Practica 1

Sensores y actuadores en ROS

Grupo Pink Floyd

Víctor Sanavia
Iker Valdezate

Sensores y actuadores en ROS 0
Edometría 2
Ejercicio1 2
Ejercicio 2 2
Ejercicio 3 3
Ultrasonidos 4
Ejercicio 1 4
Ejercicio 2 5
Ejercicio 3 6
Ejercicio 4 7
Ejercicio 5 7
Ejercicio 6 8
Ejercicio 7 9
Ejercicio 8 10
Laser 11
Ejercicio 1 11
Ejercicio 2 11
Ejercicio 3 11
Ejercicio 4 12
Ejercicio 5 13
Ejercicio 6 14
Ejercicio 7 14
Ejercicio 8 15
Ejercicio 9 16
Ejercicio 10 16
Actuadores 16
Ejercicio 1 16
Ejercicio 2 16
Ejercicio 3 16
Ejercicio 4 17
Conclusión 17

Edometría

Ejercicio1

Indique y describa la información que nos ofrece el mensaje disponible en el topic odom. Muestre algún ejemplo de captura

El topic *odom* nos proporciona información relacionada con la posición y orientación del robot respecto de su movimiento en el espacio. En este topic, podemos distinguir dos tipos de mensajes:

- **Pose** à Hace referencia a la posición y orientación del robot, representada mediante coordenadas tridimensionales (x, y, z) en cuanto a la posición, y la orientación está representada mediante un cuaternión
- **Twist** à Hace referencia a la velocidad lineal y angular del robot, representadas ambas características por coordenadas tridimensionales (x, y, z)

```
    MessageType: 'nav_msgs/Odometry'
         Header: [1×1 Header]
           Pose: [1×1 PoseWithCovariance]
          Twist: [1×1 TwistWithCovariance]
    ChildFrameId: 'robot0'
```

Ilustración 1 Topic Odom

El script que se encarga de mostrarnos esta información se llama *MensajeOdom.m*, dentro de la carpeta *SensoresOdometría*

Ejercicio 2

Mida la resolución máxima (q) de odometría lineal y angular con las diferentes combinaciones de velocidades propuestas en el simulador STDR. Construya una tabla como la que se encuentra a continuación con las 8 combinaciones propuestas

Para la realización de este ejercicio, lo que hemos hecho ha sido calcular la distancia euclídea según vamos avanzando en el mapa respecto a la posición inicial del robot, y cuando hemos recorrido un metro, el robot se detendrá y se medirán todas las resoluciones obtenidas en un array de distancias, y se elegirá la mínima de todas esas resoluciones, ya que queremos la unidad más pequeña, ya que determinará la distancia mínima por la que el robot interpreta que se ha movido en el mapa

En el simulador tenemos los siguientes resultados:

Tabla 1 Velocidades Simulador

V (ms^{-1})	**Ω ($rads^{-1}$)**	**q_lineal (m)**	**q_angular (r)**
0.1	0.0	0.0097	0
0.3	0.0	0.0293	0
0.5	0.0	0.0473	0
0.7	0.0	0.0672	0
0.9	0.0	0.0835	0
0.0	0.3	0	0.0278
0.0	0.7	0	0.0680
0.0	0.9	0	0.0895

Ejercicio 3

Mida la resolución máxima **(q)** de odometría lineal y angular con las diferentes combinaciones de velocidades propuestas en el robot real (en este caso, debe tener en cuenta las aceleraciones y deceleraciones del robot). COnstruya una tabla como la del apartado anterior con las 8 combinaciones propuestas

Tabla 2 Velocidades Robot Real

V (ms^{-1})	Ω ($rads^{-1}$)	q_lineal (m)	q_angular (r)
0.1	0.0	0.0014	0
0.3	0.0	0.0022	0
0.5	0.0	0.0014	0
0.7	0.0	0.0014	0
0.9	0.0	0.0014	0
0.0	0.3	0	0.0175
0.0	0.7	0	0.0175
0.0	0.9	0	0.0175

Ultrasonidos

Ejercicio 1

Indique y describa información que nos ofrece el mensaje disponible en el topic **sonar**. Muestre algún ejemplo de captura

Como se puede apreciar en la imagen, el sonar nos muestra el rango máximo, mínimo y el rango actual, que, en el caso de no haber nada, mostrará un *inf*

```
  MessageType: 'sensor_msgs/Range'
   ULTRASOUND: 0
      INFRARED: 1
        Header: [1×1 Header]
RadiationType: 0
  FieldOfView: 0.2618
     MinRange: 0.1000
     MaxRange: 5
        Range_: 0.8300
```

