CENG 4480

Lecture 06: PID Control

Bei Yu

香港中文大學
The Chinese University of Hong Kong

Objectives

1) *Study DC motors*
2) *Study open-loop and closed-loop control*
3) *Control methods*
   - *I) Proportional feedback control*
   - *II) PID (proportional-integral-derivative) control*

1) DC motors

For robots

Motors

- DC motors: Direct current motor, easy to control and use. For making wheeled robots
- Servo motors for making robot legs http://www.lynxmotion.com/

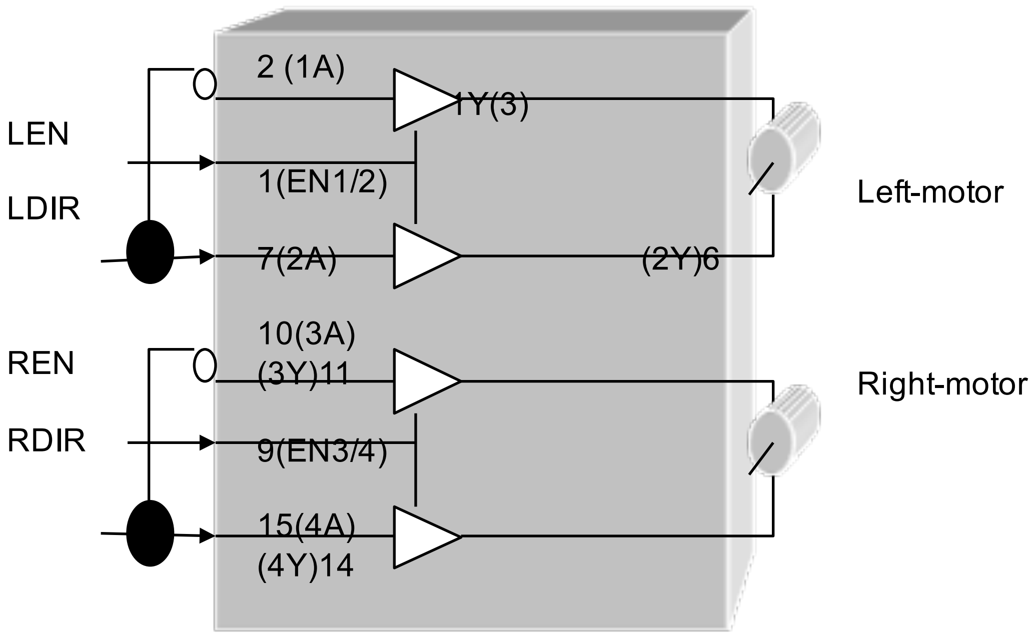
Small Direct Current D.C. motors

- Speed (~ 1200-2000 rpm).
- Operates on a 3~5Volt, Can use gear box (e.g. ratio 58:1) to increase torque
- Use H-bridge circuit to boost up current from the TLL level to motor driving level.

田宫四驱车配件15351 PRO版专用双头马达，
picture from
http://item.taobao.com/item.htm?id=1606576457&tracelog=newcardfavirate

Motor control chip

- H-bridge Chips



- L293D: H-bridge circuit, up 2A
- LDIR: left motor direction; RDIR: right motor direction
- LEN : left motor enable; REN : right motor enable

2) open-loop and closed-loop control

Feedback control

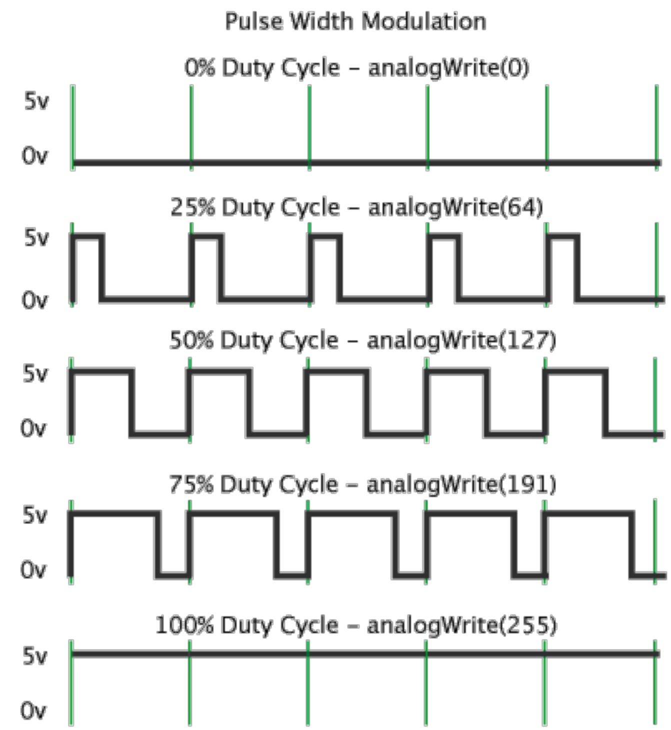
PID theory and implementation

Open-loop motor control and its problems

- Change motor supply power change speed
- Problem: How much power is right?
  - Ans: don’t know , depends on internal/external frictions of individual motors.
- Problem: How to make the robot move straight?
- How to control power (Ton) by ISR & an MCU?
  - Solution: Use feedback control to read actual wheel:
  - Slower, increase power (+ Ton)
  - Faster, reduce power (- Ton)

PWM Signal

- Pulse Width Modulation
- Analog results with digital means
- a square signal switched between on and off
- changing the portion the signal on



Exercise

- *When using the open-loop control method with a constant PWM signal for both wheels, explain why the robot would slow down when climbing up hill.*

Feedback control

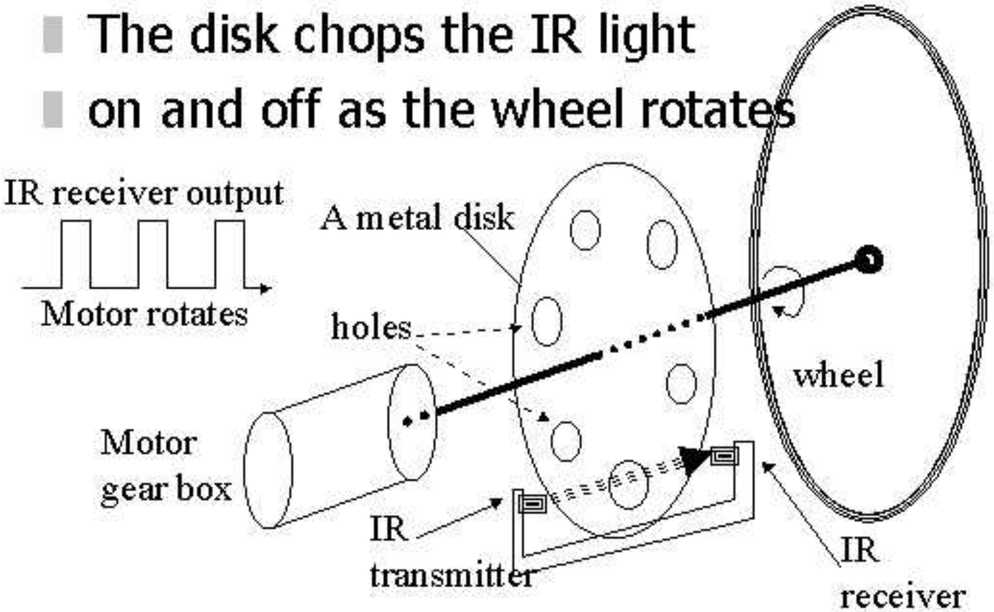
- The real solution to real speed control is feedback control
- Require speed encoder to read back the real speed of the wheel at real time.

First you need to have speed encoders

- Read wheel speed.
- Use photo interrupter
- Use reflective disk to save space
- Based on interrupts

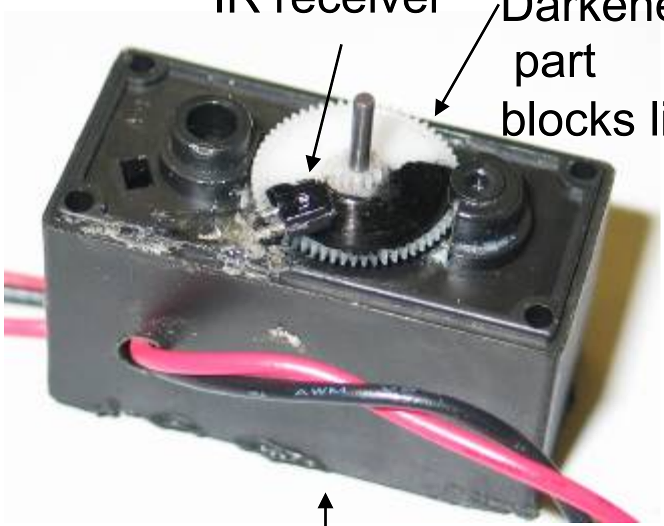
Wheel encoder

- The disk chops the IR light
- on and off as the wheel rotates

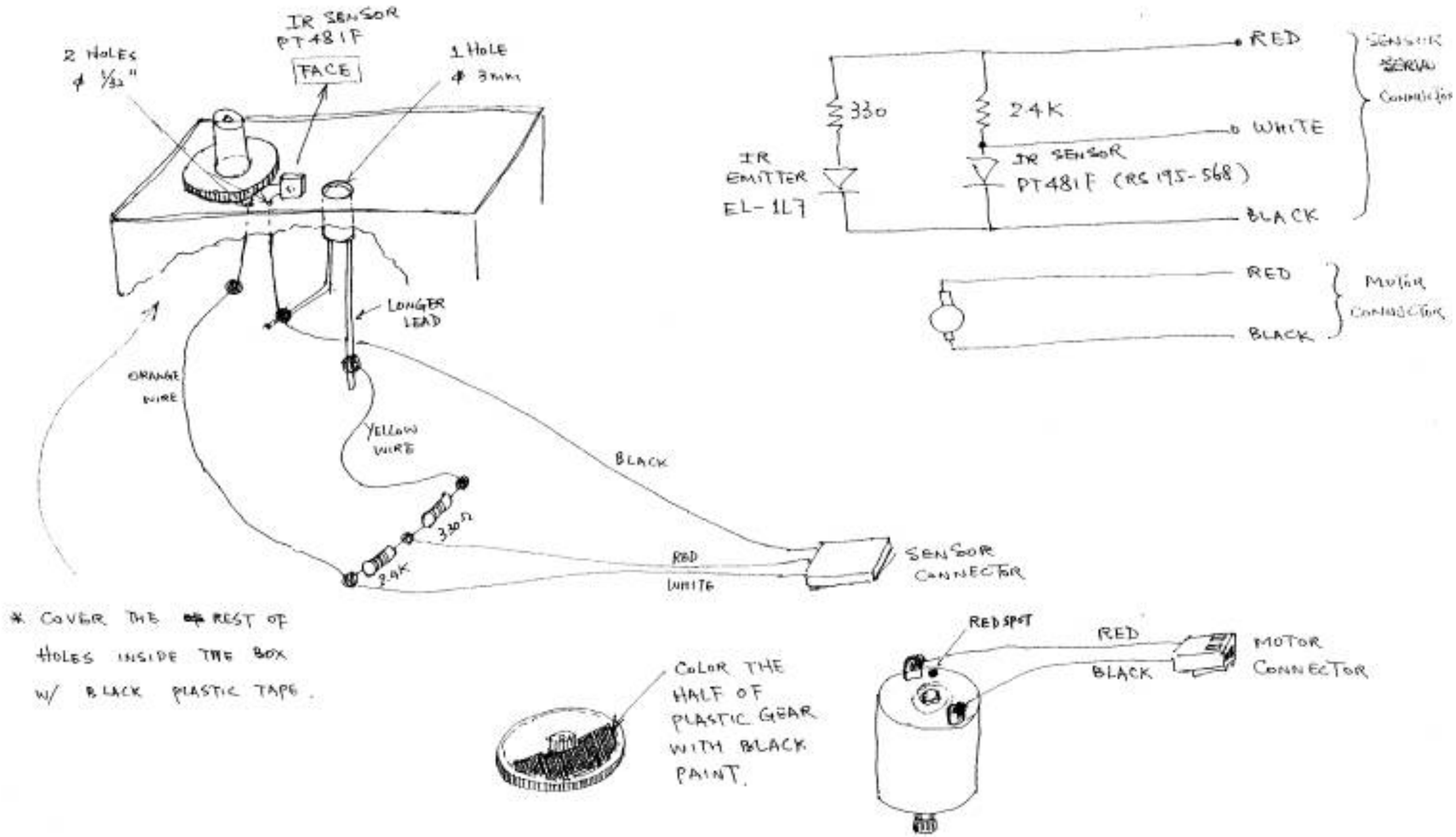


Our motor and speed encoder

Each wheel rotation= 88 on/off changes



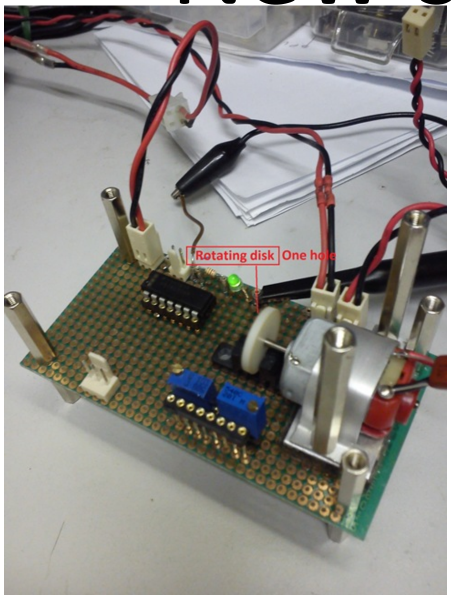
SERVO MOTOR MODIFICATION

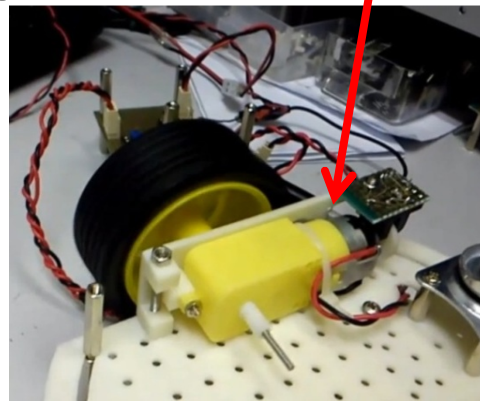


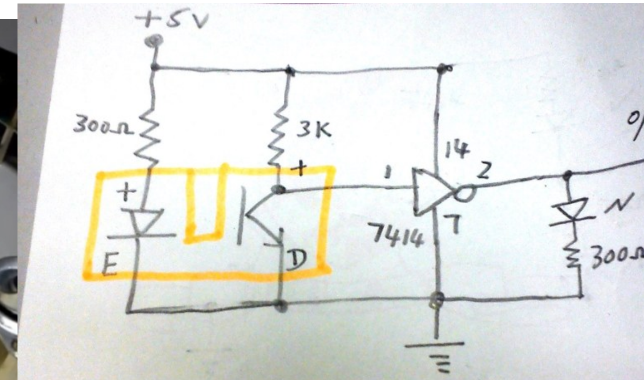
New speed el

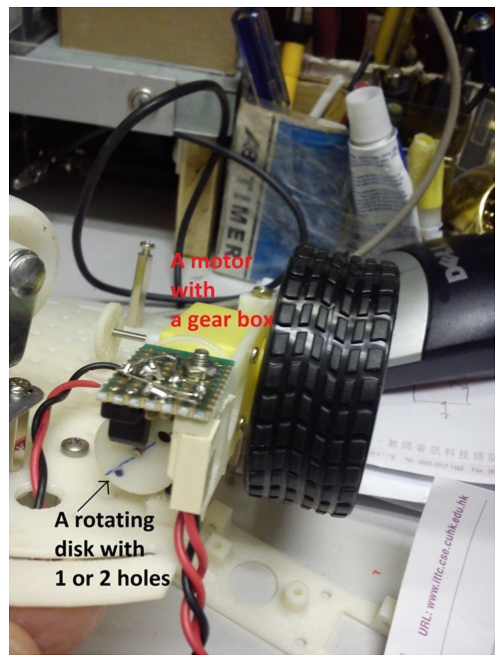
田宫四驱车配件15351 PRO版专用双头马达，
picture from http://item.taobao.com/item.htm?id=1606576457&tracelog=newcardfavirate

