Algorithm for Fall Detection based on Rule Based v12.01 2020.01.07

This document is dedicated to describe on the Algorithm of Fall Detection by Rule Based method
Here use a tiny state machine to manipulate on dynamic location changed of object

(1) Definition:

```
        //* Assume Frame Period := 100 ms // may be 50 ms depends
        //* you may define the different values on H0, H1 and H2 by your application
        H0 := 2.5 m // H0 is location of the Module installed
        H1 := 1.5 m // H1 is the hight of 1.5 m
        H2 := 0.5 m // H2 is the hight of 0.5 m
        //* Assume dropped 0.40 m when falling time within 400 ms
        JB_FALL_THRESHOLD := 0.40 m
        yBuf[0]  := y value measured at current time by Module
        yBuf[4]  := y value measured at previous 400 ms by Module
        JB_SEC_5 := 50 // 50 * Frame Period => 50 * 100 ms => 5 Seconds
```

(2) Function:

```
        input := yBuf[]
        output := return current state
        before calling this function, shift yBuf from yBuf[i] to yBuf[i+1], i from 3 to 0
                                      always put current distance into yBuf[0]
        then calling this function, jb_state = jb_fallDetectAlgorithm(yBuf)
```

(3) Appendix:
PsudoCode by Rule Based for reference

```
jb_state = jb_fallDetectAlgorithm(yBuf)
{
    //* convert y as h height from ground
    //* assume H0 is the location of the Module be installed

    h = H0 - yBuf[0];
    switch(JBG_state){
        case 0:
            if(h > H1){
                JBG_state = 1;
            }else if( (h <= H1) && (h >= H2) ){
                JBG_state = 2;
            }else if(h <= H2){
                JBG_state = 3;
            }
            break;

        case 1:
            if( ((yBuf[0] - yBuf[4]) >= JB_FALL_THRESHOLD) ){
                JBG_state = 4; // FALL event
                break;
            }
            if( (h <= H1) && (h >= H2) ){
                JBG_state = 2;
                break;
            }
            break;

        case 2:
            if( ((yBuf[0] - yBuf[4]) >= JB_FALL_THRESHOLD) ){
                JBG_state = 4; // FALL event
                break;
            }
            if( h >= H1 ){
                JBG_state = 1;
            }else if( (h <= H2) ){
                JBG_state = 3;
            }
            break;

        case 3:
            if( (yBuf[0] - yBuf[4]) >= JB_FALL_THRESHOLD) ){
                JBG_state = 4; // FALL event
                break;
            }
            if( (h <= H1) && (h >= H2) ){
                JBG_state = 2;
            }
            break;

        case 4:
            JBG_frameNumberU32Old = JBG_frameNumberU32; //
            JBG_state = 5;
            break;

        case 5: // on {FALL event} hold 5 sec for ALERT beeping then jump back to state 0
            if( JBG_frameNumberU32 >= (JBG_frameNumberU32Old + JB_SEC_5) ){
                JBG_state = 0;
            }
            break;

    }
    return JBG_state;

}
```