Ilustración 2 Topic Sonar

Ejercicio 2

En el simulador STDR, posicione el robot de tal forma que exista una distancia de 2 metros de uno de los sensores sonar concretos del robot, ¿Qué posición y sónar se ha elegido)

Hemos elegido el sonar número 3, ya que es uno de los que se encuentra en el frontal como se aprecia en la imagen a continuación, ya que se ve un sonar con un color más apagado:

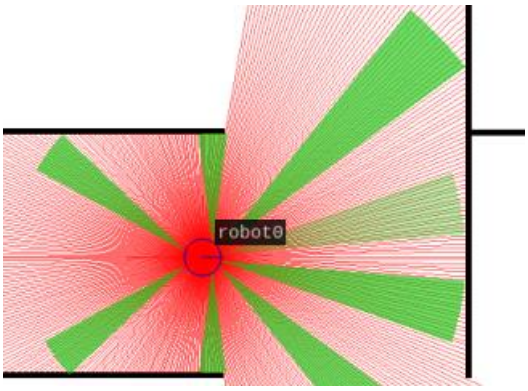

Ilustración 3 Robot en el simulador

Para mover el robot, hemos usado el script presente en el fichero *SensorUltrasonidoB.m*, que lo que hace es que va comprobando el sonar de manera constante hasta que éste sonar devuelve un resultado de 2 metros, deteniendo el robot inmediatamente.

Ejercicio 3

Obtenga 1000 medidas de distancia del sensor sónar elegido y dibuje una gráfica (comando plot en Matlab) la distancia medida. ¿Son estables esas medidas? ¿Hay ruido en la medida? En caso afirmativo, calcule el valor máximo, medio y la varianza del ruido

Para obtener las mil medidas y dibujar la gráfica, hemos utilizado el script llamado *SensorUltrasonidoC.m*, el cual mide el sonar y guarda el instante o iteración de la medida y después relaciona ambas listas, dando lugar así a la gráfica final. El resultado obtenido ha sido el siguiente:

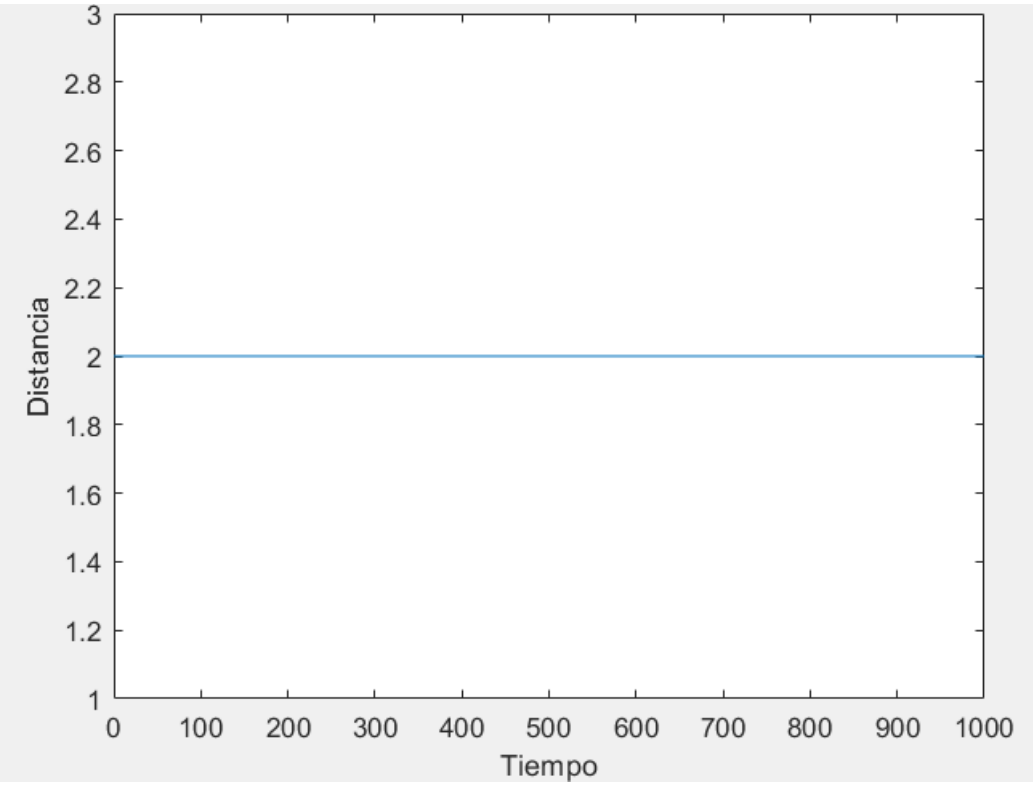

Ilustración 4 Grafico Distancia Simulador

Como podemos ver en la gráfica, el valor es constante, ya que en el apartado anterior lo movimos a una distancia en la que el sensor ultrasonido elegido (sonar 3), se sitúa a 2 metros de la pared, y como el robot no se está moviendo hacia ningún lado, la distancia respecto de la pared no va a cambiar (recalcar que estamos en un simulador, esto en la vida real no pasaría, en la vida real siempre vamos a tener un pequeño error). Entonces, sabemos que la varianza es 0 (no varía el valor), y tanto la máxima, como la mínima, como la media, valdrán 2

Ejercicio 4

Implemente un filtro media móvil con los últimos 5 valores de distancia y dibuje en una gráfica. ¿Son más estables las medidas que en el caso anterior? ¿Sería útil este método si el robot está en movimiento en lugar de permanecer estático?

