"OUR" SENSORS

In this lecture we will describe certain popular powerful sensors.

- RGB camera
- LiDAR (aka Laser scanner) (2D and 3D types)
- Depth Camera (and dual RGB-Depth)
- IMU (Inertial Measurement Unit)
- GPS (actually not covered today)

In addition to the description of those sensors we will see, in real-time, some industrial models working (Depth cameras, 2D and 3D LiDARs, and IMU.

INERTIAL MEASUREMENT UNIT (IMU)

BASIC DEFINITIONS

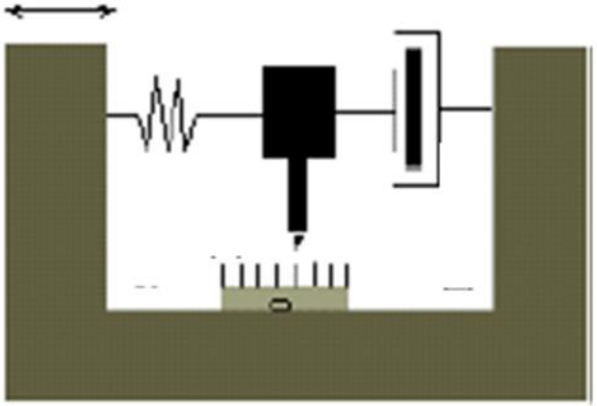
- Navigation
  - Estimate the position, orientation and velocity of a vehicle
- Inertial Navigation
  - Inertial Sensors are utilized for the navigation
- Inertial sensors
  - Based on inertial principles, acceleration and angular velocity are measured.
  - Common inertial sensors
    - Accelerometers
    - Gyros
    - (and Magnetometer)

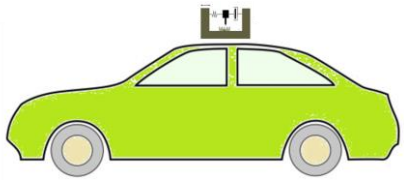
INERTIAL SENSORS

- Accelerometer measures linear acceleration (in m/s2)
- Gyroscope measures angular velocity(degrees/sec)
- Magnetometer measures magnetic field strength (e.g. in micro Tesla or Gauss)

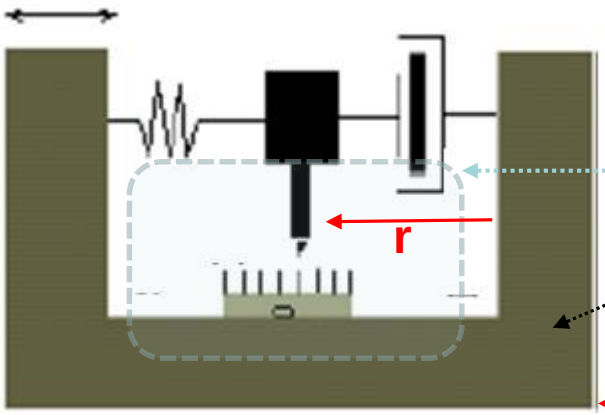
ACCELEROMETERS

Traditional devices were based on mechanical principles: By attaching a mass to a spring and a damper, measuring its relative displacement, we get a simple accelerometer. This is valid for “slow dynamics” (usually called “low frequency dynamics”) , e.g. like that of a car. Or a plane or a helicopter. ….





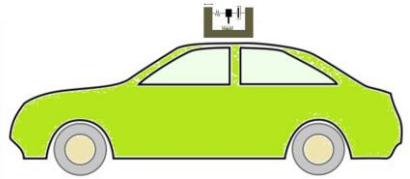
1D Accelerometer



Mass, spring and dumper

Sensor's frame

r: position of the mass relative to sensor frame.



