Percepción y control
Grado en Ingeniería de Computadores

Práctica 1: **Sensores y actuadores en ROS.**

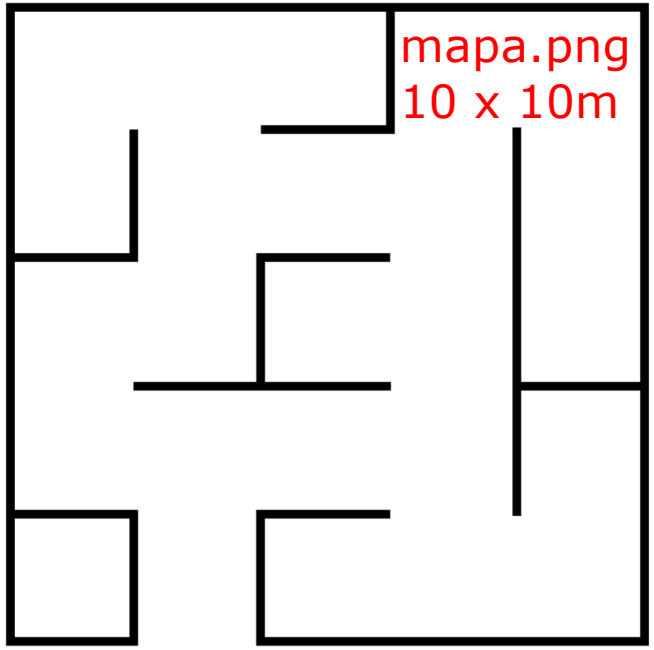
José L. Martín Sánchez, Ángel Llamazares Llamazares, Daniel Pizarro Pérez, Biel Piero Eloy Alvarado Vásquez

Departamento de Electrónica. Universidad de Alcalá

1. Introducción
2. Sensores en ROS
   1. Caracterización de los sensores de odometría.
   2. Caracterización de los sensores de distancia ultrasónicos.
   3. Caracterización del sensor de distancia láser.
3. Actuadores en ROS
4. Memoria.
5. Recordatorio: conexión al robot real.

- El **objetivo** de esta práctica es realizar una revisión completa de los conceptos aprendidos durante el Tema 2 "Sistemas de Percepción", empleando para ello la plataforma de desarrollo robótico ROS, el simulador STDR y el robot real Amigobot.
- Para ello, se va a trabajar con los topics de los sensores de distancia (**sonar**, **laser**) y odometría (**odom**) con una publicación en el topic de los motores (**cmd_vel**).

Objetivo final: disponer de funciones que permitan al robot percibir el entorno y moverse por el mismo con funciones sencillas de avanzar y girar



Se utilizará en primer lugar el simulador STDR y, posteriormente, el robot real Amigobot.

- **Caracterización de los sensores de odometría**
  - Suscripción al topic **odom**.
  - Publicación del topic **cmd_vel**.
- Moviendo el robot en línea recta por un lado y haciendo un giro por otro, a distintas velocidades, se pide:
  - Describir la información que nos ofrece el mensaje disponible en el topic **odom** (rostopic echo).
  - Medir la resolución máxima (**q**) de odometría lineal y angular con las diferentes velocidades que indican en el enunciado de la práctica para el simulador STDR y para el robot real.

V (ms^{-1})	Ω ($rads^{-1}$)	q_lineal (m)	q_angular (r)
0.1	0.0	…	…
0.3	0.0	…	…
…	…	…	…

