

CMPE58C: Sp. Tp. Mobile Location Tracking & Motion Sensing

Inertial Sensors

Can Tunca

-

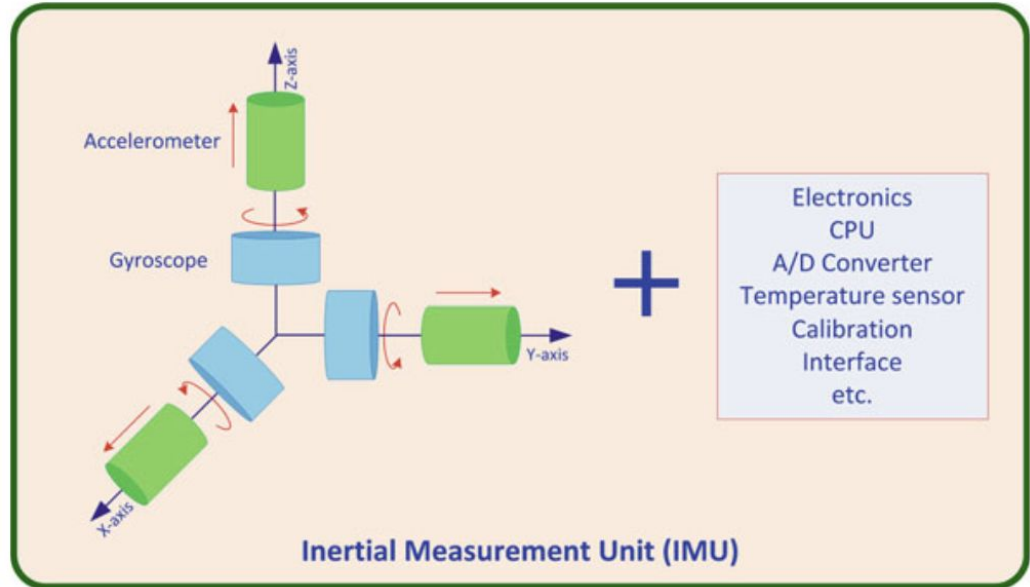
Fall 2022

Lecture Overview

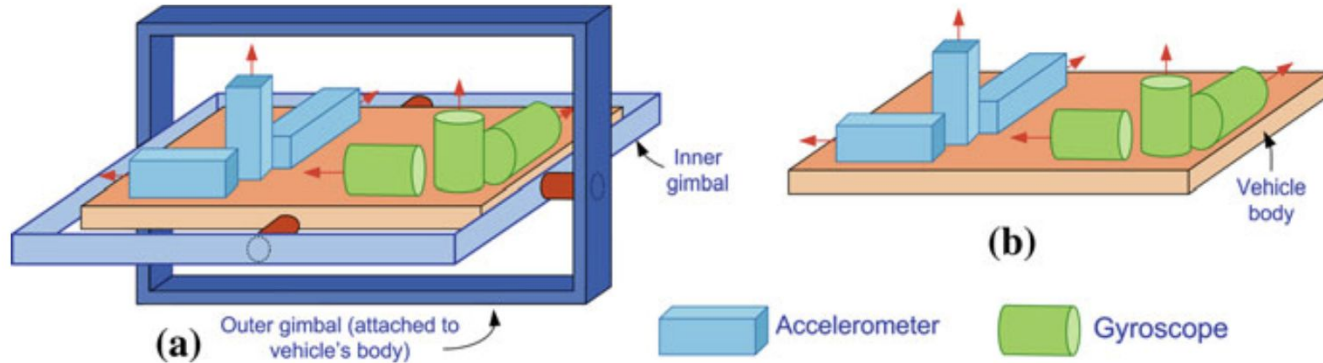
- Coordinate frames
 - ECI, ECEF, LLF, Body
- Rotation/orientation representations
 - Euler angles, rotation matrices, axis and angle, quaternions
- Other concepts
 - Inertial Measurement Unit (IMU)
 - Strapdown IMU vs Gimbal IMU
 - MEMS
 - Allan variance
 - How to calibrate sensors?
- Inertial sensors
 - Accelerometer
 - Gyroscope
 - Magnetometer
- Inertial tracking and navigation
 - Orientation/attitude estimation and tracking
 - Inertial navigation basics
 - Sensor fusion basics
 - Error sources

Inertial Measurement Unit (IMU)

- A combination of gyroscope and accelerometer (and other things...)
- Recent IMUs also contain magnetometer (compass), which is not actually an inertial sensor
- i.e. a multi-sensor module



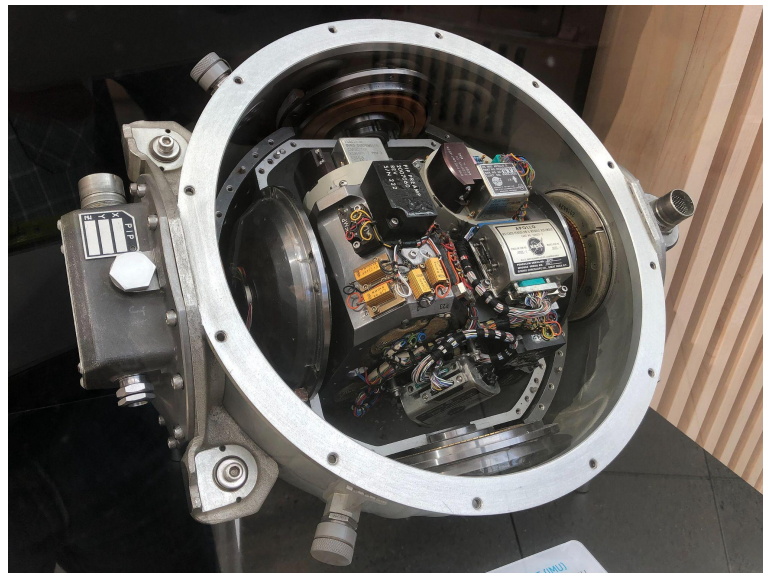
Stable-platform IMU vs Strapdown IMU



- Stable-platform IMU (aka Gimbaled IMU) **(a)**
 - Motors keep the platform aligned with the navigation frame in response to gyro measurements
 - Outputs of accelerometers can be directly used to compute velocity and displacement (without orientation compensation)
- Strapdown IMU **(b)**
 - Sensors are fixed rigidly onto the body of the device we are tracking
 - Computations required to continuously align measurements with the navigation frame

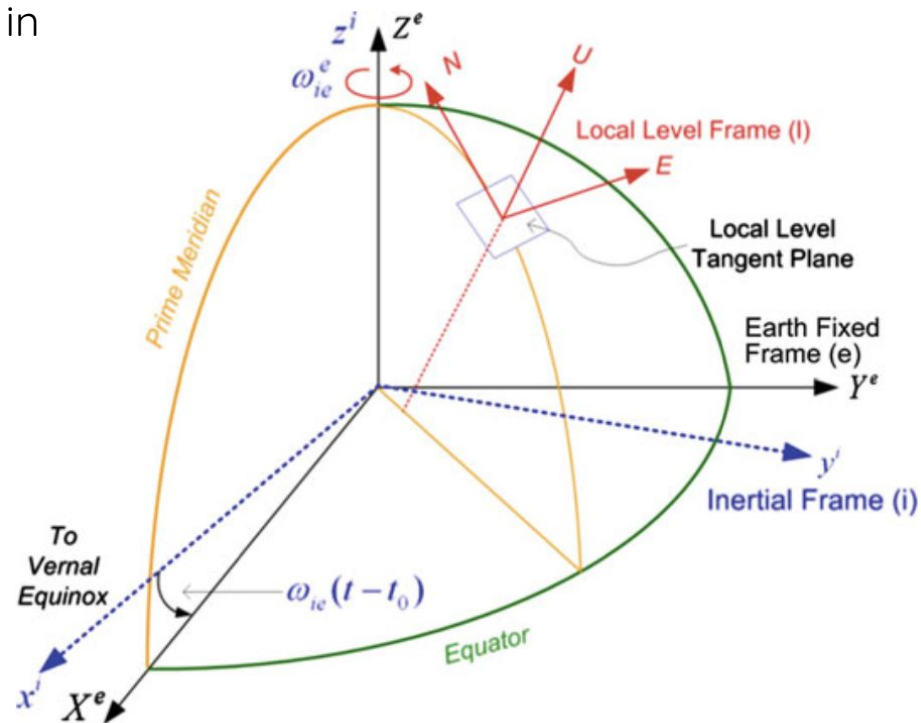
An example stable-platform IMU: Apollo IMU

- Used in Apollo space missions
- Gimbaled on three axes
- The actual IMU platform (aka Stable Member, SM) remains fixed relative to stars (i.e. the **inertial frame** of reference)
- Drifts \sim milliradian/h
 - Periodic calibration needed via optical measurements from the stars
- Costly, mechanical and big



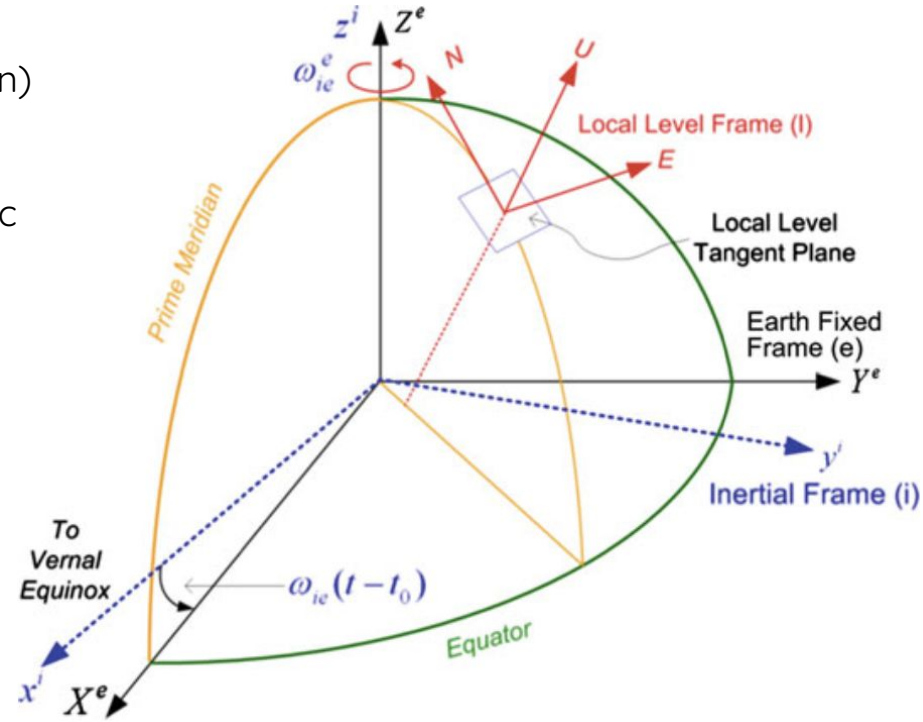
Coordinate Frames

- Used to represent the position of a point in relation to some reference
- Many alternatives, the notable ones:
 - Earth-Centered Inertial
 - Earth-Centered Earth-Fixed
 - Local Level



