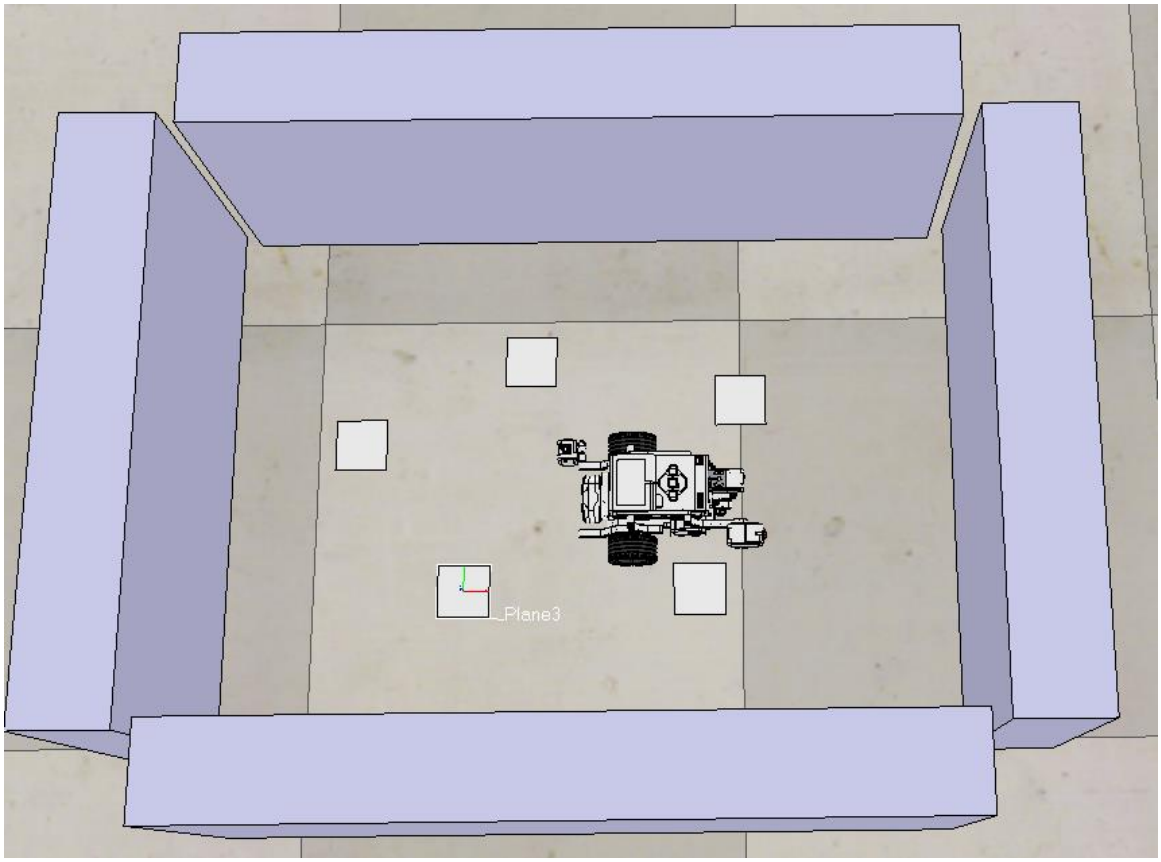


Informe de Práctica 6

Programación de Robots

David Ávila Jiménez || Pedro Antonio Aguilar Lima

Contenido

Resumen

En este informe se recoge toda la información relacionada con la Práctica 6 de programación de robots móviles. El objetivo de la práctica es aprender el funcionamiento de una arquitectura basada en comportamientos usando los LEGO EV3 en simulación mediante el entorno de programación Matlab/V-REP.

