Laboratorio 0

26/3/2023

Lucia de Oliveira 5.126.594-0
Joaquin Fontana 5.011.180-5

Indice

Parte A – Interacción con motores

Descripción del problema:

En esta parte del laboratorio se debían realizar pruebas con los motores del kit LEGO Mindstorms NXT 2.0 con el fin de familiarizarse con los mismos y las diferentes funciones que ofrece la API de LeJOS para controlarlos. Se siguió el tutorial de [2] para esta parte.

También era requerido realizar pruebas con el actuador lineal disponible en el kit, este actuador contiene una parte móvil que se extiende y se contrae, siendo programable la distancia que el actuador se mueve y en qué dirección (solo contiene dos direcciones, contraerse o expandirse).

Descripción de la solución:

Programa 1

Este programa tiene como finalidad la prueba del motor y las funcionalidades de ir hacia adelante e ir hacia atrás. También se utiliza el LCD del brick para indicar cuándo se está yendo hacia adelante y cuándo retrocediendo.

Programa 2

Este programa tiene como finalidad probar la precisión de los motores utilizando el tacómetro. Este sensor es utilizado para medir velocidades de giro, en revoluciones por minuto, así como para medir las vueltas que da un objeto. Para realizar la prueba primero se pone a girar el motor en “forward” en una velocidad fija y después de determinado tiempo se imprime el ángulo de giro, luego se para el motor y se vuelva a imprimir el ángulo.
Se puede notar que entre la primer medida impresa, 1398, y la segunda, 1438, hay una sutil pero existente diferencia de 40 unidades entre una medida y otra.
Cuando se realizó el procedimiento inverso, es decir, con el motor en “backward” se espero a llegar al grado 0 y se imprimió el tacómetro, que marcó el 0, luego se frenó el motor y se volvió a imprimir el ángulo y volvimos a notar esta diferencia, el tacómetro tiraba en la segunda medida un valor de -39 grados. De esto pudimos deducir que el tacómetro utiliza ángulos negativos cuando el motor gira en sentido antihorario.
Como conclusión general vemos que el frenado del motor no es instantáneo, habiendo un delay de aproximadamente 40 grados para la velocidad utilizada en las pruebas, y esto debería tenerse en cuenta a la hora de generar soluciones a distintos problemas utilizando estos motores.

Programa 3

En este programa probamos otra funcionalidad de los motores, estos son capaces de moverse a una velocidad continua como visto anteriormente pero también de ir a un ángulo en particular, de esta forma se avanza el motor y luego se pide que vaya al ángulo cero y que el tacómetro muestre el ángulo que detecta. En este caso en vez de imprimir el ángulo 0 imprimió 1, esto, según lo explicado en el propio tutorial, es porque el regulador del motor

tiene en cuenta cuánto el motor va a seguir girando una vez que el freno es aplicado y, por lo tanto, frena un poco antes de llegar a la medida deseada.

Programa 4

En este programa se prueba la funcionalidad para parar los motores una vez que se les dió una instrucción (como rotar una cierta cantidad de grados). El motor mantiene su dirección y velocidad hasta que un botón es presionado, en este caso el motor detiene el movimiento.

Programa 5

Este último programa tiene como objetivo medir la sincronía de los 3 motores que ofrece el kit. El objetivo es poner los 3 motores a funcionar “al mismo tiempo” (teniendo en cuenta que nunca dos instrucciones pueden ejecutarse exactamente al mismo tiempo) y mostrar, cada 200ms, el tacómetro de los 3 motores, a continuación se muestra una tabla de las medidas leídas.

	Motor A	Motor B	Motor C
Medida 1	0	0	0
Medida 2	93	94	97
Medida 3	248	249	249
Medida 4	393	393	393
Medida 5	537	537	537
Medida 6	681	681	681
Medida 7	825	825	825
Medida 8	969	969	969

Podemos observar que al principio hay un mínimo desfasaje entre los motores que luego se corrige y terminan marcado, a partir de la medida 4, los 3 motores los mismos ángulos. Las primeras medidas desiguales pueden deberse a que, como comentando antes, no se pueden activar los 3 motores realmente al mismo tiempo o porque hay una mínima diferencia entre las velocidades en los motores (a pesar de que el comando forward debería proporcionar la misma velocidad a los tres motores).