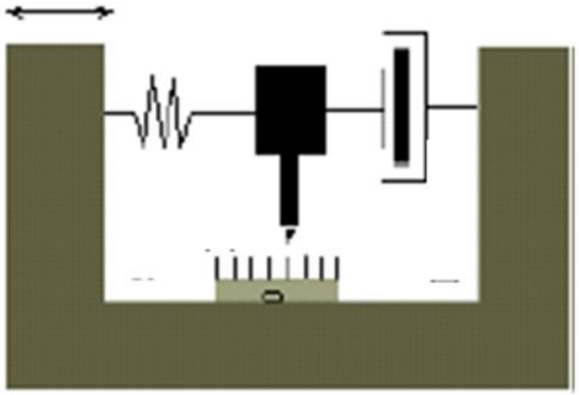
1D displacement, u(t)

u(t) : position of frame, respect to external coordinate frame, static one (*))

The figure shows a 1D accelerometer. Its objective: obtaining $r(t) = k \cdot \ddot{u}(t)$

r(t) is a displacement relative to the sensor frame,. That displacement can be measured.

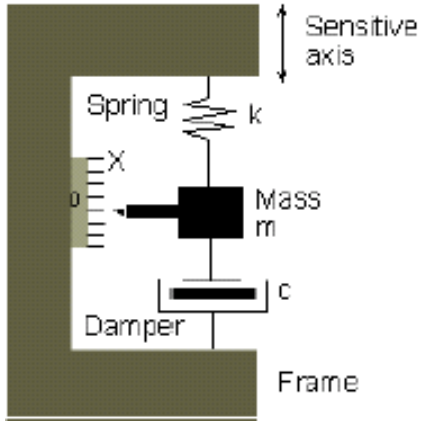
ACCELEROMETERS



The relative position of the mass **m** (respect to the frame), copies the absolute acceleration of the frame!!! (usually, it is proportional to it)

(this can be demonstrated mathematically)

BASIC ACCELEROMETER.



The balance of forces on the mass

$$m \cdot \frac{d^2\left(x+u\right)}{d^2t} + b \cdot \frac{dx}{dt} - k \cdot \left(x_0 - x\right) = 0$$

Where the equation involves the mass of the object, the friction/drag and the elastic constant of the spring, (m, b, k). The constant x0 is the position of the mass at rest. The gravity component must be included in the balance. Now we assume there is no projection of the gravity in order to understand the dynamics of the system.

The variable u(t) (the position of the frame) is forced externally and we assume the sensor does not affect it.

We want to analyze the sensor's response (signal x(t)) due to any possible "input" u(t).

BASIC ACCELEROMETER.

The analysis is simplified in the Laplace or in the Fourier domains

$$m \cdot \frac{d^2\left(x+u\right)}{dt^2} + b \cdot \frac{dx}{dt} - k \cdot \left(x_0 - x\right) = 0$$

$$-\frac{d^2u}{dt^2} = \frac{d^2x}{dt^2} + \frac{b}{m} \cdot \frac{dx}{dt} + \frac{k}{m} \cdot x = \frac{d^2x}{dt^2} + B \cdot \frac{dx}{dt} + C \cdot x$$

$$\Downarrow$$

$$-s^2 \cdot u\left[s\right] = s^2 \cdot x\left[s\right] + B \cdot s \cdot x\left[s\right] + C \cdot x\left[s\right]$$

$$x\left[s\right] = \frac{-s^2}{s^2 + B \cdot s + C} \cdot u\left[s\right]$$

(topic NOT included in the exam)

10

Basic Accelerometer.

That transfer function behaves as a double derivative for low frequency inputs.

$$x[s] = \frac{-s^2}{s^2 + B \cdot s + C} \cdot u[s]$$

$$T[\omega] = \left. \frac{-s^2}{s^2 + B \cdot s + C} \right|_{s = j \cdot \omega} = \frac{\omega^2}{-\omega^2 + B \cdot j\omega + C}$$

$$T[\omega] \to \frac{\omega^2}{C} \quad \forall \omega \to 0$$

What means that for a slowly varying input signal, u(t), the output x(t) is the second derivative of the input, i.e.

$$x(t) = \frac{d^2 u(t)}{dt^2}$$

(this can also be inferred from the BODE plots)

11

Basic (electro-mechanical) Accelerometer

The sensor is intended to be applied at low frequencies (slowly varying inputs u(t) , see note [1]). For high frequencies the output x(y) copies the input u(t). That fact can be inferred from the limit of T[ω] at high values of ω.

