Motion Control With PID Loop

Lucia Cipolina Kun[1] and Rick Anderson [2]

December 12, 2019

[1]Fubar Labs and University of Bristol
[2]Fubar Labs and Rutgers University

Agenda

Problem Definition: path control

How to Achieve Lateral Control: the Yaw angle

The PID Controller

Implementing a PID Controller

Sensor Measurement

Sensor Fusion

IMU Sensors

Demo

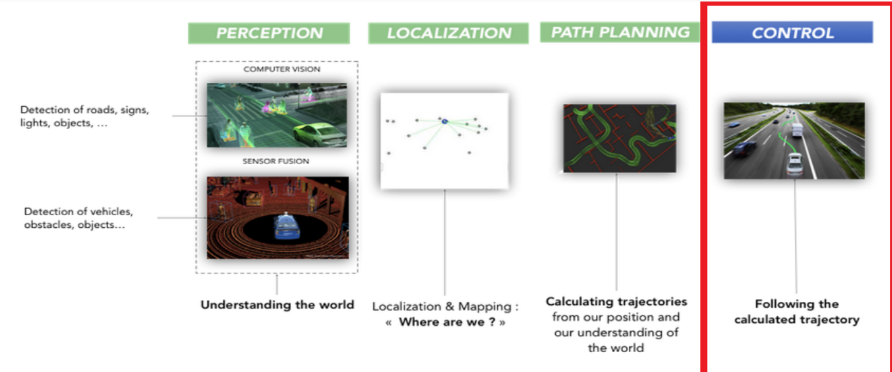
Questions

Disclaimer

The opinions expressed on this presentation are solely those of the authors and not necessarily those of their employers.

Contents

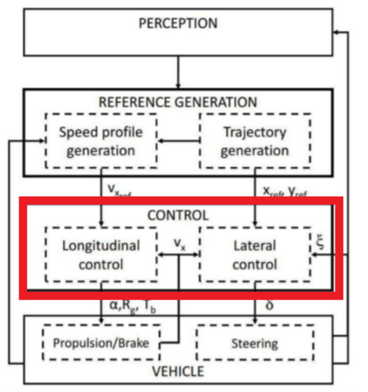
The **control module** is in charge of moving the vehicle by calculating two variables: an **angle** for the steering wheel and the **acceleration** value for each wheel.



How self-driving cars work

What types of control do we need?

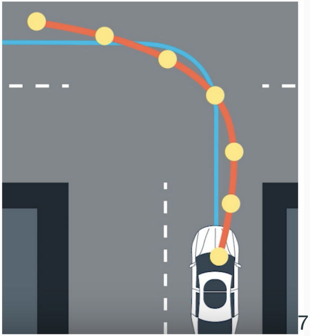
- **Longitudinal control**: control throttle to achieve desired speed.
- **Lateral control**: control angular rotation (yaw) to achieve desired trajectory.



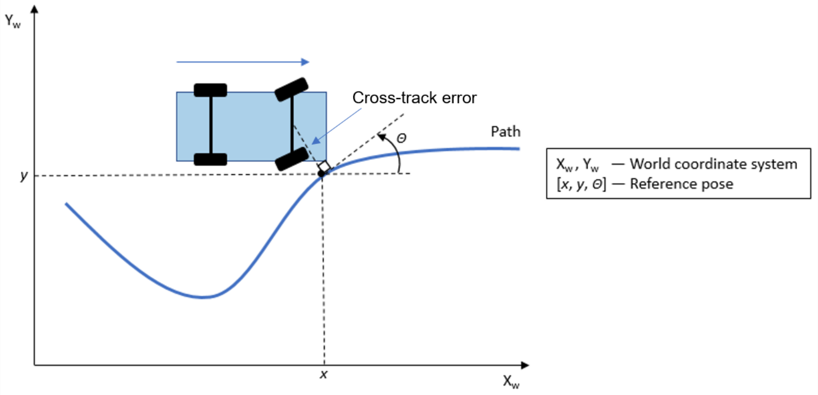
Source:Longitudinal Control for Automated Vehicle Guidance. Rachid ATTIA, Rodolfo ORJUELA and Michel BASSET