Actuador lineal

Con respecto al actuador lineal se probaron las funciones move() y moveTo(), que reciben un entero, observamos que el comportamiento de ambas funciones es igual, el actuador lineal se expande la cantidad de unidades que se le pasa a la función. Si se le pasa un entero negativo el actuador se contrae en vez de expandirse.
Se utilizó el tachocount para mostrar cuánto había avanzado y coincide exactamente con el parámetro que se le pasa a la función move o moveTo.

Observamos que la distancia máxima que se expande es 200 unidades, lo que equivale a 20 centímetros. También pudimos observar que la distancia que se mueven es relativo a la posición inicial y no es una posición absoluta, es decir, si me encuentro en la posición 100 y me muevo 50, termino en la posición 150, avanzando 50 unidades (no se contrae 50 unidades para terminar en la posición 50).

Es importante tener en cuenta que como solo se le puede pasar enteros a las funciones move y moveTo, la mínima medida que se le puede pasar es 1 (o -1 si se desea contraer), por lo tanto lo mínimo que se va a mover el actuador lineal es 0,1 centímetros. Esta puede afectar la precisión, ya que si se desea mover el actuador lineal menos de 0,1 centímetros no es posible.

Parte B – Interacción con el sensor de distancia ultrasónico

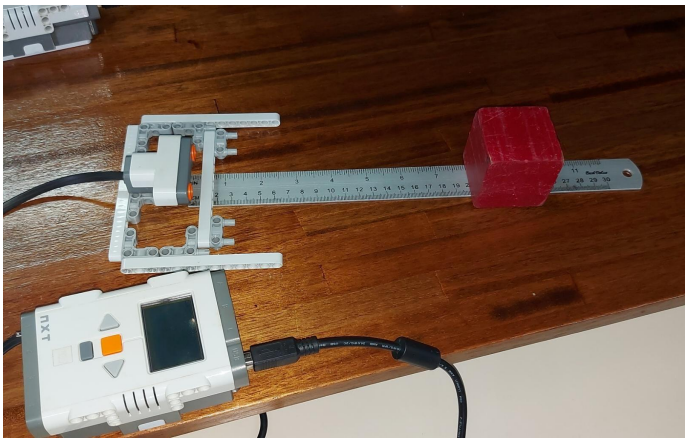
Descripción del problema:

El sensor de distancia es un dispositivo que permite medir la distancia entre el sensor y un objeto mediante ondas ultrasónicas. Sin embargo, el sensor puede tener errores debido a varios factores que afectan su lectura. En esta parte se interactuará con el sensor de distancia del kit lego Mindstorm NXT 2.0 para medir ciertos parámetros del sensor y registrar las condiciones para su correcto funcionamiento.

Descripción de la solución:

Se realizó una prueba experimental con el sensor. Se colocó el sensor frente a diferentes objetos con distintas formas y distancias. Se registró la lectura del sensor en cada caso y se comparó con la distancia real medida con una regla (esquema de trabajo: figura 1). Los resultados obtenidos fueron los siguientes:

- A menos de 5 cm las medidas son erróneas (sigue midiendo 5 cm)
- A 15 cm de distancia de un objeto con superficie plana o curva, el sensor detecta correctamente la distancia sin errores ni oscilaciones.
- A 20 cm de distancia de un objeto con superficie plana o curva, el sensor oscila entre 20-22cm sin importar la forma o el material del objeto.
- A 30 cm de distancia de un objeto con superficie plana o curva, el sensor detecta correctamente la distancia la mayoría de las veces, esporádicamente muestra valores erróneos cercanos a 20 cm.
- Alrededor de los 80 cm el sensor empieza a fallar con objetos que no son planos o son muy pequeños
- Alejando una placa plana desde 30 cm hasta 150 cm y midiendo continuamente la distancia con el sensor, las medidas son coherentes.
- A más de 150 cm de distancia de un objeto, el sensor muestra el valor máximo de 255 cm que indica que está fuera de rango. Esto ocurre incluso si el objeto es una pared lejana que debería reflejar bien las ondas ultrasónicas.



