PID CONTROLLER FOR A DIFFERENTIAL STEERING MOBILE PLATFORM

Mihai Crenganis[2], Octavian Bologa[1,2]

[1]Full Member of the Academy of Technical Sciences of Romania
[2] "Lucian Blaga" University of Sibiu.

ABSTRACT: This paper presents the steps needed to implement and tune the PID controller for a mobile platform with differential steering. It presents the steps taken in the development of a mobile platform built in the Department of Industrial Machinery & Equipment, of the Engineering Faculty of Sibiu. In a first phase, a brief introduction to PID controller is made, followed by a description of the various robot's components and traction principles underlying locomotion of mobile platforms. At the end the paper contains several principles of tuning the PID controller.

Keywords: position control, mobile robot, PID tuning.

IMPLEMENTAREA REGULATORULUI PID PE O PLATFORMA MOBILA CU TRACTIUNE DIFERENTIALA: Lucrarea de fata prezinta pasii parcursi in dezvoltarea unei platforme mobile, construita in cadrul Departamentului de Masini si Echipamente Industriale al Facultatii de Inginerie din Sibiu. Intr-o prima faza este prezentata o scurta descriere a regulatorului PID, urmata de descrierea principalelor parti componente ce stau la baza robotului dezvoltat in cadrul departamentului MEI. Finalul lucrarii contine cateva principii legate de acordarea regulatorului PID precum si rezultate obtinute ale preciziei de pozitionare.

Cuvinte cheie: controlul pozitiei, robot mobil, acordare regulator PID.

1. INTRODUCTION, PID CONTROLLER

A proportional-integral-derivative controller (PID controller) is a control loop feedback mechanism (controller) commonly used in industrial control systems. A PID controller continuously calculates an "error value" as the difference between a measured process variable and a desired setpoint. The controller attempts to minimize the error over time by adjustment of a control variable, such as the position of a control valve, a damper, or the power supplied to a heating element, to a new value determined by a weighted sum:

$$u(t) = K_p e(t) + K_i \int_0^t e(\tau)d\tau + K_d \frac{de}{dt} \qquad (1)$$

where K_p, K_i, K_d and, all non-negative, denote the coefficients for the proportional, integral, and derivative terms, respectively (sometimes denoted P, I, and D). In this model, P accounts for present values of the error (e.g. if the error is large and positive, the control output will also be large and positive), I accounts for past values of the error (e.g. if the output is not sufficient to reduce the size of the error, error will accumulate over time, causing the controller to apply stronger output), and D accounts for predicted future values of the error, based on its current rate of change.

As a PID controller relies only on the measured process variable, not on knowledge of the underlying process, it is a broadly useful controller. By tuning the three parameters of the model, one can design a PID controller for specific process requirements. The response of the controller can be described in terms of the responsiveness of the controller to an error, the degree to which the controller overshoots the setpoint, and the degree of system oscillation. Note that the use of the PID algorithm for control does not guarantee optimal control of the system or system stability.

Some applications may require using only one or two terms to provide the appropriate system control. This is achieved by setting the other parameters to zero. A PID controller will be called a PI, PD, P or I controller in the absence of the respective control actions. PI controllers are fairly common, since

derivative action is sensitive to measurement noise, whereas the absence of an integral term may prevent the system from reaching its target value due to the control action.

For the discrete time case, the term PSD, for proportional-summation-difference, is often used

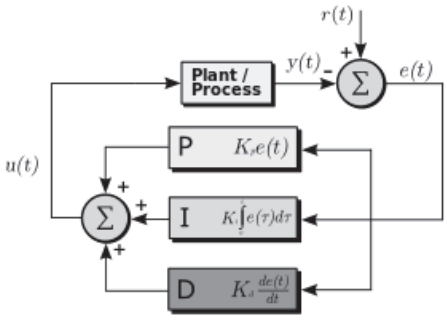

Fig.1. A block diagram of a PID controller in a feedback loop

2. MOBILE ROBOT HARDWARE

For the start of the project we started from some characteristics which the mobile robot must meet, such as the maximum dimensions and clearance: The maximum height is of 350 [mm] and maximum area of 1200 [mm] . The two major components underlying the autonomous mobile robot are: the hardware component and the software one. This paper describes only the issues underlying the hardware assembly of the robot. In figure 2 the mobile robot is presented:

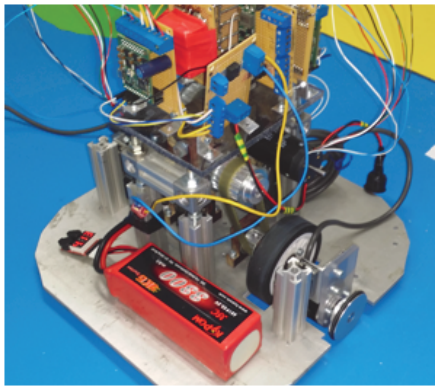

Fig.2. The mobile platform

The following hardware components of mobile robot are described and also the characteristics of these components are presented :

Regarding the mechanical part it can be said that the robot is built on several levels on a support frame or chassis made of aluminum. The total mass of the assembly is 10kg and maximum speed attainable of the robot is 2 m / s. The traction is based on two wheels driven by two MAXON electric motors fed from a DC voltage of 14.4 V. Transmission from engine to the drive wheels is via two toothed belts and two cylindrical spur gear wheels with a gear ratio of 100: 1, figure. 3.

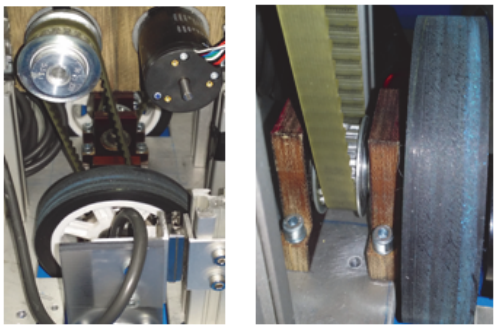

Fig.3. Belt transmision

Wheels are differential and independent powered. For the mobile robot to perform a translational motion the wheels of traction must be operated in the same direction. In order for the robot to perform a rotational movement driving wheels must be driven in different directions of rotation. Regarding the power supply should be noted that the entire voltage is achieved through a 14.4 V battery with a capacity of 3.3 Ah. The power / traction remains fed to 12 V, while the command and control is fed to a voltage of 5 V, reduced by a voltage stabilizer 12V-5V. Controlling the robot is made from a development board Arduino Mega 2560. For the detection robot opponent / obstacle were used Dives high precision sensors such as laser sensors Sick two and two ultrasonic sensors from Pololu. In order to control motors used MAXON drivers, figure 4.

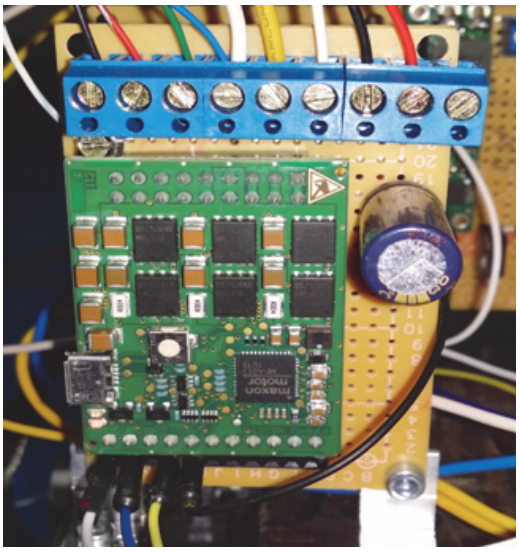

Fig.4. MAXON driver

The programming part was done in the specific programming language for the development board called Arduino IDE. The programming language is very similar to the C language. In figure 5 the control board is presented.

