6.270 Lecture 3

Control Systems

Scott Bezek
January 2011

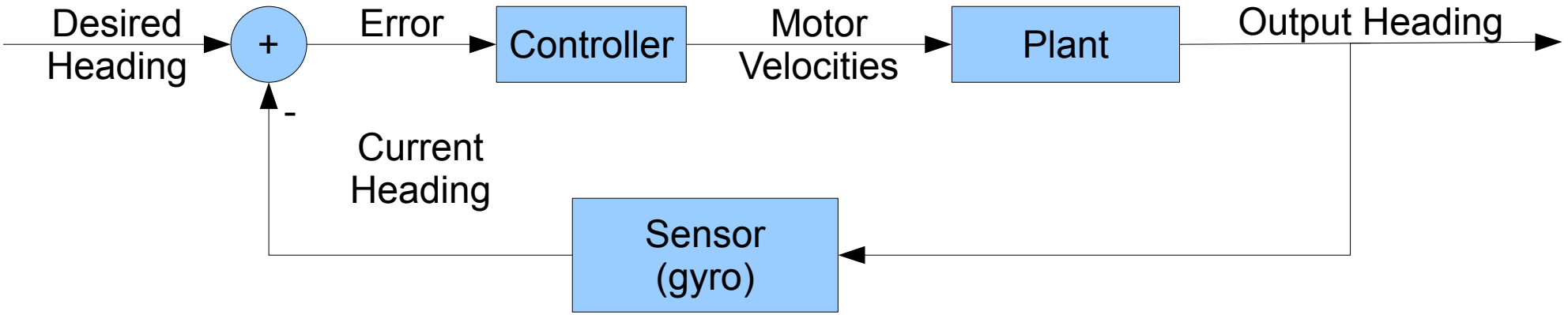
What we saw yesterday...

- Drive “straight”:
  - int umain() {
  - while(1) { //loop forever
  - if (gyro_get_degrees() > 45) {
  - motor_set_vel(0, 150);
  - motor_set_vel(1, 50);
  - } else {
  - motor_set_vel(0, 50);
  - motor_set_vel(1, 150);
  - }
  - }
  - }

We can do better than that

- With binary feedback:
  - Oversteering
  - Jerky
  - Not how you would drive a car
- What if we adjust based on amount of error?
  - Larger error → larger adjustment
  - Proportional Control!

Proportional Control



Proportional Control

- Let's write a proportional controller!
- Demo!

Proportional Control

```
#include <joyos.h>

//returns a bounded velocity
int16_t limitVel(float vel){
  if (vel < 0) return 0;
  if (vel > 255) return 255;
  return (int16_t)vel;
}

int umain(){
  int16_t forwardVel = 150;
  float desiredHeading = 45.0;

  while(true){     //loop forever
    float gain = frob_read_range(5, 50)/10.0;  //gain is configured by frob knob

    float error = desiredHeading - gyro_get_degrees();   //calculate heading error

    //Use heading error and gain to calculate motor velocities
    float leftVel  = forwardVel - error * gain;
    float rightVel = forwardVel + error * gain;

    motor_set_vel(0, limitVel(leftVel ));
    motor_set_vel(1, limitVel(rightVel));
  }
  return 0;
}

int usetup(){
  gyro_init (11, 1400000, 500L);
  return 0;
}
```

Some notes about the gyro

- gyro_get_degrees() gives absolute heading with reference to starting position
- i.e. if you rotate CCW twice, gyro_get_degrees() returns 720
- Probably want helper function to calculate heading error better
  - e.g. take heading mod 360
  - e.g. error should never be > 180 or < -180
- Calibrate it before using!

Calibrating the gyro

- Change Makefile:
  USERSRC = user/gyrotest/umain.c
- “make clean”
- “make program”
- Set robot on spinny chair (make sure it's parallel to the ground)
- Unplug USB cable, reboot HappyBoard, press Go
- Spin chair 10 revolutions at a moderate speed
- Plug in USB cable and open serial terminal
- Should see “theta = 3723” for example
- Divide by 3600 and multiply by LSB_US_PER_DEG to get new value for LSB_US_PER_DEG
- Update user/gyrotest/umain.c, reprogram HappyBoard, repeat – should get theta closer to 3600 this time

Problems with Proportional Control

- Bias – never reach desired value
- Oscillations

PID Control

- Proportional
  - Handles majority of correction
- Integral
  - Adjusts output based on magnitude and **duration** of error
  - Can reduce bias
- Derivative
  - Adjusts output based on rate of change of error
  - Slows down controller output changes
  - Can reduce amount of overshooting

Tuning PID Controller

- More complicated than proportional: 3 parameters
- See http://en.wikipedia.org/wiki/PID_controller#Loop_tuning for several tuning methods

Some ideas for driving

- Consider using multiple controllers
  - Heading controller (rotational velocity)
  - Distance controller (forward velocity)
- Update the desired heading as you drive
  - This will be covered tomorrow
- Can robot drive backwards? → maximum heading error is 90 degrees

Upcoming Events

- Another soldering workshop @ 3pm (if you missed yesterday's)
- Part 2 of C Crash Course – HappyBoard-specific and advanced topics (threading, etc) @ 7pm
- Localization and Navigation Lecture: tomorrow @ 11am

- Lab is open – work on your robots!
- Make sure your development environment is set up – we have rental HappyBoards in lab – see a TA