value from eeprom
48
49   LoadCell.begin();
50   //LoadCell.setReverseOutput();
51   unsigned long stabilizingtime = 2000; // tare preciscion can be improved by adding a few
   seconds of stabilizing time
52   boolean _tare = true; //set this to false if you don't want tare to be performed in the
   next step
53   LoadCell.start(stabilizingtime, _tare);
54   if (LoadCell.getTareTimeoutFlag()) {
55     Serial.println("Timeout, check MCU>HX711 wiring and pin designations");
56   }
57   else {
58     LoadCell.setCalFactor(calibrationValue); // set calibration factor (float)
59     Serial.println("Startup is complete");
60   }
61   while (!LoadCell.update());
62   Serial.print("Calibration value: ");
63   Serial.println(LoadCell.getCalFactor());
64   Serial.print("HX711 measured conversion time ms: ");
65   Serial.println(LoadCell.getConversionTime());
66   Serial.print("HX711 measured sampling rate HZ: ");
67   Serial.println(LoadCell.getSPS());
68   Serial.print("HX711 measured settlingtime ms: ");
69   Serial.println(LoadCell.getSettlingTime());
70   Serial.println("Note that the settling time may increase significantly if you use delay()
   in your sketch!");
71   if (LoadCell.getSPS() < 7) {
72     Serial.println("!!Sampling rate is lower than specification, check MCU>HX711 wiring and
   pin designations");
73   }
74   else if (LoadCell.getSPS() > 100) {
75     Serial.println("!!Sampling rate is higher than specification, check MCU>HX711 wiring
   and pin designations");
76   }
77 }
78
79 void loop() {
80   static boolean newDataReady = 0;
81   const int serialPrintInterval = 100; //increase value to slow down serial print activity
82
83   // check for new data/start next conversion:
84   if (LoadCell.update()) newDataReady = true;
85
```

```
  // get smoothed value from the dataset:
  if (newDataReady) {
    if (millis() > t + serialPrintInterval) {
      float i = LoadCell.getData();
      // Serial.print("Load_cell output val: ");
      Serial.println(i*30.11/1000);
      newDataReady = 0;
      t = millis();
    }
  }

  // receive command from serial terminal, send 't' to initiate tare operation:
  if (Serial.available() > 0) {
    char inByte = Serial.read();
    if (inByte == 't') LoadCell.tareNoDelay();
  }

  // check if last tare operation is complete:
  if (LoadCell.getTareStatus() == true) {
    Serial.println("Tare complete");
  }

}

```