El resultado que obtenemos es idéntico al que obtenemos en el apartado anterior, ya que seguimos en el simulador (donde no hay ruido) y el robot sigue sin moverse.

Este método resulta más útil para prevenir que medidas atípicas interfieran con el correcto funcionamiento del robot, para este tipo de sensor resulta más útil, debido a que es bastante sensible al ruido, este método no es bueno en caso de que el robot vaya a una velocidad suficientemente rápida en la que la media de medidas posea un margen de error más grande de la precisión necesitada.

Esta gráfica la hemos obtenido usando el script llamado *SensorUltrasonidoD.m*, el cual mide 1000 datos, y filtra cada 5 haciendo su media y metiéndolos en una lista de distancias filtradas.

Ejercicio 5

Repita el paso B con el robot real

Con el robot real hemos elegido el mismo sensor que en el simulador, el sonar 3, lo hemos puesto apuntando a la pizarra del laboratorio hasta que ha dado una medida de dos metros, que se podrá ver en el gráfico de abajo.

Repita el paso D con el robot real

Hemos realizado las 1000 medidas con la distancia configurada en el apartado anterior. Hemos obtenido el siguiente resultado:

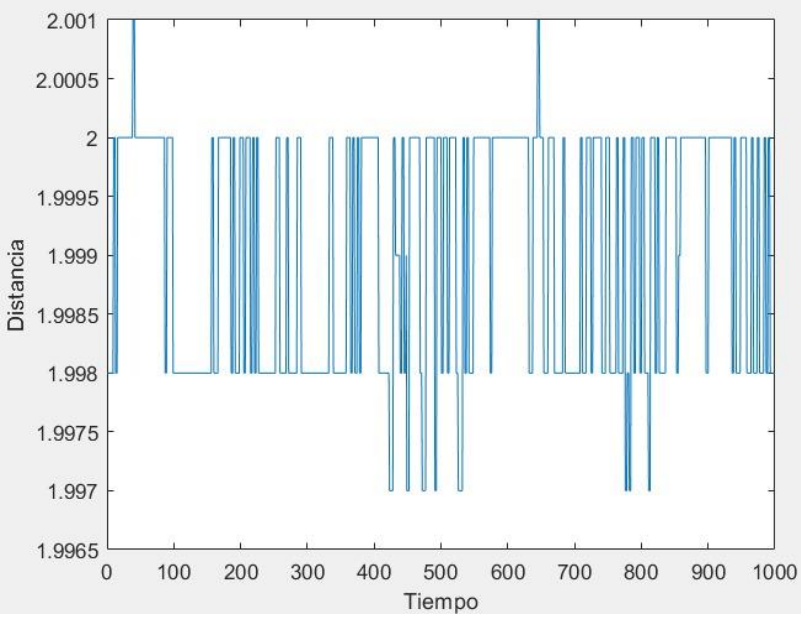

Ilustración 5 Grafico Distancia Robot Real

Nuevamente utilizando el sonar 3, lo hemos situado a 2 metros de la pared y cómo podemos observar, las medidas son bastante estables, con un error de +-0.002 debido a las posibles interferencias con el entorno.

Ejercicio 6

En el simulador, posicione al robot en la casilla X. Estando el robot perfectamente paralelo a las paredes de la celda, seleccione las medidas de los sensores sonar del robot que podrían resultar útiles para obtener las 4 rectas que definen las paredes que lo rodean. Compruebe que las orientaciones de las rectas son parlelas dos a dos y perpendiculares entre ellas. Defina una **función de calidad** para obtener el grado de confianza de dichas paredes empleando, por ejemplo, la relación entre las diferentes pendientes

Para este apartado se ha realizado el planteamiento inicial para obtener las rectas de usar los 8 sonar, Los dos azules para la recta Izquierda, morado para la recta derecha, amarillo para la recta frontal y naranja para la recta trasera.

