Arduino Day 2017

Arduino for Closed Loop Control

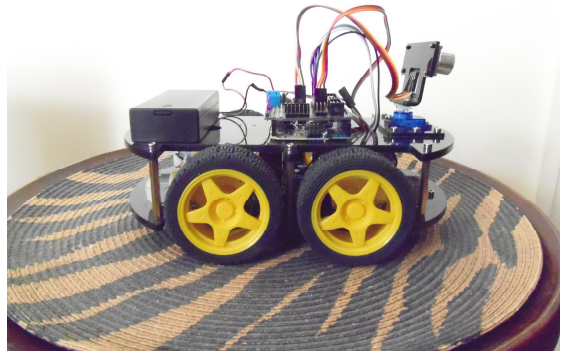
Objectives

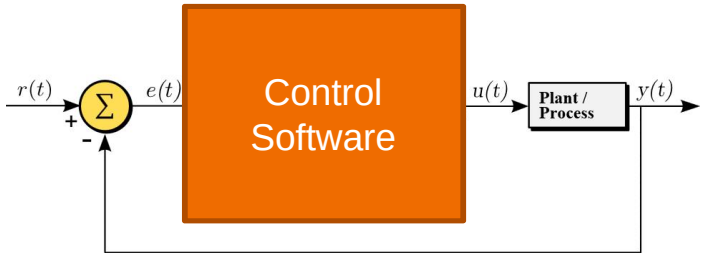
Explain Basics of Closed Loop Control

- Define Closed Loop Control
- Demo of Closed Loop Control on heading.
- PID primary tool of Closed Loop Control
- PID implementation in Code

What is Software Closed Loop Control?

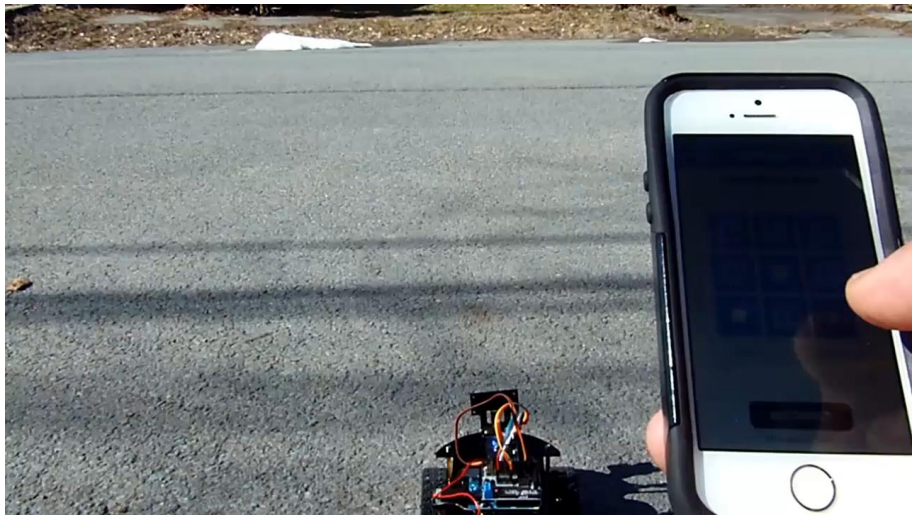
Closed loop control is when software is used to drive to zero the error between desired physical behavior and the actual measured behavior of the system.