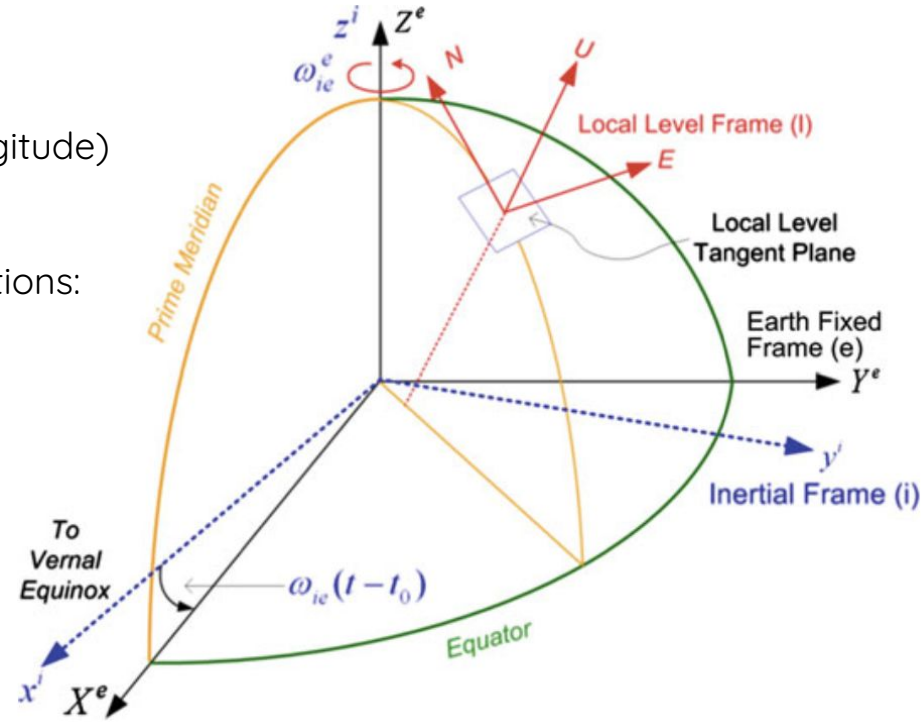
Earth-Centered Inertial Frame (ECI)

- An inertial frame is either stationary in space, or moving at constant velocity (no acceleration)
- X axis facing **vernal equinox**:
Intersection of equatorial plane with the ecliptic (plane of Earth's orbit around Sun)
- i.e. fixed relative to stars
- All IMUs measure relative to an inertial frame, along the unit's sensitive axes

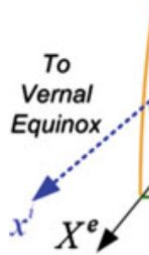


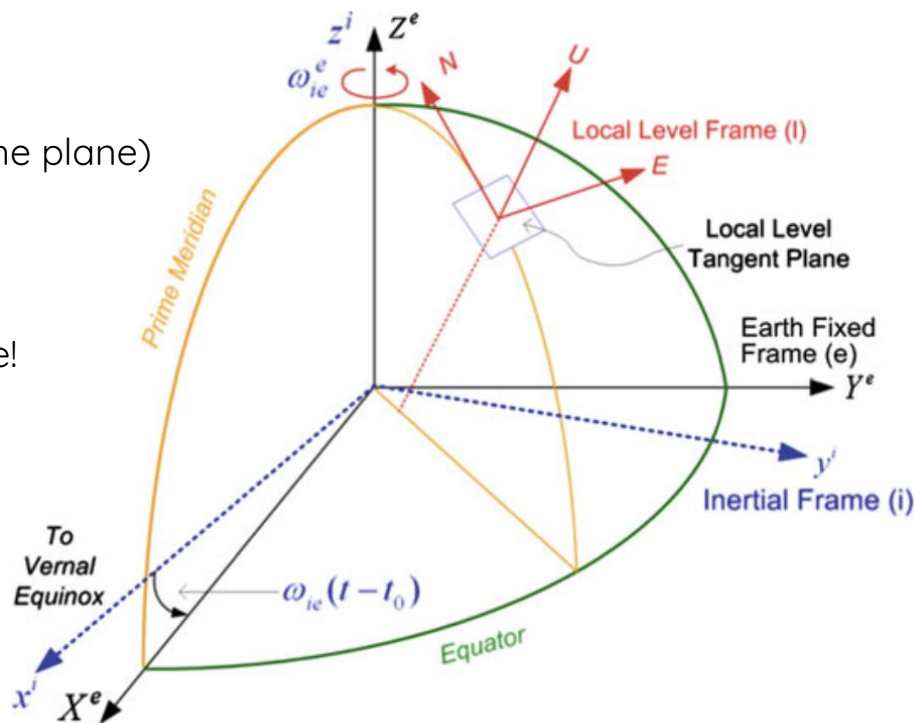
Earth-Centered Earth-Fixed Frame (ECEF)

- Shares z axis and origin with ECI
- x, y axes rotates with Earth
- x axis through Greenwich meridian (0 deg. longitude)
- Hence “Earth-fixed”
- Convenient for positioning/navigation applications:
Usually, we want to navigate relative to Earth!



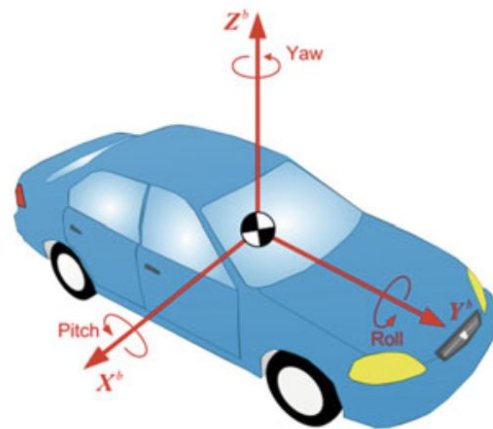
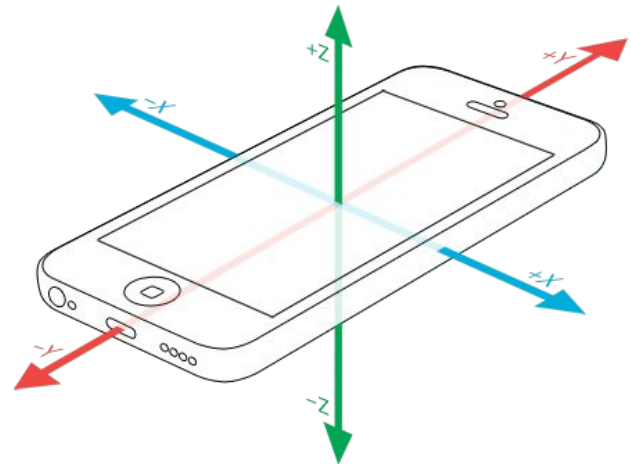
Local Level Frame (LLF)

- A tangent plane on the surface of the Earth
 - Also called Local Tangent Plane (LTP)
 - Many variants (depending on how you place the plane)
 - Most common is ENU
 - East-North-Up, corresponding to x, y, z
 - Convenient for tracking things on Earth surface!
 - Makes it easier to deal with curvature
 - Curvature negligible in small scale
 - But, has to be moved when needed
 - Could be converted to other **local** coordinate systems
(e.g. a building floor plan)
- 



Body Frame

- A frame of reference fixed on the object we are tracking
- The axes of a strapdown IMU usually coincides with body frame
 - The measurement axes of inertial sensors are also called **sensitive axes**
- Origin is typically at the center of gravity

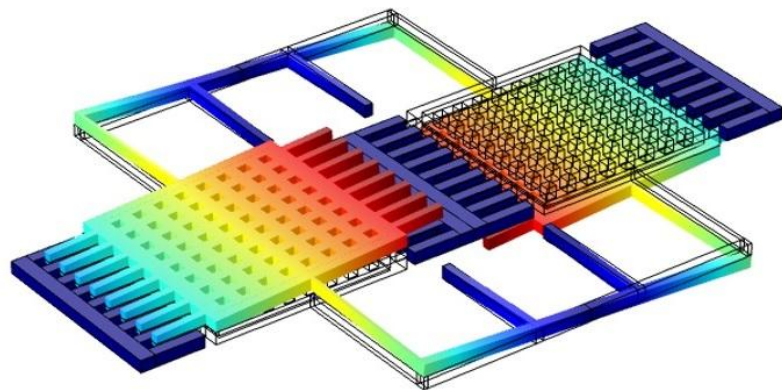


Computational Frame:

Whatever reference frame we want to track the object in
aka **Navigation Frame**

Gyroscope

- Measures rotation rate (angular velocity)
- NOT absolute angles!
- Independent measurements from three axes: x, y, z
- Useful to track a device's 3D orientation (aka **attitude**),
but we have to know initial attitude
- Rotation rate represented by ω



Traditional gyroscopes

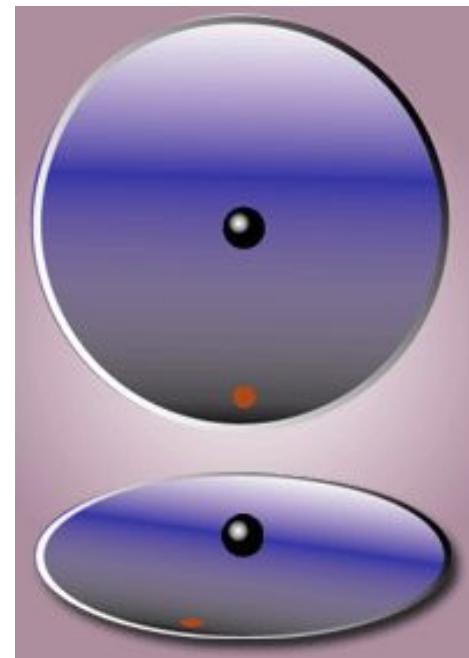
- Gyroscopes used to measure absolute orientation
- A spinning disc that remains stationary with respect to an inertial frame
- Conservation of angular momentum!