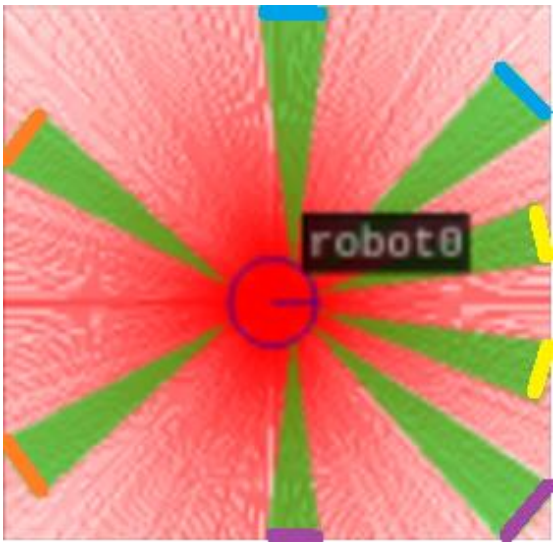

Ilustración 6 Sonars usados Primer planteamiento

Tras la creación de las rectas se han comprobado si las rectas son paralelas 2 a 2, con una función que devuelve el porcentaje de paralelismo, siendo 100 paralelas.

Al comparar la recta frontal con la trasera, el programa nos devuelve un grado de paralelismo del 99.59% este error del 0,5% se debe a las aproximaciones realizadas para los cálculos. En su contraparte tenemos las rectas izquierda y derecha, con un grado de paralelismo del 56,95% debido a que, en este caso, tanto en la recta derecha como la izquierda, se están usando puntos de dos paredes distintas, dando rectas que no definen correctamente la posición de las rectas.

Se planteará la manera de calcular las rectas laterales usando el sonar izquierdo y derecho, y asumiendo que el punto que detecta pertenece a una recta paralela al eje x.

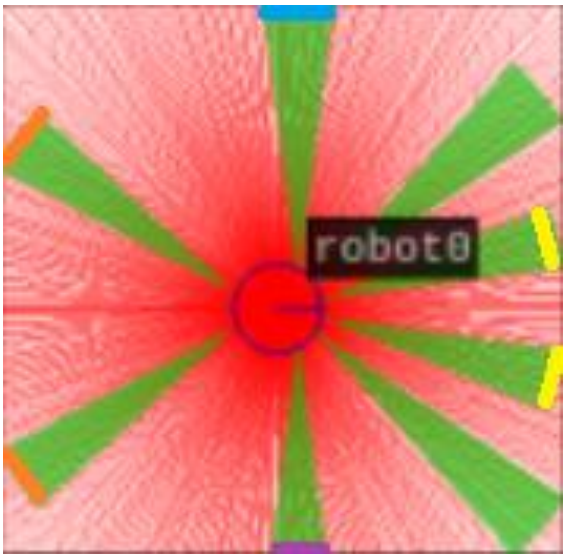

Ilustración 7 Sonars usados segundo planteamiento

Naturalmente al comparar el grado de paralelismo de las dos rectas nos da un valor del 100% por lo que no podemos fijarnos en esta métrica para ver si es una buena opción, en este caso vamos a comparar el ángulo que produce en la intersección con las rectas frontal o trasera, dando en este caso un ángulo de 89.85º, que nos indica que las rectas laterales con la frontal y trasera son perpendiculares.

El inconveniente de este método es que deja de ser útil cuando la recta real no es paralela al eje x. Por lo tanto, se combinan los dos métodos, y se usará el método que se acerque más a ser perpendicular con una de las rectas frontal o trasera, en caso de la falta de ambas rectas, se utilizará el primer método.

Una vez completado el programa se define la función de confianza como cuanto se acercan las rectas a ser paralelas, los pares de rectas paralelas y cuanto de perpendiculares son dos rectas perpendiculares, siendo 100% el valor más alto

Ejercicio 7

Diseñe una función que indique, mediante un código, el número de paredes que se encuentra el robot en sus laterales. La codificación que se puede emplear para proporcionar la salida es la que se muestra en la Figura 4. Además, indique el grado de confianza basado en la función de calidad definida en el apartado anterior.

La función está basada en coger todas las rectas detectadas y usando la función de confianza usada anteriormente en todas las parejas de rectas posibles y dando un grado de confianza basado en la media sin ponderar de todos los datos de las funciones de confianza de cada par de rectas.

