Robótica Móvil

Segundo cuatrimestre de 2025

Departamento de Computación - FCEyN - UBA

Percepción (Parte II) - clase 10

Sensado con LiDAR e IMU

Sensado

¿Qué sensores vimos hasta ahora?

- → Sensor de contacto (un botón o *bumper*).
- → Telémetro infrarrojo (IR).
- → Sonar de ultrasonido.
- → Odómetro (*encoder*).

Para poder resolver los problemas que vamos a estudiar en la segunda parte de la materia (localización, planificación de trayectorias, construcción de mapas) no alcanzan. Hoy vamos a ver:

- → IMU (Inertial Measurement Unit, giróscopo + acelerómetro).
- → LiDAR (Light Detection and Ranging).

Giróscopo: principio de funcionamiento

El giróscopo funciona gracias al efecto Coriolis:

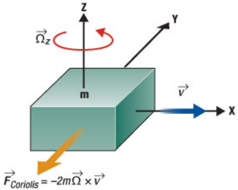


Figura: Efecto Coriolis.

- Cuando una masa m se mueve en una dirección $\mathbf{v}$ respecto de un sistema de referencia con una velocidad angular $\mathbf{\Omega_z}$, la masa experimenta una fuerza en la dirección perpendicular al eje de rotación del sistema y a la velocidad del cuerpo (efecto Coriolis).
- El desplazamiento físico de la masa causado por el efecto Coriolis puede ser leído por un sensor capacitvo que mide la capacitancia C en función de la permitividad del dieléctrico ε, el área efectiva de las placas A y la distancia entre ellas d: $C = \frac{\varepsilon A}{d}$.

Giróscopo: principio de funcionamiento

En general, los giróscopos MEMS (Microelectromechanical Systems) usan una configuración de diapasón:

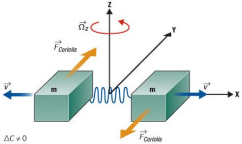


Figura: Efecto Coriolis en una configuración de diapasón.

- → Dos masas oscilan constantemente en direcciones opuestas. Al aplicar una velocidad angular al sistema, las fuerzas de Coriolis actúan en sentidos contrarios para cada masa, resultanto en un diferencia capacitiva.
- → Esta diferencia capacitiva es proporcional a la velocidad angular, y puede ser convertida en una señal analógica o digital.

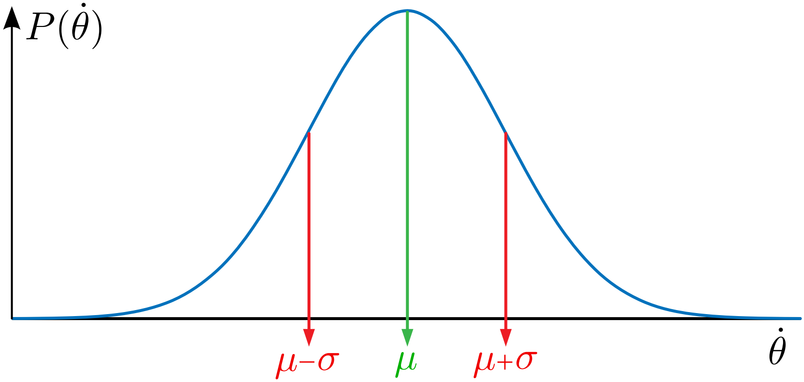
Giróscopo: modelo

- El giróscopo está sujeto a un sesgo (*bias*) y a un ruido.
- Utilizamos un modelo para describir la velocidad angular ${}^{IMU}\mathbf{\Omega} = [\omega_x,\, \omega_y,\, \omega_z]$ medida por el sensor en un instante de tiempo:

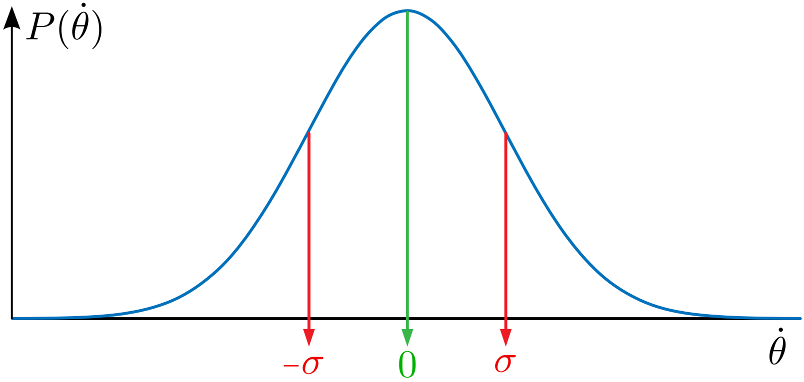
$$ {}^{IMU}\mathbf{\Omega} = {}^{IMU}\mathbf{\Omega}^* + \mathbf{b}_{\mathbf{\Omega}} + \boldsymbol{\mu}_{\boldsymbol{\Omega}} $$

a partir de la velocidad angular real del sistema ${}^{IMU}\mathbf{\Omega}^*$, un bias constante $\mathbf{b}_{\mathbf{\Omega}}$ y un ruido (gaussiano con media 0) de medición $\boldsymbol{\mu}_{\boldsymbol{\Omega}}$.