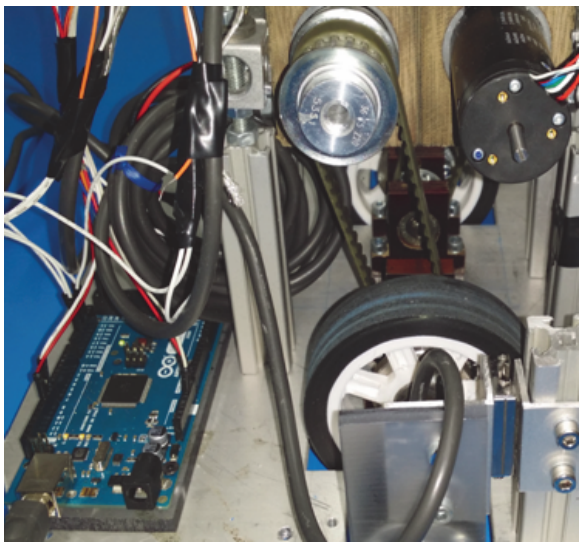

Fig.5. Arduino Mega 2560 development board

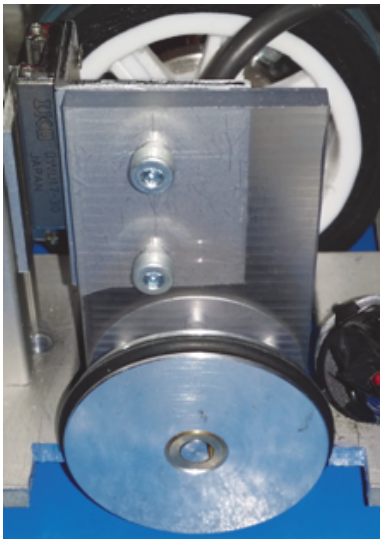

Fig.6. Digital Yumo quadrature encoder

3. PID CONTROLLER

The PID control scheme is named after its three correcting terms, whose sum constitutes the manipulated variable (MV). The proportional, integral, and derivative terms are summed to calculate the output of the PID controller. Defining u(t) as the controller output, the final form of the PID algorithm is:

$$u(t) = MV(t) = K_p e(t) + K_i \int_0^t e(\tau)\,d\tau + K_d \frac{d}{dt} e(t) \qquad (2)$$

where:

Kp: Proportional gain, a tuning parameter
Ki: Integral gain, a tuning parameter
Kd: Derivative gain, a tuning parameter
e: Error = SP-PV
t: Time or instantaneous time (the present)
τ: Variable of integration; takes on values from time 0 to the present t.

Equivalently, the transfer function in the Laplace Domain of the PID controller is:

$$L(s) = Kp + Ki/s + Kds \qquad (3)$$

where

s: complex number frequency

Proportional term:

The proportional term produces an output value that is proportional to the current error value. The proportional response can be adjusted by multiplying the error by a constant Kp, called the proportional gain constant.

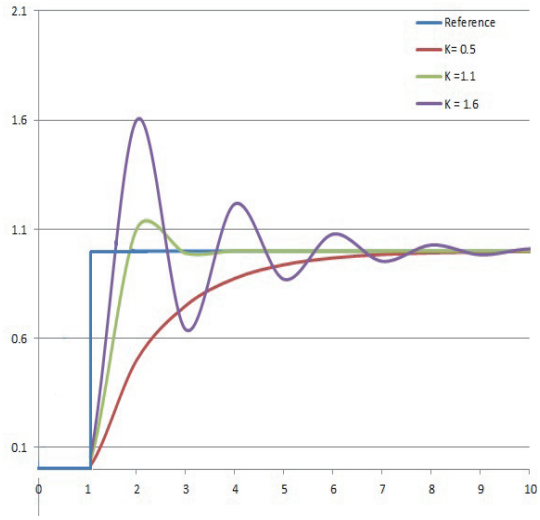

Fig.7. Plot of PV vs time, for three values of Kp (Ki and Kd held constant)

The proportional term is given by:

$$P_{out} = K_p e(t) \qquad (4)$$

A high proportional gain results in a large change in the output for a given change in the error. If the proportional gain is too high, the system can become unstable (see the section on loop tuning). In contrast, a small gain results in a small output response to a large input error, and a less responsive or less

sensitive controller. If the proportional gain is too low, the control action may be too small when responding to system disturbances. Tuning theory and industrial practice indicate that the proportional term should contribute the bulk of the output change.

Integral term:

The contribution from the integral term is proportional to both the magnitude of the error and the duration of the error. The integral in a PID controller is the sum of the instantaneous error over time and gives the accumulated offset that should have been corrected previously. The accumulated error is then multiplied by the integral gain (*Ki*) and added to the controller output.

The integral term accelerates the movement of the process towards setpoint and eliminates the residual steady-state error that occurs with a pure proportional controller. However, since the integral term responds to accumulated errors from the past, it can cause the present value to overshoot the setpoint value (see the section on loop tuning).

