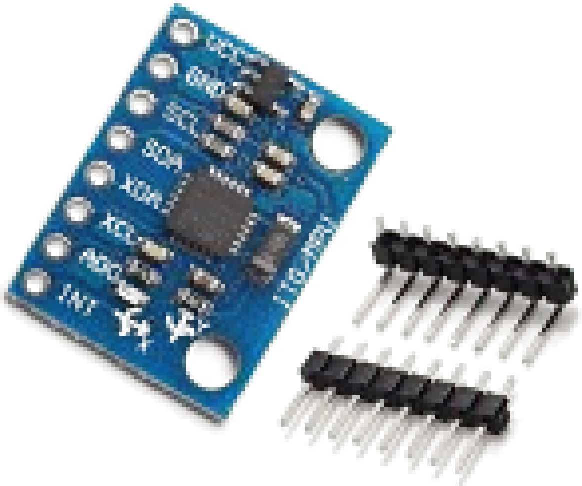
Next few sessions

- Today
  - Gyro
  - Construction
  - PID (Proportional)
- 19 Nov to 20 Dec
  - Prep for robot construction
- 21, 22, 23 Dec
  - Wireless
  - Pressure sensor
  - PID (Proportional + Derivative)
  - Underwater Robot Construction

Plan for today

- Gyro
  - Solder & Code
- Construct 2-Wheel Robot
  - Keep it simple
- PID
  - Proportional
- Preparations between now and 21 Dec

Gyro

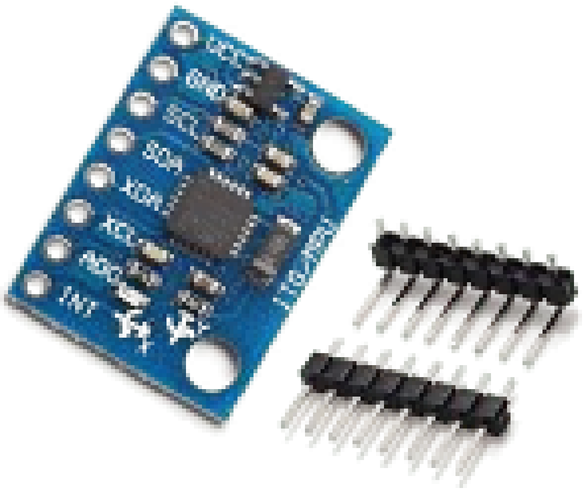


- GY 521 <= Module name
- MPU-6050 <= Chip name
- Contains:
  - 3-axis Accelerometer (...not using)
  - 3-axis Rate Gyro
- Common and cheap (~$2)
- Communicates over I2C
- You have to solder the pins yourself

BIG Three

Serial (aka UART)	• Common for: Computers, Bluetooth, GPS • Full duplex (send and receive simultaneously) • Two wires (excluding ground) • One-to-One
I2C (Inter-Integrated Circuit)	• Common for: Sensors (eg. gyro) • Half duplex (send or receive, not simultaneously) • Two wires (excluding ground) • One-to-Many
SPI (Serial Peripheral Interface)	• Common for: Sensors (eg. rfid reader) • Full duplex (send and receive simultaneously) • Three wires (excluding ground) • One-to-Many (...but require one extra wire per device)

Gyro



- Pins
  - VCC : 5V or 3.3V
  - GND : GND
  - SCL : Serial Clock
  - SDA : Serial Data
  - XDA, XCL, AD0, INT : Ignore. They have special uses which we do not need.

Code

- See “Gyro Basic” on a9i.sg/vjc for full code

```
#include<Wire.h>

const int MPU=0x68;

int16_t GyX, GyY, GyZ;

void setup(){
  Wire.begin();
  Wire.beginTransmission(MPU);
  Wire.write(0x6B); // Register 107, Power Management
  Wire.write(0);    // Disable sleep
  Wire.endTransmission(true);  // Release bus after transmission
  Serial.begin(9600);

  delay(1000);
}
```