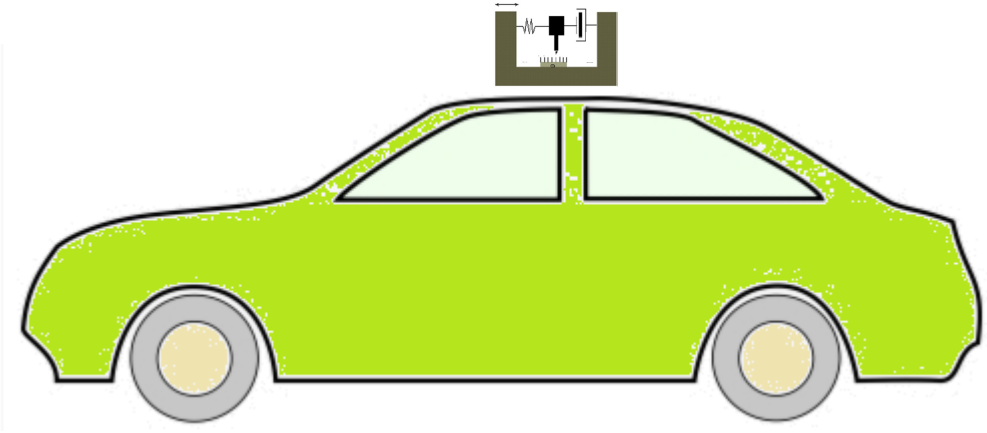
In real applications the inertial sensors involves 3 accelerometers (for axis X,Y,Z). The measurements are affected by the Gravity (as it projects forces in the directions of each axis). In order to “remove” the gravity effect from the measurements we need to know the attitude of the sensor to estimate the contributions of the gravity to the 3 acceleration measurements.

[1] The dynamics of mechanical platforms such as car, truck, helicopter, ship, are LOW FREQUENCY dynamics.