Demo of Closed Loop Control

Without Closed Loop Control

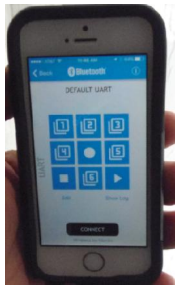


Before

With Closed Loop Control

After

Robot Car + Arduino 101



Bluetooth LE

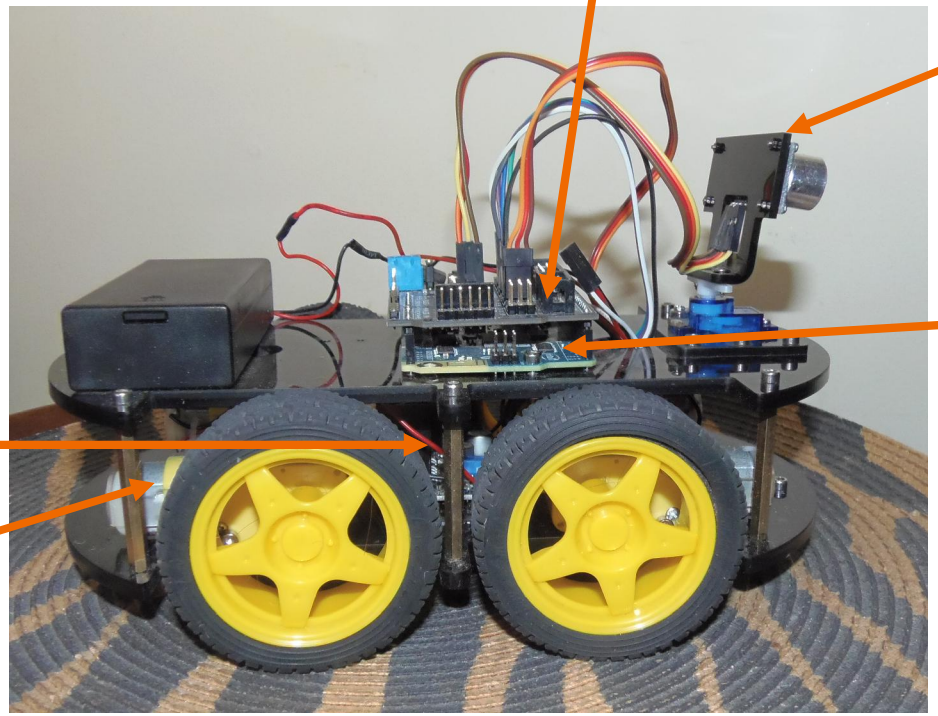
Arduino Sensor Shield V5.0

Ultrasonic Sensor and Servo

Arduino 101

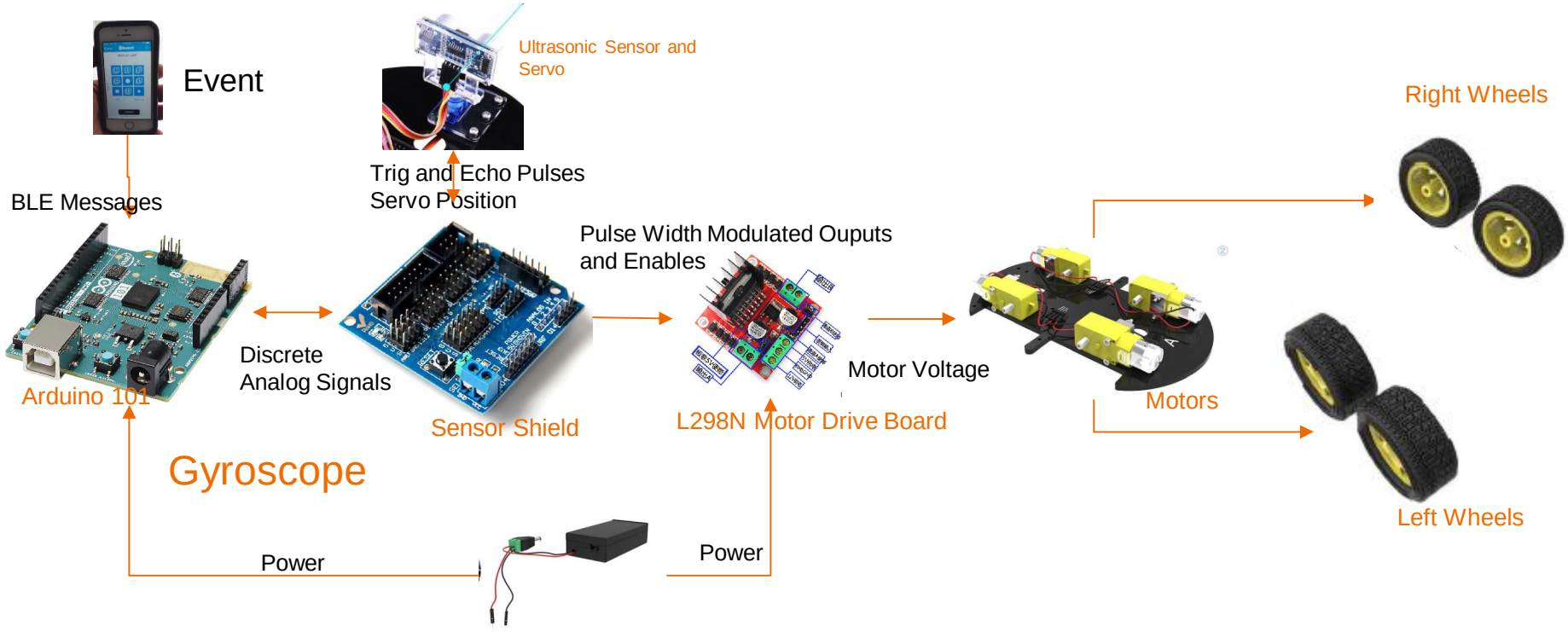