Some examples of traditional gyroscopes and illustration of conservation of angular momentum



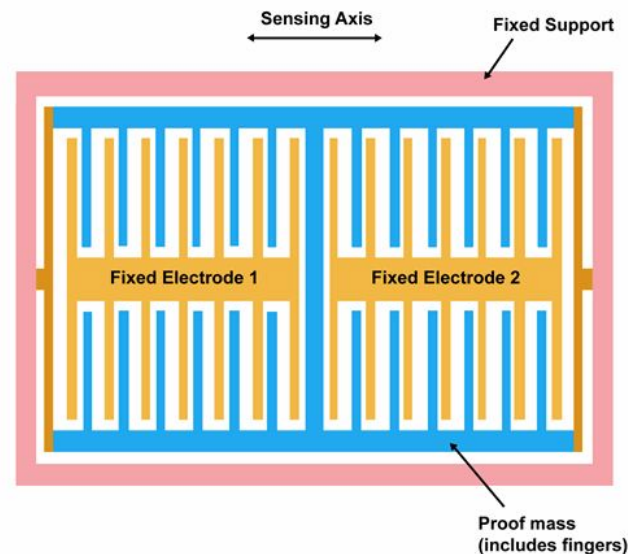
Coriolis force

- Due to rotation with respect to an inertial frame
- i.e. Due to Earth's rotation
- Gyroscopes also measure this!
 - As they measure w.r.t. an inertial reference frame
 - Could be negligible depending on sensor quality



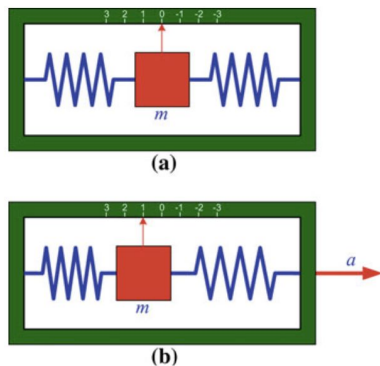
Accelerometer

- Measures acceleration $\mathbf{f} = \mathbf{a} - \mathbf{g}$
 - \mathbf{a} : Acceleration vector of the body (aka linear acceleration)
 - \mathbf{g} : Gravitational acceleration vector
- Imagine a mass on a spring

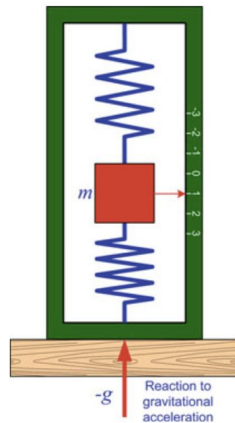


Accelerometer

- Gravity component is actually reaction to gravity ($-g$)
- We are usually interested in only body acceleration, so we try to remove gravity
- But, gravity is useful for attitude estimation! (We'll see how)



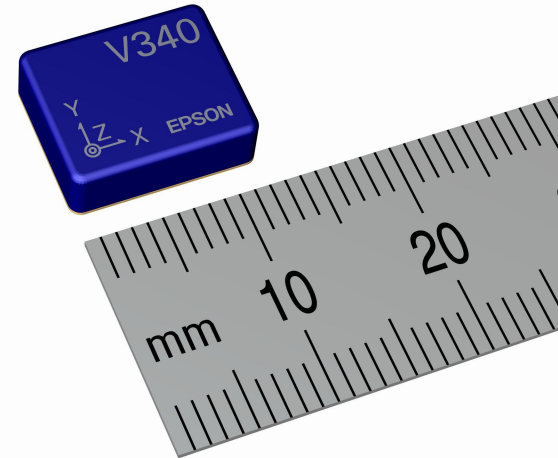
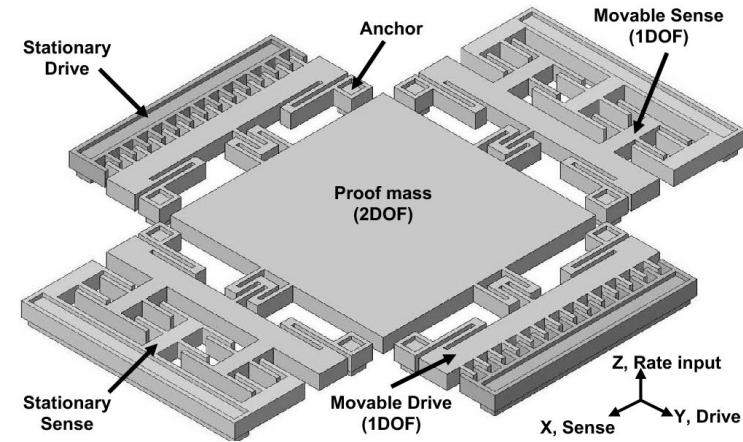
Body acceleration



Reaction to gravity

MEMS

- Micro-electromechanical systems
- Mechanical systems in the micro scale
- Allows for very small IMUs
- Low cost, mass produced
- Mechanical systems may be more accurate, but MEMS allowed inertial sensors to become widespread



Inertial Navigation: Theory

- Gyroscopes measure rotation rate
 - Could be integrated to get angular change
 - Orientation can be tracked if we know initial orientation
- Accelerometers measure acceleration
 - Acceleration can be integrated twice to get displacement
 - Position can be tracked if we know initial position
- Let's assume theory is directly applicable to practice and explore:
 - 1D case
 - 2D case
 - 3D case
- Then we'll see whether it matches reality...

1D Inertial Navigation

Integrate acceleration to compute velocity:

$$v_t = \int a_y dt = a_y t + v_0$$

↘ initial velocity

Integrate velocity to compute displacement:

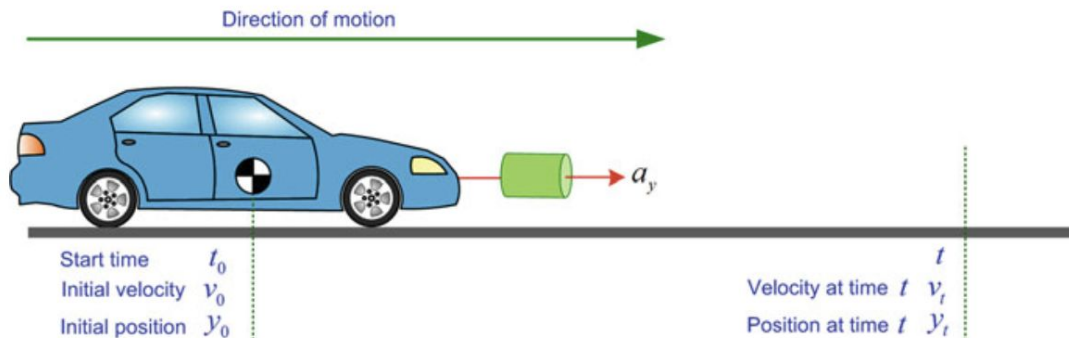
$$y_t = \int v_t dt$$

$$y_t = \int (a_y t + v_0) dt$$

$$y_t = \frac{1}{2} a_y t^2 + v_0 t + y_0$$

↘ initial position

- Assuming we know initial velocity and position
- Note: No need for orientation in 1D case
- Only accelerometer sufficient
- Theoretically we can track indefinitely



2D Inertial Navigation

Convert acceleration to computational frame:

$$a_E = a_y \sin A + a_x \cos A$$

$$a_N = a_y \cos A - a_x \sin A$$

Same principle in two axes:

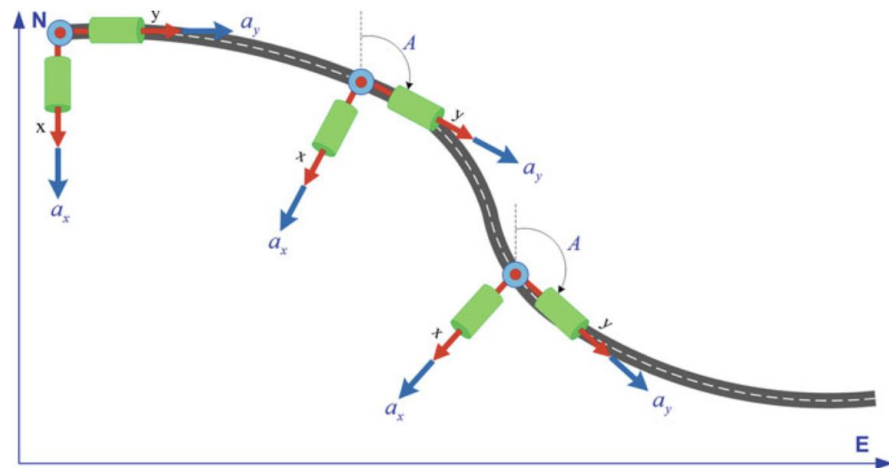
$$x_E = \int \int (a_x \cos A + a_y \sin A) dt dt$$

$$x_N = \int \int (a_y \cos A - a_x \sin A) dt dt$$

Orientation computed by:

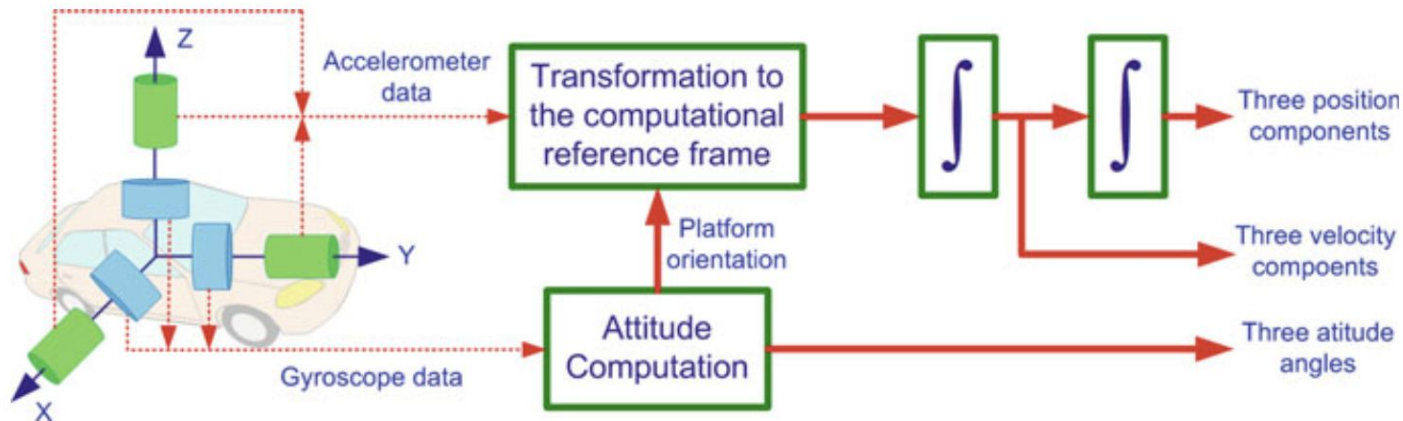
$$A(t) = \int \omega_{gyro} dt + A_o$$

- We need orientation for 2D (A)
- Hence we need gyroscope
- Same theory: We can track indefinitely