ACCELEROMETERS

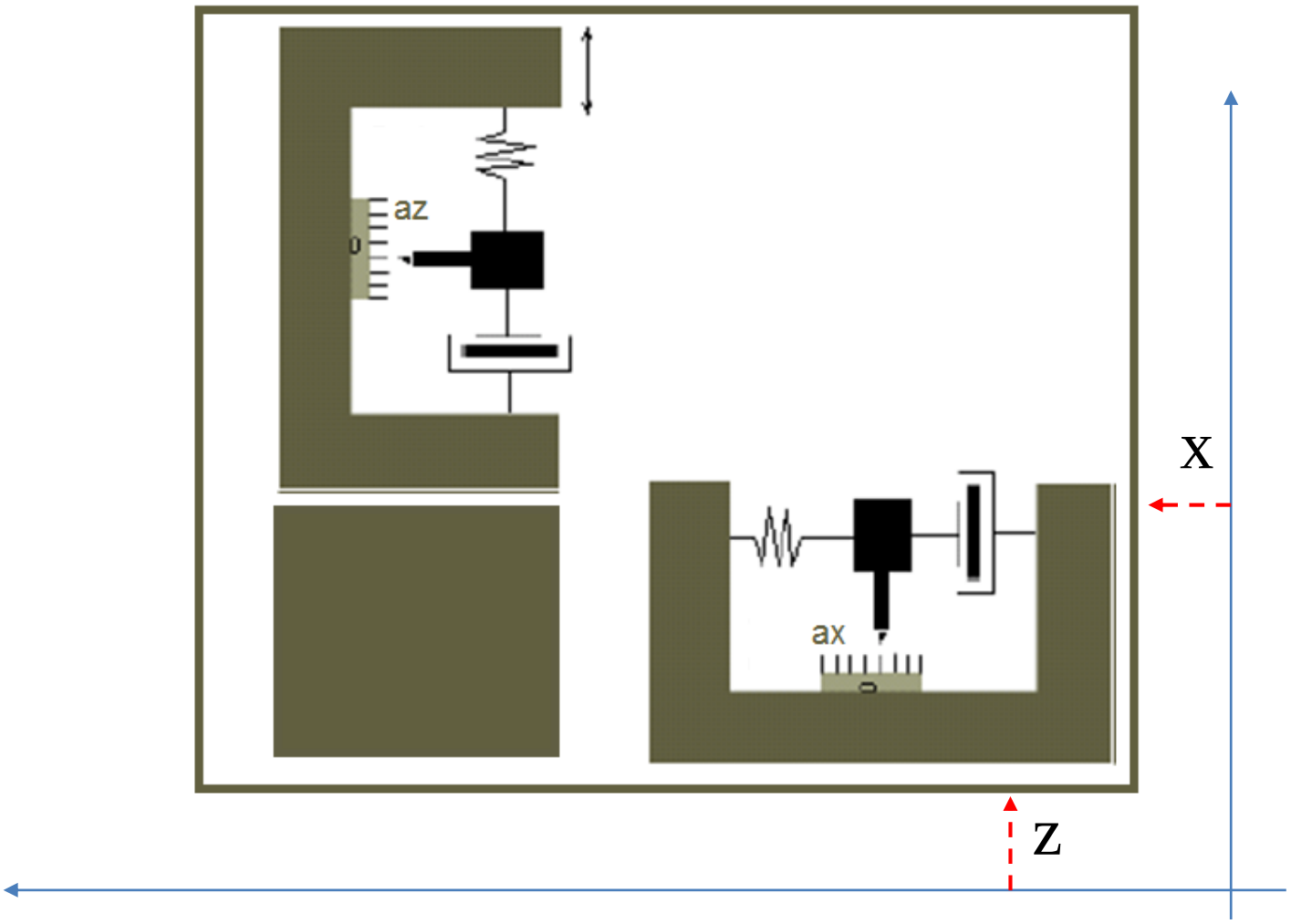
The idea is to use it in this way: Fixed to the object whose acceleration we want to measure..

(here shown in an ideal “1D universe”)



Accelerometers

We can deploy 3 “1D accelerometers”, respect to 3 orthogonal (or at least linearly independent) directions ➔ 3D accelerometers



Technical complexities

Accelerometers' measurements are affected by noise and bias, so that simple "double integration" is not a feasible way of using them, for estimating position.

They also "capture" the projection of the gravity (respect to their axes). That effect is good or bad, depending on what we try to do with the sensor.

In addition, they measure accelerations in the directions of the device's axes (IMU coordinate frame), so that 3D attitude of the device is needed to be known for full 3D position integration.

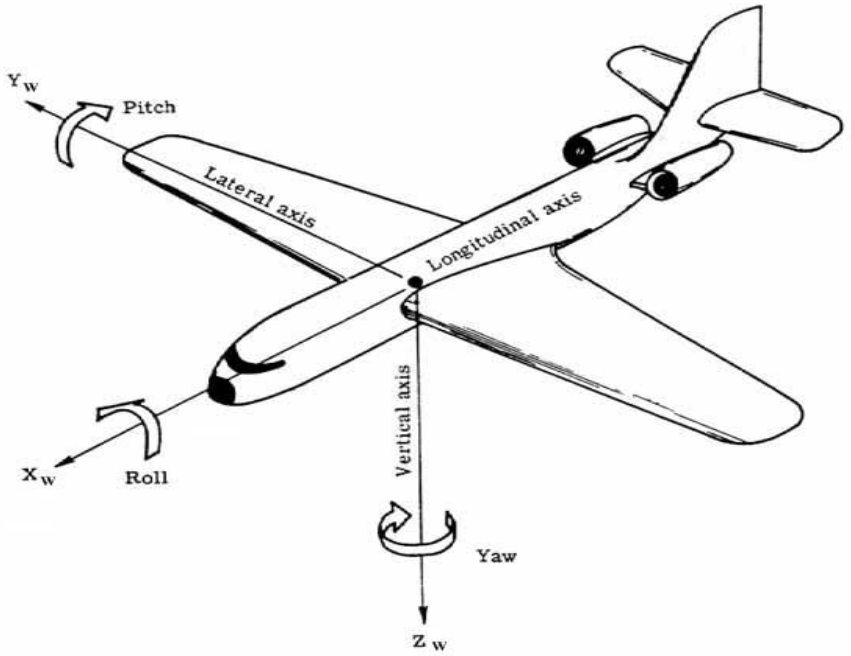
In combination with other sensors, accelerometers are well useful for estimating velocity and position in 3D.