Lateral Control: achieving desired path

- Cross-track error (CTE) measures the distance from a location to the route.
- Even when the motors are set to go the same speed (i.e. we are giving them the same power), the car does not run straight. The turn results from two motors moving at different speeds.
- Various factors in the manufacturing process cause different amounts of energy to be lost to friction in each motor.
- The motor's power levels should be adjusted so that the wheels move at the same speed rather than just being driven with the same power



Lateral Control: Cross Track Error



Source:https://www.mathworks.com/help/driving/examples/lateral-control-tutorial.html

Lateral Control: achieving desired path - cont

- The algorithm first determines the difference between the commanded speed and the actual speed. This is the error we try to minimize.
- The control value is continuously updated based on the response of the motors. This ensures that the motors are moving at the desired speed despite drag, obstacles, or other unexpected track conditions.

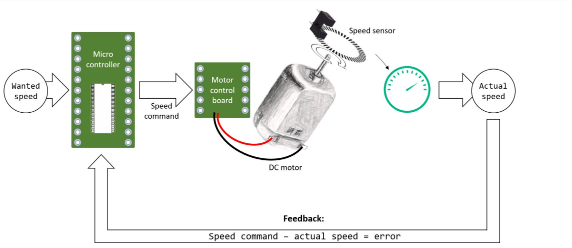


Figure 1:
Source:https://medium.com/luosrobotics/an-introduction-to-pid-control-with-dc-motor-1fa3b26ec661

Lateral Control: achieving desired path - cont

Lateral Control

- Cross track error:

$$e = \frac{ax_c + by_c + c}{\sqrt{a^2 + b^2}}$$

- Cross track steering:

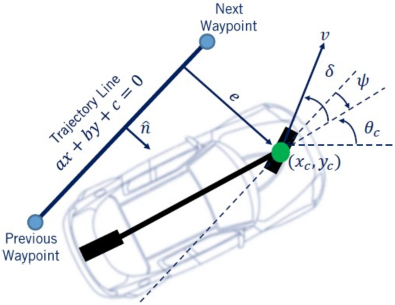
$$\tan^{-1}\left(\frac{ke}{v}\right)$$

- Heading error:

$$\psi = \tan^{-1}\left(\frac{-a}{b}\right) - \theta_c$$

- Total steering input:

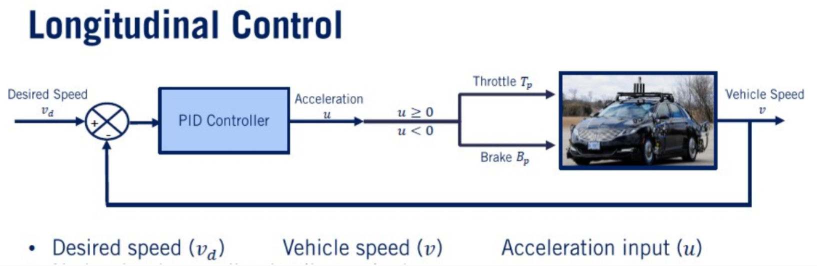
$$\delta = \psi + \tan^{-1}\left(\frac{ke}{v}\right)$$