3D Inertial Navigation

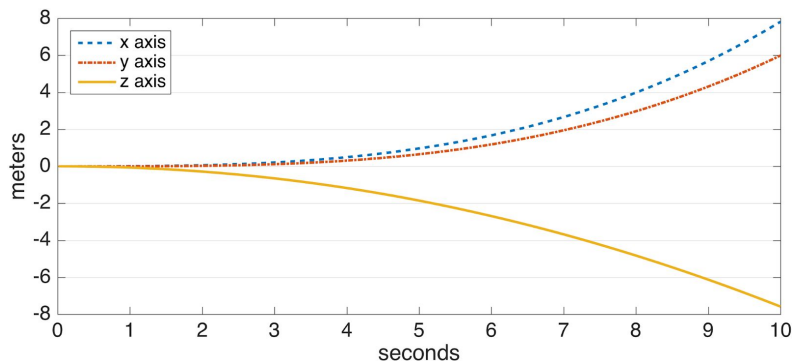
- Same principles
- 3D orientation (attitude) is required
 - Computed via gyroscope
 - Also used to rotate acceleration readings to computational frame



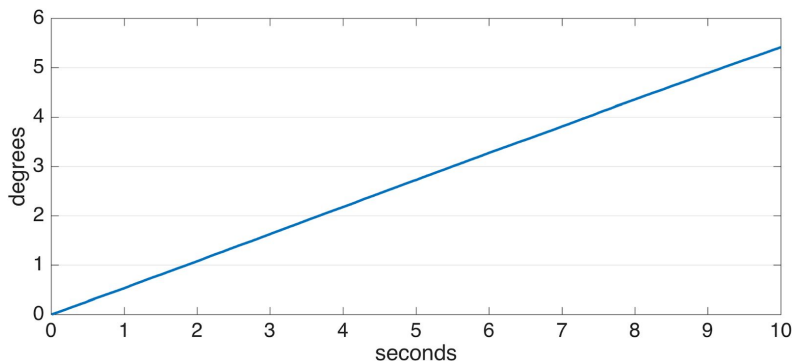
Inertial Navigation: Reality

- Example: Measurements taken from a perfectly stationary IMU
 - We'd expect all the values to remain 0
 - Reality is different
- Acceleration to displacement
 - We estimate that the device has moved away a few meters (obviously not true)
 - Double integration
 - Error is proportional to t^2
- Rotation rate to orientation
 - Similar concern, but much milder
 - Since we need to integrate only once
 - Error proportional to t (linear): manageable
 - So it is possible to track attitude...

Displacement from accelerometer

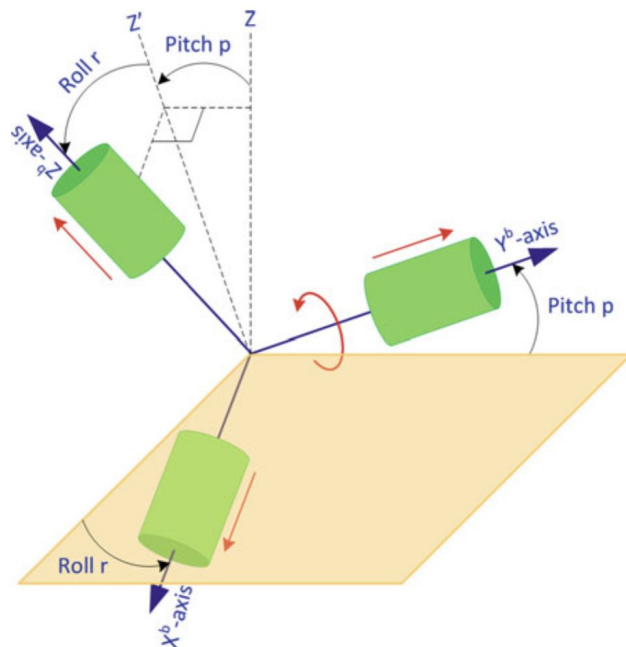


Orientation from gyroscope (only one axis)



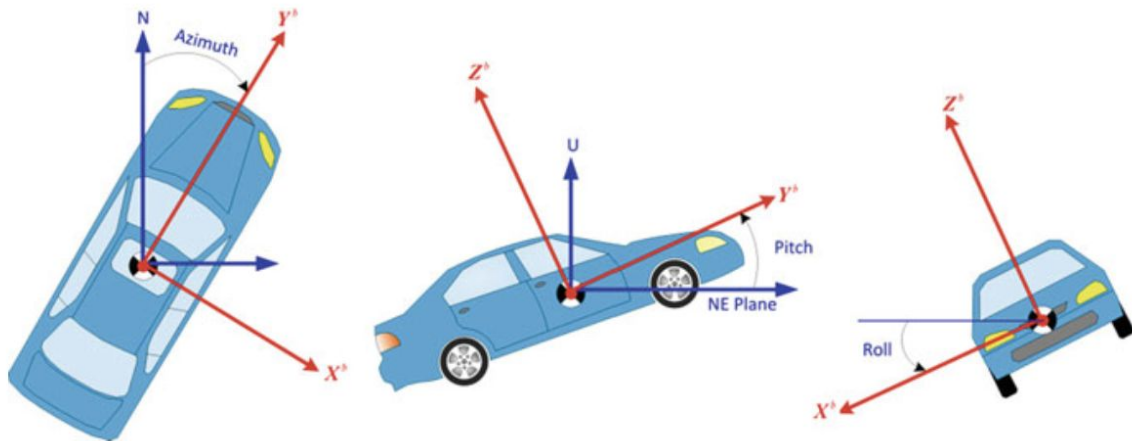
Attitude

- 3D orientation
- There are different ways to represent it
 - Euler Angles
 - Rotation Matrix (DCM)
 - Axis and Angle
 - Quaternion
- Same as rotation w.r.t. a reference frame
 - So, representing orientation is synonymous to representing rotation
- Important by itself (we'd want to know how device is oriented), also important to orient measurements from other sensors



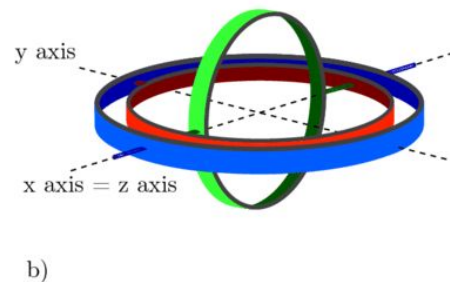
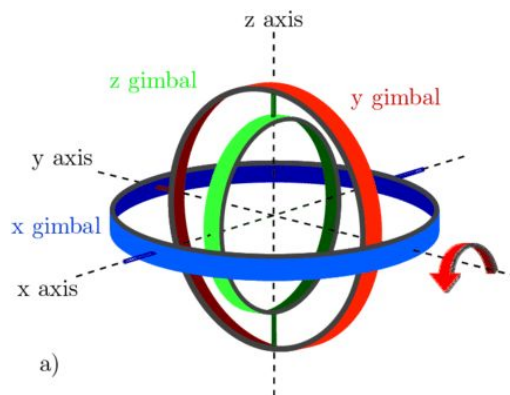
Euler Angles

- Easiest to comprehend and visualize
- 3 rotations about 3 axes: yaw, pitch, roll
- Order of rotation matters! (ambiguity)
- Susceptible to gimbal lock

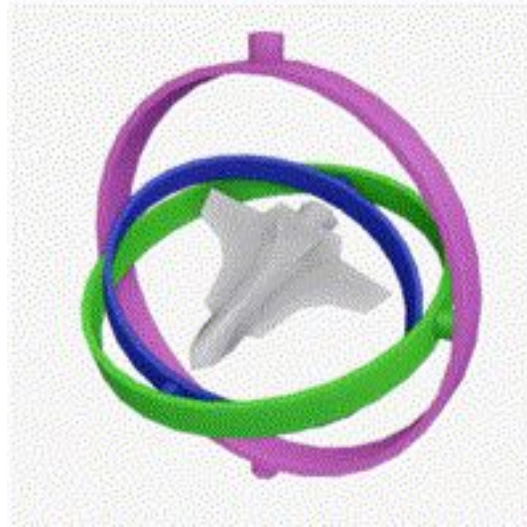


Gimbal Lock

- Happens when two of the three gimbals are parallel
- There is no gimbal available to rotate about one axis
 - Loss of one degree of freedom
 - Causes inability to describe incremental orientation changes about that axis, other values should also be changed
 - Multiple Euler angle values describe the same attitude
 - Not a physical lock, more about representation



Gimbal Lock: Example



Magenta and blue gimbals apply the same rotation,
no gimbal to rotate the airplane to its side

Another good explanation: <https://youtu.be/zc8b2Jo7mno>

Rotation Matrix

- A matrix rotating vector \mathbf{x} to a new frame: $\mathbf{x}' = \mathbf{R}\mathbf{x}$
- First, let's see how it relates to Euler angles
- Combining rotations is easy, just multiply matrices
- Order matters in the 3D case!

2D case

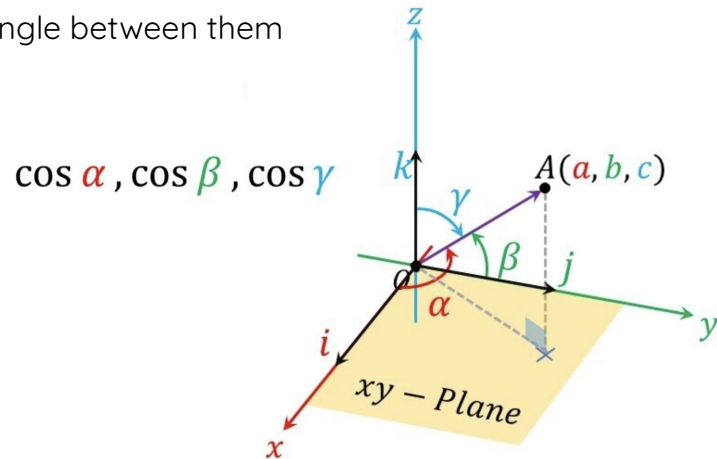
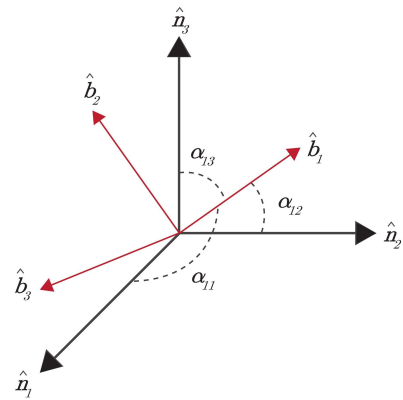
$$R(\theta) = \begin{bmatrix} \cos \theta & -\sin \theta \\ \sin \theta & \cos \theta \end{bmatrix}$$