Ejercicio 8

Diseñe una **función** que indique, mediante un código, el número de paredes que se encuentra el robot en sus laterales. La codificación que se puede emplear para proporcionar la salida es la que se muestra en la Figura 4. Además, indique el **grado de confianza** basado en la función de calidad definida en el apartado anterior

Tabla 3 Predicción de paredes con sonar en simulador y robot real

Robot_real/Simulador	Combinación real	Combinación identificada	Grado de confianza
Simulador	0	0	NaN%
Simulador	1	1	NaN%
Simulador	2	2	NaN%
Simulador	3	3	NaN%
Simulador	4	4	NaN%
Simulador	5	5	99.84%
Simulador	6	6	99.60%
Simulador	7	7	99.84%
Simulador	8	8	99.59%
Simulador	9	9	100.00%
Simulador	10	10	99.59%
Simulador	11	11	99.68%
Simulador	12	12	99.73%
Simulador	13	13	99.68%
Simulador	14	14	99.90%
Simulador	15	15	99.74%
Robot_real	0	0	NaN%
Robot_real	1	1	NaN%
Robot_real	2	2	NaN%
Robot_real	3	3	NaN%
Robot_real	4	4	NaN%
Robot_real	5	5	70.44%
Robot_real	6	6	83.46%
Robot_real	7	7	55.83%
Robot_real	8	8	67.33%
Robot_real	9	12	24.51%
Robot_real	10	13	35.46%
Robot_real	11	11	67.25%
Robot_real	12	12	83.91%
Robot_real	13	6	42.86%
Robot_real	14	14	64.52%
Robot_real	15	11	77.96%

Como podemos observar, para los 5 primeros casos, no se ha podido dar un valor de confianza, ya que se necesitan como mínimo 2 rectas, por lo tanto, el programa no es capaz de dar un número conciso. De todas formas, en el resto de las medidas, se puede observar que las medidas del simulador son muy precisas, no bajando del 99.5%, gracias al mecanismo de elección planteado anteriormente. No obstante, las medidas del robot real han sido bastante diversas, llegándose a equivocar en múltiples ocasiones, posiblemente por el ruido generado por el robot al tener poca batería.

Laser

Ejercicio 1

Indique y describa la información que nos ofrece el mensaje disponible en el topic laser. Muestre algún ejemplo de captura

En el mensaje del topic laser, tenemos disponible el rango tanto máximo como mínimo del sensor, sin embargo, la característica principal que tiene este sensor es que, en lugar de devolverte una distancia, te devuelve una lista con 400 distancias, ya que el sensor láser del amigobot tiene 400 rayos. Aquí se muestra una captura de muestra de un mensaje de este sensor:

```
    MessageType: 'sensor_msgs/LaserScan'
         Header: [1×1 Header]
       AngleMin: -3.1416
       AngleMax: 3.1416
  AngleIncrement: 0.0157
   TimeIncrement: 0
       ScanTime: 0
       RangeMin: 0.1500
       RangeMax: 8
         Ranges: [400×1 single]
    Intensities: [0×1 single]
```

Ilustración 8 Topic Laser

Ejercicio 2

Repetir el paso 2 del apartado anterior con el sensor láser

En cuanto al paso 2, se ha hecho con el mismo algoritmo, obteniendo los mismos resultados, pero ahora usando el láser número 200, porque hemos podido comprobar que ese laser es justamente el láser frontal del robot. Este código está presente en el script *SensorLaserB.m*

Ejercicio 3

Repetir el paso 3 del apartado anterior con el sensor láser

En cuanto al paso 3, se trata de coger mil medidas y representarlas mediante una gráfica, la única diferencia que tenemos con respecto al resultado obtenido con el sensor de ultrasonido es que el láser, como podemos ver en el apartado A, tiene 400 rayos, entonces nos saldrán 400 medidas en un instante de tiempo, y esas medidas siempre serán constantes al igual que el ejercicio de ultrasonidos, ya que el robot sigue

permaneciendo inmóvil, por tanto, las distancias no cambiarán. Aquí se adjunta una imagen de la gráfica obtenida:

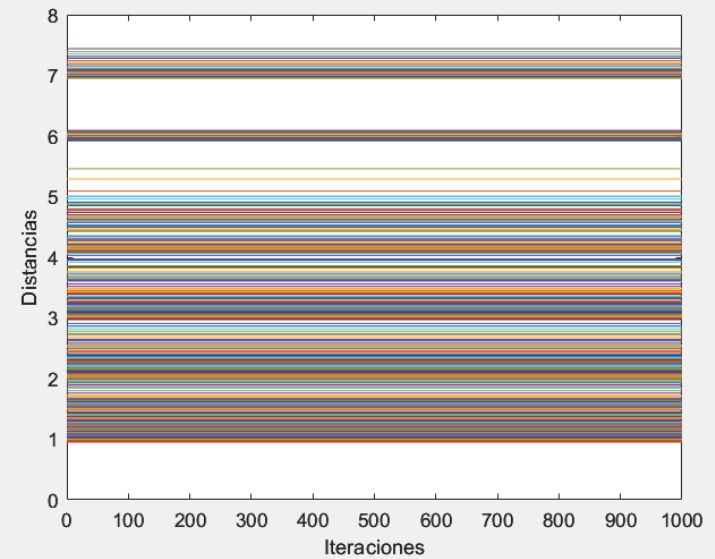

Ilustración 9 Distancia de todos los laser en simulación

Ejercicio 4

Repetir el paso 4 del apartado anterior con el sensor láser

En cuanto al paso 4, que se trata de realizar mil medidas y cada 5 hacer una media y luego mostrarlas, para este apartado, usaremos el mismo laser que usamos para el apartado A, que es el láser 200, y obtenemos el siguiente resultado:

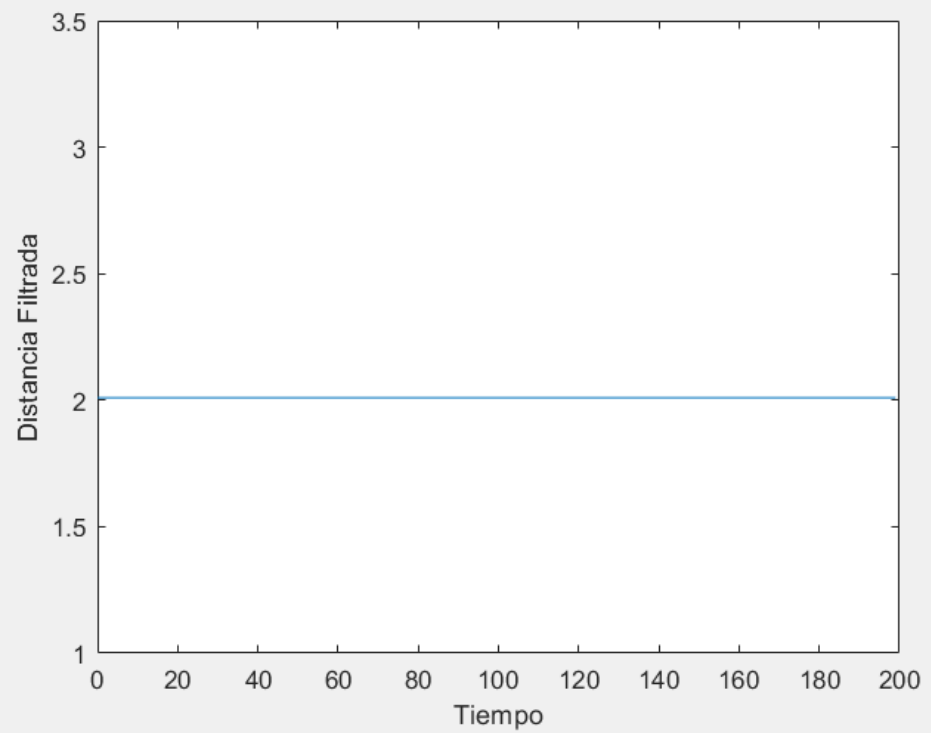

Ilustración 10 Medidas laser simulación

Las medidas obtenidas son coherentes, ya que en el simulador estamos en un lugar donde no existe ningún tipo de ruido, y es constante porque el robot permanece inmóvil, que es el mismo caso que se ha visto anteriormente con los sonars.

Ejercicio 5

Repetir el paso 1 del apartado anterior con el sensor láser usando el robot real

Se vuelve a usar el láser de la posición 200, y como observamos da una medida idéntica a la de la simulación, ya que deja de medir al detectar 2 metros.

Repetir el paso 2 del apartado anterior con el sensor láser usando el robot real

Realizamos las mil mediciones nuevamente contra la pared, a una distancia aproximada de 2 metros, dándonos este grafico con 200 datos, estos son la media de los 5 datos tomados:

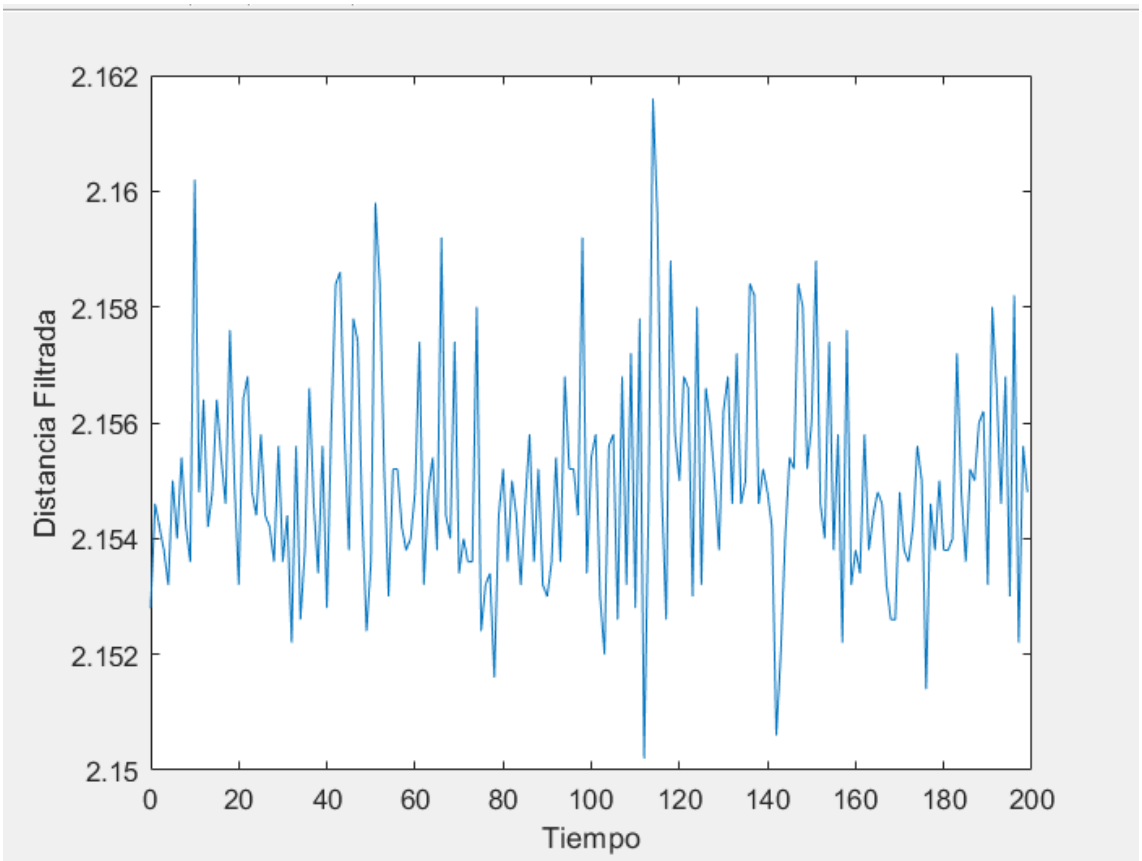

Ilustración 11 Medidas laser robot real

Como anteriormente citado, este método puede ser beneficial en caso de ser suficientemente preciso para el propósito del programa, para el láser en concreto, es menos necesario el uso de este tipo de métodos, debido a la mejor precisión del láser y debido a la cantidad de puntos que proporciona, existen mejores métodos para

detectar si un puede ser un valor correcto o es un error, como por ejemplo calcular con trigonometría el valor ideal respecto al punto a su izquierda o derecha, y comparándolo con un margen de error representativo para la aplicación del robot.

Ejercicio 6

En el simulador, posicione al robot en la casilla X. Estando el robot perfectamente paralelo a las paredes de la celda, seleccione las medidas de los sensores sonar del robot que podrían resultar útiles para obtener las cuatro rectas que definen las paredes que lo rodean. Compruebe que las orientaciones de las rectas son paralelas dos a dos y perpendiculares entre ellas. Defina una función de calidad para obtener el grado de confianza de dichas paredes empleando, por ejemplo, la relación entre las diferentes pendientes.

se plantea similar a la solución planteada con los sonar, en este caso al tener 400 puntos de los que poder elegir, se han elegido los puntos 225 y 175 para la recta frontal, 325 y 275 para la recta izquierda, 25 y 375 para la recta trasera y 125 y 75 para la recta derecha.

Para saber que ángulo tiene respecto al eje 0 se utiliza esta fórmula en la que X es la posición del láser seleccionada, con esta fórmula sacamos los ángulos de los puntos seleccionados como se muestra en la tabla inferior.

$$\left(180 + \frac{X \times 360}{400}\right) mod(360)$$

Tabla 4 posición con su ángulo

Posición	Angulo
25	202.5
75	247.5
125	292.5
175	337.5
225	22.5
275	67.5
325	112.5
375	157.5

Se han elegido estos valores, ya que proporciona un ángulo suficientemente pequeño para que sea más difícil detectar como pared, dos puntos de las rectas adyacentes en vez de frontales.

Con esto se obtienen resultados más fiables como se verá demostrado en los test posteriores.

Ejercicio 7

Diseñe una función que indique, mediante un código, el número de paredes que se encuentra el robot en sus laterales. La codificación que se puede emplear para

proporcionar la salida es la que se muestra en la Figura 4. Además, indique el grado de confianza basado en la función de calidad definida en el apartado anterior.

Se volverá a utilizar la misma función utilizada en con los sonar, siendo esta detectar la paralelidad y la perpendicularidad de dos rectas dadas, dando como función de calidad la media del valor de confianza de todas las posibles combinaciones.

