Special Section - Wall Following with PID Control

The distance data measured by the sensor can be saved as a CSV file and visualized as a graph. This makes it easier to observe changes and helps determine how to adjust the PID gain values effectively. The graph above shows the distance from the wall while performing wall following at a target distance of 10 cm. By analyzing this data, the PID gain values can be adjusted to achieve more stable control.

Code

```
//File: wfPid1_0d.nxc
//Date: 02/12/2025 07:48
//Desc: Domabot US sensor(Port4) senses a wall left of it.
//      Domabot always turns left(CCW yaw) as it moves along wall.
//      PID or bang-bang(i.e. when PID gains all are zero) is used to calculate turning speed
//Refs: wfbb0_1a3.nxc; x^2File.1.0.nxc; wf2Us0_3a1.nxc

//Global variables ------------------------------------------------------------
int xWall;  //wall distance[cm]
int dWall;  //desired wall distance[cm]
float wKp, wKi, wKd;  //wall PID gains;
float wE, wEDot, wEInt;  //wall error and its derivative and interal
float wEPrev;  //wall error previous value
float wCorr;  //wall related motor power correction [0,100]
int speedA, speedC;  //motor speed for A(right) and C(left)
int speedBase;  //motor base speed
bool orngBTNPushed, rightBTNPushed, leftBTNPushed; //NXT BTNs
int iteration;

unsigned int result;  //File handling result flag
byte fileHandle;
short bytesWritten;  //Number of bytes written
string fileName, fileHeader, text;
string strIteration, strxWall;  //string versions of data

task main() {
  //Variable initializations ---------------------------------------------------
  xWall = 0;  //initialize wall distance to 0
  dWall = 10;  //Desired distance from wall [cm]
  wKp = 1.25;  //Wall P gain e.g. (PID) = [1.5, 0.005, 30.0]
  wKi = 0.001;  //Wall I gain
  wKd = 7.5;  //Wall D gain
  wE = wEDot = wEInt = 0.0;  //initialize wall error-related values to 0
  wEPrev = 0.0;  //initialize previous wall error to 0
  speedBase = 50;  //Domabot base motor speed at 50% i.e. mid-point
  iteration = 0;
```

```
//File ---------------------------------------------------------------------
fileName = "wf_[125.001.75].csv";
result = CreateFile(fileName, 1024, fileHandle);

//Overwrite existing file
while (result == LDR_FILEEXISTS) {
  CloseFile(fileHandle);
  DeleteFile(fileName);
  result = CreateFile(fileName, 1024, fileHandle);
}

//Write column headers
fileHeader =  "Iteration, Wall distance[cm]";
WriteLnString(fileHandle, fileHeader, bytesWritten);

//Algorithm begins ---------------------------------------------------------
TextOut(0, LCD_LINE2, ">>BTN to proceed");
SetSensorLowspeed(IN_4);  //Wall on Left US (Port4)

do{
  rightBTNPushed = ButtonPressed(BTNRIGHT, FALSE);
  xWall = SensorUS(IN_4);  //for wall detection (on left, Port4)
  TextOut(0, LCD_LINE3, FormatNum("Wall = %3d cm", xWall));
} while(!rightBTNPushed);

ClearLine(LCD_LINE2);
TextOut(0, LCD_LINE2, "<<BTN to Quit");

do{  //continue wall following until left button pushed
  leftBTNPushed = ButtonPressed(BTNLEFT, FALSE);
  xWall = SensorUS(IN_4);  //left US(Port4)
  TextOut(0, LCD_LINE3, FormatNum("Wall = %3d cm", xWall));

  //(1)Calculate wall-following PID gains
  wE = xWall - dWall;
  wEInt +=wE;
  wEDot = wE - wEPrev;
  wCorr = (wKp * wE) + (wKi * wEInt) + (wKd * wEDot);

  //(1A) Check for motor staturation i.e. resulting wCorr forces
  //motor getting > 2*speedBase (if speedBase > 50, this means >100)
  if(wCorr > 0 && wCorr > speedBase) {
    wCorr = speedBase;  //saturated so set correction to speedBase
                        //So Motor C speed min will be 0
  };
  if(wCorr < 0 && wCorr < -speedBase) {
```

```
      wCorr = -speedBase;  //saturated so set correction to -speedBase
                           //So Motor C speed min will be 0
    };
    //(1B) If PID gains all zero, then wCorr = 0 so do bang-bang
    if(wCorr == 0 && xWall < dWall) {
      wCorr = -speedBase;  //Move away from wall: C = basespeed, A=0
    };
    if(wCorr == 0 && xWall >= dWall) {
      wCorr = speedBase;  //Move towards wall: A = basespeed, C=0
    };

    //(2)Command motors
    speedA = speedBase + wCorr;
    speedC = speedBase - wCorr;
    OnFwd(OUT_C, speedC);
    OnFwd(OUT_A, speedA);

    //(3) update wall errors for next derivative calculation
    wEPrev = wE;

    //Convert data to string and write to file
    strIteration = FormatNum("%d", iteration);
    strxWall = FormatNum("%d", xWall);
    text = StrCat(strIteration, ",", strxWall);

    result = WriteLnString(fileHandle, text, bytesWritten);
    if(result == LDR_EOFEXPECTED) CloseFile(fileHandle);

    iteration++;

  } while(!leftBTNPushed);  //end do-while

  //(4)User pushed <<(Left)BTN, so exit gracefully
  Off(OUT_AC);
  PlaySound(SOUND_DOUBLE_BEEP);
  Wait(5000);
  StopAllTasks();

} //end main
```