3D case

$$\begin{aligned} R &= R_z(\alpha) R_y(\beta) R_x(\gamma) = \begin{bmatrix} \cos \alpha & \overset{\text{yaw}}{-\sin \alpha} & 0 \\ \sin \alpha & \cos \alpha & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos \beta & \overset{\text{pitch}}{0} & \sin \beta \\ 0 & 1 & 0 \\ -\sin \beta & 0 & \cos \beta \end{bmatrix} \begin{bmatrix} 1 & \overset{\text{roll}}{0} & 0 \\ 0 & \cos \gamma & -\sin \gamma \\ 0 & \sin \gamma & \cos \gamma \end{bmatrix} \\ &= \begin{bmatrix} \cos \alpha \cos \beta & \cos \alpha \sin \beta \sin \gamma - \sin \alpha \cos \gamma & \cos \alpha \sin \beta \cos \gamma + \sin \alpha \sin \gamma \\ \sin \alpha \cos \beta & \sin \alpha \sin \beta \sin \gamma + \cos \alpha \cos \gamma & \sin \alpha \sin \beta \cos \gamma - \cos \alpha \sin \gamma \\ -\sin \beta & \cos \beta \sin \gamma & \cos \beta \cos \gamma \end{bmatrix} \end{aligned}$$

Rotation Matrix

- A matrix rotating vector \mathbf{x} to a new frame: $\mathbf{x}' = \mathbf{R}\mathbf{x}$
- aka **Direction Cosine Matrix** (DCM)
 - The columns correspond to old basis vectors expressed w.r.t. new basis (**Basis**: essentially a coordinate system composed of three orthogonal vectors)
 - Example: Representing the attitude of body w.r.t. navigation frame:
old basis vectors = body frame, new basis = navigation frame
 - Projection of one basis vector to the other = **cosine** of the angle between them
 - Columns are unit vectors
- No need to imagine Euler angles anymore,
think in terms of vectors!



Rotation Matrix



Strengths

- Doesn't suffer from Gimbal lock or singularity
- Easy to combine rotations (order matters!)
- Not too hard to comprehend or visualize (not as easy as Euler angles)
- Math is easy, linear algebra!

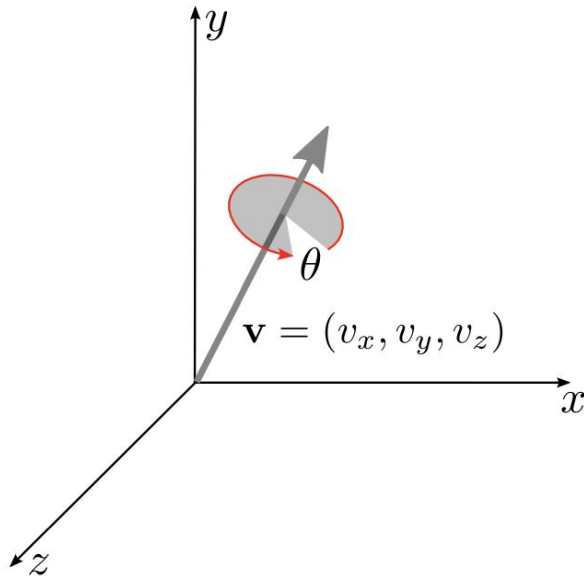


Weaknesses

- Bigger representation (9 values instead of 3 in Euler angles case)
- Not very suitable for interpolation: You need to make sure an intermediate matrix is still a rotation matrix (3 orthogonal unit vectors)

Axis and Angle

- Every rotation can be represented as some rotation θ about a vector \mathbf{v}
- No order of rotation, all at once around that “axis”
- ☒ Requires 4 values to represent: (θ, \mathbf{v})
- ☒ Easy to interpolate
- ☒ Easy to comprehend
- ☒ Combining rotations is difficult
- ☒ Math is not very convenient
(basic operations require other intermediate forms)



Quaternions

$$q = q_w + iq_x + jq_y + kq_z$$

- Essentially 4-dimensional complex numbers (i.e. 4D vectors in complex space)
- i, j, k are comparable to the imaginary part of a complex number
- The imaginary parts satisfy the following rules:

$$\begin{aligned} i &\neq j \neq k & ij &= -ji = k \\ i^2 &= j^2 = k^2 = ijk = -1 & ki &= -ik = j \\ & & jk &= -kj = i \end{aligned}$$

- A rotation-representing quaternion should have unit length:

$$\|q\| = \sqrt{q_w^2 + q_x^2 + q_y^2 + q_z^2} = 1$$

Quaternions

- You can think of quaternions as a wrapper for axis and angle representation
- It is very easy to convert axis and angle to a quaternion:

$$q(\theta, \mathbf{v}) = \underbrace{\cos\left(\frac{\theta}{2}\right)}_{q_w} + \underbrace{i v_x \sin\left(\frac{\theta}{2}\right)}_{q_x} + \underbrace{j v_y \sin\left(\frac{\theta}{2}\right)}_{q_y} + \underbrace{k v_z \sin\left(\frac{\theta}{2}\right)}_{q_z}$$

Note: \mathbf{v} should be normalized to have unit length
(this is not a requirement for axis-angle representation)

- Thus, rotation quaternions are also known as “rotation vectors”

Quaternion Algebra

- Very well defined set of operations

multiplication

$$\begin{aligned} qp &= (q_w + iq_x + jq_y + kq_z)(p_w + ip_x + jp_y + kp_z) \\ &= (q_w p_w - q_x p_x - q_y p_y - q_z p_z) + \\ &\quad i(q_w p_x + q_x p_w + q_y p_z - q_z p_y) + \\ &\quad j(q_w p_y - q_x p_z + q_y p_w + q_z p_x) + \\ &\quad k(q_w p_z + q_x p_y - q_y p_x + q_z p_w) + \end{aligned}$$

conjugate

$$q^* = q_w - iq_x - jq_y - kq_z$$

inverse

$$q^{-1} = \frac{q^*}{||q||^2}$$

Note: For rotation quaternions conjugate = inverse, since rotation quaternions have a length of 1 (unit quaternion)

rotate q_u by q

$$q'_u = qq_u q^{-1}$$







inverse rotation (of the above)

$$q_u = q^{-1} q'_u q$$

combining rotations

$$q'_u = q_2 q_1 q_u q_1^{-1} q_2^{-1}$$

Quaternions

-  Well defined math
-  Well defined interpolation operations (e.g. SLERP)
-  Does not suffer from Gimbal lock or singularity
-  Similar orientations give similar a quaternion result
-  Only 4 values (not as verbose as rotation matrices)
-  Difficult to comprehend and visualize

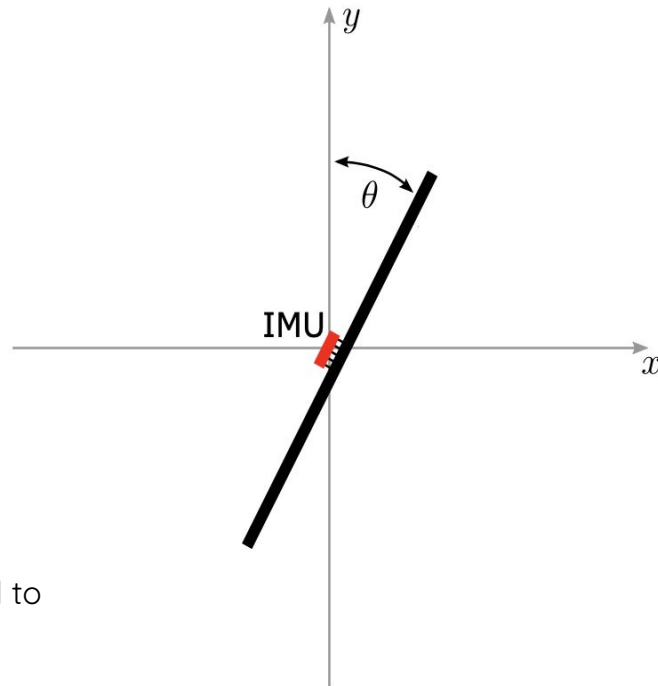
Orientation Tracking in 2D

- Imagine we are tracking only pitch...
- We already saw we can integrate gyro readings:

$$\theta_{gyro}^{(t)} = \theta_{gyro}^{(t-1)} + \tilde{\omega} \Delta t$$

current estimate previous estimate

- This is an approximation since gyro data is sampled
 - Ignoring all other sensor errors, this incurs an error proportional to time step squared $\epsilon \sim \mathcal{O}(\Delta t^2)$
 - Decreasing sampling period decreases error
 - Not too much, but unavoidable (angle estimate drifts eventually)
- Two challenges:
 - Can we keep this drift under control?
 - We need to know initial orientation (gyro tracking is relative)



Estimating Orientation via Accelerometer

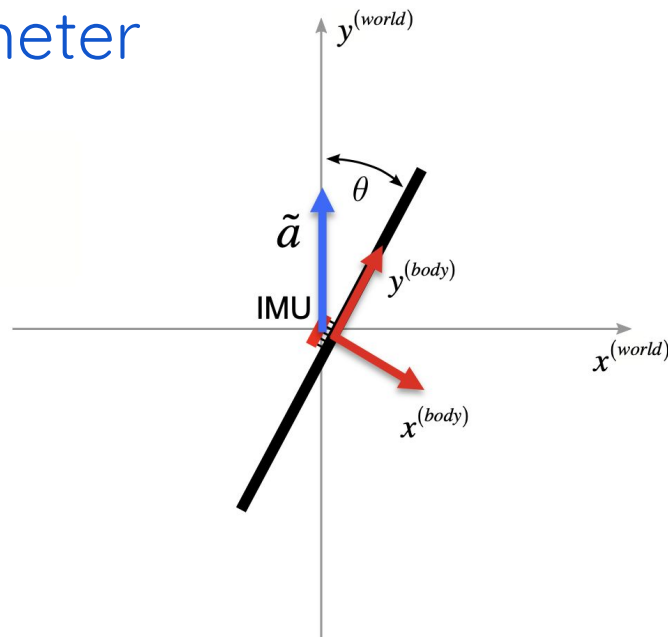
- Reminder: Accelerometers also measure gravity!
 - Specifically reaction to gravity: A vector pointing up
 - The angle this vector makes with body gives us pitch!

$$\theta_{acc} = \tan^{-1} \left(\frac{\tilde{a}_x}{\tilde{a}_y} \right)$$

- But accelerometers also measure body acceleration
 - Gravity estimation reliable only for stationary body
- Also, accelerometers are noisy sensors
(we'll see different error sources)
- We can make the following assumption:

On average, gravity is the dominating component

- Acceleration is never one way (unless you are in a space ship!)
- Noise is Gaussian distributed (zero mean)