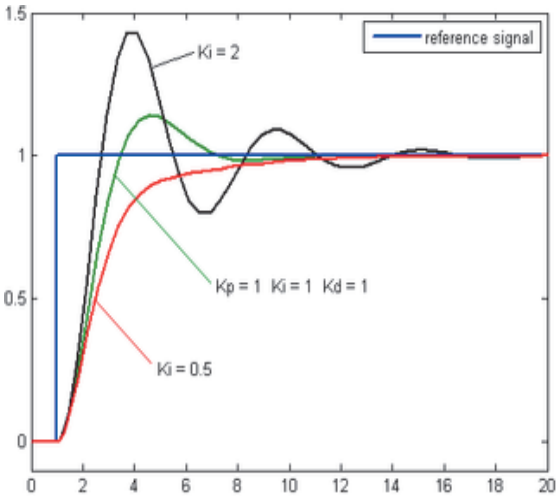

Fig.8. Plot of PV vs time, for three values of Ki (Kp and Kd held constant)

The integral term is given by:

$$I_{out} = K_i \int_0^t e(\tau)\, d\tau \tag{5}$$

Derivative term:

The derivative of the process error is calculated by determining the slope of the error over time and multiplying this rate of change by the derivative gain Kd. The magnitude of the contribution of the derivative term to the overall control action is termed the derivative gain, Kd.

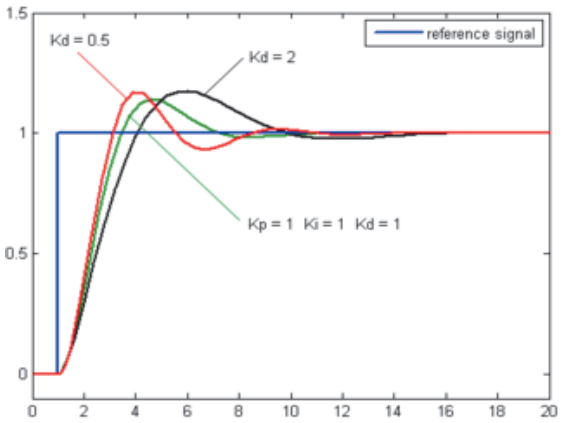

Fig.9. Plot of PV vs time, for three values of Kd (Kp and Ki held constant)

The derivative term is given by:

$$D_{out} = K_d \frac{d}{dt} e(t) \tag{6}$$

Derivative action predicts system behavior and thus improves settling time and stability of the system. An ideal derivative is not causal, so that implementations of PID controllers include an additional low pass filtering for the derivative term, to limit the high frequency gain and noise. Derivative action is seldom used in practice though - by one estimate in only 25% of deployed controllers - because of its variable impact on system stability in real-world applications.

4. LOOP TUNING

Tuning a control loop is the adjustment of its control parameters (proportional band/gain, integral gain/reset, derivative gain/rate) to the optimum values for the desired control response. Stability (no unbounded oscillation) is a basic requirement, but beyond that, different systems have different behavior, different applications have different requirements, and requirements may conflict with one another.

PID tuning is a difficult problem, even though there are only three parameters and in principle is simple to describe, because it must satisfy complex criteria within the limitations of PID control. There are accordingly various methods for loop tuning, and more sophisticated techniques are the subject of patents; this section describes some traditional manual methods for loop tuning.

Designing and tuning a PID controller appears to be conceptually intuitive, but can be hard in practice, if multiple (and often conflicting) objectives such as short transient and high stability are to be achieved. PID controllers often provide acceptable control using default tunings, but performance can generally

be improved by careful tuning, and performance may be unacceptable with poor tuning. Usually, initial designs need to be adjusted repeatedly through computer simulations until the closed-loop system performs or compromises as desired.

Some processes have a degree of nonlinearity and so parameters that work well at full-load conditions don't work when the process is starting up from no-load; this can be corrected by gain scheduling (using different parameters in different operating regions).

4.1 Manual tunning

If the system must remain online, one tuning method is to first set *Ki* and *Kd* values to zero. Increase the *Kp* until the output of the loop oscillates, then the *Kp* should be set to approximately half of that value for a "quarter amplitude decay" type response. Then increase *Ki* until any offset is corrected in sufficient time for the process. However, too much *Ki* will cause instability. Finally, increase, *Kd* if required, until the loop is acceptably quick to reach its reference after a load disturbance. However, too much *Kd* will cause excessive response and overshoot. A fast PID loop tuning usually overshoots slightly to reach the setpoint more quickly; however, some systems cannot accept overshoot, in which case an over-damped closed-loop system is required, which will require a *Kp* setting significantly less than half that of the *Kp* setting that was causing oscillation.