L298N Motor Drive Board

4 Motors



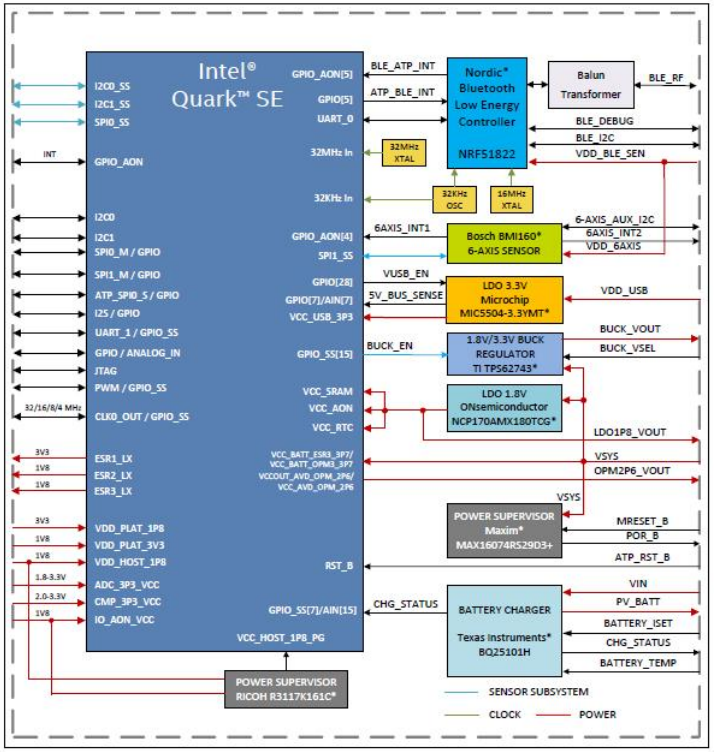
Elegoo Robot Car - UNO + Arduino 101

Hardware Block Diagram



Features Intel Curio of the Arduino 101

Figure 1-1. Intel® Curie™ module block diagram

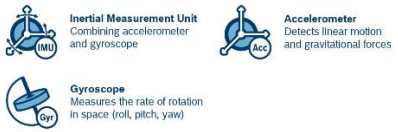


Intel Quark microcontroller
32 bit – Pentium x86 core.