Code

- See “Gyro Basic” on a9i.sg/vjc for full code

```
void loop(){
  readGyro();

  Serial.print("Gyroscope: ");
  Serial.print("X = ");
  Serial.print(GyX);
  Serial.print(" | Y = ");
  Serial.print(GyY);
  Serial.print(" | Z = ");
  Serial.println(GyZ);

  delay(500);
}
```

Code

- See “Gyro Basic” on a9i.sg/vjc for full code

```
void readGyro() {
  Wire.beginTransmission(MPU);
  Wire.write(0x43);  // Register 67, Gyroscope measurement
  Wire.endTransmission(false);
  Wire.requestFrom(MPU,6,true);

  GyX = (Wire.read()<<8|Wire.read());
  GyY = (Wire.read()<<8|Wire.read());
  GyZ = (Wire.read()<<8|Wire.read());
}
```

Gyro

- Solder…
- Wire…
- Test out code
- What do you see printed on the console?
  - Try to make sense of it

Gyro Zero Error

- This is a **rate** gyro; it reports rate of turn...
- ...but getting non-zero reading, even when gyro is stationary
- How to solve?
  - Same as with any other instruments:
    - Determine zero error (Make multiple measurements and find average for better accuracy)
    - Subtract zero error from readings

Code

- See "Gyro Intermediate" on a9i.sg/vjc

```
void calibrateGyro() {
  int32_t GyX_total=0, GyY_total=0, GyZ_total=0;
  int counts = 80;

  for (int a=0; a<counts; a++) {
    readGyro();
    GyX_total += GyX;
    GyY_total += GyY;
    GyZ_total += GyZ;
    delay(50);
  }

  GyX_error = GyX_total / counts;
  GyY_error = GyY_total / counts;
  GyZ_error = GyZ_total / counts;
}
```

Code

- See “Gyro Intermediate” on a9i.sg/vjc

```
int16_t GyX, GyY, GyZ;
int16_t GyX_error=0, GyY_error=0, GyZ_error=0;

void readGyro() {
  Wire.beginTransmission(MPU);
  Wire.write(0x43);
  Wire.endTransmission(false);
  Wire.requestFrom(MPU,6,true);

  GyX = (Wire.read()<<8|Wire.read()) - GyX_error;
  GyY = (Wire.read()<<8|Wire.read()) - GyY_error;
  GyZ = (Wire.read()<<8|Wire.read()) - GyZ_error;
}
```

Code

- See "Gyro Intermediate" on a9i.sg/vjc

```
void setup(){
  Wire.begin();
  Wire.beginTransmission(MPU);
  Wire.write(0x6B);
  Wire.write(0);
  Wire.endTransmission(true);
  Serial.begin(9600);

  delay(1000);
  calibrateGyro();
}
```

Gyro

- Test it out again after correcting for zero error
- Readings should be close to zero when gyro is stationary

Heading

- Rate of turn is nice to have, but we really want to know the direction the robot is facing
- To convert rate (eg. degrees/s, m/s)…
- ...into total change (eg. Angle, displacement)…
- ...we need to integrate (...from calculus)
- Integration just means to add together small changes… easy to do in code

Heading

- How to get heading?
  - Read rate of turn (degrees/s)
  - Multiply by the time since last reading (s)
  - Get change of angle (degrees)
  - Add up change of angle
  - Done!
  - (Somewhere along the line, you’ll also need to convert the units into degrees)

Code

- See “Gyro Complete” on a9i.sg/vjc

```
void updateHeading() {
  static unsigned long lastTime = 0;
  unsigned long currentTime;
  unsigned long duration;

  currentTime = millis();

  if (lastTime == 0) {
    lastTime = currentTime;
    return;
  }

  duration = currentTime - lastTime;
  lastTime = currentTime;

  HeadingX += (GyX / VALUE_PER_DEGREE) * (duration / 1000.0);
  HeadingY += (GyY / VALUE_PER_DEGREE) * (duration / 1000.0);
  HeadingZ += (GyZ / VALUE_PER_DEGREE) * (duration / 1000.0);
}
```