Real life is, usually, in 3D (at least!)

3D Attitude Representation

In 2D we need just one angle: Yaw (or heading)

In 3D we need to specify 3 angles.
There are different conventions for representing orientation in 3D. The following convention for *Roll*, *Pitch* and *Yaw*, is one of the most used (but not the only one)

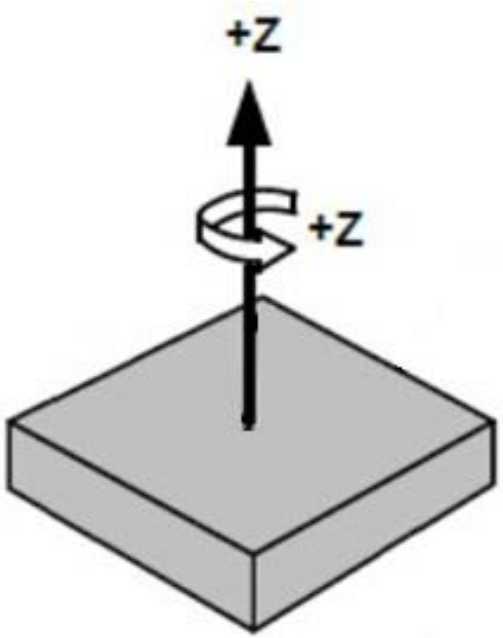


(Image from NASA, “mtp.jpl.nasa.gov/notes/pointing/Aircraft_Attitude2.png”)

1D Gyroscope

It can measure the angular rate respect to its axis

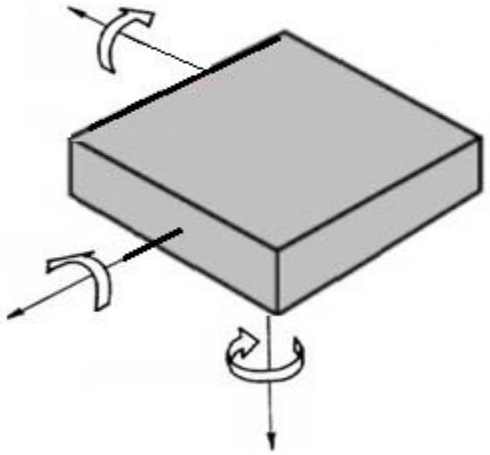
In a platform which operates in pure 2D (x,y, and heading), a 1D gyro with a vertical axis would be sufficient .