Parameter	Rise time	Overshoot	Settling time
K_p	Decrease	Increase	Small change
K_i	Decrease	Increase	Increase
K_d	Minor change	Decrease	Decrease

Parameter	Steady-state error	Stability
K_p	Decrease	Degrade
K_i	Eliminate	Degrade
K_d	No effect in theory	Improve if K_d small

Fig.10. Effects of *increasing* a parameter independently

4.2 Ziegler–Nichols method

Another heuristic tuning method is formally known as the Ziegler–Nichols method, introduced by John G. Ziegler and Nathaniel B. Nichols in the 1940s. As in the method above, the *Ki* and *Kd* gains are first set to zero. The proportional gain is increased until it reaches the ultimate gain, *Ku*, at which the output of the loop starts to oscillate. *Ku* and the oscillation period *Tu* are used to set the gains as shown:

Control Type	K_p	K_i	K_d
P	$0.50K_u$	-	-
PI	$0.45K_u$	$1.2K_p/T_u$	-
PID	$0.60K_u$	$2K_p/T_u$	$K_pT_u/8$

Fig.10. Ziegler–Nichols method

5. IMPLEMENTING THE PID CONTROLLER

Here is a simple software loop that implements the PID algorithm:

```
previous_error = 0
integral = 0
start:
  error = setpoint - measured_value
  integral = integral + error*dt
  derivative = (error - previous_error)/dt
  output = Kp*error + Ki*integral +
Kd*derivative
  previous_error = error
  wait(dt)
 goto start
```

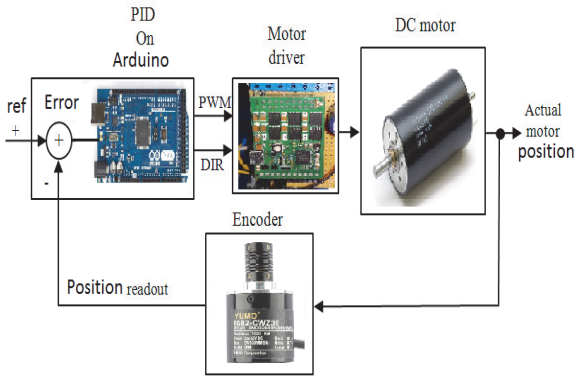

Fig.11. Bloc diagram for PID implementation

Cascade control

One distinctive advantage of PID controllers is that two PID controllers can be used together to yield better dynamic performance. This is called cascaded PID control. In cascade control there are two PIDs arranged with one PID controlling the setpoint of another. A PID controller acts as outer loop controller, which controls the primary physical parameter, such as fluid level or velocity. The other controller acts as inner loop controller, which reads the output of outer loop controller as setpoint, usually controlling a more rapid changing parameter, flowrate or acceleration. It can be mathematically proven that the working frequency of the controller is increased and the time constant of the object is reduced by using cascaded PID controllers.

REFERENCES

Ang, K.H., Chong, G.C.Y., and Li, Y. (2005). PID control system analysis, design, and technology, IEEE Trans Control Systems Tech, 13(4), pp.559-576. http://eprints.gla.ac.uk/3817/1/IEEE3.pdf

Araki, M. "PID Control" (PDF).

Arkin, R. C. "*Behavior-Based Robotics",* MIT Press 1998;

Atherton, Drek P (December 2014). "Almost Six Decades in Control Engineering". Control Systems, IEEE. doi:10.1109/MCS.2014.2359588.

Bechhoefer, John. "Feedback for Physicists: A Tutorial Essay On Control". Reviews of Modern Physics (APS Physics) 77 (3): 783–835.

Bennett, Stuart (1993). A history of control engineering, 1930-1955. IET. p. p. 48. ISBN 978-0-86341-299-8.

Bennett, Stuart (1996). "A brief history of automatic control" (PDF). IEEE Control Systems Magazine (IEEE) 16 (3): 17–25.