Orientation Tracking in 2D: Sensor Fusion

- Gyroscopes give us good short term estimates
 - Drifts long term (due to approximation and sensor errors)
 - Relative tracking, we need to know initial orientation
- Accelerometers give us good long term estimates
 - Noisy short term
 - Affected via body acceleration
 - But gives us absolute estimates
- They complement each other!
- A good opportunity for sensor fusion

Complementary Filter

$$\theta^{(t)} = \alpha \left(\theta^{(t-1)} + \overset{\substack{\text{from gyroscope} \\ \uparrow}}{\tilde{\omega} \Delta t} \right) + (1 - \alpha) \overset{\substack{\text{from accelerometer} \\ \uparrow}}{\text{atan2}(\tilde{a}_x, \tilde{a}_y)}$$

- A basic but effective form of sensor fusion
- Essentially a weighted combination of both estimates
- α is blending weight (between 0-1), typically set to a high value to favor gyro short term
- Intuition:
 - Remove drift from gyro via high-pass filtering (big α)
 - Remove noise from accelerometer via low pass-filtering (small $1 - \alpha$)
- A form of exponential filter
 - If you pass previous estimate and latest value as two factors, this becomes an exponential filter
- The same principle can be applied to 3D (we'll see it next)

Estimating Attitude via Accelerometer in 3D

- With an accelerometer we can only estimate pitch and roll (not yaw)
 - Reaction to gravity is pointing UP (no component on the Earth-tangential plane)
- aka **tilt** = pitch + roll
- Let's first work with Euler angles $\overset{\text{roll}}{\mathbf{R}_z(-\theta_z)} \overset{\text{pitch}}{\mathbf{R}_x(-\theta_x)} \overset{\text{yaw}}{\mathbf{R}_y(-\theta_y)}$
- Acceleration measurement = vector UP rotated from world to body

$$\begin{aligned}
 \hat{\mathbf{a}} = \frac{\tilde{\mathbf{a}}}{\|\tilde{\mathbf{a}}\|_2} &= \underbrace{\begin{pmatrix} \cos(-\theta_z) & -\sin(-\theta_z) & 0 \\ \sin(-\theta_z) & \cos(-\theta_z) & 0 \\ 0 & 0 & 1 \end{pmatrix}}_{\mathbf{R}_z(-\theta_z)} \underbrace{\begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos(-\theta_x) & -\sin(-\theta_x) \\ 0 & \sin(-\theta_x) & \cos(-\theta_x) \end{pmatrix}}_{\mathbf{R}_x(-\theta_x)} \underbrace{\begin{pmatrix} \cos(-\theta_y) & 0 & \sin(-\theta_y) \\ 0 & 1 & 0 \\ -\sin(-\theta_y) & 0 & \cos(-\theta_y) \end{pmatrix}}_{\mathbf{R}_y(-\theta_y)} \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix} \\
 &= \begin{pmatrix} -\cos(-\theta_x) \sin(-\theta_z) \\ \cos(-\theta_x) \cos(-\theta_z) \\ \sin(-\theta_x) \end{pmatrix}
 \end{aligned}$$

Estimating Attitude via Accelerometer in 3D

$$\hat{a} = \frac{\tilde{a}}{\|\tilde{a}\|} = \begin{pmatrix} -\cos(-\theta_x)\sin(-\theta_z) \\ \cos(-\theta_x)\cos(-\theta_z) \\ \sin(-\theta_x) \end{pmatrix}$$

- **Roll:**

Note: **atan2** is typically used to avoid ambiguity

$$\theta_z = -\tan^{-1} \left(-\frac{\hat{a}_x}{\hat{a}_y} \right) = -\text{atan2}(-\hat{a}_x, \hat{a}_y)$$

- **Pitch:**

$$\frac{\hat{a}_z}{\sqrt{\hat{a}_x^2 + \hat{a}_y^2}} = \frac{\sin(-\theta_x)}{\underbrace{\sqrt{\cos^2(-\theta_x) (\sin^2(-\theta_z) + \cos^2(-\theta_z))}}_{=1}} = \tan(-\theta_x)$$

$$\theta_x = -\text{atan2}(\hat{a}_z, \sqrt{\hat{a}_x^2 + \hat{a}_y^2})$$

We can then employ two complementary filters to fuse gyro and accelerometer

Estimating Attitude in 3D via Quaternions

- Convert gyro output $\tilde{\boldsymbol{\omega}} = (\tilde{\omega}_x, \tilde{\omega}_y, \tilde{\omega}_z)$ to instantaneous rotation quaternion by

$$q_{\Delta} = q \left(\underset{\substack{\uparrow \\ \text{angle}}}{\Delta t \|\tilde{\boldsymbol{\omega}}\|}, \frac{\tilde{\boldsymbol{\omega}}}{\|\tilde{\boldsymbol{\omega}}\|} \right)$$

axis

- Integrate instantaneous rotation to accumulated rotations from all previous timesteps by

$$q_{\boldsymbol{\omega}}^{(t+\Delta t)} = q^{(t)} q_{\Delta}$$

- A vector quaternion can then be rotated from body to world frame by

$$q_{\mathbf{u}}^{(world)} = q_{\boldsymbol{\omega}}^{(t+\Delta t)} q_{\mathbf{u}}^{(body)} q_{\boldsymbol{\omega}}^{(t+\Delta t)^{-1}} = q^{(t)} q_{\Delta} q_{\mathbf{u}}^{(body)} q_{\Delta}^{-1} q^{(t)^{-1}}$$

Vector quaternion: A vector converted to a quaternion - scalar component zero, vector component is the vector

Tilt Correction via Quaternions

- We now know how to track attitude with gyroscope only, but how to fuse with accelerometer?
- Accelerometer output $\tilde{\mathbf{a}} = (\tilde{a}_x, \tilde{a}_y, \tilde{a}_z)$ converted to a vector quaternion by

$$q_{\mathbf{a}}^{(body)} = 0 + i\tilde{a}_x + j\tilde{a}_y + k\tilde{a}_z$$

- We can then rotate this from body to world using our last estimate from gyro

$$q_{\mathbf{a}}^{(world)} = q_{\omega}^{(t+\Delta t)} q_{\mathbf{a}}^{(body)} q_{\omega}^{(t+\Delta t)-1}$$

- If our estimate is correct, this should point UP (assuming only gravity - no noise, no external acc.)

$$q_{\mathbf{a}}^{(world)} = q_{up}^{(world)} = 0 + i0 + j9.81 + k0$$

- Our goal is to compute a quaternion q_t that would rotate $q_{\mathbf{a}}^{(world)}$ to UP:

$$q_{up}^{(world)} = q_t q_{\mathbf{a}}^{(world)} q_t^{-1}$$

Tilt Correction via Quaternions

- A solution: Calculate its rotation axis \mathbf{n} and rotation angle ϕ , then convert into quaternion

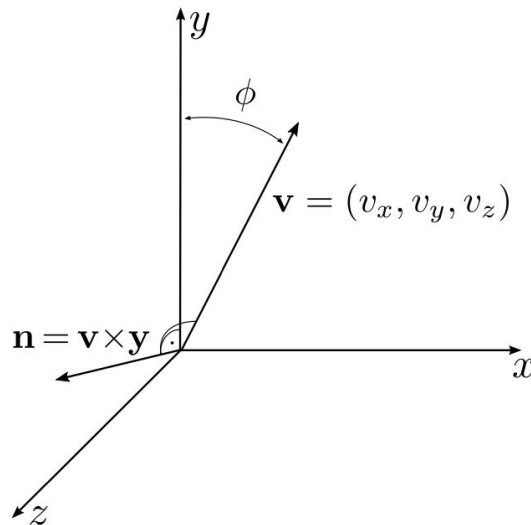
$$q_t = q\left(\phi, \frac{\mathbf{n}}{\|\mathbf{n}\|}\right), \quad \mathbf{n} = \begin{pmatrix} v_x \\ v_y \\ v_z \end{pmatrix} \times \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix} = \begin{pmatrix} -v_z \\ 0 \\ v_x \end{pmatrix}, \quad \phi = \cos^{-1}(v_y)$$

normalized acceleration vector cross product

- Then, we can apply complementary filtering

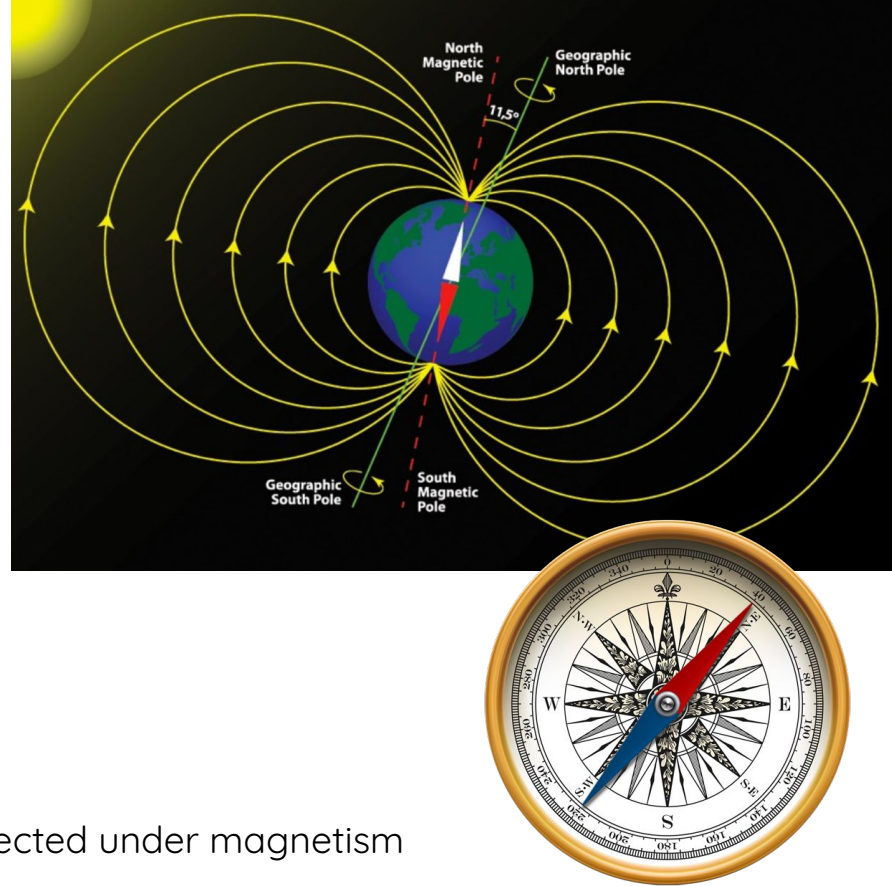
$$\overset{\text{final fused estimate}}{q_c^{(t+\Delta t)}} = q\left((1 - \alpha)\phi, \frac{\mathbf{n}}{\|\mathbf{n}\|}\right) q_\omega^{(t+\Delta t)}$$

- Intuition: Rotate “a bit” into the direction of tilt correction
- aka Mahony filter
- (Similar to 2D case) this can’t correct for yaw...