Source:https://github.com/enginBozkurt/SelfDrivingCarsControlDesign

Longitudinal Control: achieve desired speed

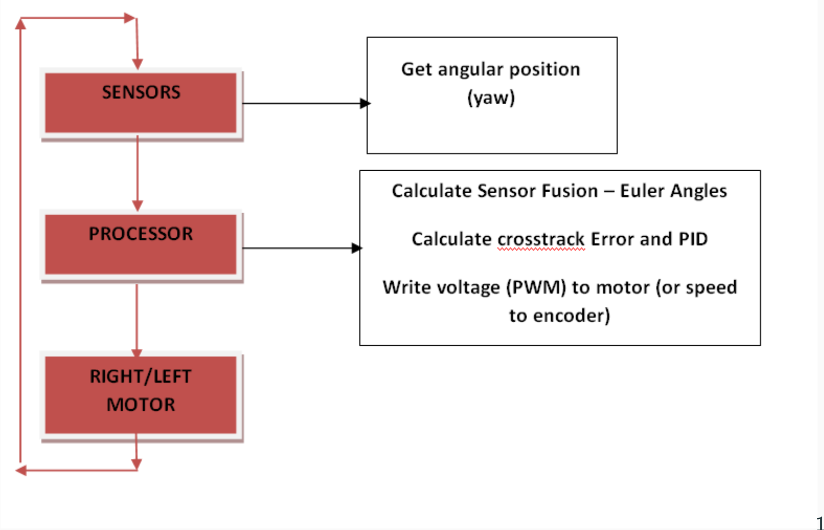
In this case, the car needs to achieve a reference speed using throttle and brake commands. When the vehicle is subjected to different loads and resistances (for example changing steepness of the road), the acceleration should be changed by the cruise controller accordingly.



Source:https://github.com/enginBozkurt/SelfDrivingCarsControlDesign

Achieving Control: the Big Picture

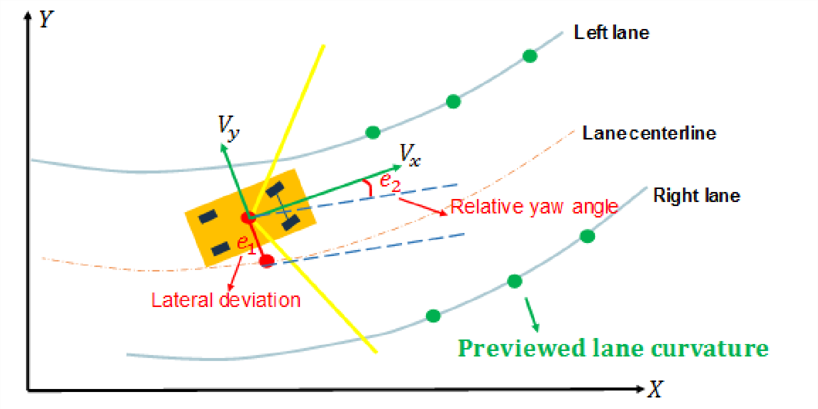
The main steps to achieve motion control are as follows:



Contents

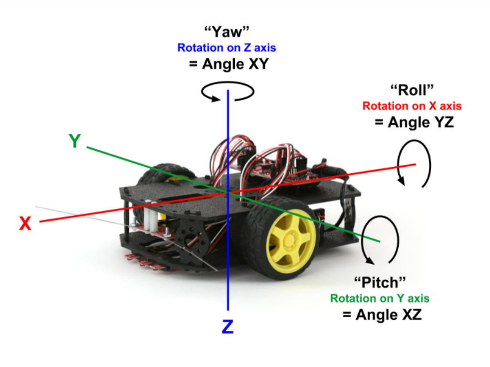
How to Achieve Lateral Control: the Yaw Angle

In terms of *Euler angles* the rotation over the vertical axis (i.e. left-to-right) is called "yaw".



Lateral control with the Yaw Angle

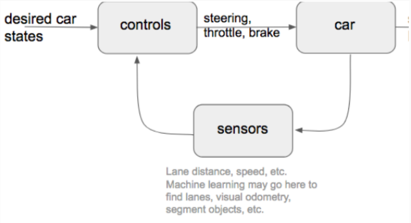
- There are several ways to represent rotations: Euler angles, Tait-Bryan angles, quaternions and rotational matrices. Each system has its own advantages and disadvantages.
- Note that when the car is on a level surface, the yaw value cannot be accurately determined because the acceleration due to Earth's gravity is acting in the same direction (i.e. downward) as the Z axis.



Contents

How do we achieve control? The PID controller

- A control loop feedback system is a system that runs in a close loop, with a desired set-point to reach.
- The command given to the actuator to reach the desired set-point depends on the feedback error. The feedback is the difference between the sensor value (called process value) and the target value (set-point).
- The resulting error is computed and re-injected into the initial order as a command that automatically corrects and adjusts the value of the actuator in order to reach the set-point.