Ejercicio 8

Comprueba los resultados de la función diseñada en el apartado anterior tanto con el simulado STDR como con el robot real y con las 16 posibles combinaciones. Complete la tabla siguiente indicando los resultados.

Tabla 5 Predicción de paredes con láser en simulador y robot real

Robot_real/Simulador	Combinación real	Combinación identificada	Grado de confianza
Simulador	0	0	NaN%
Simulador	1	1	NaN%
Simulador	2	2	NaN%
Simulador	3	3	NaN%
Simulador	4	4	NaN%
Simulador	5	5	99.91%
Simulador	6	6	98.73%
Simulador	7	7	99.27%
Simulador	8	8	99.10%
Simulador	9	9	98.71%
Simulador	10	10	99.92%
Simulador	11	11	99.24%
Simulador	12	12	99.24%
Simulador	13	13	99.31%
Simulador	14	14	99.30%
Simulador	15	15	99.27%
Robot_real	0	0	NaN%
Robot_real	1	1	NaN%
Robot_real	2	2	NaN%
Robot_real	3	3	NaN%
Robot_real	4	4	NaN%
Robot_real	5	5	98.57%
Robot_real	6	6	98.12%
Robot_real	7	7	97.94%
Robot_real	8	8	93.13%
Robot_real	9	9	97.84%
Robot_real	10	10	98.61%
Robot_real	11	11	96.93%
Robot_real	12	12	97.45%
Robot_real	13	13	96.77%
Robot_real	14	14	95.29%
Robot_real	15	15	98.49%

Dados los datos podemos concluir que las medidas realizadas por laser son más similares a las de la simulación respecto a los sonar, teniendo unas predicciones mayores que el 90%

Ejercicio 9

¿Tiene más o menos ruido que el sensor sónar?

Una vez finalizadas las medidas, se concluye que el láser es mucho menos sensible al ruido que el sonar, ya que el sonar llegaba a tener interferencias con los ruidos generados por el robot, en cambio, al láser, frecuentemente detecta distancias con valor *inf*, pero este problema se puede solucionar volviendo a realizar la medición.

Ejercicio 10

¿Cómo es el grado de confianza empleando este sensor?

El grado de confianza del láser es considerablemente superior al sonar, ya que el laser no ha detectado ningún caso con la opción incorrecta y ha dado todas las medidas con un porcentaje de fiabilidad mayor al 90%.

Actuadores

Ejercicio 1

Diseñe una función avanzar que reciba como parámetro de entrada la distancia a avanzar

Esta función codificada en el fichero *ActuadoresRos.m*, dentro de la carpeta *ActuadoresRos*, pide como parámetro una distancia a recorrer, después se le asigna una velocidad constante y se comprueba mediante la odometría la distancia recorrida por el robot, una vez que se supere esa distancia se le enviará un mensaje parando así el robot

Ejercicio 2

Diseñe una función girar que reciba como parámetro de entrada el ángulo a girar

Esta función está codificada en el mismo fichero que la función anterior y tiene un comportamiento similar, se le envía una velocidad angular constante, y una vez que el robot esté girando se comprueba cuanto ha girado haciendo uso de los sensores de odometría. El caso en el que el ángulo sea positivo, cuando el ángulo recorrido sea mayor que el objetivo, el robot se detendrá, y en caso contrario cuando el Angulo introducido en el parámetro de entrada sea negativo

Ejercicio 3

Empleando el simulador STDR, y utilizando las funciones avanzar y girar, navegue con el robot desde la esquina inferior derecha (4,-4,0) hasta la salida utilizando los puntos centrales de las casillas como coordenadas de destino.

Para realizar el recorrido hemos usado las dos funciones anteriormente mencionadas, poniendo como parámetros de entrada las distancias y ángulos respectivos para hacer el recorrido, y hemos comprobado que funciona

Ejercicio 4

Realice un recorrido con el robot real concatenando los siguientes tramos

- Tramo recto de 2m
- 1 giro de 90º
- 1 tramo recto de 1m
- 1 giro de -90º
- 1 tramo recto de 1m

Para verificar el funcionamiento de este ejercicio, hemos grabado un vídeo realizando el recorrido, cuyo enlace está aquí

Conclusión

A lo largo de este trabajo hemos realizado distintas pruebas con los múltiples aspectos que presenta el robot, se ha podido comprobar cómo funciona el desplazamiento y la dificultad de la precisión en este, sobre todo al girar un ángulo concreto.

Se ha podido comprobar dos tipos de sensores, ultrasonidos y láser, ambos tienen sus ventajas y desventajas. El ultrasonido, está siempre en una posición fija y es muy sensible al ruido, pudiendo dar medidas incorrectas. En cambio, el laser es más flexible, ya que en este caso proporciona 360 medidas, pero genera valores nulos en algunos instantes.