•









[Demo movie](#)

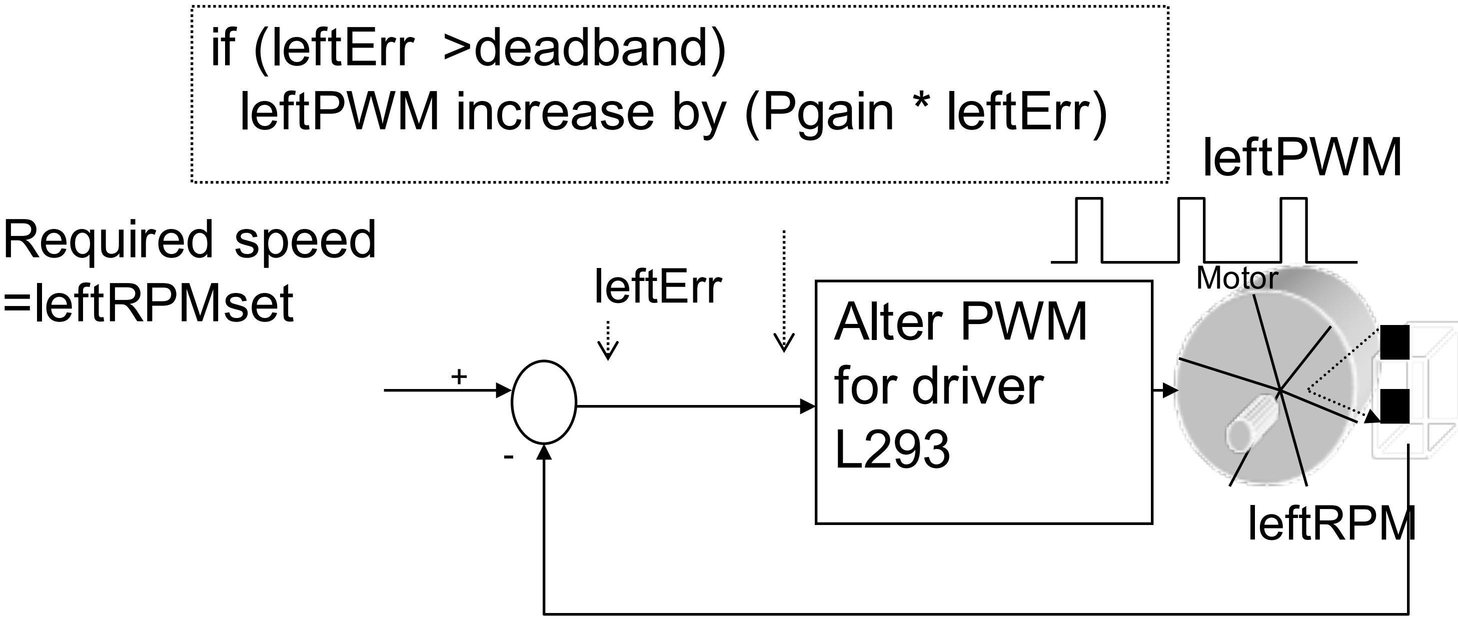
3) Control methods

I) Proportional feedback control

II) PID (proportional-integral-derivative) control

I) Proportional closed-loop feed back control system

- Show the left motor control only

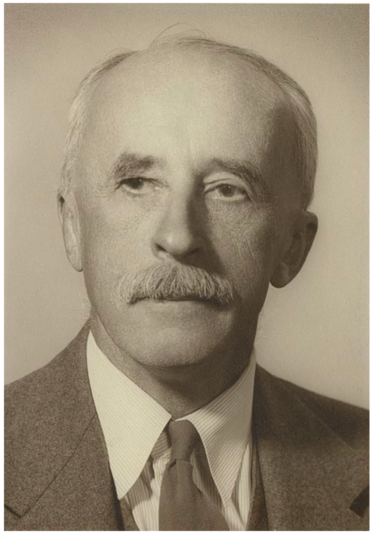


II) PID (proportional-integral-derivative) control

A more formal and precise method

Used in most modern machines

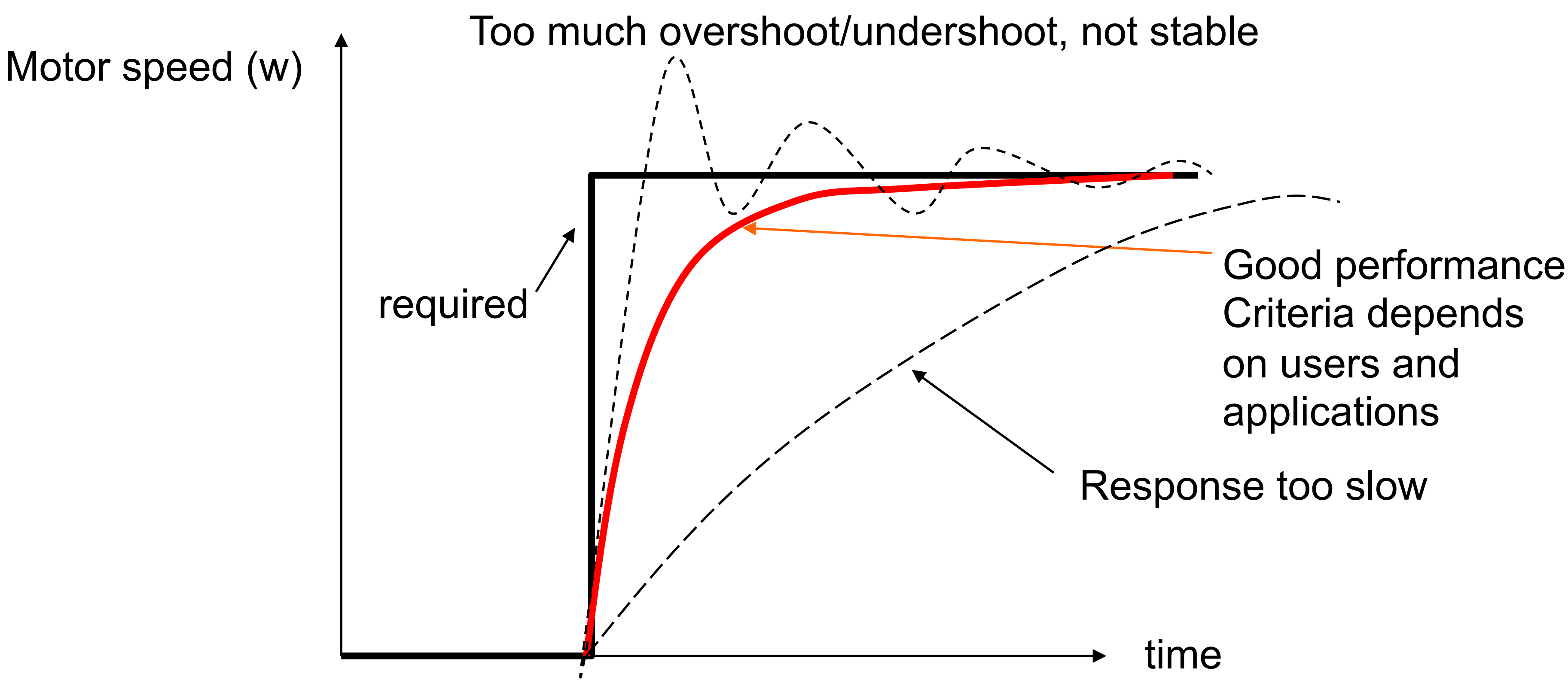
History of PID



- By <u>Nicolas Minorsky in 1922</u>
- observations of a helmsman
- steered the ship based on
  - the current course error
  - past error
  - the current rate of change

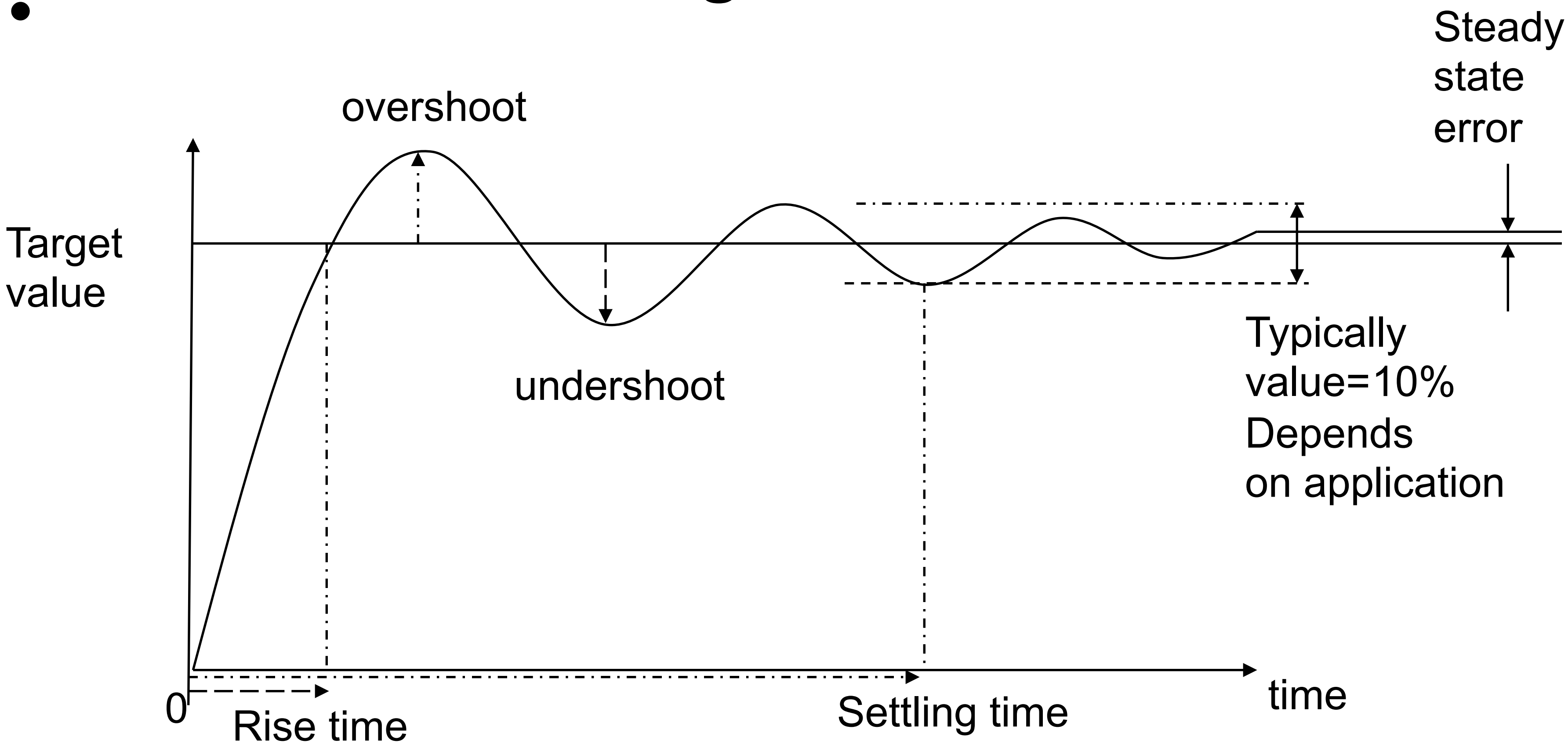
Introduction

- Control for better performance
- Use PID, choose whatever response you want



Values to evaluate a control system Exercise 2: Describe the terms n the following diagrams