Code

- See “Gyro Complete” on a9i.sg/vjc

```
void loop(){
  readGyro();
  updateHeading();

  count++;
  if (count % 40 == 0) {
    Serial.print("Gyroscope: ");
    Serial.print("HX = "); Serial.print(HeadingX);
    Serial.print(" | HY = "); Serial.print(HeadingY);
    Serial.print(" | HZ = "); Serial.println(HeadingZ);
  }
  delay(READ_DELAY);
}
```

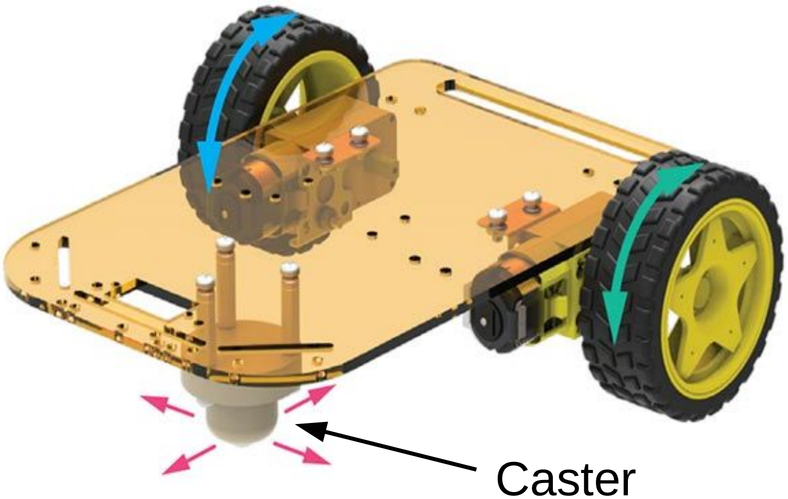
Completed Gyro Code

- On power-up or reset; Spend 4 seconds calibrating zero error. (Make sure gyro is perfectly still during this time!)
- Prints heading every 1 second
- Heading should stay at zero if gyro is not moving
- Heading should change to 90 if you turn it by 90 degrees

Improving Gyro Code

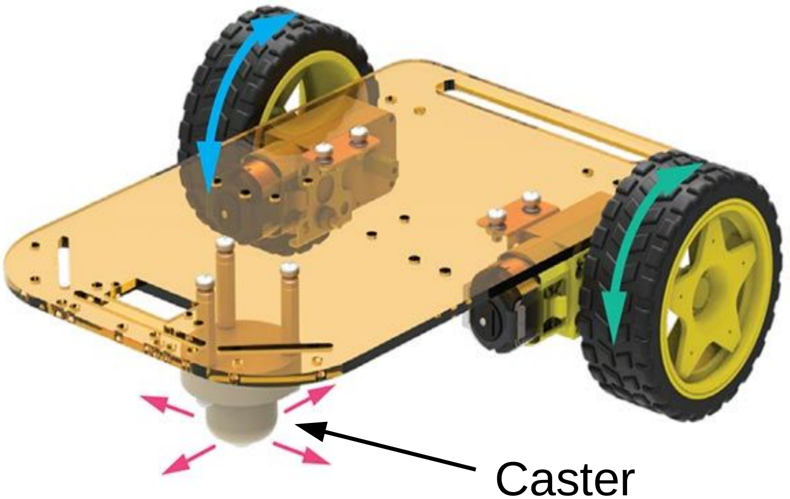
- What we are doing is a form of numerical integration
- We did it in a very “crude” way; read up on other methods of numerical integration
- We can also improve accuracy by decreasing the delay between reads (improve resolution)
- We can also get a more accurate timing using “micros()” instead of “millis()”

Construct 2-Wheeled Robot



- AKA Differential Drive Robot
- Usually have a caster, but we're skipping this to keep it simple
- Easy to control, and movements are easy to model mathematically
- Interesting to study its motion, but let's leave that for another time