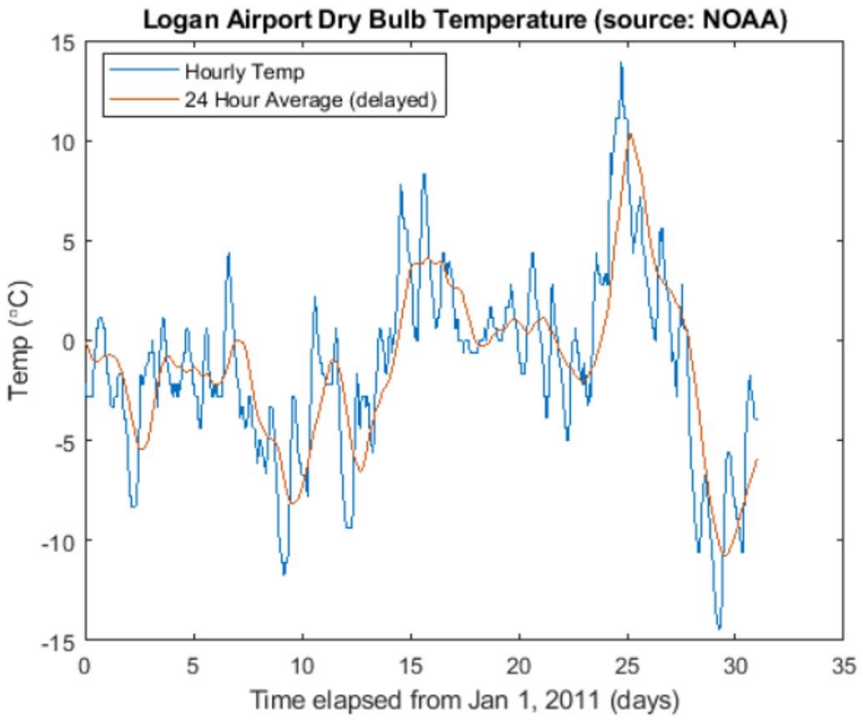
- ¿Qué es la resolución?
  - Mínima diferencia entre dos valores
  - Normalmente: limite inferior del rango dinámico = resolución
  - *Por ejemplo, en sensores digitales coincide con la resolución del A/D:* 5V / 255 = 19,6mV (8 bit)
  - *En otras palabras: "¿Cuánto tiene que cambiar la magnitud que está midiendo para que el sensor detecte una variación?"*
- Ejemplo: La temperatura real de una habitación cambia desde **21ºC** a **22ºC** de manera continua.
  - Sensor 1 - Mide inicialmente **21ºC** y, cuando cambia, mide **22ºC**

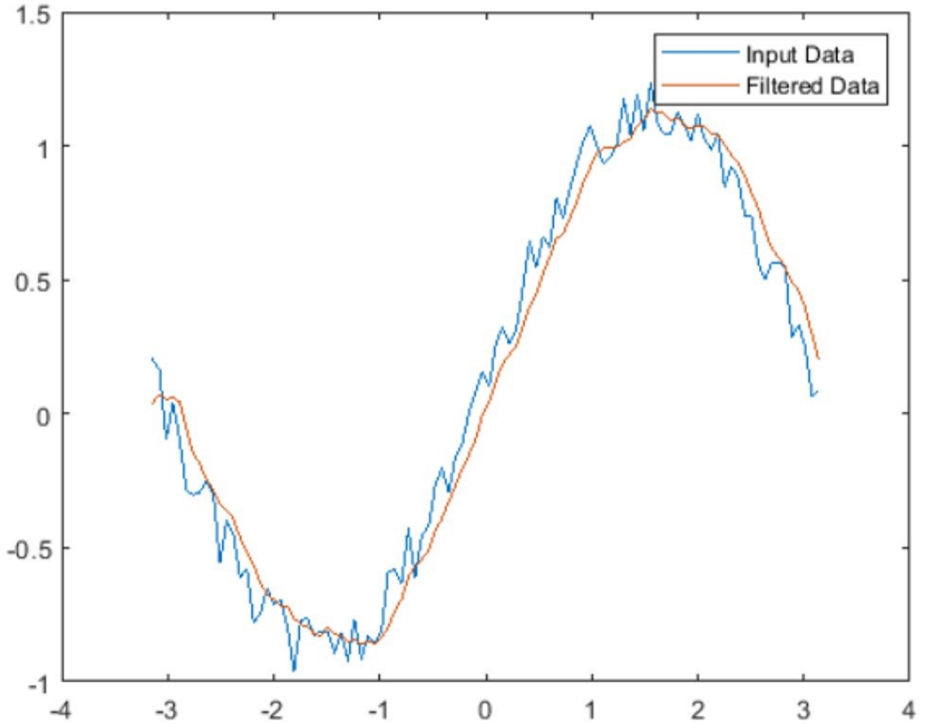
    => *Sensor 1 - Resolución = 1ºC*
  - Sensor 2 - Mide inicialmente **21ºC** y, cuando cambia, mide **21,5ºC** y, posteriormente, **22ºC**