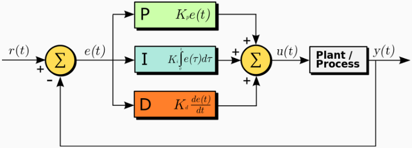
PID Controllers: The Theory

- The PID (Proportional Integral and Derivative) controller minimizes the error rate $e(t)$ over time by applying a dynamic adjustment to the controlled variable (the steering angle or the target speed).
- The correction value $u(t)$ is calculated using 3 terms: the *Proportional*, *Integral* and *Derivative* coefficients applied to the location error $e(t)$:

$$\boxed{u(t) = K_\mathrm{p} e(t) + K_\mathrm{i} \int_0^t e(t')\,dt' + K_\mathrm{d} \frac{de(t)}{dt}} \tag{1}$$

where K_p, K_i and K_d have to be calibrated.

The algorithm works in a loop:



PID Controllers - Interpretation of Parameters

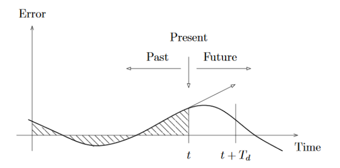
A PID controller takes control action based on past, present and prediction of future control errors.

$$u = -\Bigg(\underbrace{K_p e_{ct}}_{\substack{\text{Proportional}\\ \text{(current)}}} + \underbrace{K_i \int e_{ct}(t)dt}_{\substack{\text{Integral}\\ \text{(past)}}} + \underbrace{K_d \dot{e}_{ct}}_{\substack{\text{Derivative}\\ \text{(future)}}}\Bigg)$$

Proportional - get rid of the current error!

Integral - if I am accumulating error, try harder!

Derivative - if I am going to overshoot, slow down!

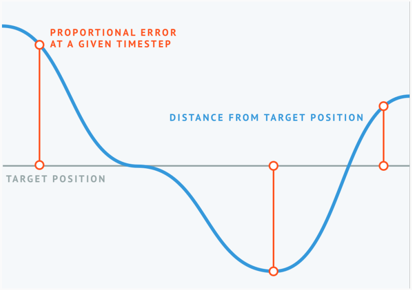


Source: R. M. Murray, Z. Li and S. S. Sastry, A Mathematic
Introduction to Robotic Manipulation

Source:https://courses.cs.washington.edu/courses
/cse490r/19sp/site/resources/lec14$_p id_p ure_p ursuit.pdf$

PID Controllers - Interpretation of Parameters

- The **proportional** term considers "how far" the measured process variable has moved away from the desired set point.
- The correction is applied in the same proportion of the error, but in the opposite direction. As that error value grows or shrinks, the amount added to or subtracted from the error similarly grows or shrinks both immediately and proportionately.
- **Problem:** There's a natural overshooting effect that will cause the car to swivel hard left and right eventually driving the car off-track.

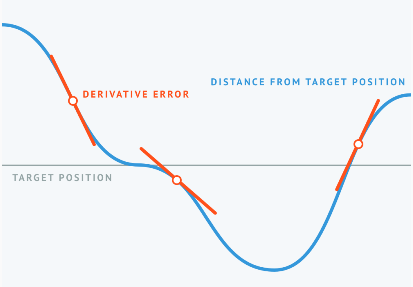


PID Controllers - Interpretation of Parameters -cont

- The **differential** component is the best estimate of the future trend of the error, based on its current rate of change.
- The derivative term considers "how fast" the error value changes at an instant in time. The derivative computation yields a rate of change or slope of the error curve. An error that is changing rapidly yields a large derivative regardless of whether a dynamic event has just begun or if it has been underway for some time.
- A way to cancel the overshoot effect is to introduce a temporal derivative of the *error* term.

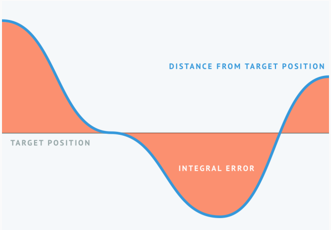
$$\int_0^t e(t')\,dt \simeq \frac{error(t) - error(t-1)}{\delta t} \tag{2}$$

- As the error becomes smaller over time the counter steering won't be as sharp helping the converge the movement to the target trajectory.



