Percepción y Control

PRÁCTICA 1

SENSORES

Y

ACTUADORES

**Ingeniería de Computadores**

**Pedro Barquín Ayuso**

**Miguel Ballesteros García**

**1.-Pruebas con el interfaz position2d**

Realizar las siguientes pruebas sobre el simulador empleando el interfaz position2d:

a) Diseñar una función avanzar que se reciba como parámetro de entrada la distancia a avanzar (1m, 2m...). Para ello se puede emplear la función set\_cmd\_pose y set\_cmd\_vel. Ya queel robot real Amigobot acepta únicamente comandos de velocidad, se recomienda usar comando

únicamente de velocidad set\_cmd\_vel.

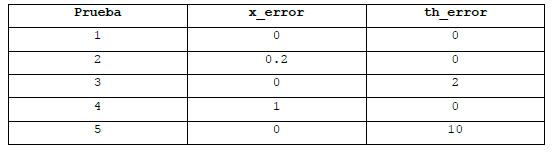
b) Diseñar una función girar que se reciba como parámetro de entrada el ángulo a girar (45º, 90º,...). Para ello se puede emplear la función set\_cmd\_pose y

set\_cmd\_vel.

c) Utilizando las funciones avanzar y girar navegar con el robot desde la esquina inferior derecha (2,-2,0) hasta la salida utilizando los puntos centrales de las casillas como coordenadas de destino. Además, observe de qué manera influye el error de odometría en la ejecución de la trayectoria anterior modificando dicho error (odom\_error) en el fichero practica1.world, tanto para los valores de

avance (x\_error) como para los giros (th\_error).Comprobar el efecto para las siguientes combinaciones, grabando 1 vídeo por cada una de las pruebas.

**-Se adjunta video (video1c) y su código correspondiente (avanzarGirar/ejemplo1.c).**



**Caso 1 corresponde con el video video1c**

**Caso 2 corresponde con el video video2c**

**Caso 3 corresponde con el video video3c**

**Caso 4 corresponde con el video video4c**

**Caso 5 corresponde con el video video5c**

d) Analiza el valor del parámetro stall cuando hay un error de odometría nulo y un error de odometría suficientemente grande como para chocarse durante diferentes recorridos.

¿Qué información proporciona? **Nos informa de la situación del motor de las ruedas del robot**

¿Para qué sería útil este parámetro? **Podríamos saber si el motor esta funcionando y como consecuencia el robot debería está en movimiento (flag a 0) o si el motor está bloqueado y el motor está quieto (flag a 1)**

¿Qué valor tomaría este parámetro en aquellas situaciones en las que el robot se traslade por suelos resbaladizo? **En estas situaciones el flag estaría a 0 debido a que el motor se mueve, aunque debido al deslizamiento el robot no avance.**

e) Realizar un recorrido con el robot real concatenando los siguientes tramos:

1 tramo recto de 1m

1 giro de 90º

1 tramo recto de 0.5m

1 giro de -90º

1 tramo recto de 1m

**Se adjunta video (prueba\_real).**

Comprobar el error final que se ha obtenido e indicar con qué combinación de error de odometría simulado se correspondería.

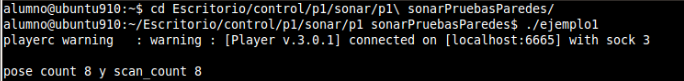
**No pudimos tomar correctamente la medición con el robot por lo que no podemos concretar el error final obtenido. Pero por lo observado en la simulación vemos que el robot tiene un pequeño margen de error en los dos movimientos por lo que el error final sería la suma de los dos.**

**2.-Pruebas con el interfaz sonar**

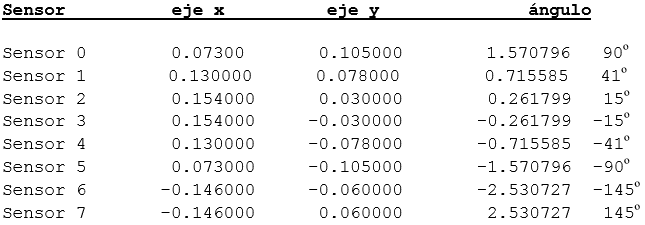
**Se adjunta código correspondiente a la parte del sonar (sonar/ejemplo1.c)**

a) Comprobar la información que proporcionan los ultrasonidos sobre el simulador Stage.