-



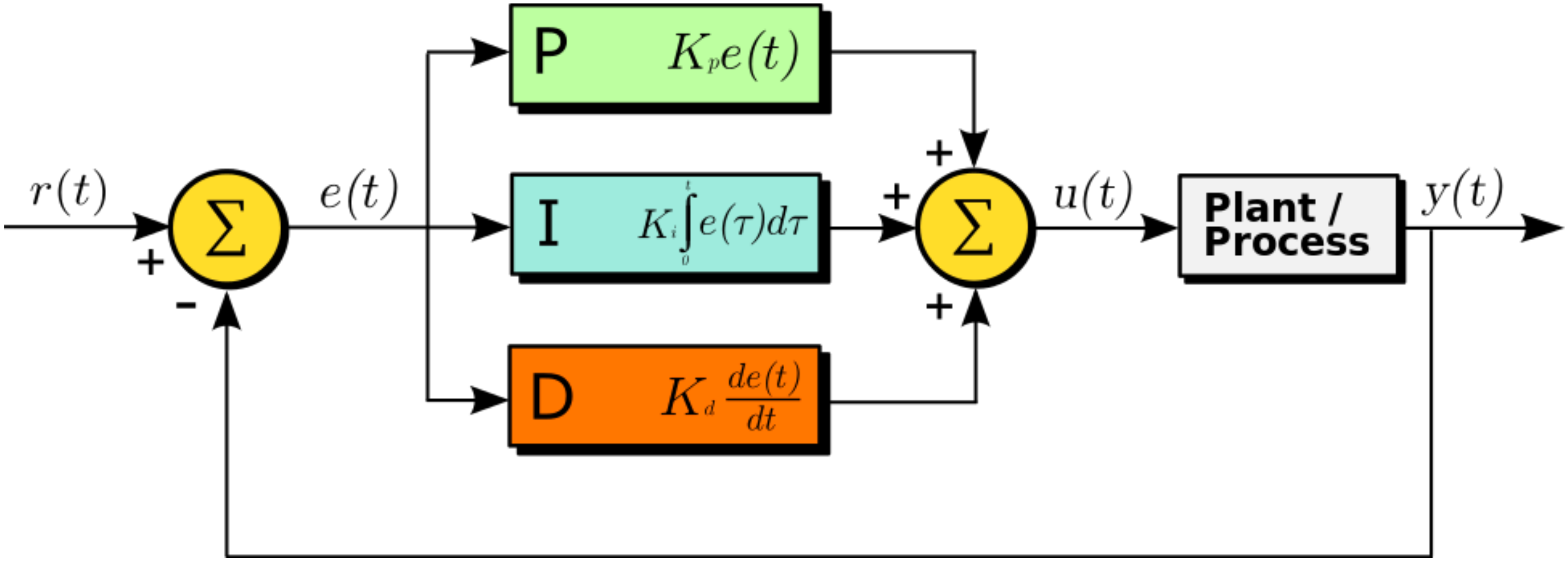
PID Control

$$u(t) = K_p e(t) + K_i \int_0^t e(t)dt + K_d \frac{de(t)}{dt},$$

where

- $e(t)$: error value
- $u(t)$: control variable
- K_p: coefficient for the proportional (P)
- K_i: coefficient for the integral (I)
- K_d: coefficient for the derivative (D)

PID Control


P
$K_p e(t)$
I
$K_i \int_0^t e(\tau) d\tau$
D
$K_d \frac{de(t)}{dt}$
$r(t)$
$e(t)$
$u(t)$
$y(t)$
Plant / Process
+
-
+
+
+

Use of PID
control terms are intertwined

http://en.wikipedia.org/wiki/PID_controller

- *Kp (Pgain):* ***Proportional Gain*** *- Larger Kp typically means faster response since the larger the error, the larger the Proportional term compensation. An excessively large proportional gain will lead to process instability and oscillation.*
- *Ki (Igain):* ***Integral Gain*** *- Larger Ki implies steady state errors are eliminated quicker. The trade-off is larger overshoot: any negative error integrated during transient response must be integrated away by positive error before we reach steady state.*
- *Kd (Dgain):* ***Derivative Gain*** *- Larger Kd decreases overshoot, but slows down transient response and may lead to instability due to signal noise amplification in the differentiation of the error.*

Effects of increasing parameters

http://en.wikipedia.org/wiki/PID_controller

•

Parameter	Rise Time	Overshoot	Settling Time	Steady state error
Kp (Pgain)	Decrease step1	Increase	Small Change	Decrease
Ki (Igain)	Decrease	Increase	Increase	Eliminate step3
Kd (Dgain)	Small Change	Decrease step2	Decrease	Small Change

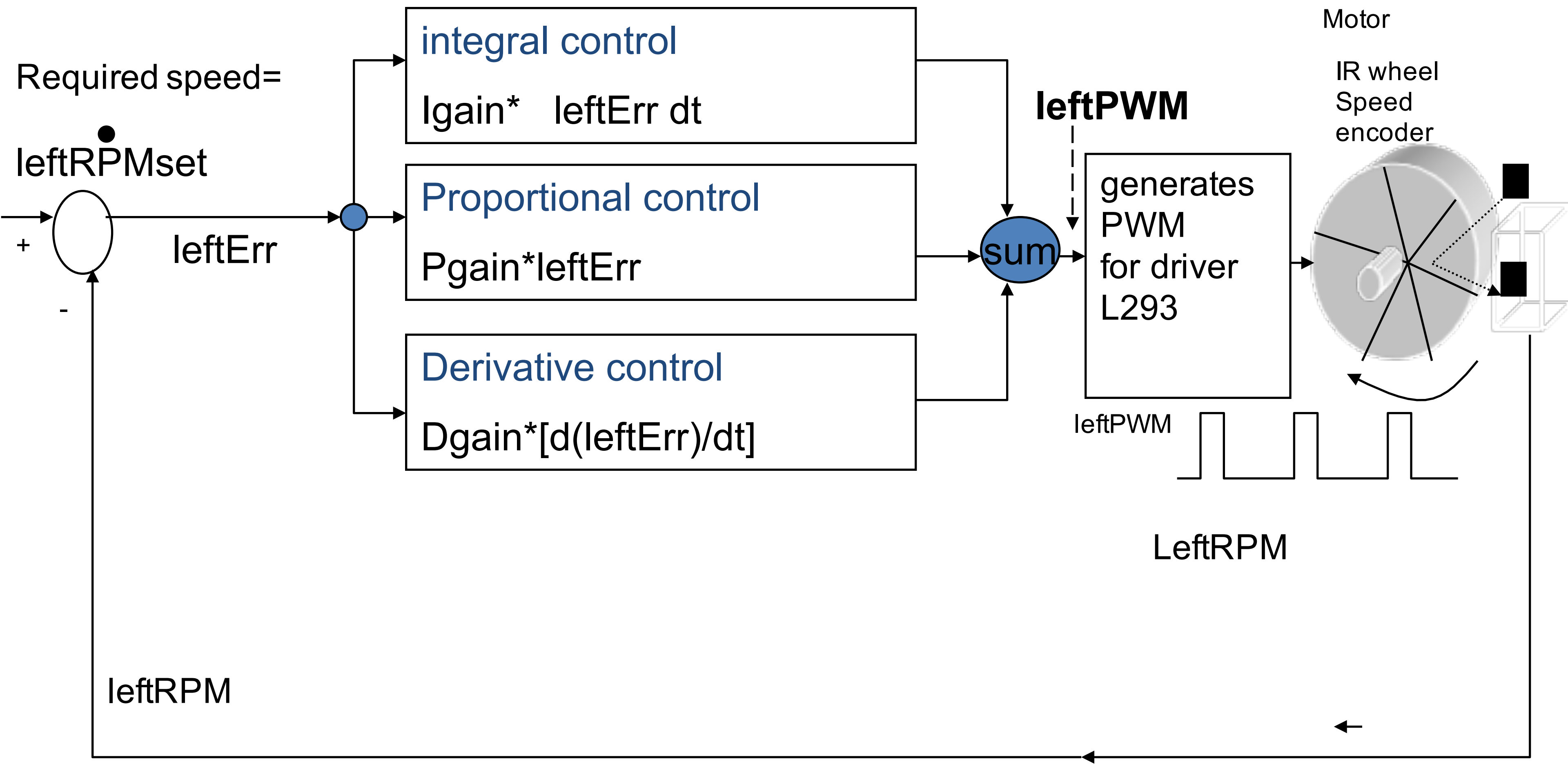
Software

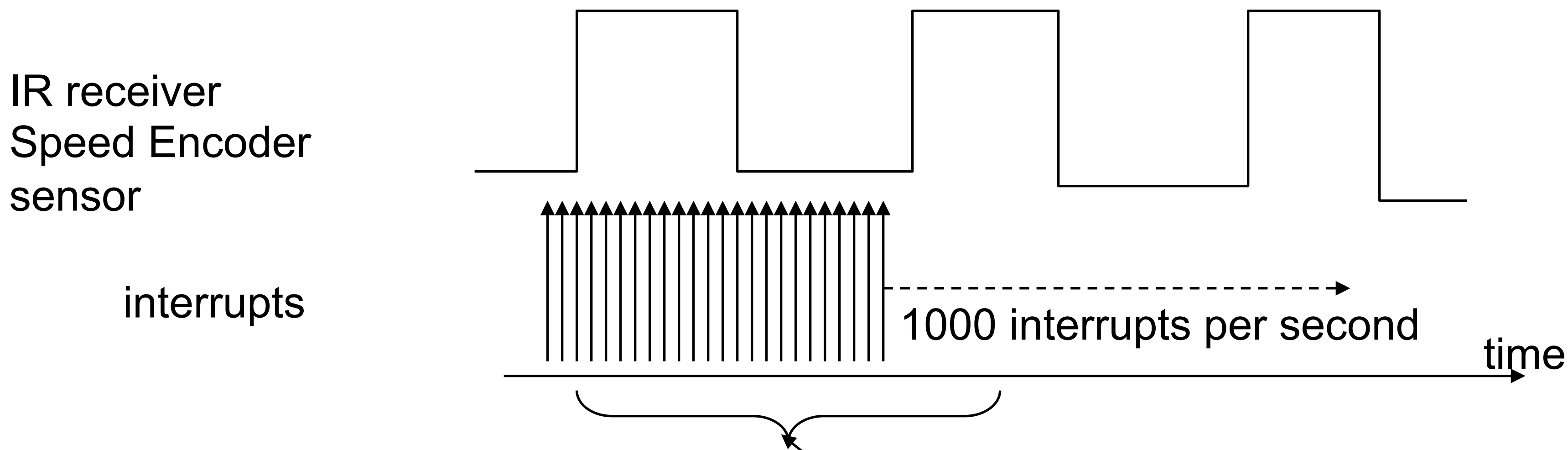
PIDrobotdemo.c

(at course webpage)

http://www.cse.cuhk.edu.hk/~khwong/www2/ceng2400/PIDRobotDemo093.c

control method:
PID (proportional-integral-derivative) control





PID control algorithm using interrupt

```
Main( )
{
Setup( );
:
:
}
```

```
_IRQ( )//1000Hz
{
:
read wheel speed
PID
:
}
```

Overview

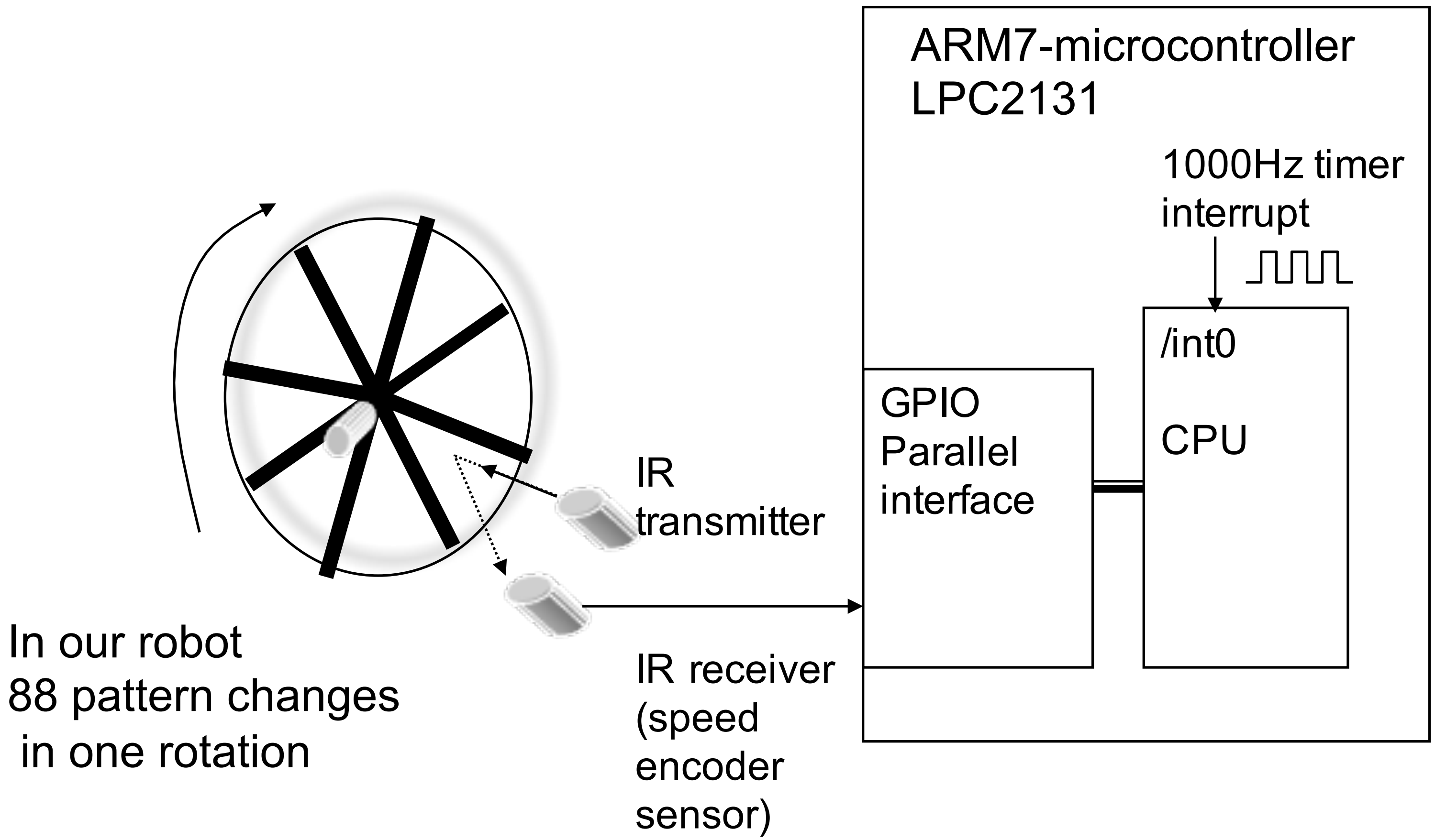
- //////////////main //
- Main()
- { setup()
- forward (Lstep, Rstep, Lspeed, Rspeed).....
- }
- ////////////subroutine ///
- forward (Lstep, Rstep, Lspeed, Rspeed)
- { lcount=0
- if (lcount>=Lstep)
- Stop left motor
- ...same for right motor...
- .}
- //////////timer triggered , interrupt service routine, 1000Hz///////
- __irq exception// Interrupt() running at 1000HZ
- { lcount++
- read wheel speeds
- PID control to achieve L/Rspeed
- ....same for right motor....
- }