(figura 1: disposición de los elementos para la toma de medidas)

Se concluye que el sensor tiene un rango efectivo entre 5 y 150 cm aproximadamente. Dentro de este rango, el sensor es preciso. Sin embargo, fuera de este rango, el sensor puede mostrar valores erróneos o indicar que no hay ningún objeto. Se recomienda usar el sensor solo para medir distancias cortas y evitar objetos muy lejanos o muy cercanos al límite del rango.

Parte C - Interacción con el sensor de color

Descripción del problema:

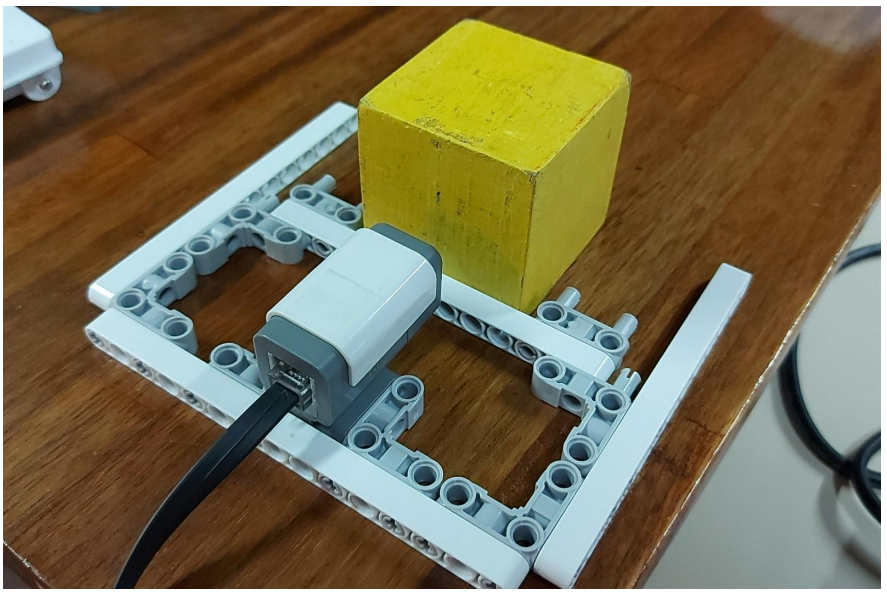
El objetivo de esta sección es utilizar el sensor de color para identificar y mostrar seis categorías de color diferentes. Para ello, se debe medir como afectan distintos factores a las medidas realizadas por el sensor. Algunos de estos factores pueden ser: la iluminación del ambiente, la distancia del sensor al objeto y la orientación del objeto.

Descripción de la solución:

Para asegurar la estabilidad del sensor y garantizar que la distancia al objeto al medir su color fuera constante, se construyó una estructura utilizando piezas de lego. Esto se consideró para evitar posibles variaciones en las condiciones en que se toman las medidas. Una vez completada la estructura, se procedió a tomar medidas en la configuración que se muestra en la (figura 2), utilizando cubos de seis colores diferentes: negro, azul, rojo, verde, amarillo y magenta. Para llevar a cabo esta tarea se utilizó la función **`getColorID()`** la cual devuelve un número entero que identifica el color [3]. Sin embargo, después de realizar varias medidas, observamos que el sensor presentaba algunos errores al clasificar los colores. En particular, clasificaba el color magenta como rojo y el verde como negro. Para solucionar este problema, decidimos modificar el programa que ejecuta el brick.

La solución consistió en que cuando el sensor mide color rojo, discriminamos según el valor RGB de azul. Si este valor era mayor a 75, clasificábamos el color como magenta, de lo contrario lo clasificamos como rojo. Para distinguir entre verde y negro, utilizamos el valor RGB de verde. Si este valor era mayor a 50, clasificábamos el color como verde, de lo contrario lo clasificábamos como negro. Con esta modificación, el sensor pudo clasificar correctamente los seis colores de los cubos utilizados en nuestra experimentación.

Los valores que elegimos para discriminar (75 de azul en el caso de rojo/magenta y 50 de verde en el caso de verde/negro) los identificamos experimentalmente, es decir, tomamos medidas RGB de los cubos en conflicto y comparamos sus RGB, identificamos la componente en la cual diferían más y cuáles eran los valores que tomaban y llegamos a las medidas comentadas anteriormente.