384kb of flash
80kb of SRAM

ARC EM-4 based Sensor Subsytem

Six-axis accelerometer/gyroscope (Inertial Measurement Unit)

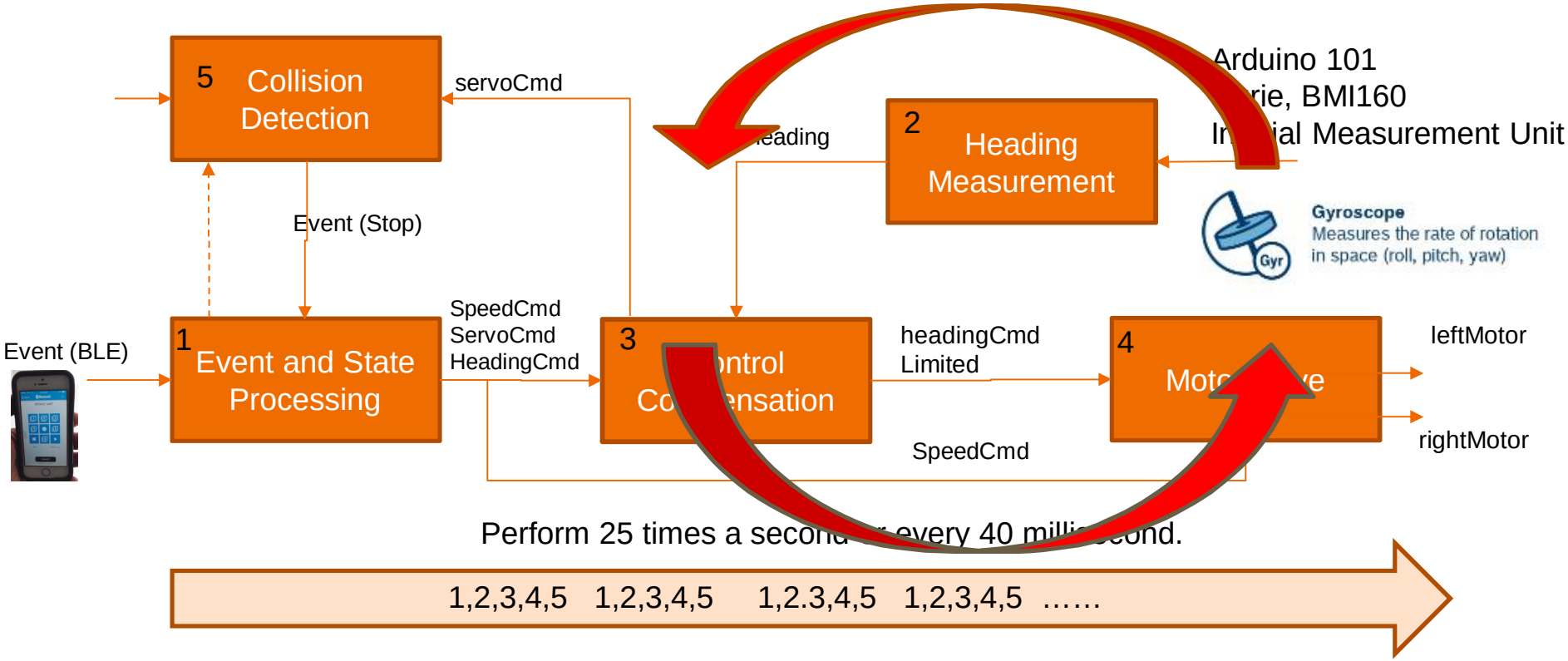


Bluetooth Low Energy

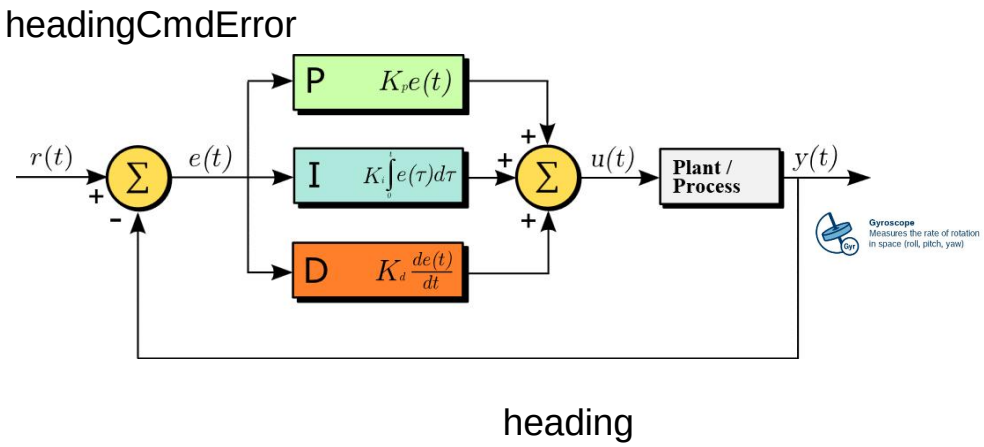
Pattern Matching Engine (Neural Net)

USB, I2C,I2S, UART, SPI, DMA Controller, GPIO, PWM