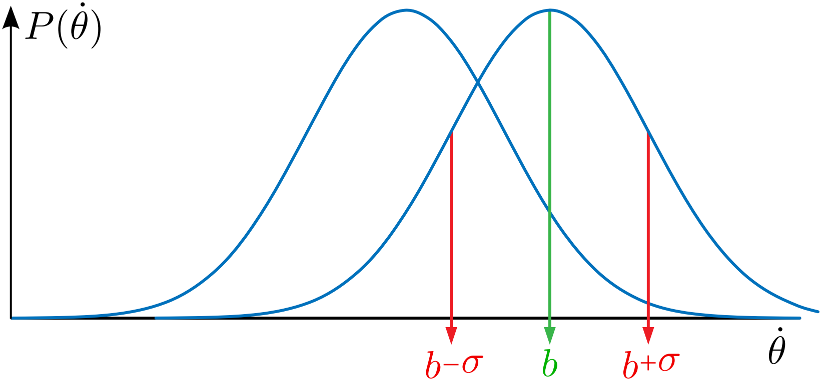
Ruido y Sesgo



Ruido y Sesgo



Ruido y Sesgo



¿Cómo se podría corregir esto?
Tenemos que restar el bias!

Giróscopo: calibración

¿Por qué el **giróscopo** necesita calibrarse?

- → Errores de calibración de fábrica.
- → Cambios de temperatura (*warm-up effect*).
- → Cambios en la fuerza de gravedad (dependiendo de la aplicación).
- → Si queremos obtener la velocidad angular real ${}^{IMU}\mathbf{\Omega}^*$, hace falta poder estimar los otros términos.
- → $\boldsymbol{\mu_\Omega}$ lo modelamos como un ruido gaussiano con media cero. Un promedio del mismo tenderá a $\mathbf{0}$ cuando el n tienda a ∞.
- → Entonces, suponiendo que podemos dejar el giróscopo completamente quieto en un intervalo de tiempo (de calibración), y tomamos el promedio de n mediciones:

$$\frac{\sum {}^{IMU}\mathbf{\Omega}}{n} = \frac{\sum {}^{IMU}\mathbf{\Omega}^*}{n} + \frac{\sum \mathbf{b_\Omega}}{n} + \frac{\sum \boldsymbol{\mu_\Omega}}{n} = \frac{\mathbf{0}}{n} + \frac{n \cdot \mathbf{b_\Omega}}{n} + \frac{\mathbf{0}}{n} = \mathbf{b_\Omega}$$

obtenemos una estimación del sesgo o bias del sensor.

Giróscopo: integración

- El girósocopo nos da una estimación de la velocidad angular para cada instante de tiempo.
- Sin embargo, nos interesa saber la orientación del robot, no su velocidad angular.
- Podemos estimar la orientación integrando numéricamente la velocidad angular en pequeños incrementos:

$$\Delta\theta = \Delta t \left({}^{IMU}\boldsymbol{\Omega} - \mathbf{b}_{\boldsymbol{\Omega}} \right)$$
$$\theta_{t_{k+1}} = \theta_{t_k} + \Delta\theta$$

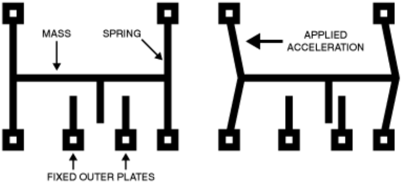
donde ${}^{IMU}\boldsymbol{\Omega}$ es la última velocidad angular medida, $\mathbf{b}_{\boldsymbol{\Omega}}$ es el bias previamente estimado y $\Delta t = t_{k+1} - t_k$ es el tiempo transcurrido entre la última medición y la actual.

¿Qué sucede cuando trabajamos con cuaterniones?

$$Q_{t_{k+1}} = Q_{t_k} \cdot \Delta Q$$

Acelerómetro: principio de funcionamiento

El acelerómetro nos permite medir las aceleraciones.



- → Un acelerómetro MEMS típicamente consiste de una masa con la libertad de moverse en una dimensión, y dos conjuntos de placas capacitivas: uno de ellos sujeto al sustrato, y otro sujeto a la masa.
- → Una aceleración del sistema, produce un desplazamiento de la masa, generando una diferencia en la capacitancia entre las placas fijas y móviles.
- → A partir de esa diferencia en la capacitancia se estima la aceleración.