(figura 2: disposición del sensor de color para la toma de medidas)

Parte D – Interacción con la cámara

Descripción del problema:

El objetivo de esta parte es familiarizarse con la cámara para que esta pueda reconocer 6 colores: negro, rojo, azul, verde, amarillo y magenta. La cámara debe ser calibrada para reconocer estos colores ante una iluminación específica, que se busca que no varíe demasiado mientras las pruebas son realizadas.
Luego se debe generar un programa para que en el brick se imprima la información de los objetos detectados, se debe imprimir la cantidad de objetos reconocida por la cámara, el color de los mismos y las coordenadas del centro de su boundingbox, que es el rectángulo en el cual está inscripto el objeto.

Descripción de la solución:

Se utilizó la aplicación NXTCamView para realizar la calibración de la cámara. La aplicación permite la calibración de hasta 8 colores distintos, para lograr esto se toman capturas de la cámara (fotos), que son desplegadas en la aplicación, luego se selecciona el color que se quiere configurar (identificados del 1 al 8) y se “pinta” en la imagen píxeles de ese color, el software a su vez colorea los píxeles que considera parecidos a este y además establece un rango para los valores de RGB que se puede ver en el panel de configuración de los colores.

Este procedimiento se realiza para cada uno de los colores que se quiere configurar, teniendo en cuenta que no puede haber solapamiento entre los colores, es decir, no puede haber dos colores que tengan un rango de RGB similar, esto es porque en caso de que la medida RGB de un objeto cayera en este rango solapado, la cámara no podría decidir cuál de los dos colores es. Esto último supuso una gran dificultad a la hora de configurar los 6 colores ya que es muy difícil que ningún rango se solape, en especial con los colores rojo y magenta esto fue particularmente complicado.

Para resolver el problema con estos dos colores tomamos capturas específicas solamente con cubos de colores rojo y magenta e intentamos establecer los rangos a partir de estas imágenes de la forma más abarcativa posible sin que llegase a haber solapamiento entre ellos, esto mejoró notablemente la distinción de los colores rojo y magenta.

Una herramienta que es de gran utilidad para refinar la detección de colores es la que permite trackear los objetos que la cámara está viendo, esta herramienta muestra los boundingbox de los objetos, el color de los mismos, sus coordenadas y la cantidad de píxeles que ocupan, permitiendo además filtrar por la cantidad de píxeles para evitar ruido. Observamos utilizando esta herramienta que por el tipo de luz y el color de la mesa, se detectaban en la misma píxeles amarillos, que pueden ser eliminados fácilmente aplicando un filtro en la cantidad de píxeles o el área del objeto.

Luego de que la calibración fuera correcta se creó un programa para que identificara objetos e imprimiera el color de los mismos, un problema que se presentó al realizar esto es que la

cámara detecta hasta 8 objetos y ocurría frecuentemente que las agrupaciones de píxeles de color que detectaba en un cubo no estaban lo suficientemente juntas para que las considerara un único objeto, ocasionando que mismo cuando la cámara solo estaba viendo un cubo, dijera que estaba detectando hasta 8 objetos distintos. Luego de aplicar el filtro de área, cuando se imprimían los colores de estos objetos, los colores coincidían en su amplia mayoría con el color del cubo que estaba viendo la cámara, habiendo ocasionalmente interferencias de otros colores.

En el caso del cubo verde, el cubo utilizado para las pruebas es un color bastante oscuro, ocasionando a veces que con la sombra se confundieran algunos píxeles con el color negro. En el caso del rojo y del magenta podría pasar que alguna agrupación de píxeles, generalmente de tamaño menor que el resto, apareciera del color equivocado. Es decir, si el cubo era rojo aparecían algunos píxeles identificados con magenta y si el cubo era magenta aparecían algunos píxeles identificados como rojo.

Cabe destacar que en la aplicación de escritorio, al sacar las capturas, la configuración de color que teníamos marcaba casi perfectamente los colores, identificando casi la totalidad del cubo para la mayoría de colores y no confundía colores, sacando algunos píxeles muy puntales, a la hora de probar la cámara con el programa realizado y a través del brick no se percibía tanta precisión a la hora de la detección de colores y las áreas.