ADC Unit, Analog Compartors, RTC

Software needs to take action in “Real-Time”



Control Compensation PID

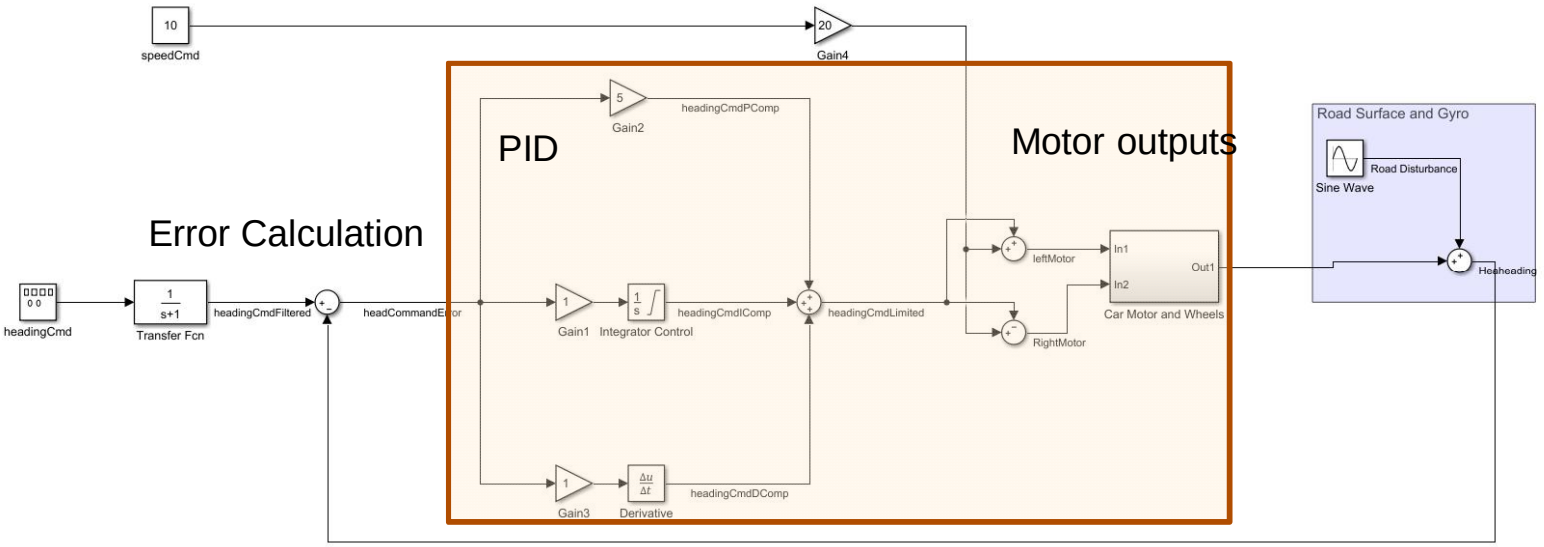


P accounts for present values of the error. For example, if the error is large and positive, the control output will also be large and positive.