PID Controllers - Interpretation of Parameters - cont

- The **integral** term addresses "how long" the measured process variable has been away from the desired set point. It considers all past values of the *error* and it's measured by the integral or the sum of the crosstrack errors over time.
- The reason we need it is that there's likely residual error after applying the proportional control. This ends up causing a bias over a long period of time that avoids the car to get in the exact trajectory.
- This integral term seeks to eliminate this residual error by adding a historic cumulative value of the error.



3 variations

1. *P* **is sometimes used**
2. *PI* **is most often used**
3. *PID* **is sometimes used**

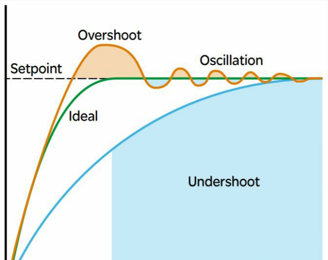
- **P-Only control** — Uses only the proportional term. It is the simplest form of control which makes it the easiest to tune. It provides an initial and rapid kick in response to both disturbances and set point changes, but it is subject to offset.
- **PI Control** — is the most common configuration of the PID controller in industry. It supplies the rapid initial response of a P-Only controller, and it addresses offset that results from P-Only control.
- **PID Control** — This configuration uses the full set of terms and it allows for more aggressive Proportional and Integral terms without introducing overshoot.

PID Tuning approaches

1. **Trial and error**
2. **Tuning algorithms**
3. **Self Tuning**

PID Controllers - Manual Tuning of the Gains

- **Triak and error:** Start with the proportional control first setting the rest to zero. Then tune the rest one by one. and then adjust the rest of the parameters to fine-tune.
- Graph the results. If P is too high, the car becomes too sensitive and tends to over-correct, eventually it will cause overshoots, and you will have high frequency oscillations.
- You can lower P to reduce the oscillations, but reduce it too much and your car will undershoot.

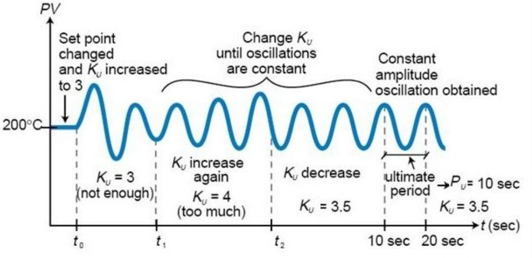


PID Controllers - Tuning Algorithms

The most common algorithms used for PID tuning is the **Ziegler-Nichols**

- The process starts with a proportional-gain only system. Increase the P gain until the system exhibits oscillations that are sustained (i.e. stable in terms of amplitude) and regular (i.e. stable in terms of period); the oscillation does not need to be centered around the setpoint.

Example:



- Once you have sustained, regular oscillation, you record the proportional gain and measure the oscillation period. These values are referred to as the ultimate gain (K_U) and the ultimate period (P_U), respectively.
- Calculate K_P, K_I, and K_D according to the following table:

	K_P	T_I	T_D
P-only control	$K_U/2$		
PI control	$K_U/2.2$	$P_U/1.2$	
PID control	$K_U/1.7$	$P_U/2$	$P_U/8$

- Note that the integral and derivative columns are labeled T_I and T_D instead of K_I and K_D. These T variables refer to the time constant used when calculating the integral or derivative. The following relationships can be used to determine K_I and K_D:

$$K_I = K_p \left(\frac{T}{T_I} \right) \tag{3}$$

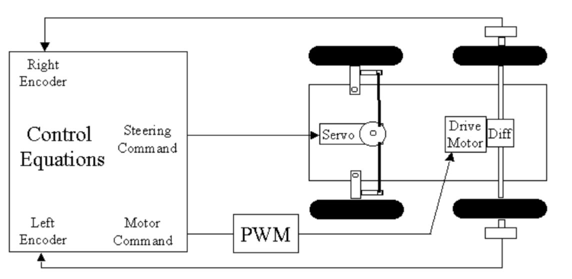
$$K_D = K_p \left(\frac{T_D}{T} \right) \tag{4}$$

- where T is the PID interval, i.e. the amount of time between successive executions of the routine that calculates the PID output and updates the control circuitry.