Bennett, Stuart (June 1986). A history of control engineering, 1800-1930. IET. pp. 142–148. ISBN 978-0-86341-047-5.

Bennett, Stuart (November 1984). "Nicholas Minorsky and the automatic steering of ships" (PDF). IEEE Control Systems Magazine 4 (4): 10–15. doi:10.1109/MCS.1984.1104827. ISSN 0272-1708.

Brian P. Gerkey. Richard T. Vaughan. Andrew Howard. The „*PlayerfStage Project: Toofs for Multi-Robot and Distributed Sensor System"s*. In Proc. of the Intl. Conf. on Advanced Robotics (ICAR). pages 317-323. Coimbra. Portugal. July 2003;

Cooper, Douglas. "Integral (Reset) Windup, Jacketing Logic and the Velocity PI Form". Retrieved 2014-02-18.

Cooper, Douglas. "PI Control of the Heat Exchanger". Practical Process Control by Control Guru. Retrieved 2014-02-27.

Gregory Dudek. Michael Jenkin. „*Computaţional Principles of Mobile Robotics",* Cambridge University Press. 2000;

Jinghua Zhong (Spring 2006). "PID Controller Tuning: A Short Tutorial" (PDF). Retrieved 2011-04-04.

Kebriaei, Reza; Frischkorn, Jan; Reese, Stefanie; Husmann, Tobias; Meier, Horst; Moll, Heiko; Theisen, Werner. "Numerical modelling of powder metallurgical coatings on ring-shaped parts integrated with ring rolling". Material Processing Technology 213 (1): 2015–2032. |accessdate= requires |url= (help)

Kikuo Fujimura. „*Motion Planning in Dynamic Environments",* Springer-Verlag Tokyo. 1991;

King, Myke. Process Control: A Practical Approach. Wiley, 2010, p. 52-78

Li, Y. and Ang, K.H. and Chong, G.C.Y. (2006) PID control system analysis and design - Problems, remedies, and future directions. IEEE Control Systems Magazine, 26 (1). pp. 32-41. ISSN 0272-1708

Li, Y., et al. (2004) CAutoCSD - Evolutionary search and optimisation enabled computer automated control system design, Int J Automation and Computing, vol. 1, No. 1, pp. 76-88. ISSN 1751-8520. http://userweb.eng.gla.ac.uk/yun.li/ga_demo/

Liang, Y.-L.() et al. (2009). "Controlling fuel annealer using computational verb PID controllers". Proceedings of the 3rd international conference on Anti-Counterfeiting, security, and identification in communication (IEEE): 417–420.

Neuhaus, Rudolph. "Diode Laser Locking and Linewidth Narrowing". www.toptica.com. Missing or empty |url= (help); |accessdate= requires |url= (help)

Tim Wescott (October 2000). "PID without a PhD" (PDF). EE Times-India.

Vaughan, R. T., B. Gerkey. P.,. Howard, A. „*On device abstractions for portable, reusable robot code",* In Proc. of the IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS). Pag. 21-2427, Las Vegas, October 2003;

Veselý, V., Rosinová, D.: Robust PSD Controller Design, Editors: Fikar, M., Kvasnica, M., In Proceedings of the 18th International Conference on Process Control, Tatranská Lomnica, Slovakia, 565–570, 2011

Y Li, KH Ang, GCY Chong, Patents, software, and hardware for PID control: An overview and analysis of the current art, Control Systems, IEEE, 26 (1), 42-54. http://eprints.gla.ac.uk/3816/1/IEEE2pdf.pdf

Yang, T. (June 2005). "Architectures of Computational Verb Controllers: Towards a New Paradigm of Intelligent Control". International Journal of Computational Cognition (Yang's Scientific Press) 3 (2): 74–101.

"A Brief Building Automation History". Retrieved 2011-04-04. (Bennett 1993, p. 67)

"Discrete PI and PID Controller Design and Analysis for Digital Implementation". Scribd.com. Retrieved 2011-04-04.

"Introduction: PID Controller Design". University of Michigan.

"PID process control, a "Cruise Control" example". CodeProject. 2009. Retrieved 4 November 2012.

*** http://en.wikipedia.org/wiki/Odometry

***http://en.wikibooks.org/wiki/Robotics/Types_of_Robots/Wheeled;

***https://en.wikipedia.org/wiki/PID_controller