I accounts for past values of the error. For example, if the voltage output is not sufficiently strong, the integral of the error will accumulate over time, and the controller will respond by applying a stronger action

D accounts for possible future trends of the error, based on its current rate of change

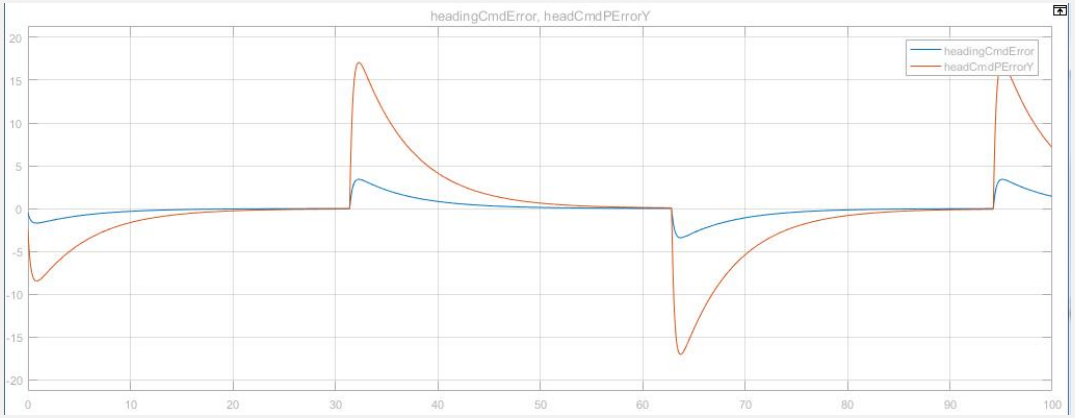
Closed Loop Control Model



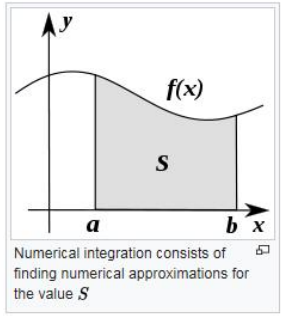
Gyroscope (IMU) heading value

Proportional Implementation

```
// Proportional Compensation Formula K=Gain T=Period or Interval
// y(n) = x(n) * K ;  K=5

// Code
headCmdPErrorY = headCmdErrorProportionalGain * headCmdError;
```