Interrupt rate 1000Hz

__IRQ see next slide

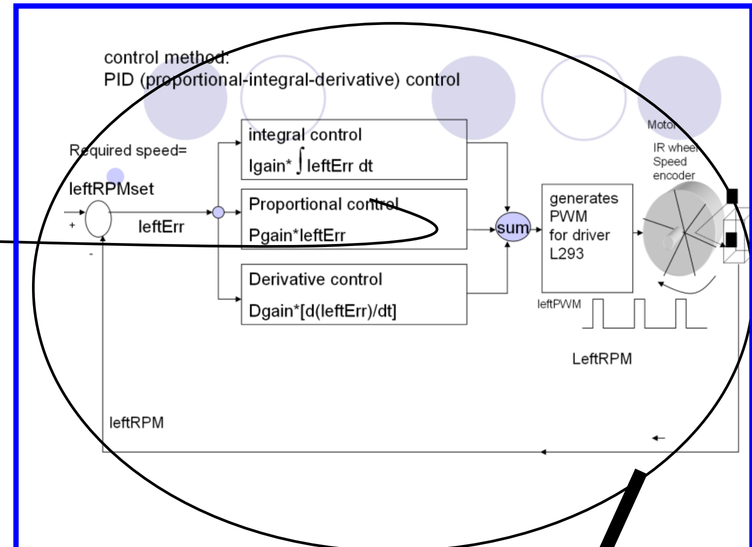
Speed encoder interfacing (show left motor only)

- Block diagram



Exercise 3

_irq interrupt programming method for the main PID loop, using a counter (intcount)

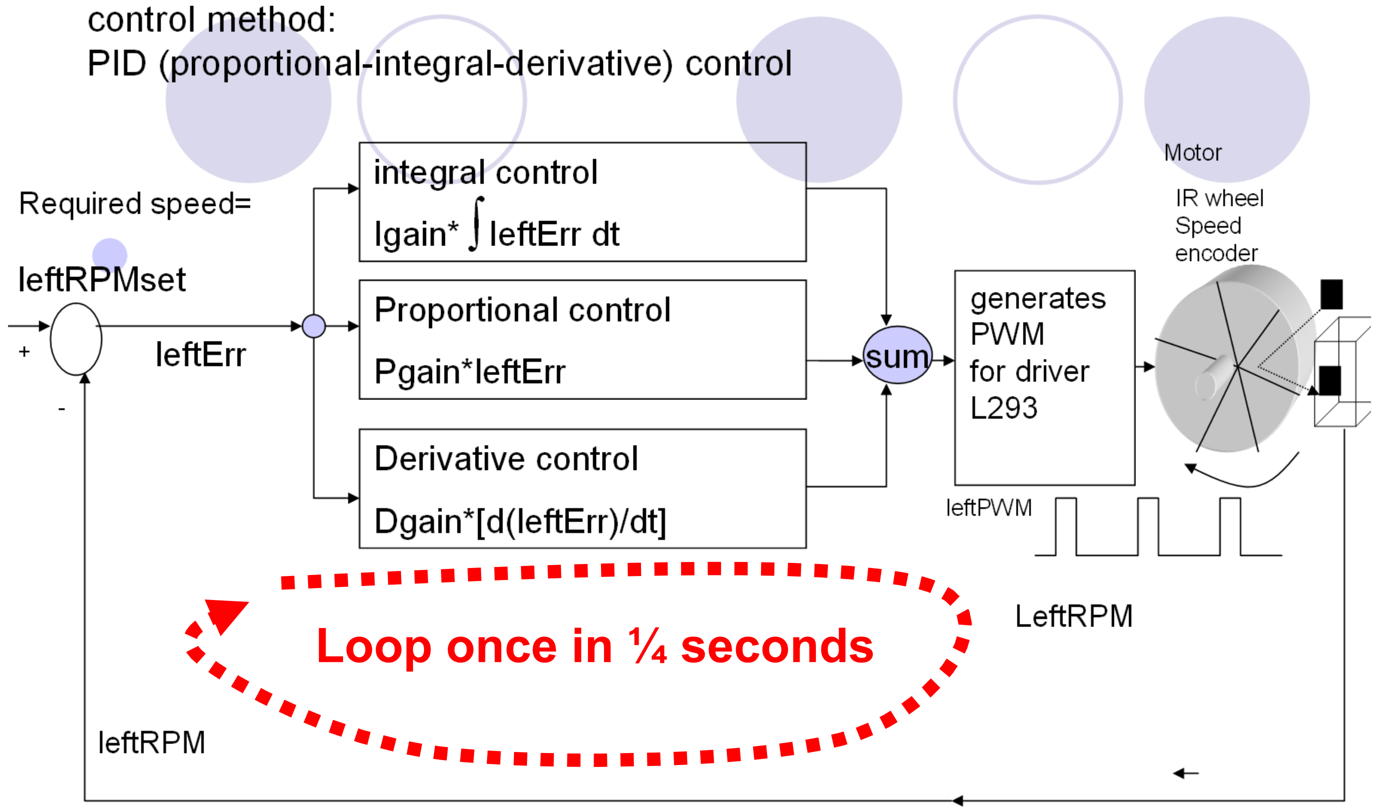


- __irq() exception //timer interrupt 1000HZ,
- { intcount++; //intcount
- //increases at 1000 times/sec
- //read motor speed:
- //update the motor speed in every 1/4 seconds
- if(intcount==250) // happens at every ¼ seconds
- {
- read wheel speed
- Control the PWM using PID
- intcount=0;
- }
- }
- Question: If "if(intcount==250) " is changed to "if(intcount==500) "
- What happens?

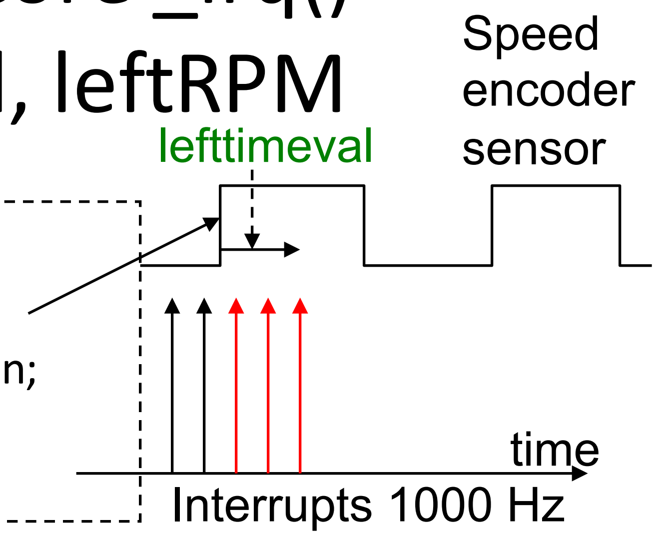
PID CORE

The interrupt service routine enables the loop to run 4 times in a second

-



Speed control interrupt core _irq() part 1: Find motor speed, leftRPM



- void __irq IRQ_Exception()

Part1 Read Wheel speeds

- {ms++; intcount++;
- //get the current wheel sensor values
- lcur=IO0PIN & LWheelSen;rcur=IO0PIN & RWheelSen;
- if(lcur!=lold) { lcount++;lold=lcur;lefttimeval++;}
- :
- //update motor speed by PID feed back control in every ¼ Seconds

Part 2 PID control

- if(intcount==250) { //calculate the speed of left motor in RPM
- leftRPM=lefttimeval;
- : PID core : See following slides
- intcount=0;
- lefttimeval=0;
- }
- }

- lcount :steps of wheel
- lcur : sensor read 1 or 0
- lefttimeval: time lapsed since sensor last change of state
- leftRPM : left wheel RPM

Part 2 :Algorithm for PID core

Set_point

- For every ¼ seconds
  – Find error=(desired value - measured value)
  – {If (error>dead band)
    - { find
      – Error,
      – Accumulated error (add up all previous errors)
      – Derivative error (current error – previous error)
      – PWM+=Kp* Error+
      – Ka*(Accumulated error)+
      – Kd* Derivative error ;
    - }
    - Else
      – Error <= dead_band, error too small do nothing

In our experiment leftPWM=276000 at the beginning and 192800 at steady state

part 2: PID core

PID core

```
1)  if(intcount==250) {//for every ¼ seconds
2)    //caculate the speed of left motor in RPM
3)    leftRPM=lefttimeval;
4)    leftErr =  leftRPMset - leftRPM;      //caculate left error
5)    if((leftErr<DeadBand*(-1))||(leftErr>DeadBand)) //see next slide
6)    {  //if |left error| > deadband
7)      leftP = Pgain * leftErr;//calculate P Proportional term
8)      leftI = Igain * leftaccErr;           //calculate I Integral term
9)      leftD = Dgain * (leftErr - leftlastErr);           //calculate D Derivative term
10)     leftPWM += (leftP + leftI + leftD);//update left motor PWM using PID
11)     if(leftPWM>PWM_FREQ)
12)        leftPWM=PWM_FREQ;//prevent over range (max.=PWM_FREQ)
13)     if(leftPWM<0) leftPWM = 0;    // (min. = 0)
14)        leftaccErr += leftErr;              // accumulate error
15)       leftlastErr = leftErr;//update left last error
16)      :
17)      : // handle  right motor similarly…….
18)     current_leftRPM = leftRPM*240/88;
19)     current_leftPWM = leftPWM;//
20)     lefttimeval = 0;
21)   }
```

P=Proportional

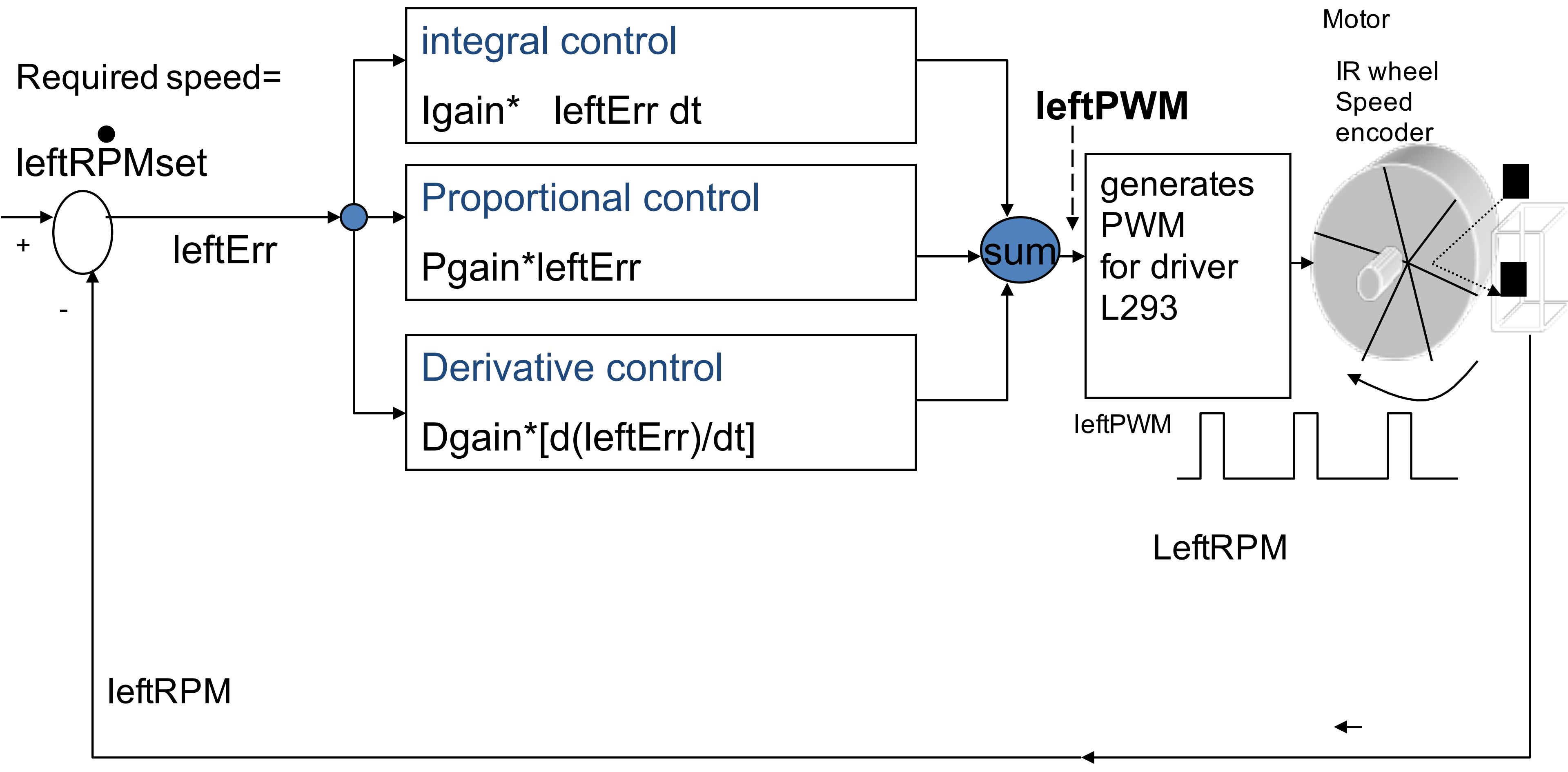
I Integral

D Derivative

because each rotation has 88 counts, the ISR loop is in ¼ seconds, each minutes 60 seconds. see line 18

Pgain, Igain, Dgain are constants found by a trial and error method, here we have
Pgain = 8000;
Igain = 6000;
Dgain = 5000;

control method:
PID (proportional-integral-derivative) control



Inside the PID core, we will study these lines

```
5) if((leftErr<DeadBand*(-1))||(leftErr>DeadBand))
:
:
7) leftP = Pgain * leftErr;          //calculate P Proportional term
8) leftI = Igain * leftaccErr;       //calculate I Integral term
9) leftD = Dgain * (leftErr - leftlastErr);//calculate D Derivative term
10) leftPWM += (leftP + leftI + leftD);//update motorPWM by PID
:
:
14) leftaccErr += leftErr;           // accumulate error
```

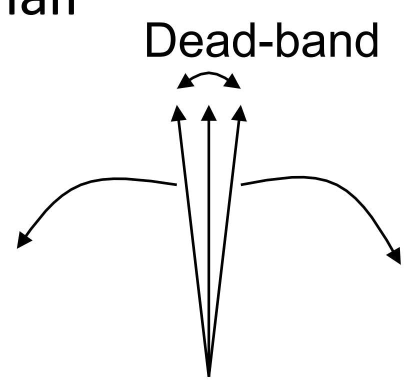
Dead band

```
line5) if((leftErr<DeadBand*(-1))||(leftErr>DeadBand))
```

- **Dead-band** : *A **Dead-band** (sometimes called a **neutral zone**) is an area of a signal range or band where no action occurs* :

```
leftErr =  leftRPM - leftRPMset; //calculate left error
    if(leftErr>DeadBand )
    { activate motor}
```

- only enable motor when leftErr> a small value (deadband, ie =1 in our robot)
- Otherwise may oscillate when leftErr is small