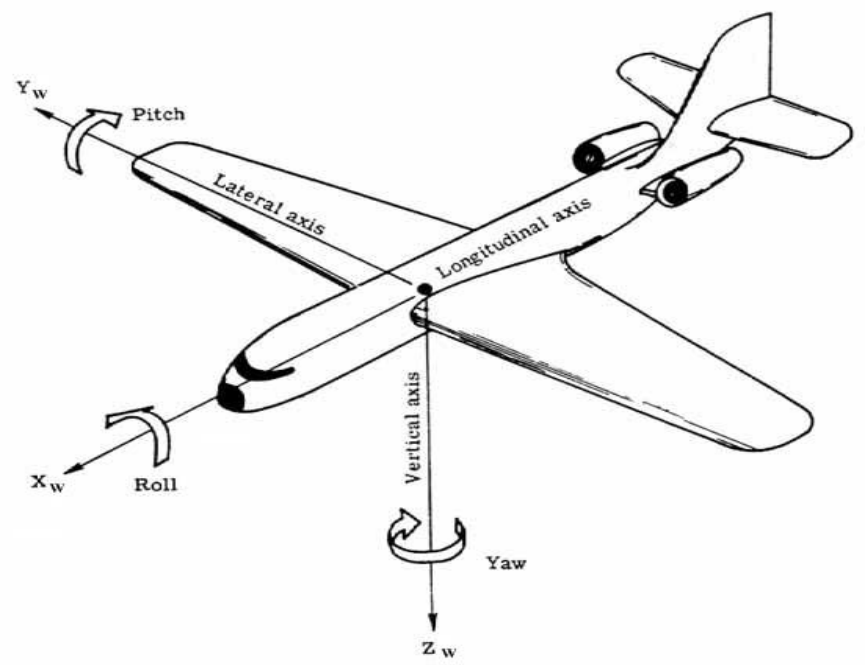


It measures the angular rate respect to its axis.

3D Gyroscopes

For a platform operating in full 3D, we use three 1D gyros (we call that configuration "3D gyros"), usually aligned with the local coordinate frame of the platform.





GYROSCOPES

gyro model: $\tilde{\omega} = \omega + b + \eta$ $\quad \eta \sim N\left(0, \sigma^2_{gyro}\right)$

ω: true angular velocity; b: bias; η: additive, zero-mean Gaussian noise

- 3 DOF = 3-axis gyros that measures 3 orthogonal axes, assuming no "cross talk"
- Bias is temperature-dependent and may change over time; can be approximated as a constant (or assumed to be "slowly time varying").
- Polluted with additive measurement noise.

→ Do not PANIC: All these issues are real, but are usually treated by proper processing.

GYRO INTEGRATION (DEAD RECKONING)

- Can be integrated for estimating 3D attitude.

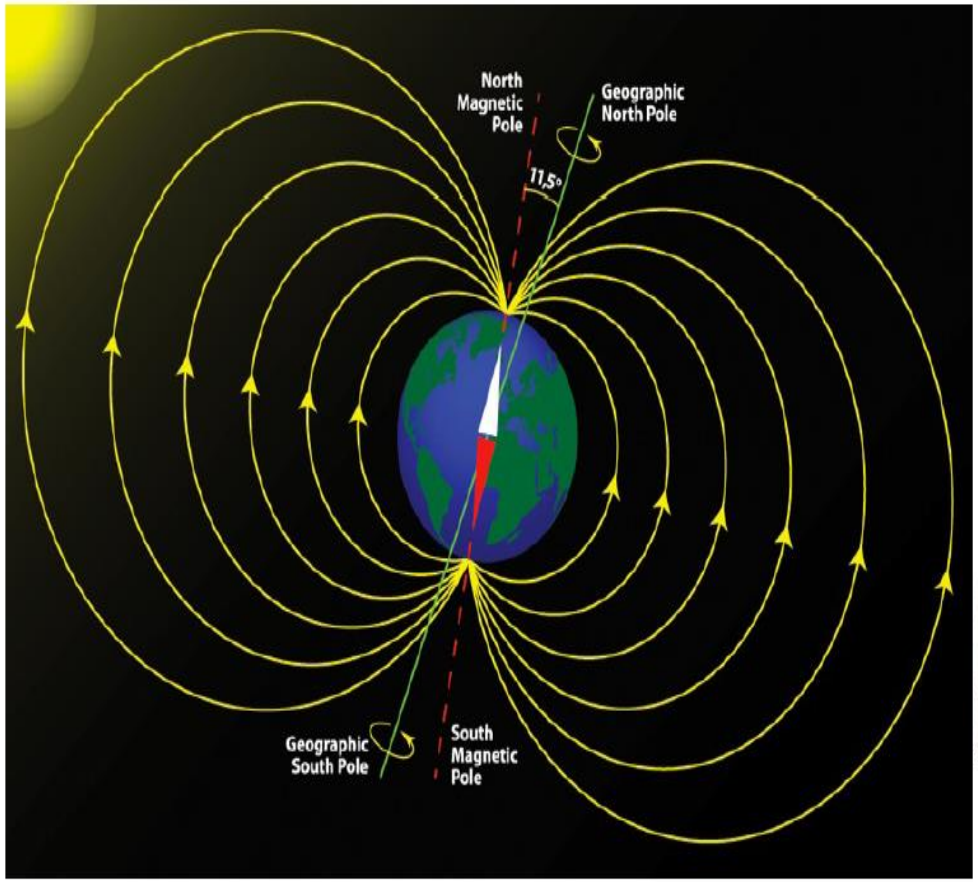
- Bias and noise variance can be estimated, other sensor measurements are usually used (used) to correct drift (sensor fusion)
- Accurate in short term, but not adequate in long term, due to drift.