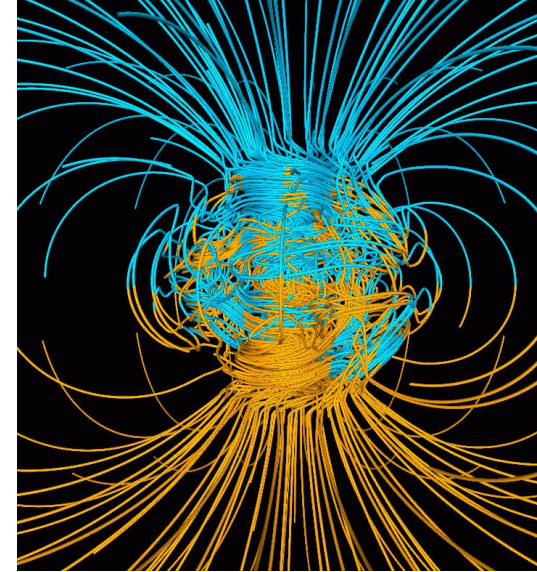
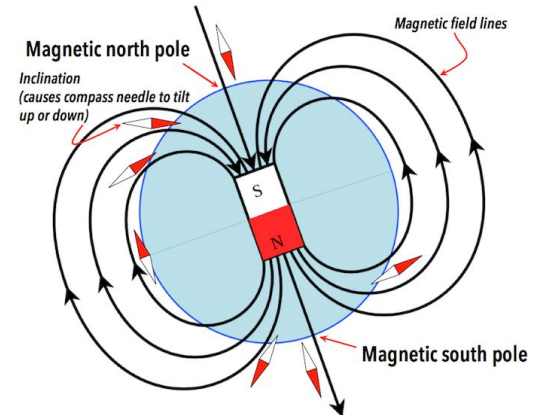
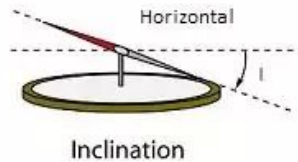
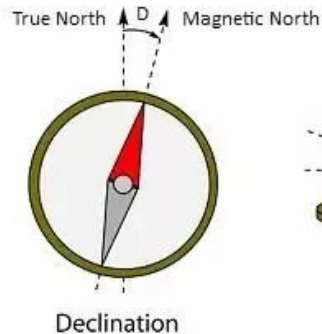
Magnetometer

- Essentially a compass
- Measures Earth's magnetic field
- In units of Gauss or tesla (T)
- 3 axes (the magnetic field is 3D!)
- Measures both strength and direction
- Can be used to estimate **yaw**!
 - Similar sensor fusion methods apply
- Sensor operation principles
 - Hall effect: Current in a conductor gets deflected under magnetism
 - Magnetoresistance: Resistance difference due to magnetism



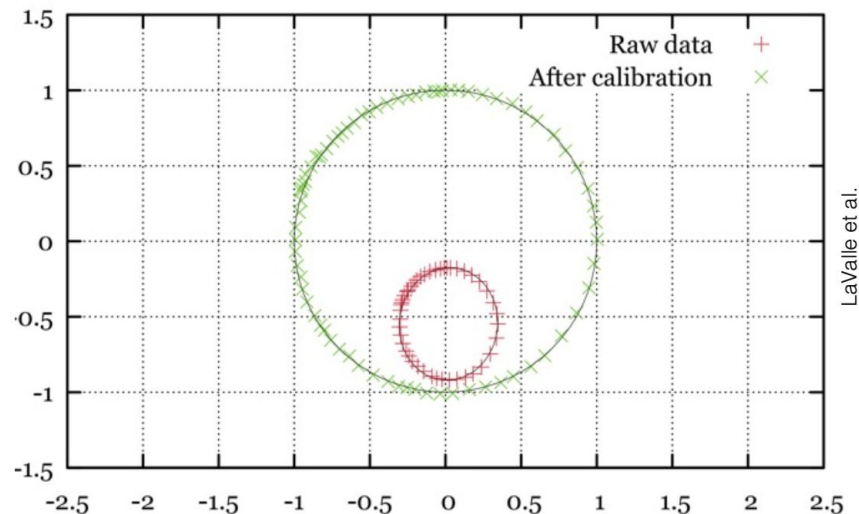
Earth's Magnetic Field

- Earth's magnetic field is very complex!
- Generally points to magnetic north
 - Difference between magnetic north and true north is **declination**
- Field is not oriented horizontally flat
 - The vertical angle is **inclination**
- Actual declination and inclination depends on place on Earth
 - Modelable, but we need to know rough location



Magnetometer Weaknesses

- Affected by external factors, especially indoors (metallic objects etc.)
- Requires frequent calibration (e.g. figure 8 motion)

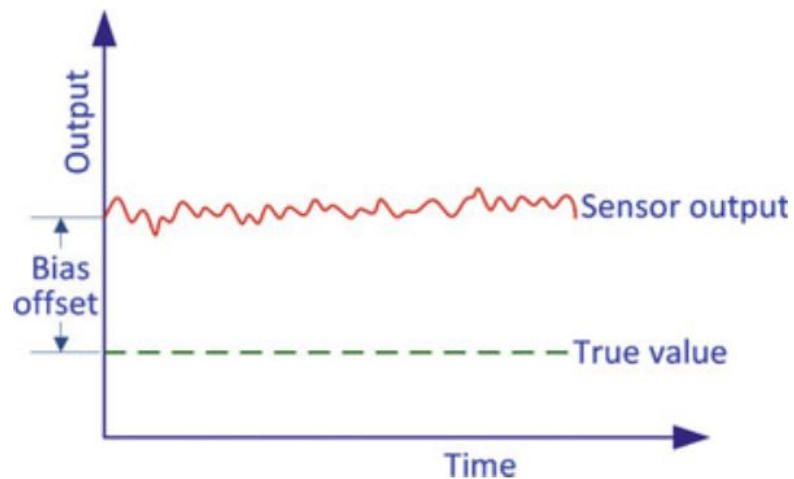


Inertial Sensor Errors

- There are many error sources which may affect inertial tracking badly
- Modelling these errors will be important for sensor/data fusion
- Two main categories: Systematic errors and random errors
 - Systematic errors may be mitigated via calibration
 - Random errors can't be estimated and removed, they can only be modelled

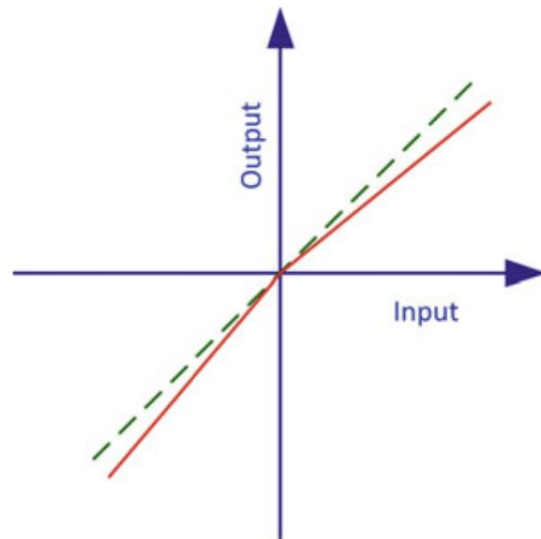
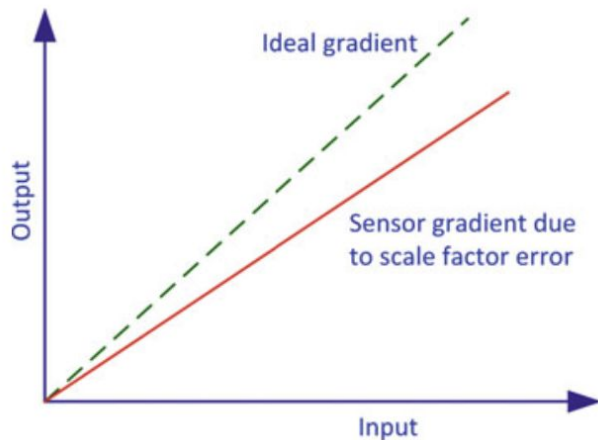
Inertial Sensor Errors: Bias

- An offset between measurement and actual
- Independent of the actual input (force or angular rate)
- Changes with time and environmental factors (like temperature)
- A very common error, especially for gyroscopes



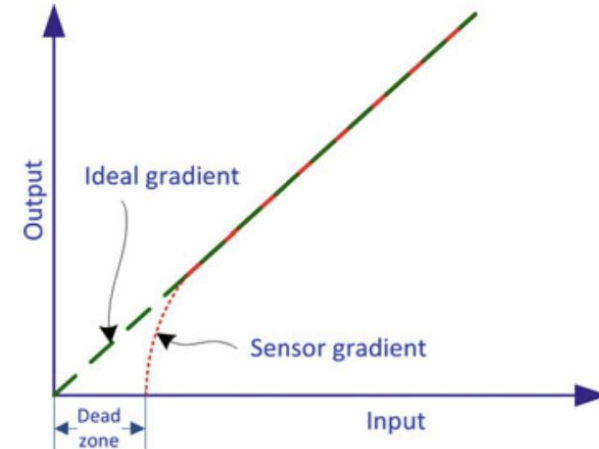
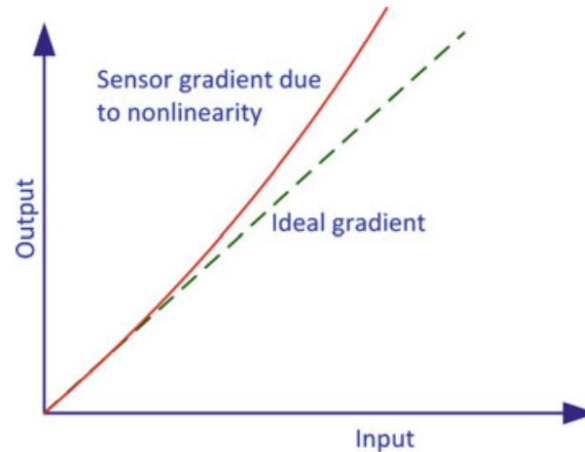
Inertial Sensor Errors: Scaling Error

- Deviation of the input-output gradient
- Proportional to actual input
- A systematic over/under-estimation
- May exhibit different characteristics based on sign