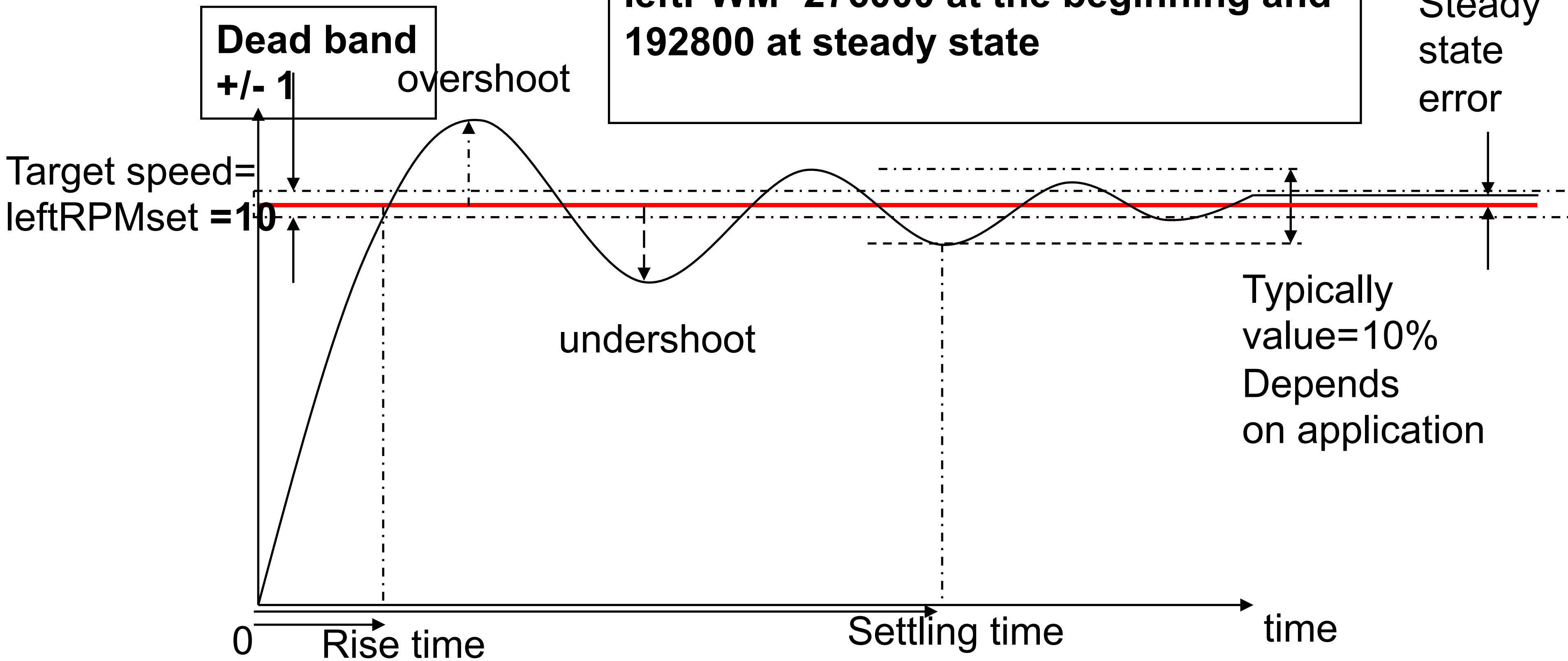
Exercise 4:

Discuss what will happen if dead-band is changed, say (a) 0.5 or (b) 2.

Example of a dead band ;do nothing “if 10-1<leftPRM <10+1

In our experiment leftPWM=276000 at the beginning and 192800 at steady state

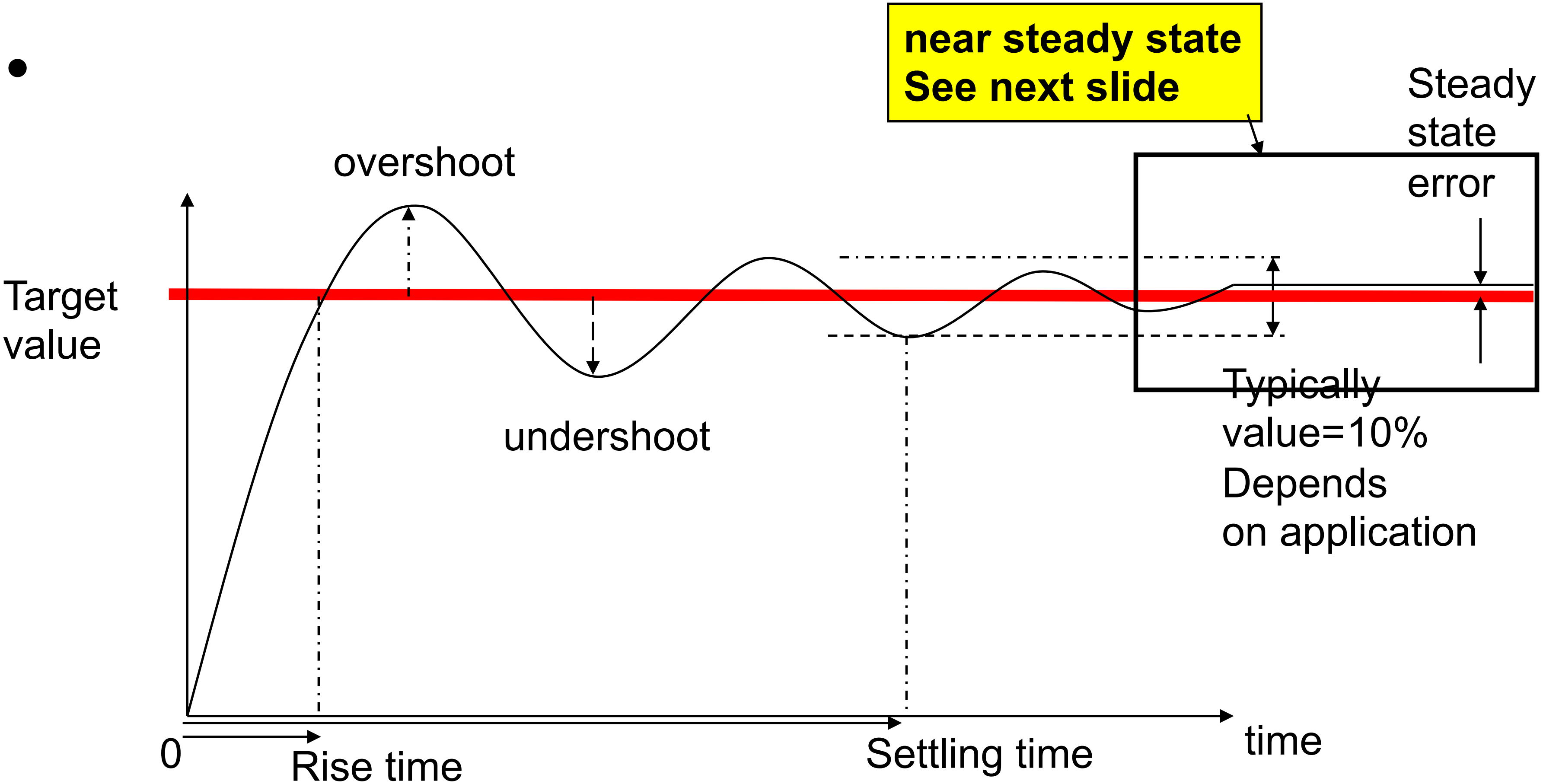
-



When leftRMPset=10, the real RPM is RPM*240/88=27.3., because each rotation has 88 counts, the ISR loop is in ¼ seconds, each minutes 60 seconds. see line 18

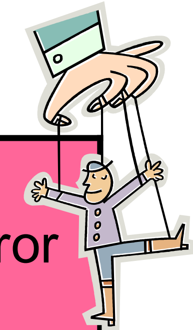
Parameters for evaluating a control system

•

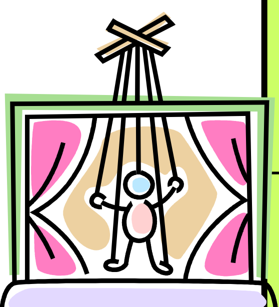


(line 7) Effects of increasing Kp(P)

http://en.wikipedia.org/wiki/PID_controller



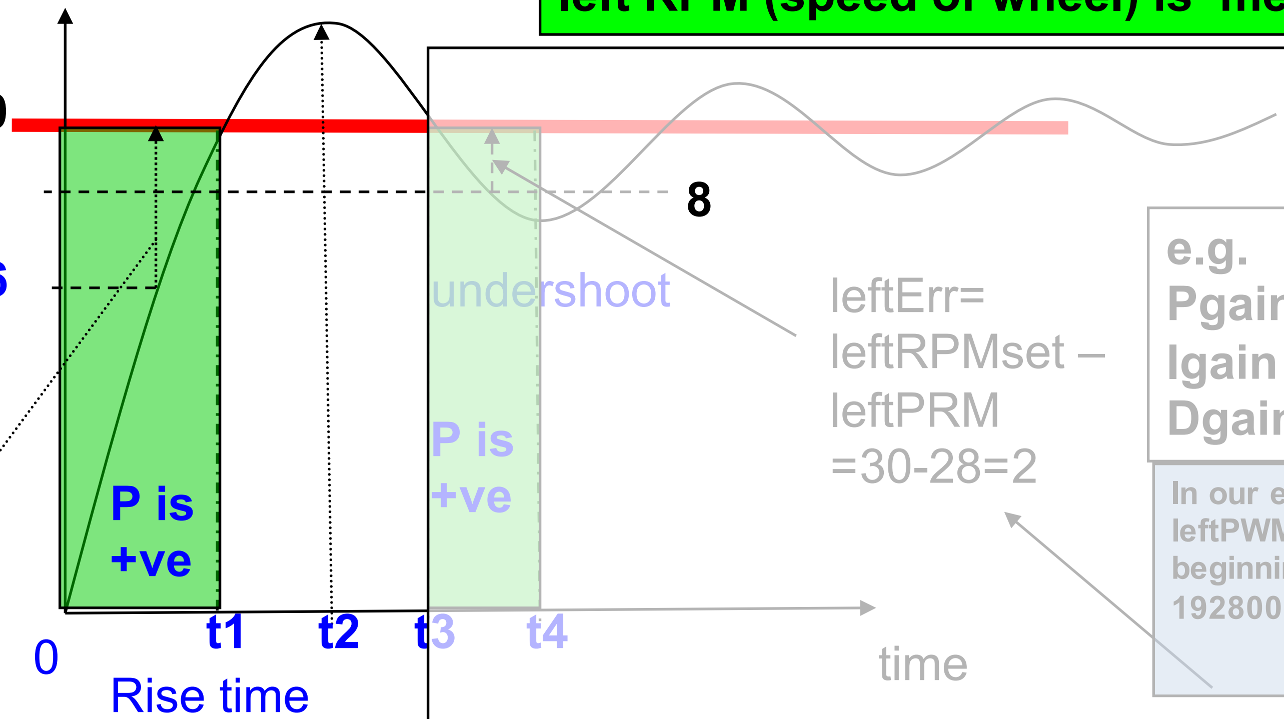
Parameter	Rise Time	Overshoot	Settling Time	Steady state error
(1) Kp (Pgain)	Decrease step1	Increase	Small Change	Decrease
Ki (Igain)	Decrease	Increase	Increase	Eliminate step3
Kd (Dgain)	Small Change	Decrease step2	Decrease	Small Change



Example: t0→t1, The proportional term when the measurement is below the set point (leftRPMset), proportional P term is +ve

(leftErr = leftRPMset-leftRPM)

Usage of the P proportional term
Each ¼ seconds a new
left RPM (speed of wheel) is measured



leftErr=
leftRPMset –
leftPRM
=10-6=4

leftErr=
leftRPMset –
leftPRM
=30-28=2

e.g.
Pgain = 8000;
Igain = 6000;
Dgain = 5000;

In our experiment leftPWM=276000 at the beginning and 192800 at steady state

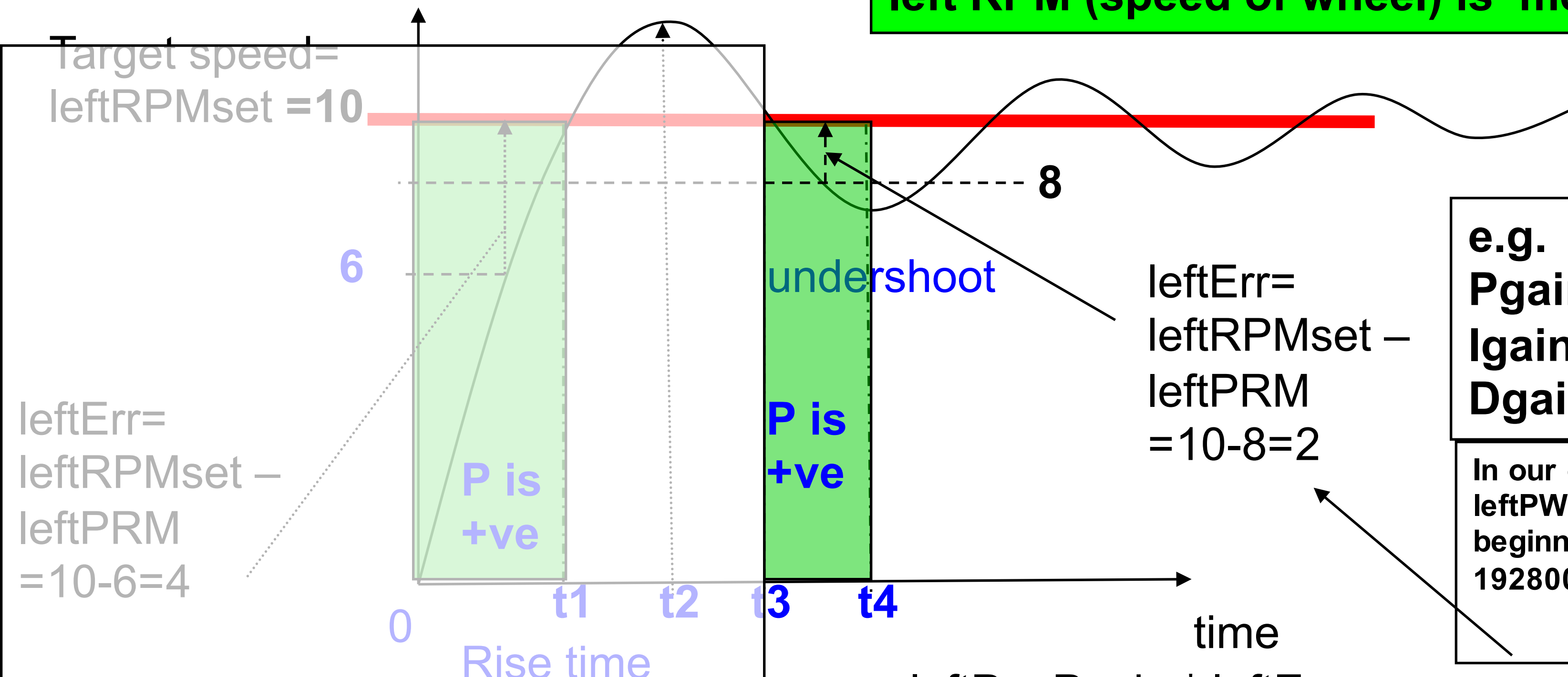
leftP = Pgain * leftErr;
leftP= 8000* (10-6)= 8000*4 is positive
This term pushing up "leftPWM"
(more energy delivered to the wheel).

leftP = Pgain * leftErr;
leftP= 8000* (0-8)= 8000*2 is positive
This term pushing up "leftPWM"
(more energy delivered to the wheel).