**La función que pide en este apartado es infoSensores.**



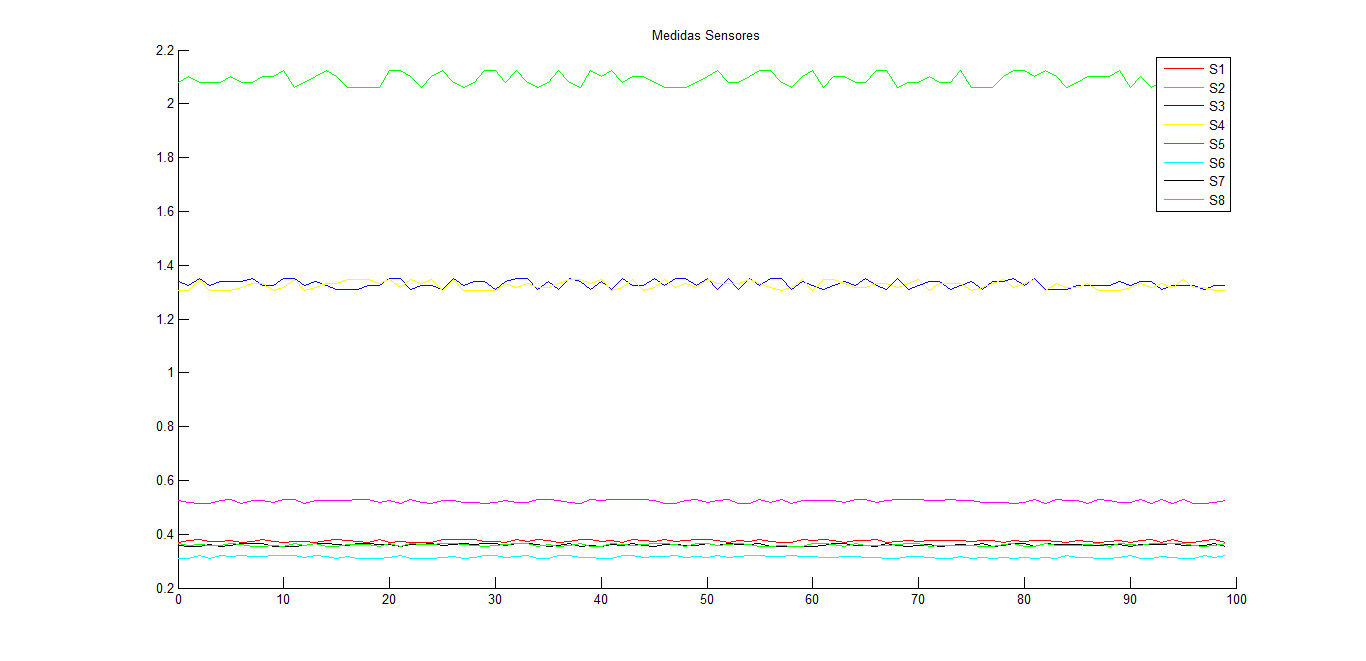
b) Obtener la posición y orientación de los sensores de ultrasonidos sobre el robot mediante la función get\_geom, completar la siguiente tabla.