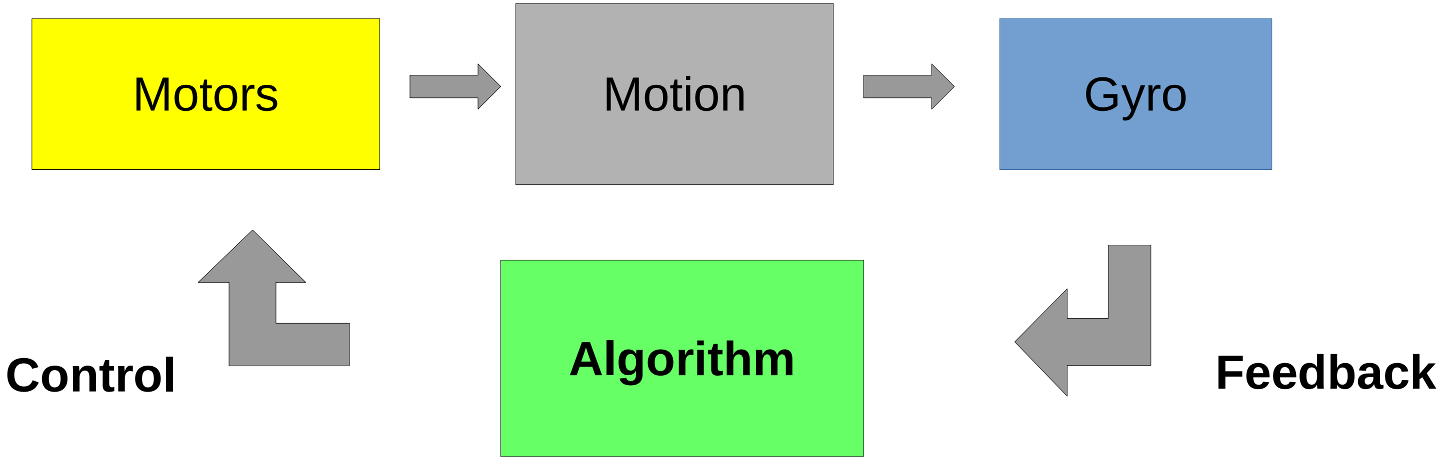
Construct 2-Wheeled Robot



- Mount the motors on with a bit of hot glue
- Keep center of gravity slightl behind wheels
- Make sure gyro is not dangling loosely
- Be quick. We are not using this robot after today. Don't bother making it pretty.

Feedback Control

- “Real Robots Don’t Drive Straight”
  (Fred G. Martin) (2007 AAAI Spring Symposium)
- Make use of feedback to control movement

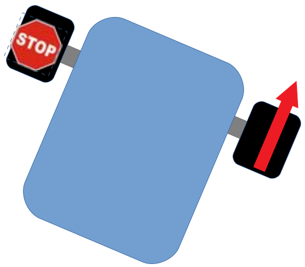


Feedback Control

- Simplest algorithm...

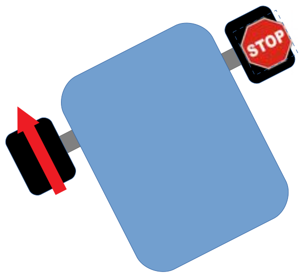
```
if heading > 0:
  // Turn Left
  Right_Motor_Power = 100
  Left_Motor_Power = 0

else:
  // Turn Right
  Right_Motor_Power = 0
  Left_Motor_Power = 100
```