Parte E – Variando el ambiente

Descripción del problema:

Cuando la iluminación ambiental cambia, el espectro de luz que incide en el objeto que se está midiendo también cambia. Esto puede afectar las medidas realizadas por el sensor. En esta sección se experimentará variando la luz ambiente para observar como afecta a las medidas realizadas.

Descripción de la solución:

Durante la medición de colores utilizando un sensor de color, se observó que la iluminación del ambiente tenía un efecto en los valores medidos de los colores. Esto significa que el color medido puede variar dependiendo de la intensidad y la temperatura del color de la luz ambiente presente en el entorno.

Para resolver este problema, se tomaron medidas de los valores RGB de los 6 colores a clasificar en condiciones de iluminación controlada. Se calculó el promedio de estos valores RGB para cada color, y se utilizó ese valor como valor de referencia para clasificar los colores.

Para clasificar los colores, se toma una medida del valor RGB del color a clasificar. Se calcula la distancia a los valores de referencia utilizando la norma 2 (también conocida como la distancia euclidiana). El color se clasifica según qué valor de referencia sea el más cercano al valor medido.

Es importante destacar que, aunque la solución presentada permite corregir los efectos de la iluminación ambiental en las mediciones de color, aún hay margen para mejorar el sistema de clasificación de colores.

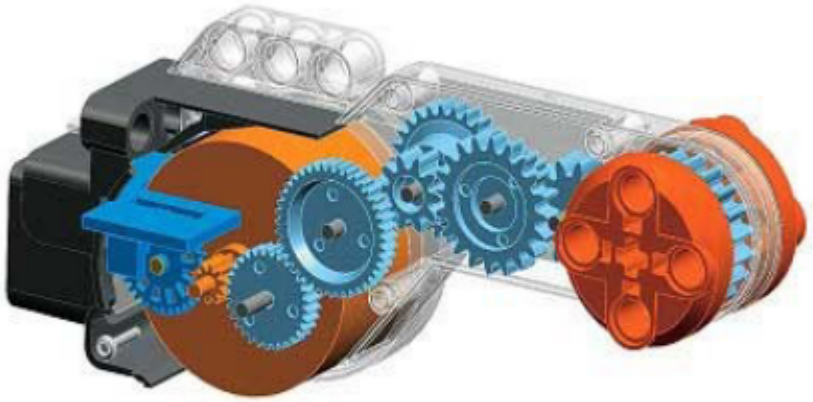
Una posible mejora a futuro podría ser la realización de mediciones de los colores bajo diferentes condiciones de iluminación para crear varios sets de valores de referencia específicos para cada entorno de iluminación. Esto permitiría una clasificación aún más precisa de los colores en diferentes situaciones de iluminación. En nuestro caso esto no se pudo realizar ya que observamos que la luz ambiente que detecta el sensor depende ampliamente del color que esté mirando. Por ejemplo, si hay poca iluminación pero el color es amarillo o si hay iluminación pero tenemos un color oscuro (como rojo o verde), los valores de luz ambiente dan igual, haciendo imposible determinar solo por la luz ambiente si hay mucha luz o poca al momento de realizar la medición.

Parte F - Cuestionario

Motores

¿Qué tipo de motores tiene el kit robótico utilizado en el laboratorio 0?

El kit robótico tiene servo motores, estos motores pueden ser llevados a posiciones angulares específicas al enviar una señal codificada [1]. Los motores del kit tienen funcionalidades para rotar a un ángulo específico así como para darles una dirección y que giren de forma continua a una velocidad dada. Los motores del Lego NXT poseen además un sensor que les permite contar los grados de giro realizados, llamado tacómetro. Estos motores además contienen una reducción integrada formada con engranajes por dentro del motor como se ve en la imagen:



¿Qué tipo de sensor es el tacómetro? ¿Cuál es su principal uso?

El tacómetro es un sensor interno ya que se encuentra dentro del motor, es pasivo ya que no cambia el ambiente, solo realiza mediciones, es local ya que mide las vueltas de un motor en particular, por último es digital, ya que nos marca con un número los grados que se giró.