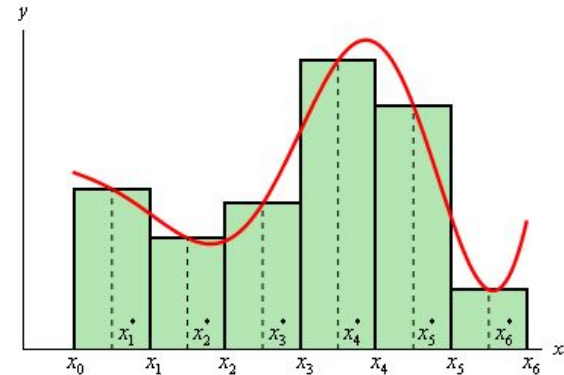
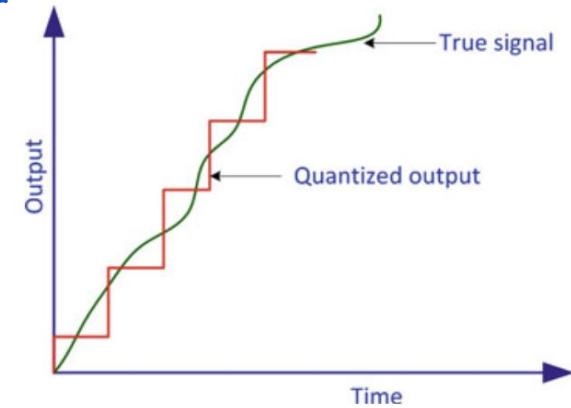
Inertial Sensor Errors: Non-linearity

- Non-linearity between input and output
- Similar to scaling error (amount changes with input)
- Dead zone: A specific deficiency around zero



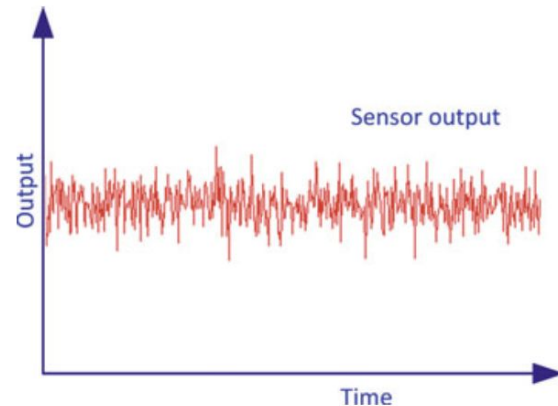
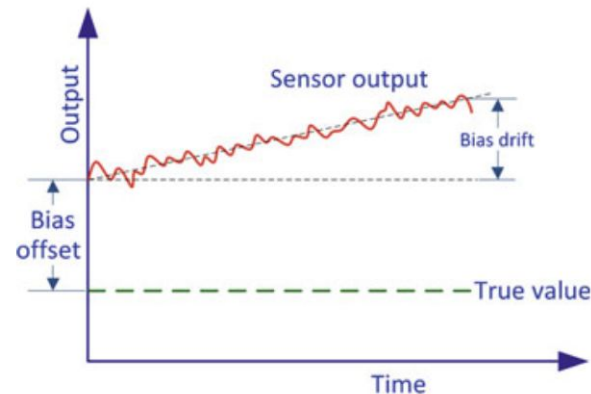
Inertial Sensor Errors: Quantization Error

- A common error source for all digital systems
- Due to discretization
- Based on sensor sensitivity and resolution
- Even if no quantization error, sampling creates a similar effect
 - Values between samples missed
 - Causes errors in integration
 - Decreases as sampling rate increases
 - Different integration schemes exist to reduce this, but not a big factor in the face of other errors



Inertial Sensor Errors: Random Errors

- **Bias drift:** Bias itself may change over time
- **Noise:** aka white noise, usually modelled as Gaussian
- Bias drift and noise are random errors
- Different from systematic errors:
 - Impossible to estimate and remove
 - But can be modeled (usually with a Gaussian, especially for noise)



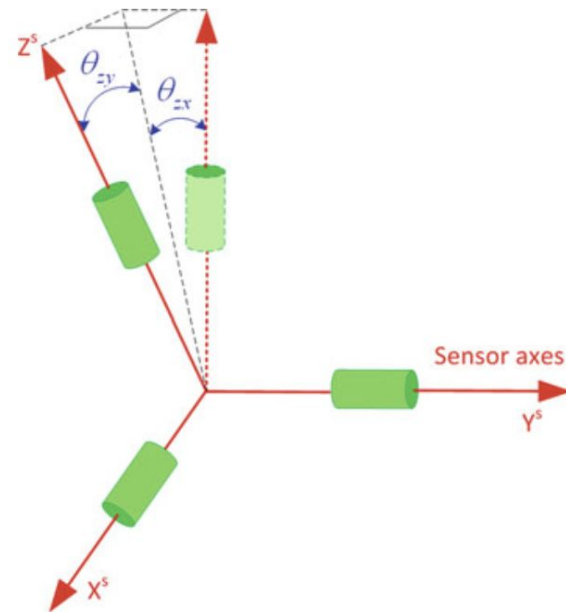
Inertial Sensor Errors: Other Errors

- **Non-orthogonality error:**

Sensor axes are not perfectly orthogonal to each other

- **Misalignment error:**

Sensor axes are not perfectly aligned with the body



How to quantify sensor errors?

- A common technique: **Allan Variance**
- A time domain analysis technique for characterizing noise and stability
- Can be applied to any time-series data sampled at uniform intervals
 1. Divide data into n bins of length t
 2. Average the data in each bin to obtain a list of averages

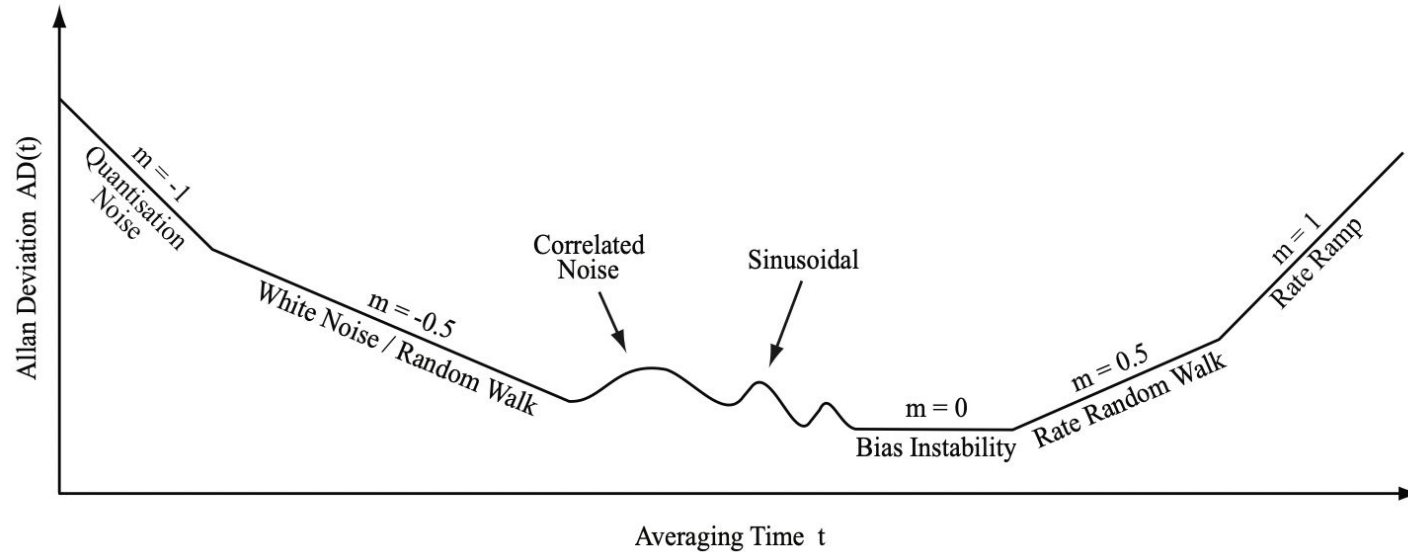
$$(a(t)_1, a(t)_2, \dots, a(t)_n)$$

3. Calculate Allan Variance and Deviation by

$$\text{AVAR}(t) = \frac{1}{2 \cdot (n-1)} \sum_{i=1}^{n-1} (a(t)_{i+1} - a(t)_i)^2 \quad \text{AD}(t) = \sqrt{\text{AVAR}(t)}$$

4. Plot Allan Deviance as a function of t on a log-log scale

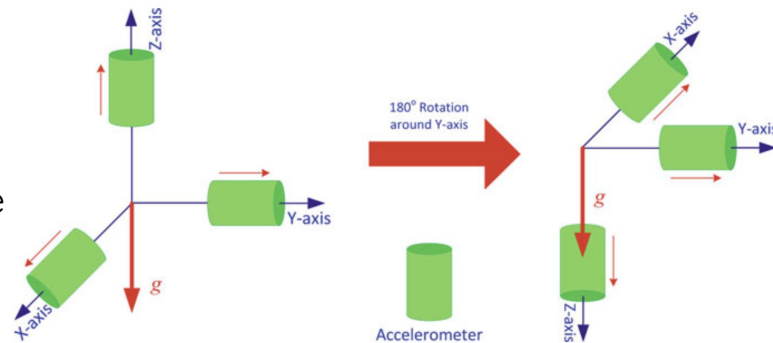
Allan Variance Plot



- Different processes appear in different regions of t
- Fitting curves around different regions give us numerical error characteristics

Inertial Sensor Calibration

- Six-Position Static Test
 - Mount sensor on a level table with each sensitive axis alternatively pointing up and down
 - Six positions for three axes
 - Especially important for accelerometers
- Angle Rate Tests
 - Rotate IMU with known rotation rates
 - Especially important for gyroscopes
 - Big and costly devices



Classification of IMUs

Performance	Strategic grade	Navigation grade	Tactical grade	Commercial grade ^a
Positional error	30 m/h h < 100 m/h	1 nmi ^b /h or .5 m/s	10–20 nmi/h	Large variation
Gyroscope drift	0.0001–0.001	<0.01 °/h	1–10°/h	0.1°/s
Gyroscope random walk	–	<0.002°/√h	0.05– <0.02°/√h	Several °/√h
Accelerometer bias	0.1–1	<100 µg	1–5 mg	100–1,000 µg
Applications	Submarines Intercontinental ballistic missile	General navigation high precision georeferencing mapping	Integrated with GPS for mapping Weapons (short time))	Research Low cost navigation pedometers, Antilock breaking active suspension, airbags

^a Also called automotive grade

^b 1 nautical mile (nmi) ≈ 6,076 ft ≈ 1,851 m

Conclusion

- A very good talk on inertial sensors and sensor fusion
<https://youtu.be/C7JQ7Rpwn2k>
- Inertial sensors are good for short term tracking, but not sufficient by themselves
- In the next lecture, we'll talk about more sophisticated methods of sensor fusion
- Not just fusion of raw sensor data, but “data fusion” - Fusion of high level estimates