****

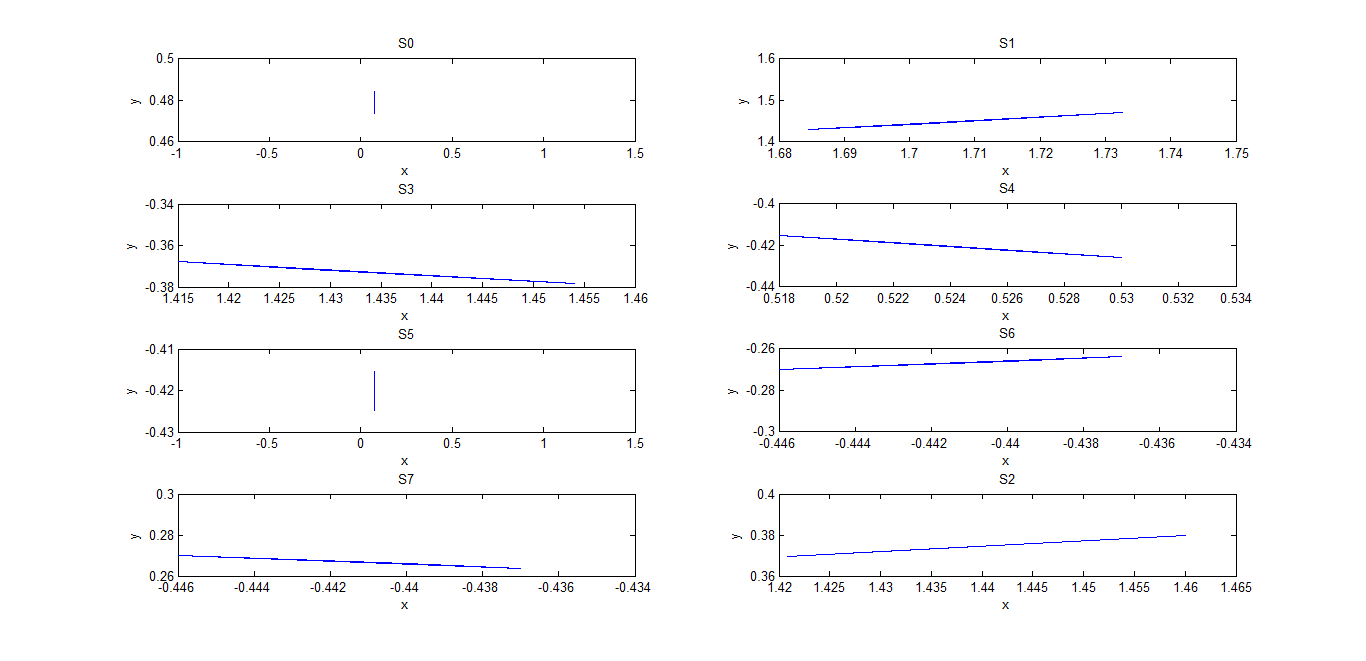
c) Obtener 100 medidas de cada uno de los sensores sónar y dibujar en una gráfica las coordenadas x e y. ¿Son estables las medidas? Calcular el valor medio de las 100 medidas en x e y.

**El código está adaptado para hacer 20 medidas en vez de 100 para reducir tiempo, ya que lo necesitamos para apartados posteriores. La función correspondiente este apartado es la de recalcular. Las medidas en el simulador son estables porque no tienen error por lo que hemos metido un error aleatorio para asemejarlo a la realidad mediante la función aleatorioSonar.**

**100 medidas de distancias de cada sensor.**



**Valores X e Y de las 100 medidas de cada sensor.**

****

d) Obtener un indicador de la dispersión de los datos mediante el valor de la desviación típica Sx y Sy de los sensores ultrasónicos.

**El código correspondiente a este apartado se encuentra en las funciones mediax, mediay, sx, sy.**

e) Obtener el valor de la covarianza para saber si las medidas obtenidas están relacionadas entre sí.

**El código correspondiente a este apartado se encuentra en la función sxy.**

f) Obtener el coeficiente de correlación de Pearson para saber si el ajuste de la nube de puntos a la recta de regresión obtenida es satisfactorio.