    => *Sensor 2 - Resolución = 0,5ºC*

- Caracterización de **los sensores de distancia ultrasónicos**
  - Suscripción a los topic **sonar**.
- Posicionando el robot de tal forma que exista una distancia de 2m desde uno de los sensores sónar hasta una pared, se pide:
  - Describir la información que el topic **sonar** e indicar que sónar se ha elegido.
  - Obtener 1000 medidas de distancia, dibujarlas y analizarlas.
  - Implementar un filtro de media móvil de 5 valores, dibujar el resultado y analizarlo.
- Posicionando el robot en las distintas casillas del mapa dado:
  - Seleccionar los sensores sonar que podrían ser útiles para obtener las cuatro rectas que definen las paredes que lo rodean e identificar el tipo de casilla en el que está el robot.
  - Comprobar que las orientaciones de las rectas son paralelas dos a dos y perpendiculares entre ellas y definir una **función de calidad** para obtener el grado de confianza de dichas paredes empleando, por ejemplo, la relación entre las diferentes pendientes.

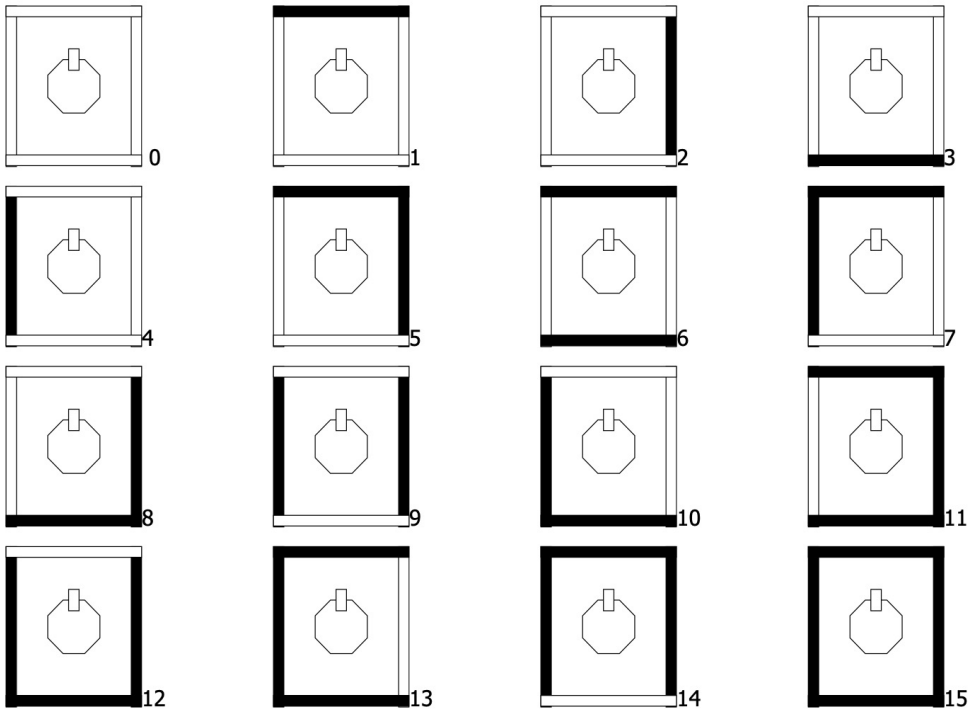
- ¿Qué es un filtro de media móvil?





https://es.mathworks.com/help/matlab/ref/filter.html

- Diseñar una **función** que indique, mediante un código, el número de paredes que se encuentra el robot en sus laterales, indicando el **grado de confianza** basado en la función de calidad definida en el apartado anterior.



- Comprobar los resultados de la función diseñada para el simulado STDR y para el robot real.