GYROSCOPES

- Gyroscopes measure 3D angular rates
- Can be integrated for estimating 3D attitude.

- Bias and noise properties (e.g. variance) can be estimated, other sensor measurements are used to mitigate the drift
- Accurate in short term, but not feasible in long term, due to drift.

MAGNETOMETER



MAGNETOMETER

- Measure earth's magnetic (field in Gauss or uT.)

3 respect to 3 orthogonal axes of the device.

- Actual direction depends on latitude and longitude
- Distortions due to metal / electronics objects nearby.

SENSOR FUSION

- Fusing gyro and accelerometer data gives 6 DOF sensor fusion.
- Can correct tilt (i.e., pitch & roll) only. No information about yaw

- Magnetometer can allow to include yaw in estimates.

Now, we see one working

(in real time, read by a process, shown in Matlab (friendly)

IMU model: Microstrain GM03

(operating @166hz, but I can be set to higher ones, up to 1Khz)

- Now, we inspect accelerometers, gyroscopes and magnetometers working in real time.
- We could see typical polluting noises and bias

Many models of different cost, quality, etc.
Example- low cost one: InvenSense MPU 9250,

https://invensense.tdk.com/products/motion-tracking/9-axis/mpu-9250/

EXAMPLE- INVENSENSE MPU 9250

Multiple modes of operation (for signal ranges):

- Accelerometers: [-2,+2], [-4,+4], [-8,+8], [-16,+16] G
- Gyroscopes: [-250,+250], [-500,+500], [-1000,+1000], [-2000,+200] degrees/second.
- Magnetometers: [-4800,+4800] μT (microTesla)

- BTW: Arduino compatible via IC2 interface.
- Not industrial grade , but useful for many projects.

Attitude estimation based on integration of measured local angular rate.

- The angular velocities provided by the IMU's gyroscopes can be integrated to predict the attitude of the unit, in the navigation (aka "global") coordinate frame.,

- The sensor measures those physical variables in its local coordinate frame. Consequently, as the body of the sensor moves and rotates, the sensor's coordinate frame does vary as well. This fact implies that some extra processing is needed to express those angular velocities (measured in the sensor's coordinate frame) in some fixed and external coordinate frame.

Integrating the angular rates, directly in the local CF, does not make any sense or practical purpose.

There is a transformation which converts those local angular rates, into angular rates expressed in a Navigation CF.

Attitude Estimation based on Integration of measured local angular rate.

