### **−**Table of Contents

* [Code tweaks](http://robot.cmpe.boun.edu.tr/~cmpe565/doku.php?id=mehmetalpsarkisla:assignment4#code_tweaks)
* [ku and pu values](http://robot.cmpe.boun.edu.tr/~cmpe565/doku.php?id=mehmetalpsarkisla:assignment4#ku_and_pu_values)
* [pidP pidI and pidD values](http://robot.cmpe.boun.edu.tr/~cmpe565/doku.php?id=mehmetalpsarkisla:assignment4#pidp_pidi_and_pidd_values)
* [P control](http://robot.cmpe.boun.edu.tr/~cmpe565/doku.php?id=mehmetalpsarkisla:assignment4#p_control)
* [PI control](http://robot.cmpe.boun.edu.tr/~cmpe565/doku.php?id=mehmetalpsarkisla:assignment4#pi_control)
* [PID control](http://robot.cmpe.boun.edu.tr/~cmpe565/doku.php?id=mehmetalpsarkisla:assignment4#pid_control)
* [lastError](http://robot.cmpe.boun.edu.tr/~cmpe565/doku.php?id=mehmetalpsarkisla:assignment4#lasterror)
* [totalError](http://robot.cmpe.boun.edu.tr/~cmpe565/doku.php?id=mehmetalpsarkisla:assignment4#totalerror)

## Code tweaks

I used dynamically changing offset using the robotYaw angle and divide it to input\_scan.angle\_increment. This gave me an approximate number of points to correct offset.

int offset = 100 + (PI/2-robotYaw)/input\_scan.angle\_increment;

I added a Special case for error = 0 since error = 0 only if there is no wall around robot range. So it must really be infinite not 0.

// check for min and max values

if (lDist > input\_scan.range\_max ) {

lDist = input\_scan.range\_max;

} else if(lDist == 0) {lDist = 1;

}

else if (lDist <input\_scan.range\_min) {

lDist = input\_scan.range\_min;

}

if (rDist > input\_scan.range\_max ) {

rDist = input\_scan.range\_max;

} else if(rDist == 0) {rDist = 1;}

else if (rDist < input\_scan.range\_min) {

rDist = input\_scan.range\_min;

} std::cout<< "l " << rDist << " r " << lDist << std::endl;

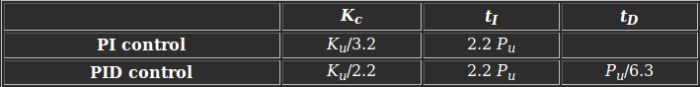
Formun Üstü

Formun Altı

## ku and pu values

float **ku** = 2.5;   
float **pu** = 2.458;

After I take the PID values with Ziegler-Nichols Method I used Tyreus-Luyben Tuning Chart:



Formun Üstü

Formun Altı

## pidP pidI and pidD values

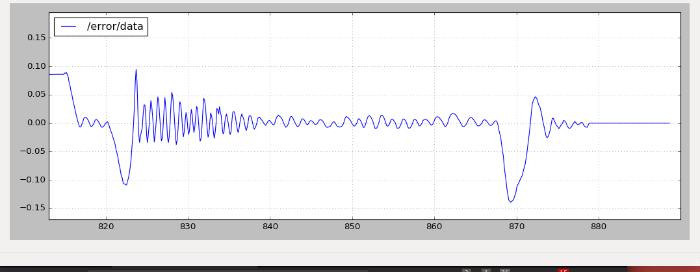
pidD = pidP \* td ;   
pidI = pidP / ti

|  | **pidP** | **pidI** | **pidD** |
| --- | --- | --- | --- |
| **P** | ku/2 | 0 | 0 |
| **PI** | ku/3.2 | (ku/3.2) / ti | 0 |
| **PID** | ku/2.2 | (ku/2.2) /ti | (ku/2.2) \* td |

Formun Üstü

Formun Altı

## P control

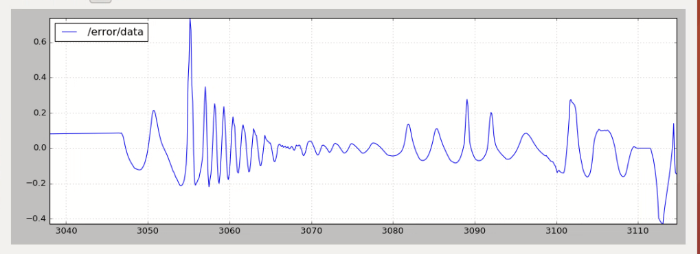


<https://youtu.be/iJlU61AnOZo>

Formun Üstü

Formun Altı

## PI control

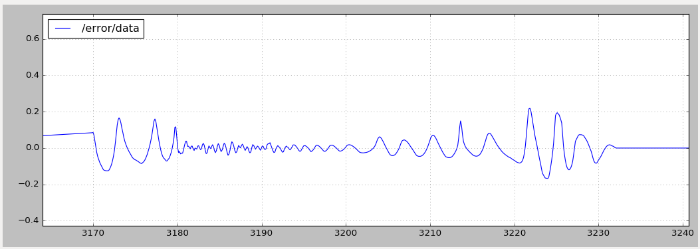


<https://youtu.be/RZm7Q4NoPSU>

Formun Üstü

Formun Altı

## PID control

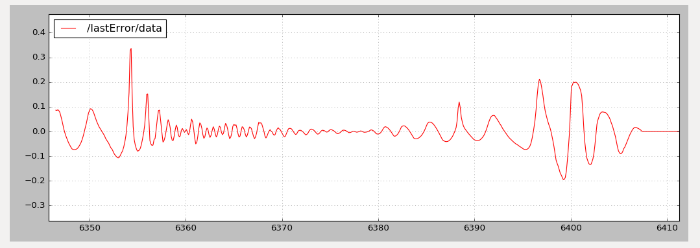


<https://youtu.be/HN3Fpp0_YTA>

Formun Üstü

Formun Altı

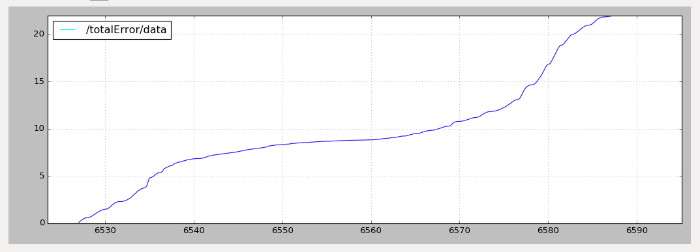
## lastError



Formun Üstü

Formun Altı

## totalError



Formun Üstü

Formun Altı