Heading > 0

Heading ≤ 0

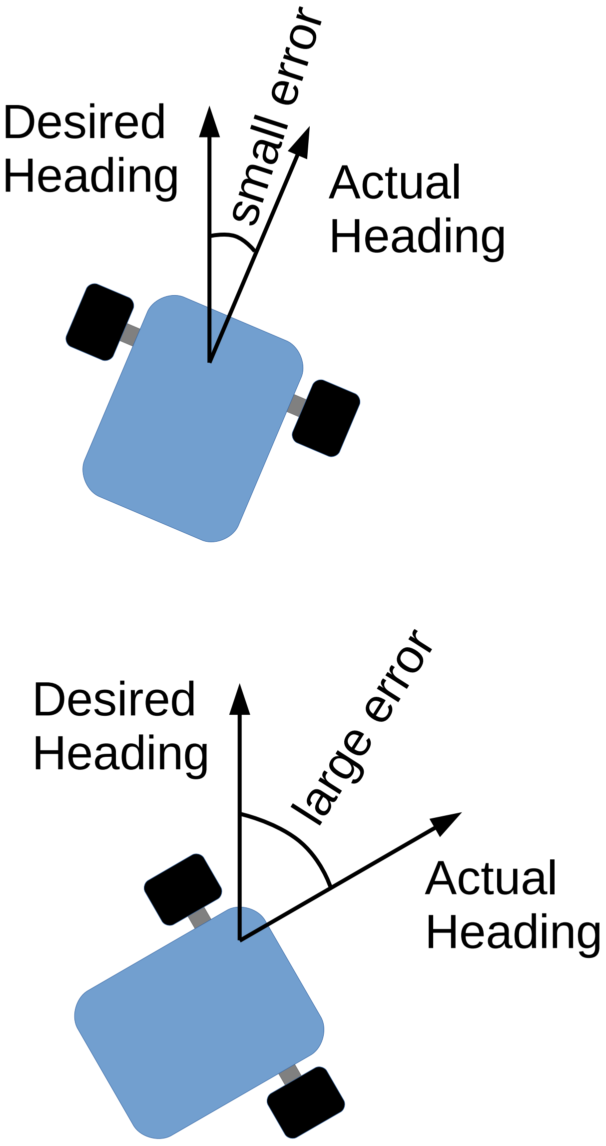


- Two states algorithm; Turn left or Turn right

Error

Difference between what you want and what you have

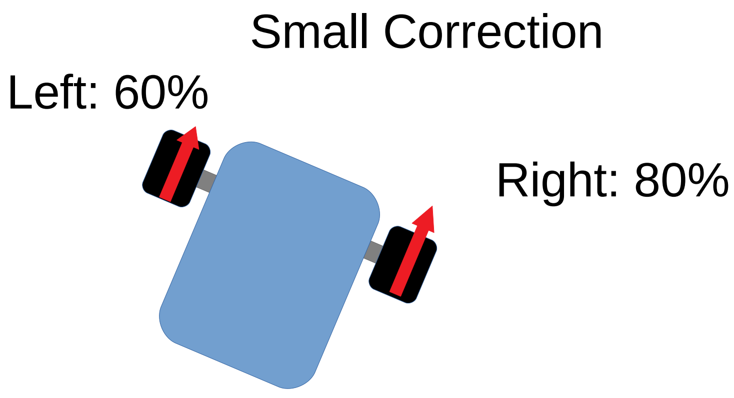
Error = Actual - Desired



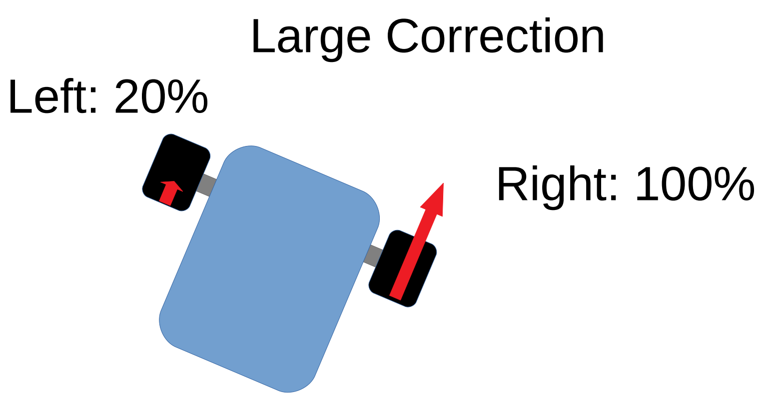
Feedback Control

Correction

Adjustment made to try and reduce error.



Here we are applying a difference between left and right motor...