The local angular velocities, $(\omega_x, \omega_y, \omega_z)$, which are provided by the on-board gyroscopes, need to be transformed to angular velocities expressed in the global coordinate frame. The transformation is evaluated by the following expression

$$\frac{d\varphi_x}{dt} = \omega_x + \left(\omega_y \cdot \sin(\varphi_x) + \omega_z \cdot \cos(\varphi_x)\right) \cdot \tan(\varphi_y)$$

$$\frac{d\varphi_y}{dt} = \left(\omega_y \cdot \cos(\varphi_x) - \omega_z \cdot \sin(\varphi_x)\right)$$

$$\frac{d\varphi_z}{dt} = \left(\omega_y \cdot \sin(\varphi_x) + \omega_z \cdot \cos(\varphi_x)\right) / \cos(\varphi_y)$$

in which $\varphi_x, \varphi_y, \varphi_z$ are the attitude angles and $\frac{d\varphi_x}{dt}, \frac{d\varphi_y}{dt}, \frac{d\varphi_z}{dt}$ their time derivatives , all of them being expressed in the global coordinate frame

This is continuous time process, which can be approximated by a discrete time one, for performing its integration, via the following recursive step, based on the Euler's approximation:

$$\varphi_x(t+\Delta t)=\varphi_x(t)+\left(\omega_x(t)+\left(\omega_y(t)\cdot\sin\left(\varphi_x(t)\right)+\omega_z(t)\cdot\cos\left(\varphi_x(t)\right)\right)\cdot\tan\left(\varphi_y(t)\right)\right)\cdot\Delta t$$

$$\varphi_y(t+\Delta t)=\varphi_y(t)+\left(\left(\omega_y(t)\cdot\cos\left(\varphi_x(t)\right)-\omega_z(t)\cdot\sin\left(\varphi_x(t)\right)\right)\right)\cdot\Delta t$$

$$\varphi_z(t+\Delta t)=\varphi_z(t)+\left(\omega_y(t)\cdot\sin\left(\varphi_x(t)\right)+\omega_z(t)\cdot\cos\left(\varphi_x(t)\right)\right)/\cos\left(\varphi_y(t)\right)\cdot\Delta t$$

(in discrete time k)

$$\varphi_x[k+1] = \varphi_x[k] + \left(\omega_x[k] + \left(\omega_y[k]\cdot\sin\left(\varphi_x[k]\right) + \omega_z[k]\cdot\cos\left(\varphi_x[k]\right)\right)\cdot\tan\left(\varphi_y[k]\right)\right)\cdot T$$

$$\varphi_y[k+1] = \varphi_y[k] + \left(\left(\omega_y[k]\cdot\cos\left(\varphi_x[k]\right) - \omega_z[k]\cdot\sin\left(\varphi_x[k]\right)\right)\right)\cdot T$$

$$\varphi_z[k+1] = \varphi_z[k] + \left(\omega_y[k]\cdot\sin\left(\varphi_x[k]\right) + \omega_z[k]\cdot\cos\left(\varphi_x[k]\right)\right)\Big/\cos\left(\varphi_y[k]\right)\cdot T$$

Example in Matlab.

The following piece of code implements one step of the numerical integration (Euler Method) of the attitude equation.

```
Function NewAttitude  = IntegrateOneStepOfAttitude( gyros, dt, CurrentAttitude )

        % for a small delta time , dt
        % CurrentAttitude is the current (initial) attitude, in radians
        % gyros:vector with the gyros measurements, scaled in rad/sec

ang = CurrentAttitude ;     % current global Roll, Pitch, Yaw  (at time t)
wx = gyros(1);                      %local roll rate
wy = gyros(2);                      %local pitch rate
wz = gyros(3);                      %local yaw rate

% ------------------------------
cosang1=cos(ang(1)) ;
cosang2=cos(ang(2)) ;
sinang1=sin(ang(1)) ;

roll        = ang(1) + dt * (wx + (wy*sinang1 + wz*cosang1)*tan(ang(2))) ; %(*)
pitch       = ang(2) + dt * (wy*cosang1 - wz*sinang1)                     ;
yaw         = ang(3) + dt * ((wy*sinang1 + wz*cosang1)/cosang2)           ; %(*)
% ------------------------------

NewAttitude= [roll,pitch,yaw];          % new global Roll, Pitch, Yaw (at time t+dt)
return ;


% (*): you could see that if ang(2) is close to pi/2 or –pi/2, the term tan( ) and 1/cos( ) would have numerical problems.
```

We end here, our discussion about IMUs.

(we may play a bit more, with the IMU)