Robot_real/Simulador	Combinación real	Combinación identificada	Grado confianza
Simulador	0	…	…
Simulador	1	…	…
…	…	…	…

- Caracterización del **sensor láser**
  - Suscripción al topic **láser**.
- Repetir el estudio realizado para los sensores de distancia ultrasónicos y comparar los resultados.

- Utilizando el topic cmd_vel, se pide:
  - Diseñar una función **avanzar** que reciba como parámetro la distancia a avanzar (2m, 4m...).
  - Diseñar una función **girar** que se reciba como parámetro el ángulo a girar (45º, 90º,...).
  - Empleando el simulador STDR, y utilizando las funciones **avanzar** y **girar**, navegar con el robot desde la esquina inferior derecha (4,-4,0) del mapa hasta la.
  - Realizar un recorrido con el robot real concatenando los siguientes tramos:
    - *Un tramo recto de 2m*
    - *Un giro de 90º*
    - *Un tramo recto de 1m o 1 giro de -90º*
    - *1 tramo recto de 1m*
  - Comprobar el error final que se ha obtenido con el robot real, indicando cuál es el error de odometría global que se ha cometido.

- Entrega de **un único archivo** (en un archivo comprimido .zip / .rar) de la práctica que contenga:
  - Un informe/memoria (en .pdf), que incluya los datos / tablas solicitadas en el guion, capturas de pantalla de los códigos programados, los resultados numéricos o gráficos obtenidos, los problemas encontrados y soluciones propuestas, las respuestas a aquellas preguntas / comparativas / razonamientos que se indican en el guion y unas conclusiones. Para obtener la máxima puntuación, es imprescindible haber **completado y justificado TODOS los apartados** solicitados en el enunciado de la práctica.
  - Archivos fuente de los apartados implementados en Matlab (.m / .mlx)

- Recordatorio de la Guía Rápida de Introducción a ROS (apartados 10 y 11)
  - Es necesario estar conectado a la red Wifi Amigobot:
    - *ESSID: Amigobot WiFi*
    - *pass: Robotica1718!*
  - Es necesario cambiar IP ***ROS_MASTER_URI*** por la IP del robot*:*
    - *El roscore se encuentra en el robot real.*
  - Es necesario cambiar los topics:

	Topic Simulador	Topic Robot Real
Odometría	/robot0/odom	/pose
Comandos de velocidad	/robot0/cmd_vel	/cmd_vel
Datos del sonar i (i desde 0 hasta 7)	/robot0/sonar_i	/sonar_i
Datos del láser	/robot0/laser_1	/scan
Habilitación de los motores	No hay topic	/cmd_motor_state

```
%% DECLARACIÓN DE SUBSCRIBERS
odom = rossubscriber('/robot0/odom'); % Subscripción a la odometría
laser = rossubscriber('/robot0/laser_1', rostype.sensor_msgs_LaserScan);
sonar0 = rossubscriber('/robot0/sonar_0', rostype.sensor_msgs_Range);

%% DECLARACIÓN DE PUBLISHERS
pub = rospublisher('/robot0/cmd_vel', 'geometry_msgs/Twist'); %

%% Nos aseguramos recibir un mensaje relacionado con el robot
while (strcmp(odom.LatestMessage.ChildFrameId,'robot0')~=1)
    odom.LatestMessage
end
```

```
%% DECLARACIÓN DE SUBSCRIBERS
odom=rossubscriber('/pose'); % Subscripción a la odometría
laser = rossubscriber('/scan', rostype.sensor_msgs_LaserScan);
sonar0 = rossubscriber('/sonar_0', rostype.sensor_msgs_Range);

%% DECLARACIÓN DE PUBLISHERS
pub = rospublisher('/cmd_vel', 'geometry_msgs/Twist'); %
pub_enable = rospublisher('/cmd_motor_state', 'std_msgs/Int32');

msg_enable_motor = rosmessage(pub_enable);

%Activación de los motores enviando enable_motor = 1
msg_enable_motor.Data=1;
send(pub_enable,msg_enable_motor);

%% Nos aseguramos recibir un mensaje relacionado con el robot
while (strcmp(odom.LatestMessage.ChildFrameId,'base_link')~=1)
    odom.LatestMessage
end
```