...but it can also be a turn in rudder, pivoting of wheels, etc

Proportional Control

- Relationship between error and correction?
- Proportional Control
  - Correction is proportional to error
  - Big error => Big correction
  - Small error => Small correction

Correction = Kp * Error

Kp : Gain Constant

Proportional Control

Examples:

Correction = 2 * Error   (Kp = 2)

When error is 2 degrees, the right wheel will move 4% faster and the left wheel 4% slower

Correction = -3 * Error   (Kp = -3)

When error is 2 degrees, the right wheel will move 6% slower and the left wheel 6% faster

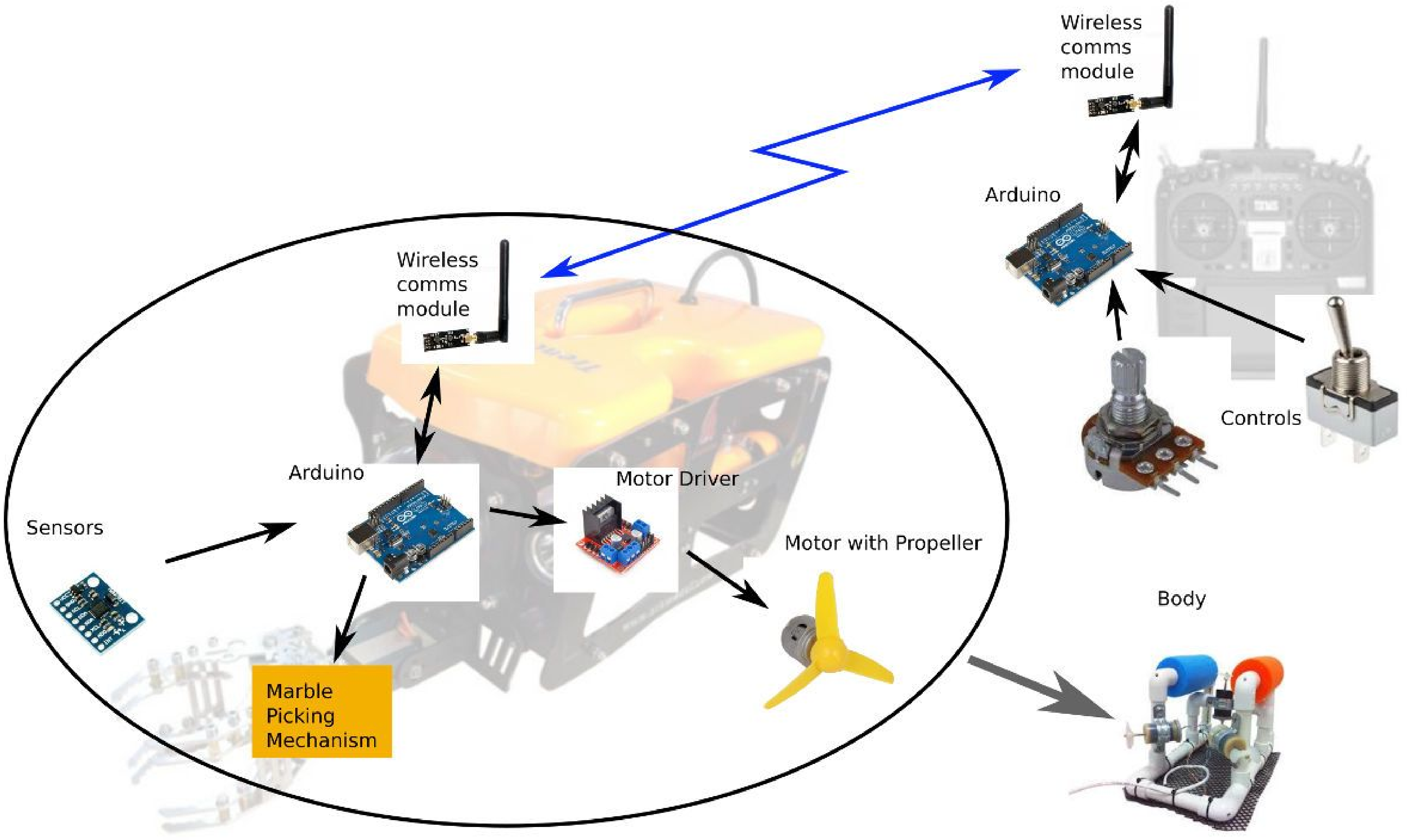
Proportional Control

- You already know how to...
  - Control motors
  - Get heading from Gyro
- Now apply the proportional control formula to make your robot drive straight
- I could give you the answer, but you'll learn more if you figure it out yourself. Ask if you need help.