**El código correspondiente a este apartado se encuentra en la función r**

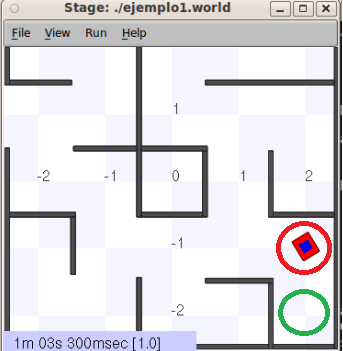
g) Posicionar al robot en la casilla central (0, 0, 0). Estando el robot perfectamente paralelo a las paredes de la celda, seleccionar las medidas de los 8 sensores sonar del robot para obtener las cuatro rectas que definen las paredes que lo rodean. Obtener el coeficiente de correlación para cada recta. Comprobar que las orientaciones de las rectas son paralelas dos a dos y perpendiculares entre ellas. Definir una función de calidad para la obtención de dichas paredes, empleando para ello el coeficiente de correlación de Pearson y la relación entre las diferentes pendientes.

**La función de calidad es calidadPearson**

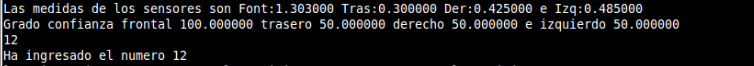
h) Diseñar una función que indique mediante un código el número de paredes que se encuentra el robot en sus laterales. La codificación que se puede emplear para proporcionar la salida es la que se muestra en la siguiente tabla.

Además, indicar el grado de confianza basado en la función de calidad definida en el apartado anterior.

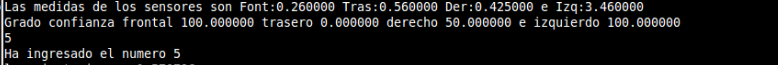
**La función pedida en este apartado es paredesAlrededor. Adjuntamos dos ejemplos de este apartado unido al anterior.**

****

**Lugar de medida verde**



**Lugar de medida rojo**

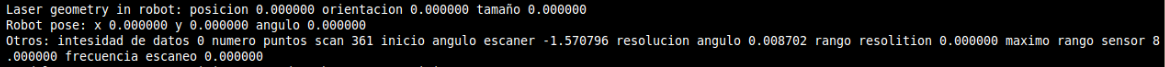


**3.-Sensores de distancia láser**

**Se adjunta código correspondiente a la parte del sonar (laser/ejemplo1.c)**

1-Obtener información sobre la geometría del láser (get\_geom), es decir, la ubicación del mismo, orientación, etc.

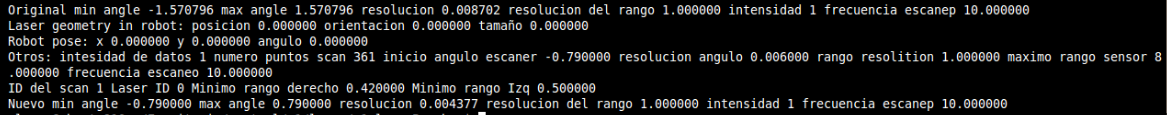
**La función de este apartado es mostrarInfoLaser.**



2-Obtener información sobre su configuración (get\_config), ángulo máximo y mínimo, resolución, intensidad, frecuencia, etc..

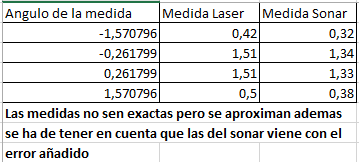
3-Configurar el láser (set\_config), ángulo máximo y mínimo, resolución, intensidad, frecuencia, etc..

**Estos comando se usan en el código del laser dicho al principio del punto 3. Primero se cambian los valores con el comando set\_config y después se imprimen por pantalla mediante get\_config y se observa su correcto funcionamiento**.



**4.-Pruebas con el interfaz laser**

Repetir las mismas pruebas del sónar pero empleando las medidas del sensor láser. Comprobar la gran similitud que existe entre ambos sensores.



**Se observa la una gran similitud como dice el enunciado, varíando un poco las medidas debido al error aleatorio introducido al sonar y un pequeña variación en el ángulo de los dos sensores, que hace que no sea exactamente el mismo.**