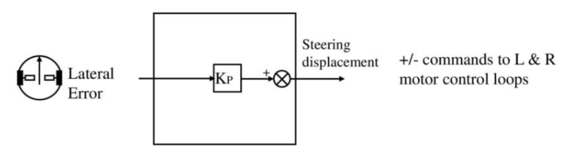
Contents

PID Controllers: The Practice - Steering Control

The steering is done by a servo which goes to the POSITION commanded by the control equations



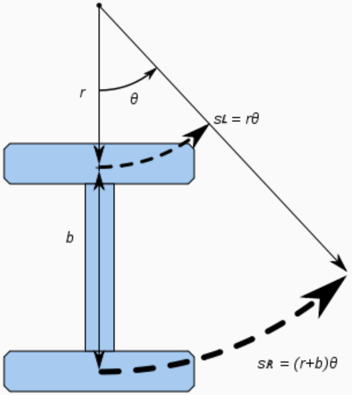
The general loop is as follows:



Steering Controller

- **Input**:
  - Desired Angle (from guidance controller)
  - Actual Angle (from sensors)
- **Output**:
  - The steering angle is the output of the PID equation
  - We correct the speed of both wheels based on the error:
  - Correction term on the left wheel
  - Correction term on the right wheel

Steering Controller

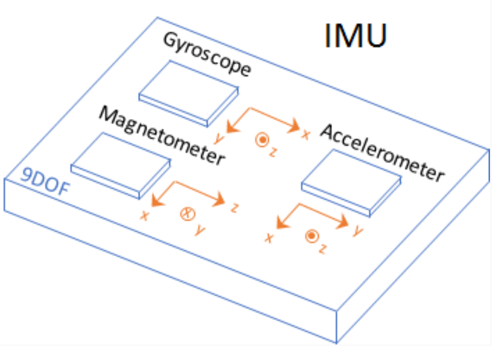


- The initial speed is S_0. We would like to turn θ radians to the left to correct for the error.
- The left wheel needs to travel $S_L = r\theta$ the right wheel needs to travel $S_R = (r + b)\theta$
- To maintain the same overall speed S_0, we need $S_0 = (r + b/2)\theta$, the left wheel will need to travel at $S_L = S_0 - (b/2)\theta$ the right wheel will need to travel at $S_R = S_0 + (b/2)\theta$.

Contents

How to Measure YAW - Intertial Measurement Units

- An IMU is an electronic device mounted on a platform.
- The IMU consists of individual sensors that report various information about the platform's motion.
- IMUs combine multiple sensors, which can include accelerometers, gyroscopes, and magnetometers.



Intertial Measurement Units - Cont

Measurements returned from an IMU model use the following unit and coordinate conventions.

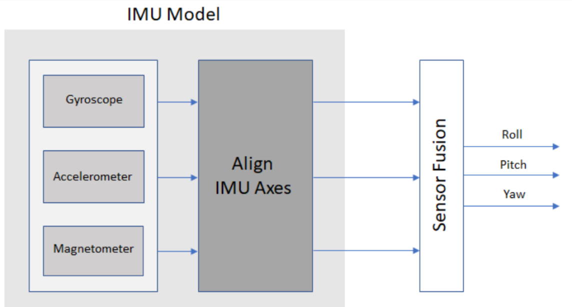
Output	Description	Units
Acceleration	Current accelerometer reading	m/s^2
Angular velocity	Current gyroscope reading	rad/s
Magnetic field	Current magnetometer reading	μT

To correctly measure yaw, we use not only the gyroscope data but also the accelerometer and magnetometer's output. The data returned by IMUs is *fused* together to correct for noise.