Exercise 5: Fill in ?__: t0→t1, The proportional term when the measurement is below the set point (leftRPMset), proportional P term is +ve

•

Usage of the P proportional term
Each ¼ seconds a new
left RPM (speed of wheel) is measured



e.g.
Pgain = 8000;
Igain = 6000;
Dgain = 5000;

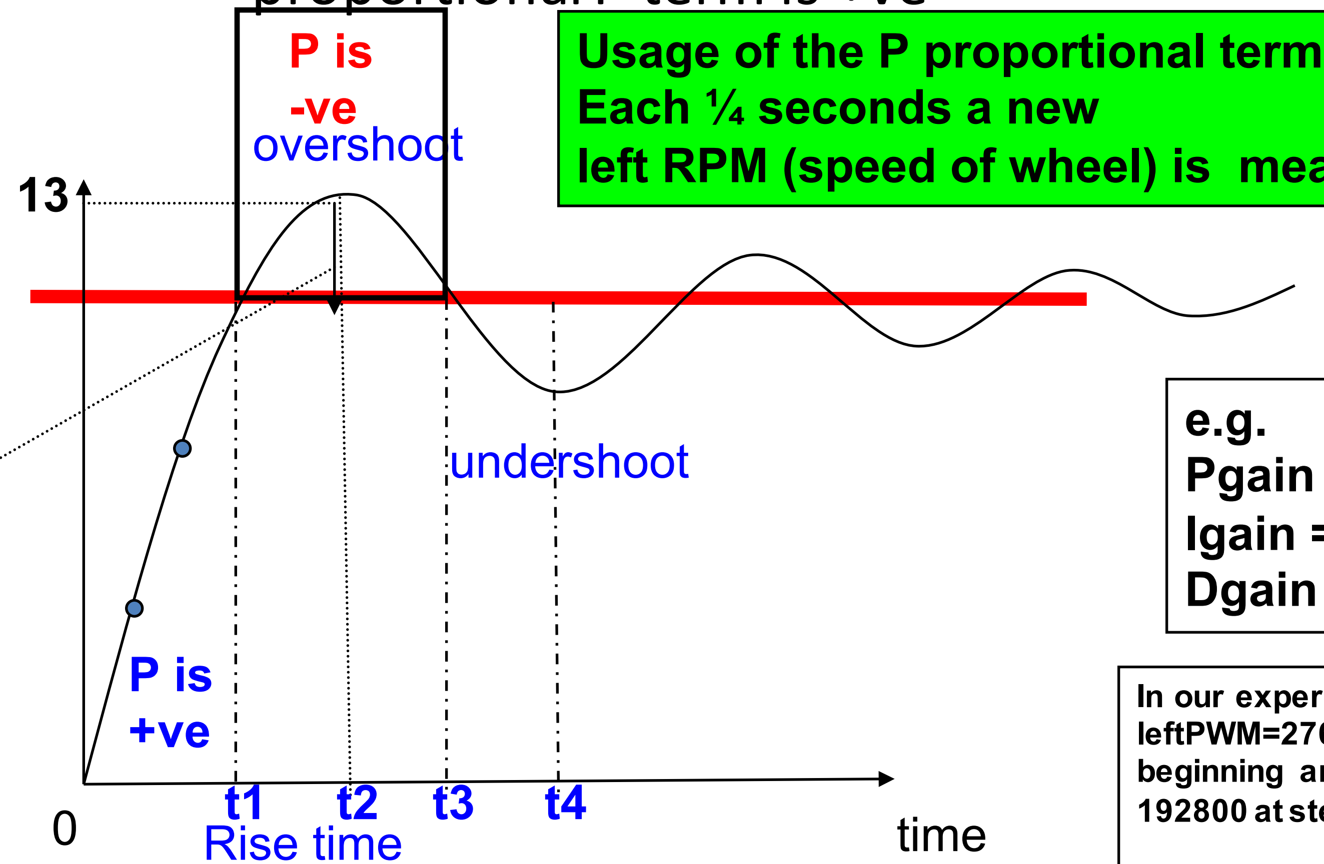
In our experiment leftPWM=276000 at the beginning and 192800 at steady state

leftP = Pgain * leftErr;
leftP= 8000* (10-6)= 8000*4 is positive
This term pushing up "leftPWM"
(more energy delivered to the wheel).

leftP = Pgain * leftErr;
leftP=?_____________ ,is +ve or -ve?
"leftPWM" is increasedor decreased?__
more/less energy delivered to wheel?__

Example: t1→t3,The proportional term when the measurement is below the set point (leftRPMset), the proportional P term is +ve

•



Usage of the P proportional term
Each ¼ seconds a new
left RPM (speed of wheel) is measured

Target speed=
leftRPMset **=10**

leftErr=
leftRPMset –
leftPRM
=10-13= -3

e.g.
Pgain = 8000;
Igain = 6000;
Dgain = 5000;

In our experiment leftPWM=276000 at the beginning and 192800 at steady state

leftP = Pgain * leftErr;

leftP= 8000* (10-13)= 8000*(-3) is negative

<u>**This term lowering down "leftPWM" (less energy delivered to the wheel).**</u>

Understanding PID –a little summary for the P proportional term

- When the measurement is below the set point (leftRPMset)
  - The motor is currently too slow
  - The P term calculated is +ve, need to push the speed higher ↑
- When the measurement is above the set point (leftRPMset)
  - The motor is running too fast
  - The P proportional term is -ve, lowering down the speed. ↓
- Increase in Pgain (Kp) will decrease rise time (meaning faster to reach set point) , decrease steady state error (study it later)
  - It also increases overshoot

(line 9) Effects of increasing Kd (D)

http://en.wikipedia.org/wiki/PID_controller

Parameter	Rise Time	Overshoot	Settling Time	Steady state error
Kp (Pgain) Igain	Decrease step1	Increase	Small Change	Decrease
Ki (Igain) gainI	Decrease	Increase	Increase	Eliminate step3
Kd (Dgain)	**Small Change**	**Decrease step2**	**Decrease**	**Small Change**

Derivative term

- Derivative control
  – Dgain*[d(leftErr)/dt]
- d(leftErr)/dt =
  – =Derivative term
  – =current_Err - last_Err
  – =leftErr – leftlastErr ; in our program

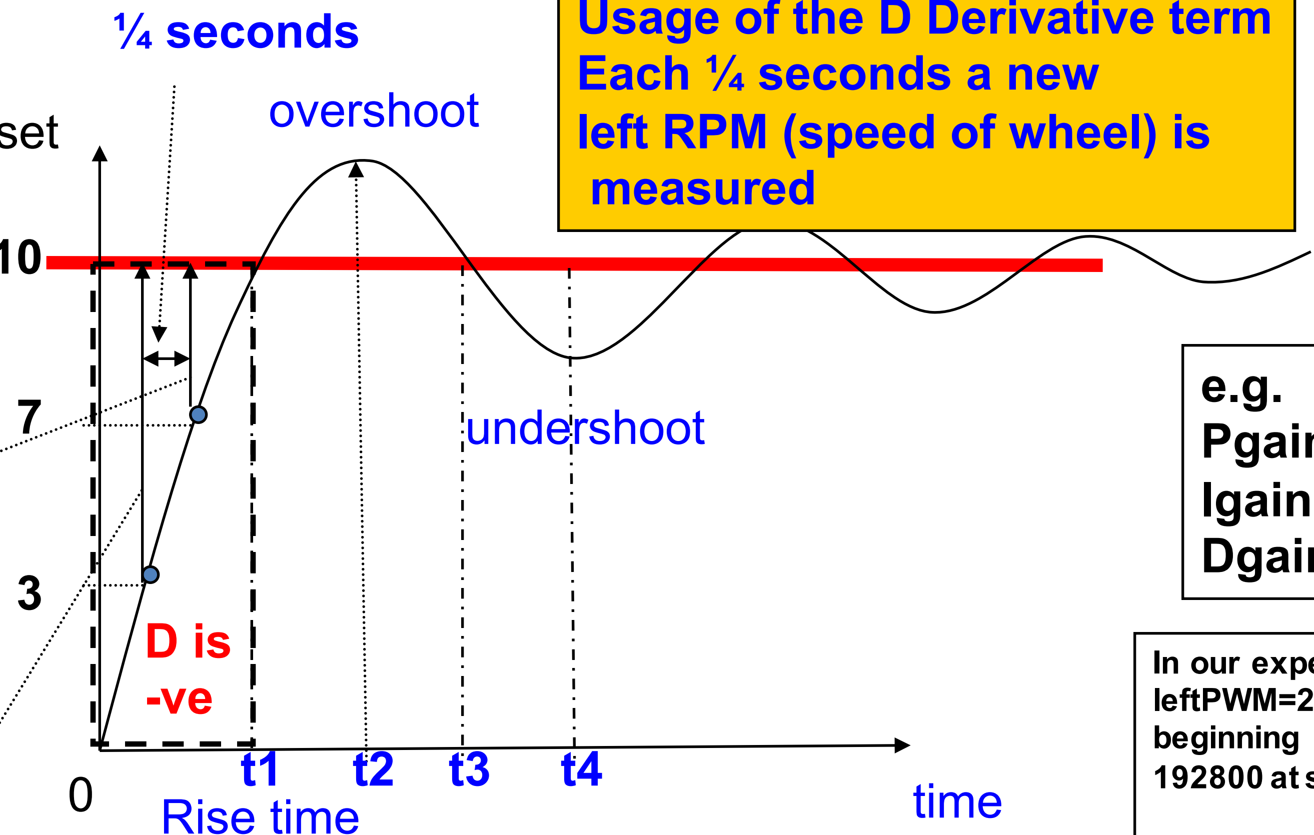
Example: time 0→ t1, the Derivative term (=current-previous) When the measurement is rising, the Derivative term is -ve

- Target Value= leftRPMset =10

leftRPMset **=10**

leftErr= leftRPMset – leftPRM =10-7=3

leftlastErr= leftRPMset – eftlastPRM =10-3=7



Usage of the D Derivative term Each ¼ seconds a new left RPM (speed of wheel) is measured

e.g.
Pgain = 8000;
Igain = 6000;
Dgain = 5000;

In our experiment leftPWM=276000 at the beginning and 192800 at steady state

leftD = Dgain * (leftErr – leftlastErr);

leftD= 5000* (3-7)= 5000*(-4) is negative

To do:This term lowering down “leftPWM” (less energy delivered to the wheel).

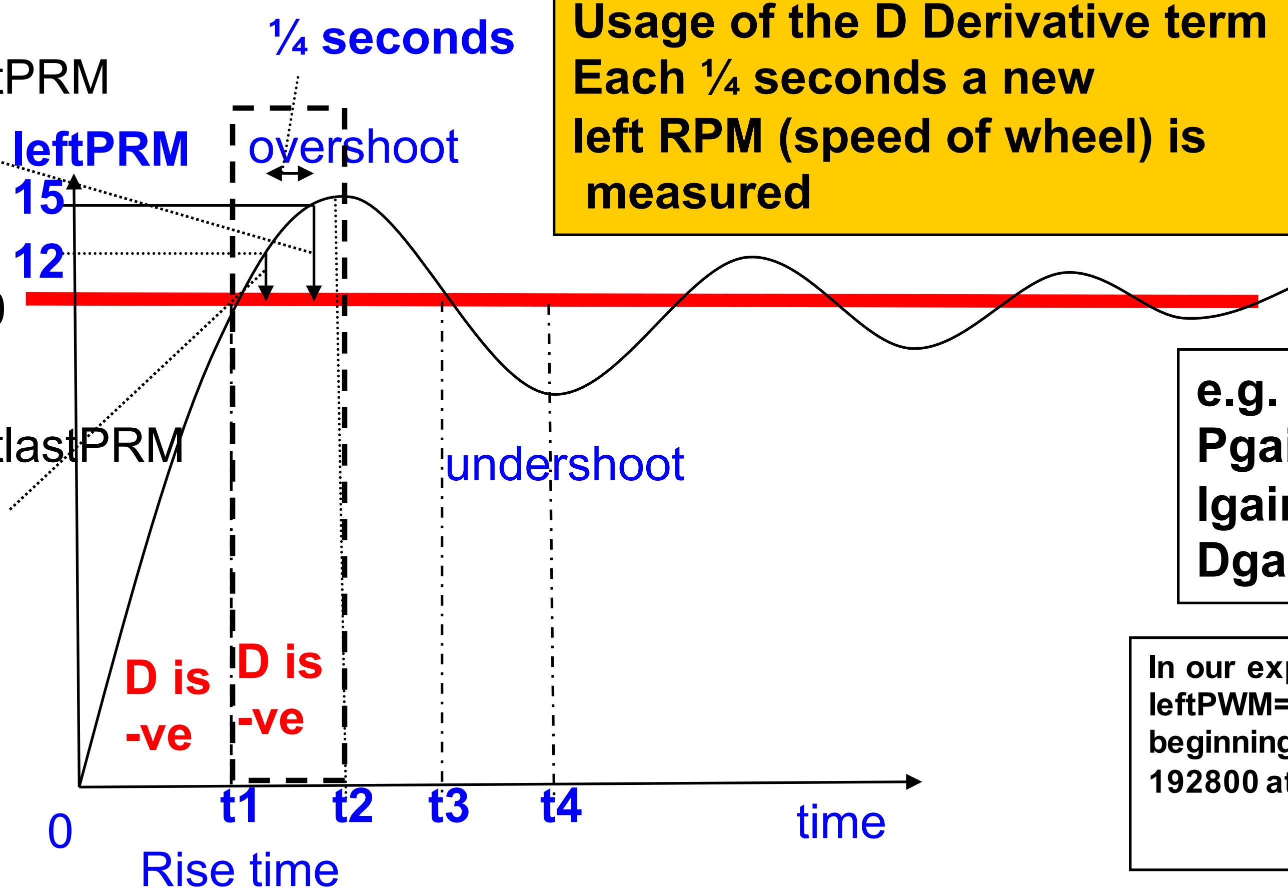
Example: time t1→t2, the Derivative term