Acelerómetro: modelo

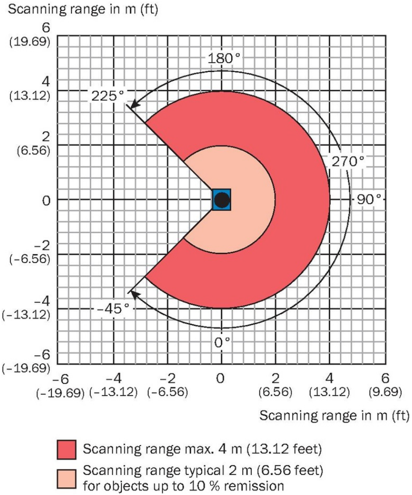
Al igual que con el giróscopo, utilizamos un modelo para describir la aceleración ${}^{IMU}\mathbf{a} = [a_x,\ a_y,\ a_z]$ experimentada por el sensor en un instante de tiempo:

$$
{}^{IMU}\mathbf{a} = {}^{IMU}\mathbf{a}^* - {}^{IMU}\mathbf{g}^* + \mathbf{b}_a + \boldsymbol{\mu_a}
$$

donde ${}^{IMU}\mathbf{g}^*$ es la aceleración de la gravedad, $\boldsymbol{\mu_a}$ es el ruido gaussiano con media 0 de la medición y $\mathbf{b}_a$ un desvío constante (bias) de los valores.

En general se puede considerar que $\mathbf{b}_a = 0$ ya que suele ser despreciable.

Telémetro láser (o láser)



Datos importantes:

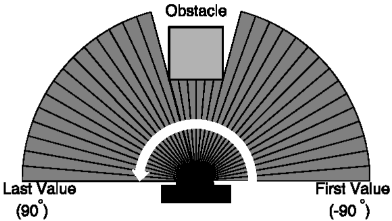
→ Apertura angular (min, máx).

→ Resolución o incremento.

→ Rango.

Telémetro láser: mediciones

- Una medición en un instante de tiempo determinado, se suele representar con un arreglo de n posiciones.
- Cada posición representa la medición en una dirección angular φ distinta, y cada valor indica la distancia r que midió el láser en esa dirección particular.
- Luego, para cada instante de medición obtenemos una colección de coordenadas polares (r_i, φ_i).



El ángulo φ_i correspondiente a la posición i del arreglo se puede encontrar como:

$$\varphi_i = \varphi_{min} + i\frac{\varphi_{max} - \varphi_{min}}{n}$$

Telémetro láser: representación en coordenadas polares

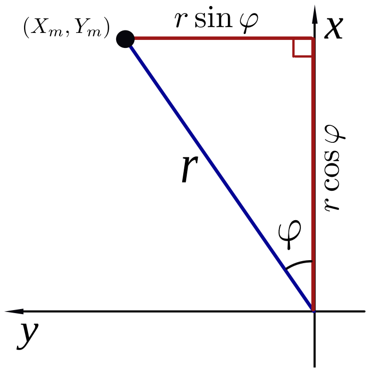
- Cuando detectamos obstáculos, es útil saber su posición en coordenadas cartesianas (x_i, y_i), no polares.
- Para pasar un punto representado en coordenada polares (r, φ) a una representación cartesiana (x, y) hacemos:

$$X_m = r \cdot \cos(\varphi)$$

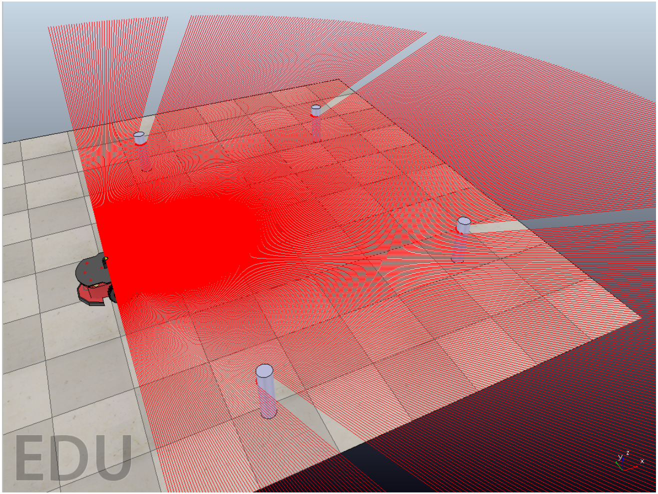
$$Y_m = r \cdot \sin(\varphi)$$

$$r = \sqrt{X_m^2 + Y_m^2}$$

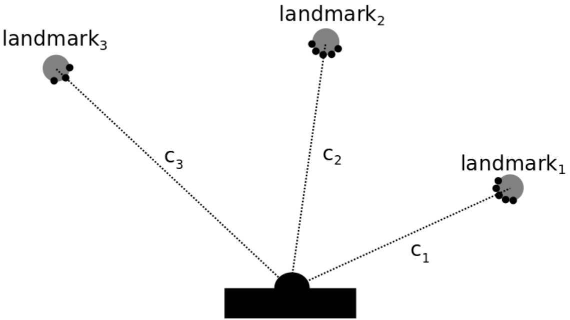
$$\varphi = atan2(Y_m, X_m)$$



Detección de postes mediante un sensor LiDAR 2D



Detección de postes mediante un sensor LiDAR 2D



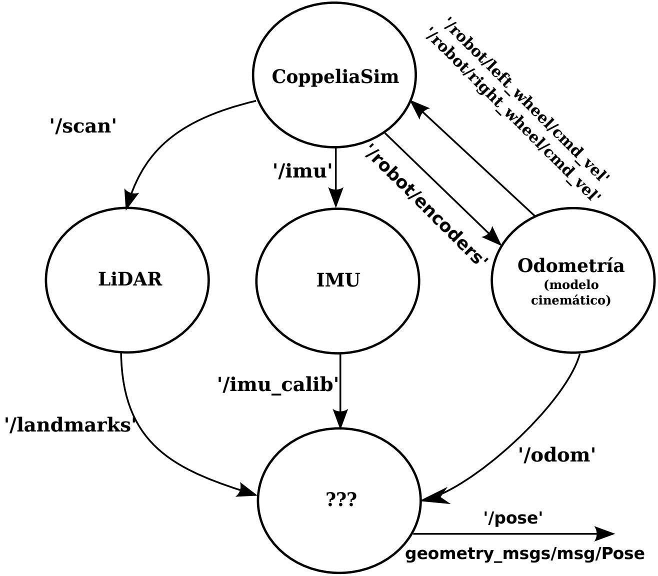
l_k^i : son las k mediciones del landmark i $\qquad$ $l_{min} = min\_dist(l_k^i, laser)$

$$c_i = l_{min} + normalize(l_{min}) \times radio(\text{landmark}_i)$$

Por último, esto está escrito en coordenadas del sensor, para pasarlo a coordenadas del robot finalmente hay que hacer:

$$^{R}c_i = {}^{R}T_S\, {}^{S}c_i$$

Para el Taller, ¿qué vamos a ver ahora?



Mensajes de la IMU

sensor_msgs/msg/Imu

std_msgs/Header header
geometry_msgs/msg/Quaternion orientation
geometry_msgs/msg/Vector3 angular_velocity
geometry_msgs/msg/Vector3 linear_acceleration

std_msgs/Header

uint32 seq
time stamp
string frame_id

La información de la orientación no viene dada directamente por el sensor (porque mide velocidades, no orientaciones). Por lo tanto, **depende de la integración que utilicemos sobre las mediciones.**

$$Q_{t_{k+1}} = Q_{t_k} \cdot \Delta Q$$

Mensajes del sensor LiDAR

sensor_msgs/msg/LaserScan

std_msgs/Header header
float32 angle_min
float32 angle_max
float32 angle_increment
float32 range_min
float32 range_max
float32[] ranges

std_msgs/Header

uint32 seq
time stamp
string frame_id

La información va a estar en relación al marco de coordenadas del láser. Se aplican transformaciones para **traducir la información al marco de coordenadas del robot**.

$$^{R}c_i = {}^{R}T_S \, {}^{S}c_i$$

Mensajes de las referencias/landmarks

robmovil_msgs/msg/LandmarkArray

std_msgs/Header header
Landmark[] landmarks

robmovil_msgs/msg/Landmark

float32 range
float32 bearing

Un poco de la librería geométrica tf2

https://wiki.ros.org/tf2

tf2::Vector3:

- → **vec1 + vec2** : suma de vectores
- → **vec1** · **vec2** : multiplicación de vectores **elemento a elemento**
- → **vec1.dot(vec2)** : producto escalar entre vectores
- → **vec1** · **scalar** : multiplicación de vectores por un escalar
- → **vec1 / scalar** : división de vectores por un escalar
- → **vec1.length()** : norma 2 del vector
- → **vec1.normalize()** : se normaliza el vector

tf2::Quaternion delta_q;
delta_q.setRPY(roll, pitch, yaw);

- → calcula ΔQ cuando queremos resolver $Q_{t_{k+1}} = Q_{t_k} \cdot \Delta Q$