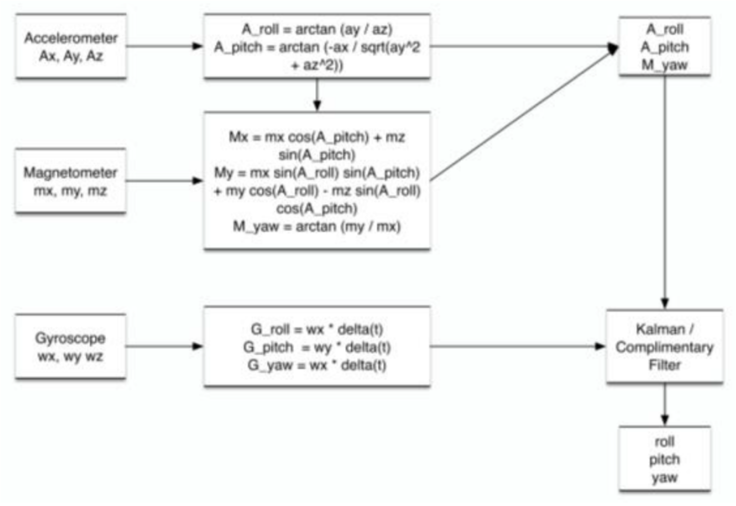
Contents

Sensor Fusion Filters

- Sensor fusion is combining of sensory data to extract one measurement such that the resulting information has less uncertainty than would be possible when these sources were used individually.
- With sensor fusion, the drift from the gyroscopes integration is compensated for by reference vectors, namely gravity, and the earth magnetic field. This results in a drift-free orientation.



The integration of the gyroscope angular speed adds up the sensor noise creating and increasing drift over time.



Contents

IMU Sensors

Choosing an IMU

Principal considerations:

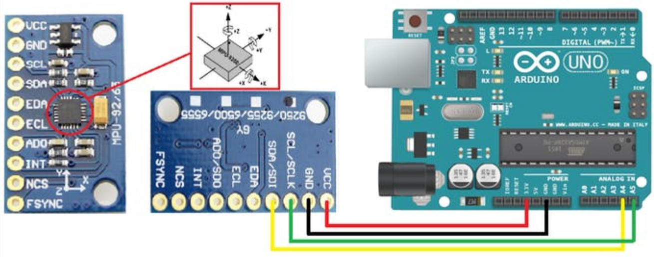
- Price
- Range and resolution
- Degrees of freedom
- Interface (analog/digital, bus type, etc)

Additional considerations:

- Noise distribution
- Power consumption
- Gyro temperature offset

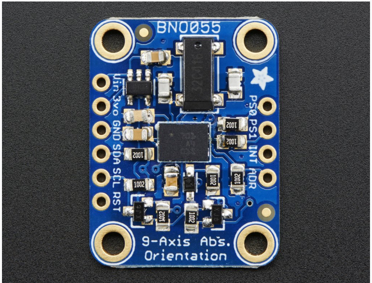
MPU 9250

- This is a 9DOF sensor.
- The communication is done via I2C protocol.
- The output is the "raw" values that have to be processed into yaw-pitch-roll
- cheapest one in the market usd2.5 +shipping on Amazon



Adafruit's BNO055

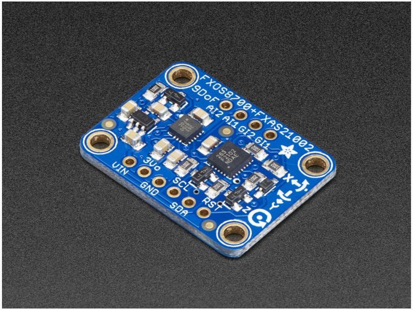
- Comes with the sensor fusion incorporated on the chip.
- Fastest calculation of fused Euler angles since calculations are inside the unit.
- Output can be set to Quaternions or raw data, but calculations are black box.
- Price is on the expensive side: usd35 + shipping



Sensors We Like - Cont

Adafruit’s NXP 9DoF

- Simple 9DOF sensor but comes with a good library.
- The library provided works well for Euler angles
- Price is inexpensive: usd15 + shipping



Contents

Demo Time!

show the IMU sensor with Processing

Contents

Questions ?