When the measurement is rising, the Derivative term is -ve

leftErr= leftRPMset -leftPRM =10-15= -5

leftRPMset **=10**

leftlastErr= leftRPMset -leftlastPRM =10-12= -2



Usage of the D Derivative term
Each ¼ seconds a new left RPM (speed of wheel) is measured

e.g.
Pgain = 8000;
Igain = 6000;
Dgain = 5000;

In our experiment leftPWM=276000 at the beginning and 192800 at steady state

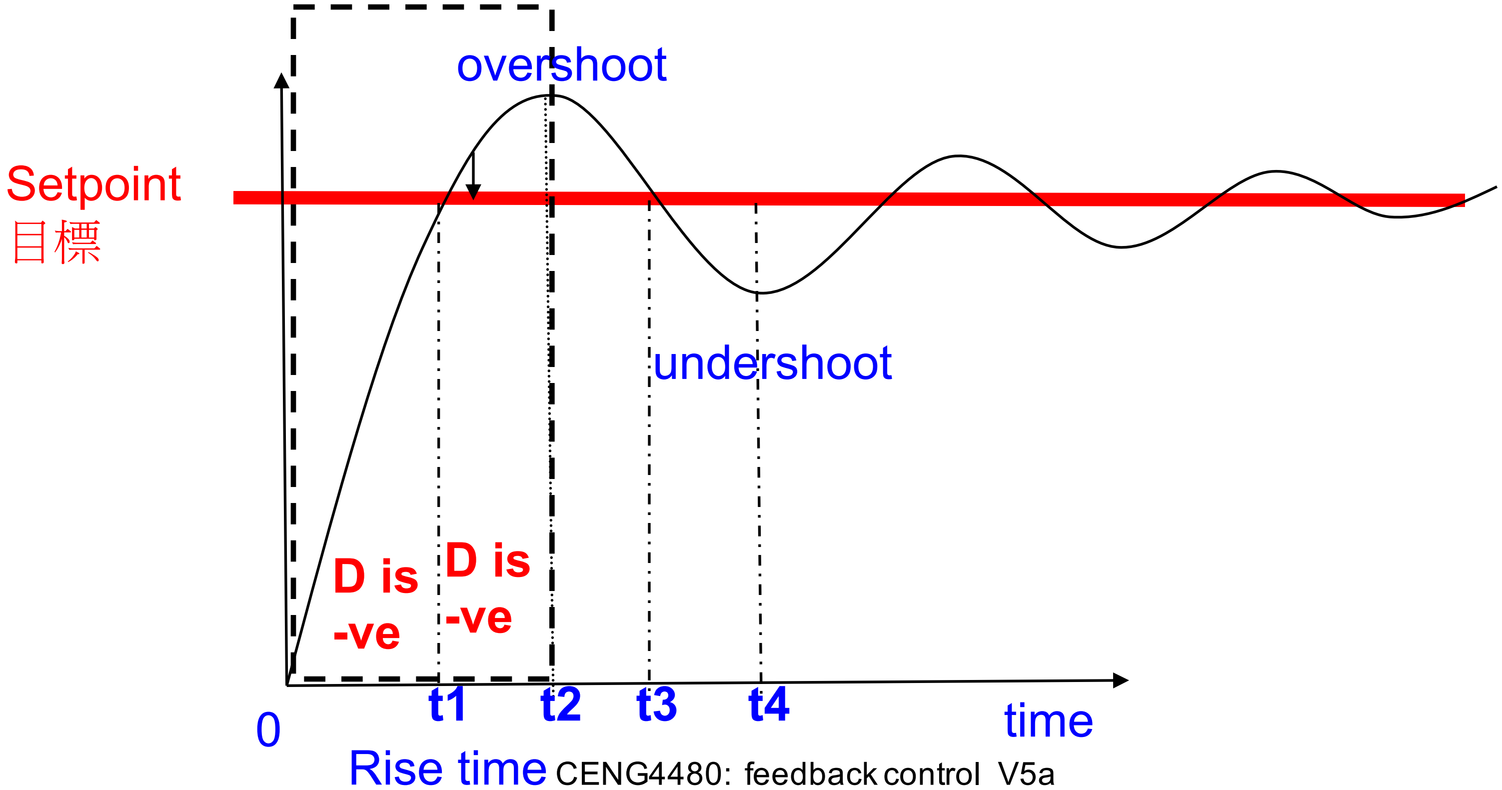
leftD = Dgain * (leftErr – leftlastErr);

leftD= 5000* (-5- (-2))= 5000*(-3) is negative

This term lowering down "leftPWM" (less energy delivered to the wheel).

Little Summary: Negative D **Derivative term**

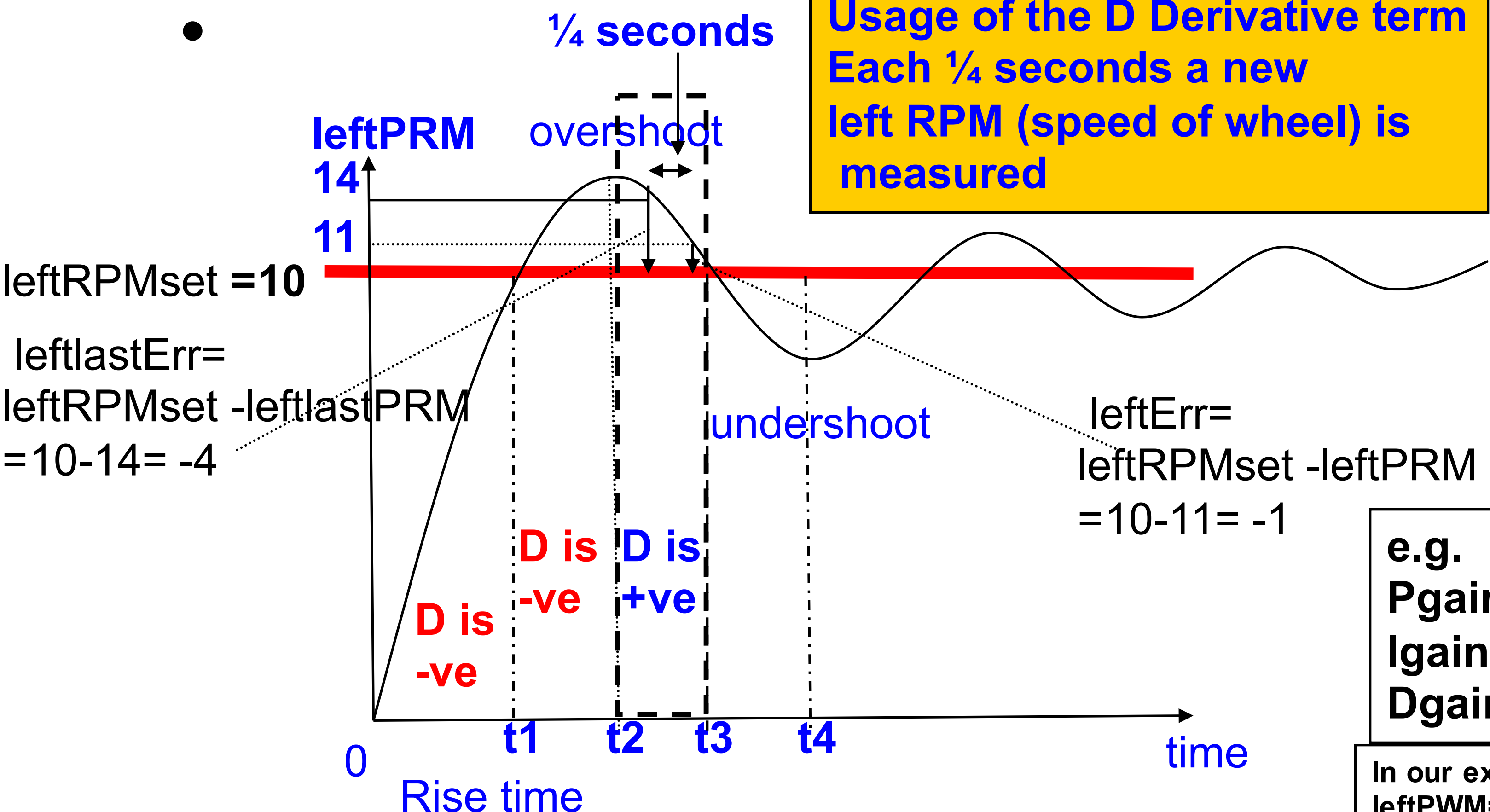
- When the measurement (leftRPM) is rising, the change (change= current-previous) of error (error = setpoint-leftRPM) is -ve = d(Err/dt)=-ve
- **D Derivative term=** Kd*d(Err/dt) is –VE
- Decrease energy to motor → decrease overshoot



Example: time t2→t3, the Derivative term

When the measurement is falling, the Derivative term is +ve

Usage of the D Derivative term
Each ¼ seconds a new left RPM (speed of wheel) is measured



e.g.
Pgain = 8000;
Igain = 6000;
Dgain = 5000;

In our experiment leftPWM=276000 at the beginning and 192800 at steady state

leftD = Dgain * (leftErr – leftlastErr);

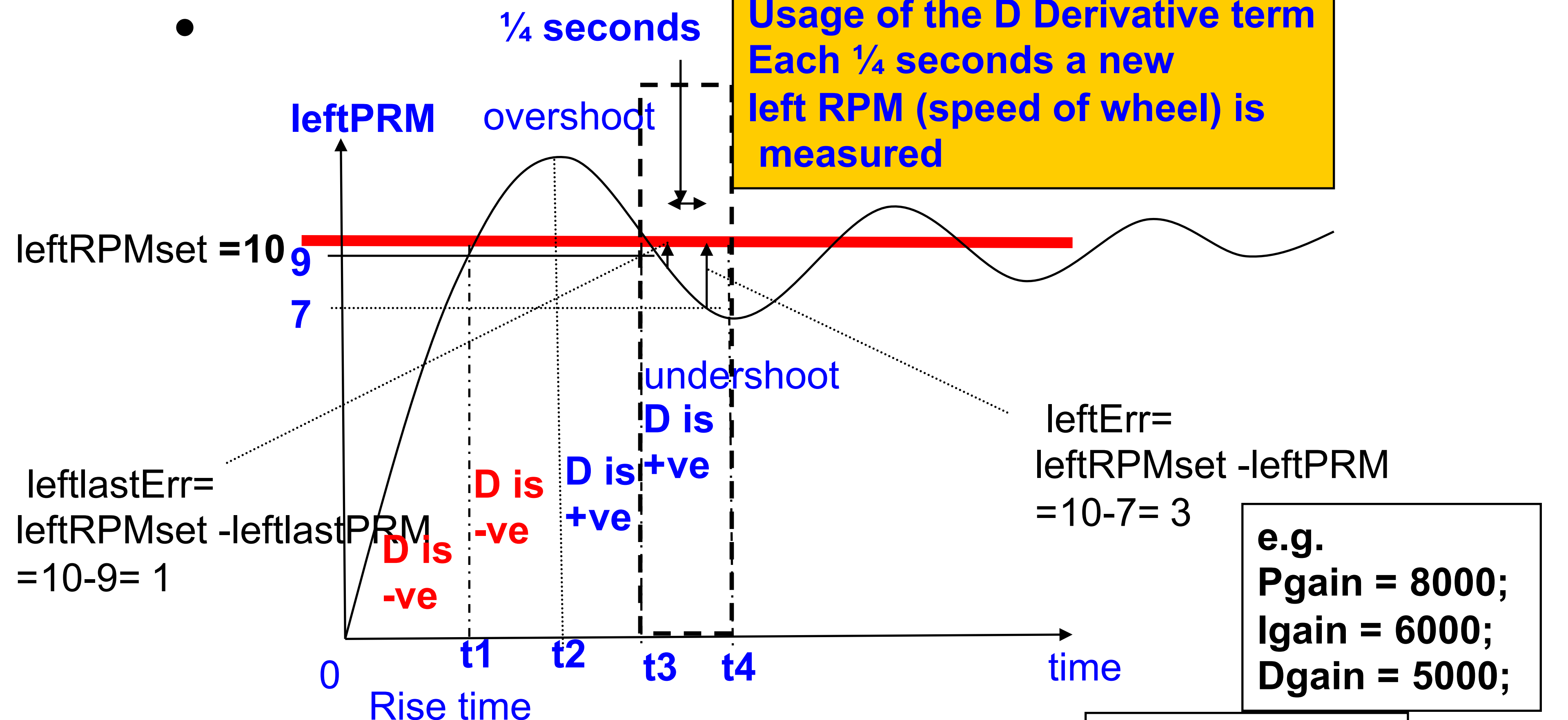
leftD= 5000* (-1- (-4))= 5000*3 is positive

This term pushing up "leftPWM" (more energy delivered to the wheel).