El tacómetro no es más que un instrumento de medición que se usa en elementos rotatorios, para determinar la cantidad de giros que éste realiza en su propio eje y por ende, se acompaña también de la velocidad de éstos. [4]

Sensor de color

¿Cuál es la característica del ambiente que más afecta el resultado de este sensor? ¿Por qué?

La característica que más afecta el sensor es la distancia al objeto, realizando pruebas experimentales podemos ver que cuando la distancia es aproximadamente de entre 1 y 2 centímetros el sensor de color funciona correctamente en la mayoría de los casos, sin

embargo, cuando alejamos el sensor del objeto comienza a dar medidas erróneas. Esto se debe a que el sensor de color del kit de LEGO está pensando para utilizarse cerca de los objetos.

¿Alguna otra?

Realizando pruebas para la parte E probamos alumbrando los objetos con una linterna para ver si daba medidas erróneas, notamos que para ciertos colores, como rojo y magenta si se les apuntaba con una cierta inclinación el sensor confundía estos colores, sin embargo con otras inclinaciones no, y para el resto de colores no notamos ninguna inconsistencia.
Por otro lado probamos tapando el sensor, despojándolo de toda luz, pero siguió funcionando de manera correcta.
Entendemos que la luz puede afectar al sensor pero no de forma significativa, al menos a una distancia corta, como fueron las pruebas realizadas.

Otra prueba realizada fue cambiar la orientación de los objetos, que en un principio estaban perpendiculares al sensor y fueron levemente rotados ocasionando que los colores no se distinguieran correctamente, en el caso del magenta distinguía el cubo como rojo, por ejemplo. Entendemos que la orientación dle cubo si es crítica a la hora identificar el color, por lo tanto de usarse el sensor se debería intentar que los objetos a medir queden los más perpendicular posible al sensor de color.

Cámara

¿Que ventajas y desventajas ofrece el uso de la cámara frente al uso del sensor de color? ¿La característica del ambiente que respondió en la pregunta anterior, afecta en igual medida si usa la cámara?

Una ventaja importante del sensor de color frente a la cámara es que el sensor de color del kit ya tiene configurado un modo para reconocer colores, con este modo puede reconocer correctamente los colores negro, azul, verde, amarillo, rojo, blanco y marrón, por lo tanto, excepto para el color magenta es una muy buena forma de reconocer los colores. Mientras que la cámara requiere una configuración que lleva un tiempo y además no es tan precisa como el sensor de color.

Es importante tener en cuenta que la cámara puede empeorar significativamente sus medidas cuando la luz ambiente es baja, ya que la falta de iluminación afecta la calidad de la imagen. En cambio, el sensor de color no se ve afectado significativamente como se vio en la parte anterior.

Por otro lado, el sensor de color solo funciona si el sensor se encuentra del objeto a una distancia muy corta. Sin embargo, esta característica del ambiente no es un problema en el caso de la cámara, que se puede usar desde una distancia mayor. A su vez la cámara puede reconocer hasta 8 objetos al mismo tiempo, lo que permite obtener más datos con una única medición, mientras que el sensor de color solo puede detectar de un objeto a la vez. Además, la cámara nos puede dar las coordenadas del objeto en la imagen, lo que nos permitiría, por ejemplo, ubicarlo en el espacio.

References

[1] Facultad de Ingenieria. (2019, April 11). *Robótica Móvil*. Eva. Retrieved March 24, 2023, from https://eva.fing.edu.uy/pluginfile.php/25433/mod_resource/content/6/teorico/RBC23_Clase02_Robotica%20Movil.pdf

[2] LEGO. *Controlling the Motors*. leJOS. Retrieved March 24, 2023, from https://lejos.sourceforge.io/nxt/nxj/tutorial/MotorTutorial/ControllingMotors.htm

[3] LEGO. *leJOS API*. leJOS. Retrieved March 24, 2023, from https://lejos.sourceforge.io/nxt/nxj/api/index.html

[4] MaterialesLaboratorio. *Tacómetro - ¿Qué es, cómo funciona y para qué sirve?* Materiales de Laboratorio. Retrieved March 24, 2023, from https://materialeslaboratorio.com/tacometro/