Preparations (19 Nov – 20 Dec)

- Arrange your own meetings
- Plan design
- Buy parts
- Learn 3D modeling
- Optional
  - Experiment with pressure sensor and wireless
  - Start fabricating underwater robot

Underwater Robot Design



Design Questions to Consider

- How should the robot move?
  - It doesn’t have to be the same as everyone else!
- What is the control scheme?
- How can I make adjustments to...
  - Motor position
  - Buoyancy
  - Center of gravity
  - Center of buoyancy

Available Parts

- What I can provide...
  - Motors (R280)
  - Film canister
  - Propellers
- What you already have...
  - Arduino
  - H-Bridge
  - Gyro
  - Wireless and pressure sensor (...in club room)

Parts to Buy / Prepare

- Shared
  - Marbles (buy)
  - Marbles holder* (...can 3D print or fabricate from plastic sheets)
  - Window frame*
  - Basket area*

* See competition rules document for details

Parts to Buy / Prepare

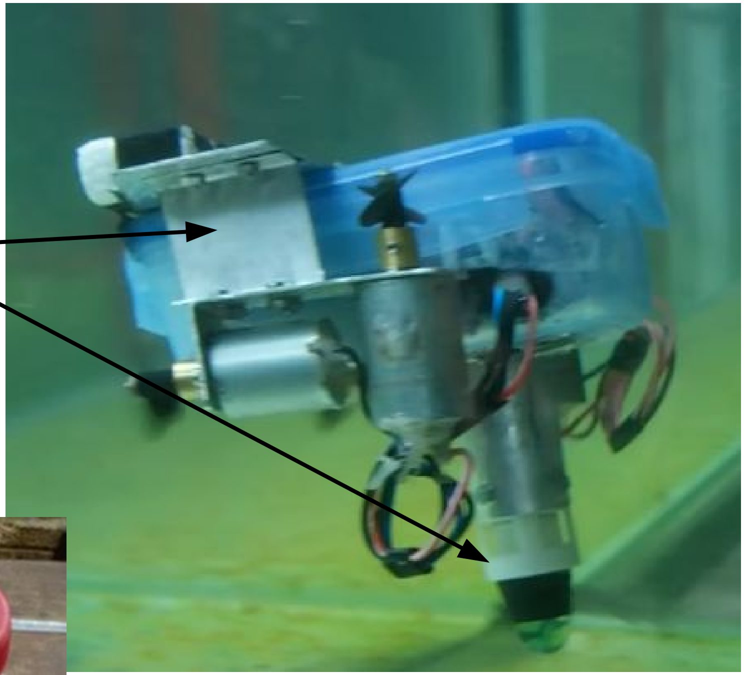
- Team
  - Robot body (...probably a food storage container)
  - Marble picking mechanism (...plan it out and buy what you need)
  - Ballast (...your robot will probably be too buoyant to submerge; try adding fishing weights, nuts, bolts)
  - Battery holders? (9V batteries are very weak)
  - Different motors (...if you don't want to use the R280)

Learn 3D Modeling

Free Options

- Onshape (https://www.onshape.com)
  - Professional Quality. Try the online tutorials.
- Tinkercad (https://www.tinkercad.com/)
  - Easy to use, targeted at kids. Some people have made amazing designs here.
- FreeCAD (https://www.freecadweb.org/)
  - Not the most polished. Hard to start.
- Many others...
  - SketchUp, DesignSpark, Fusion360, OpenSCAD

Why 3D Print?



You can 3D print these!

You can 3D print these!

You can 3D print these!

3D Modeling / Printing Tips

- Keep it simple
- Keep it small (...printing is really slow)
  - 10cm is ok
  - 15cm is questionable
  - 20cm is (...probably) a bad idea
- Combine it with other materials
  - eg. Use acrylic sheets for long straight sections, and 3D printed parts for the brackets

Contact

- Email me at any time if you need advice:
- cort@aposteriori.com.sg

Copyright

- Created by A Posteriori LLP
- Visit http://aposteriori.com.sg/ for more tips and tutorials