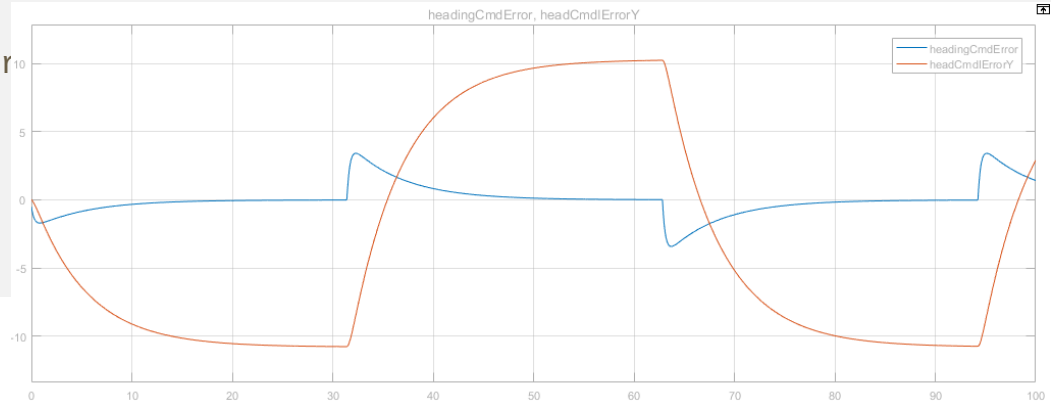


Integrator Implementation



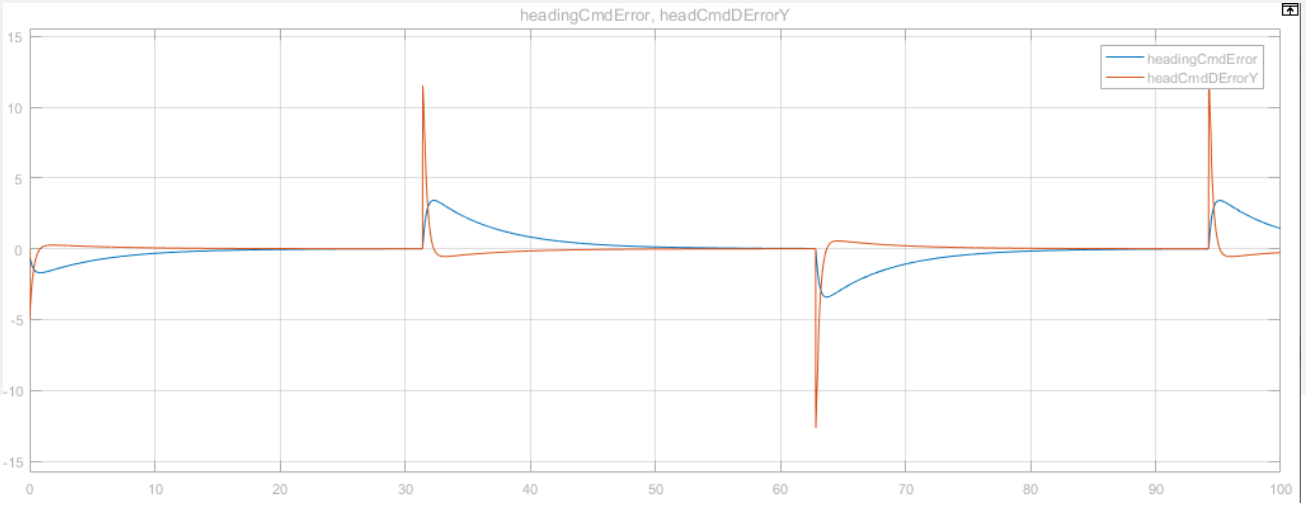
Numerical integration consists of finding numerical approximations for the value S

```
// Trapezoidal Integrator Formula K=Gain T=Period or Interval
// x(n) = y(n-1) + K*T/2 * u(n-1)  K=1  T=.04sec
// y(n) = x(n) + K*T/2*u(n)

headCmdErrorIntegralGain = 1;
headCmdErrorX = headCmdErrorYpv + (headCmdErrorIntegralGain*(Period/2) * headCmdErrorUpv);
headCmdErrorUpv = headCmdError;
headCmdErrorY = headCmdErrorX + (headCmdErrorIntegralGain*(Period/2)*headCmdError);
headCmdErrorYpv = headCmdErrorY;

// Optional Limit on the Past Value to pr
 if ( headCmdErrorYpv > 250 ) {
             headCmdErrorYpv = 250;}
 else if (headCmdErrorYpv < -250 ) {
             headCmdErrorYpv = -250;
 }
```



Derivative Implementation

```
// Derivative Compensation Formula K=Gain T=Period or Interval
// y(n) = K*x(n)/T – K*x(n-1)/T;  K=0

headCmdDErrorY = (headCmdErrorDerivativeGain * headCmdError / Period )-
(headCmdErrorDerivativeGain * headCmdDErrorXpv / Period );
headCmdDErrorXpv = headCmdError;
```



Summary

Basics of Closed Loop Control

- Closed Loop Control Defined
- Demo of Closed Loop Control on heading.
- PID Explained
- PID implementation in Code

MAKE

GIVE

TOOL UP

SHARE

LEARN

Thank You

SUPPORT

PARTICIPATE

CHANGE

PLAY