Exercise 6:Fill in ?_ time t2→t3, the Derivative term

When the measurement is falling, the Derivative term is +ve

-



e.g.
Pgain = 8000;
Igain = 6000;
Dgain = 5000;

In our experiment leftPWM=276000 at the beginning and 192800 at steady state

leftD = Dgain * (leftErr – leftlastErr);

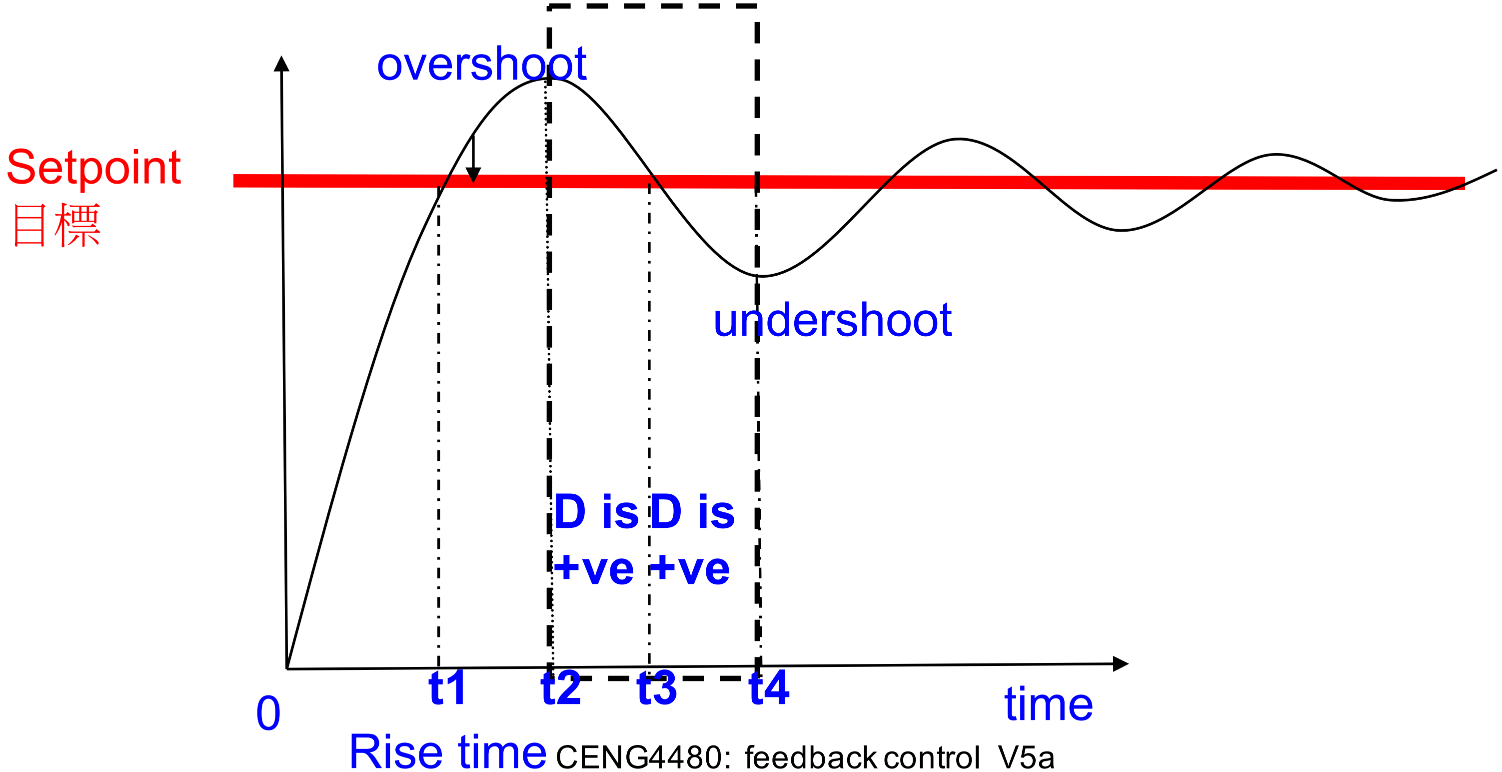
leftD= ?_____________, which is +ve or –ve?_____

"leftPWM" increased or decreased?_____

More/less energy delivered to the wheel?___

Little Summary: Positive D **Derivative term**

- When the measurement (leftRPM) is falling, the change (change= current-previous) of error (error = setpoint-leftRPM) is +ve
- **D Derivative term=** Kd*d(Err/dt) is +VE
- Increase energy to motor → decrease undershoot



Understanding PID –a little summary for the D derivative term

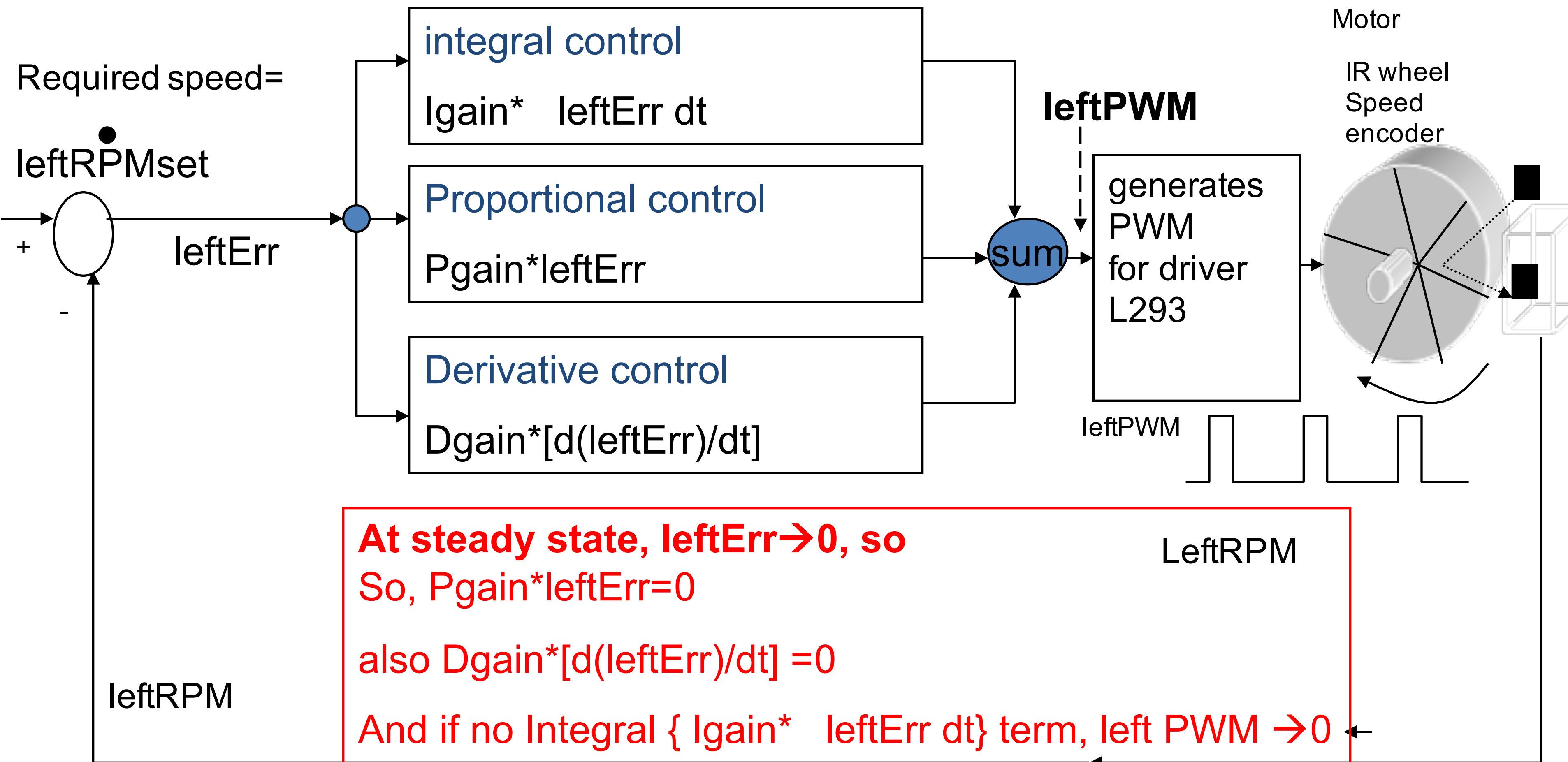
- When the measurement (leftRPM) is rising,
  - The motor is gaining speed
  - The D derivative term is –ve, so lowering the motor speed → decrease overshoot
- When the measurement (leftRPM) is falling,
  - The motor is reducing speed
  - The Derivative term is +ve, so pushing the motor speed higher → decrease undershoot
  - In conclusion, the gradient of the error (Err) determines the adjustment. Depends on whether d(Err)/dt is +ve or –ve.
- Increase in Dgain (Kd) will decrease overshoot/undershoot and settling time (system more stable)

(line 8)Effects of increasing Ki (I)

http://en.wikipedia.org/wiki/PID_controller

Parameter	Rise Time	Overshoot	Settling Time	Steady state error
Kp (Pgain)	Decrease step1	Increase	Small Change	Decrease
Ki (Igain)	**Decrease**	**Increase**	**Increase**	**Eliminate step3**
Kd (Dgain)	Small Change	Decrease step2	Decrease	Small Change

control method:
PID (proportional-integral-derivative) control



At steady state, leftErr→0, so

So, Pgain*leftErr=0

also Dgain*[d(leftErr)/dt] =0

And if no Integral { Igain* leftErr dt} term, left PWM →0

It is a problem.

Time → near steady state

leftRPMset=leftRPM

hence leftErr =0, leftlastErr=0→ leftP=0, leftD=0

In our experiment leftPWM=276000 at the beginning and 192800 at steady state

e.g.
Pgain = 8000;
Igain = 6000;
Dgain = 5000;

- leftErr = leftRPMset – leftRPM=0
- So as leftlastErr = 0
- Therefore
- Steps
- 7) leftP = Pgain * leftErr//=0 //calculate P Proportional term
- 8) leftI = Igain * leftaccErr; //calculate I Integral term
- 9) leftD = Dgain * (leftErr - leftlastErr)//=0 //calculate D Derivative term
- 10) leftPWM += (leftP + leftI + leftD);//update left motor PWM using PID

near steady state : leftP=0 leftD=0

The only valid term is the integral term "leftI"

- leftPWM += leftI
- **The main idea is to create a small term (leftI) to maintain the leftPWM value**

The integral term is found by adding all previous errors

same as leftaccErr=sum{leftErr(t=0)+leftErr(t=1)+.. +leftErr(t=now)}

- 14) leftaccErr += leftErr;//LeftaccErr is the summation of all previous errors
- 8) leftI=Igain*leftaccErr// integral term,
- :
- 10) leftPWM += leftI // near steady state, only leftI is valid
- :

e.g.
Pgain = 8000;
Igain = 6000;
Dgain = 5000;

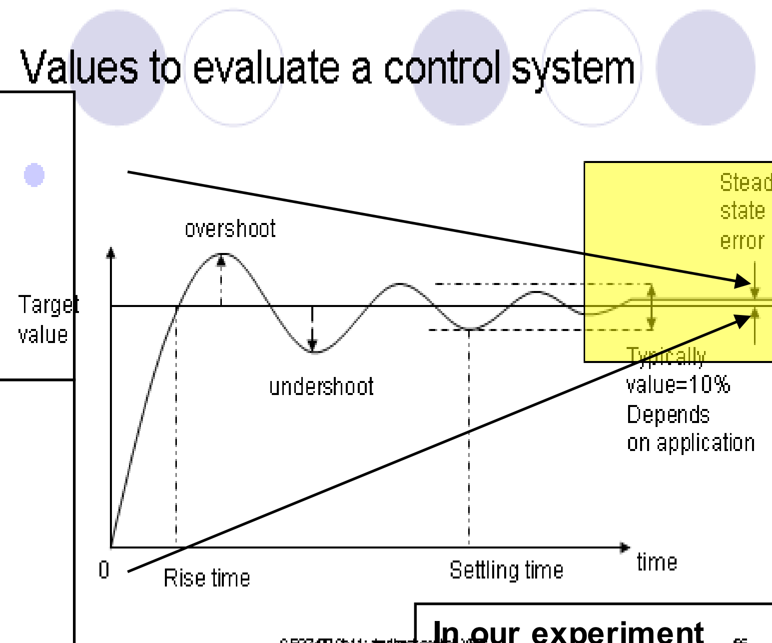
- Don't worry it will not become infinitive, one measure to safeguard this is:
  - 11) if(leftPWM>PWM_FREQ);// PWM_FREQ=maximum PWM allowed
  - 12) leftPWM=PWM_FREQ;//prevent over range
  - (max.=PWM_FREQ)
- Also, because near steady state , leftaccErr will adjust itself automatically, see next slide

In our experiment leftPWM=276000 at the beginning and 192800 at steady state

How the integral term adjusts itself automatically near steady state

Pgain = 8000;
Igain = 6000;
Dgain = 5000;

- Near steady state
  - 8) leftI=Igain*leftacceErr
  - 10) leftPWM += leftI
- If measured speed (leftRPM) > set point
  - leftErr is -ve
  - leftaccErr (Acculumation) decreases, hence reducing leftaccErr at a suitable value to maintain leftPWM
- If measured speed (leftRPM) < set point
  - leftErr is +ve
  - leftaccErr (Acculumation) increases, hence increasing leftaccErr at a suitable value to maintain leftPWM

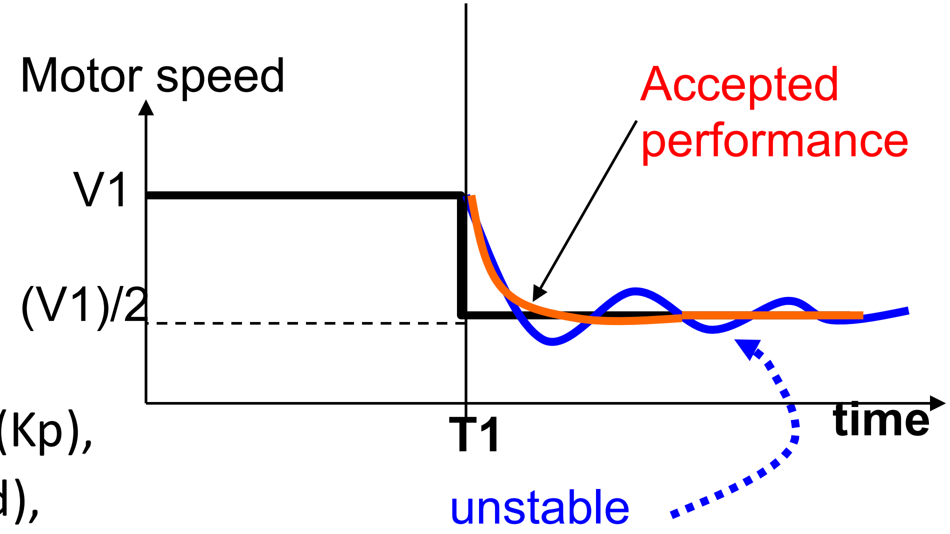


In our experiment leftPWM=276000 at the beginning and 192800 at steady state

Understanding PID –a little summary for the I Integral term

- When the measurement is below the set point (leftRPMset)
  - The motor is slow
  - The I Intergral term is +ve, pushing the speed higher
- When the measurement is above the set point (leftRPMset)
  - The motor is running too fast
  - The I intergral term is -ve, lowering down the speed.
- Increase in Igain (Ki) will result in
  - It is similar to the P term (see above two points)
  - So increase overshoot, settling time
  - and decrease rise time
  - Note: the main function of Integral control is to reduce steady state error

PID Tuning (usually done by trail and error)



- Tune (adjust manually)
  - step1) Pgain proportional_gain (Kp),
  - step2) Dgain derivative_gain (Kd),
  - step3) Igain integral_gain (Ki)
- Set constant speed V1 for a while (5 seconds) and reduced to (V1)/2 at T1
  - Record the speed by the computer after T1 and see if the performance is ok or not
    - Yes (accept Kp,Ki,Kd)
    - No (tune Kp,Ki,Kd again)

done

Summary

- Studies PID control theory and implementation