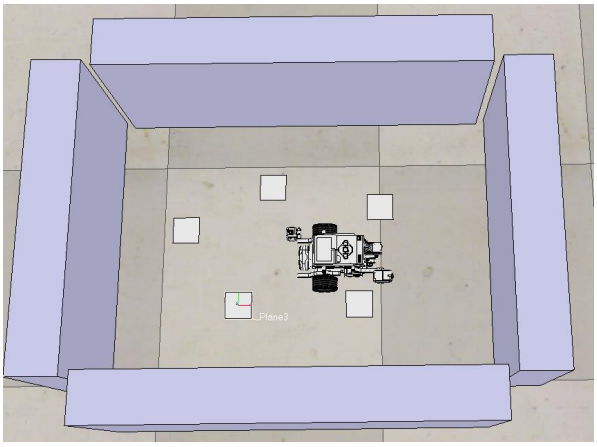
La tarea por desarrollar en esta práctica es una "recolección de alimentos", en donde se detallarán después una serie de requisitos a seguir en la práctica. Dos detalles iniciales a tener en cuenta en esta práctica, en la ejecución el robot será ayudado por el usuario que realiza la prueba, quitando la comida que el robot vaya encontrando en el transcurso de la prueba. Además, el proceso que realiza el robot será grabado y se subirá el mejor intento resultante de las pruebas que se realicen con el robot.

Introducción

En este apartado comenzaremos hablando un poco del escenario usado en las prácticas y de los requisitos que se deben tener en cuenta para su realización. También se comentará el esquema general que se debería seguir.

Escenario

Para esta práctica se volverá a utilizar el escenario usado en las prácticas 2, 3 y 5, que se nos proporcionaba en el campus virtual de la asignatura. En este caso, para esta última práctica se trabajará la zona que se encuentra cerrada por 4 paredes de color violeta, en cuyo interior tenemos una serie de objetos blancos que serán en nuestro caso los "alimentos" a recoger. El escenario puede ser cambiado de forma que la tarea se vuelva más complicada, pero nunca más fácil de resolver. En nuestro caso no hemos decidido probar otra configuración del escenario, pues con la usada por defecto se comprueba que el robot realiza bien su trabajo, aunque al ser de forma aleatoria, algunos intentos serán mejores que otros. También, se nos recomienda dejar el robot al comienzo lo más céntrico posible, en nuestro caso se cumple ese detalle, tal como se puede observar en la foto incluida abajo.



Restricciones

En la práctica se nos han impuesto una serie de restricciones o requisitos que se deben cumplir y que son los que se relatan a continuación:

- Cuando un alimento es detectado (mediante el sensor de luz) el robot deberá de encender su luz del panel central y detenerse. En caso de que el papel haya sido quitado y el botón del panel central sea pulsado de nuevo, podrá reanudar la marcha.
- El robot deberá evitar los obstáculos que haya en el entorno haciendo uso del sonar. Como el sonar tiene un rango limitado, cuando no perciba nada devuelve un 255 y debería ser tratado como un obstáculo (después se explicará que esto se ha seguido con pinzas.

- El tiempo máximo de trabajo del robot debe ser de dos minutos, aunque debería ser lo más bajo posible.

Código de la práctica

En el enunciado de las prácticas nos propone un esquema general a seguir en el código de desarrollo del programa. En nuestro caso, dicho esquema ha sido seguido, pero tomando nuestro propio punto de vista debido a que resultada más sencillo y cómodo realizarlo de dicha manera.

En el caso del bucle principal, mostrando por pantalla el estado de los sensores, se cumple. En el momento que se habla de implementar tres subrutinas, una por cada comportamiento del robot, se ha realizado mediante bucles while, aunque teniendo en cuenta la jerarquía de cada una para así conseguir el buen funcionamiento del robot.

```
TextOut(0,LCD_LINE1,'-- Practica 6 --');
TextOut(0,LCD_LINE2,'Presione el boton central para');
TextOut(0,LCD_LINE3,'comenzar con la prueba');


while(~ButtonPressed(BTNCENTER))
    % Esperamos a que se pulse el boton central
end
ClearScreen();

ResetRotationCount(OUT_A); % Establece a 0 los encoder de los dos motores
ResetRotationCount(OUT_C); % A izquierda, C derecha
SetSensorLight(IN_1); %Inicia el sensor de luz
SetSensorUltrasonic(IN_2); %Inica el sonar
t_ini = CurrentTick();  % Obtiene el tiempo de simulacion actual
tiempo = 120000; % Tiempo en milisegundos que debe durar el programa
con_Tiempo=t_ini; %Contador de tiempo que se usará en el programa
N=0; %Contador usado en el programa
marca_sonar_antigua = 30; %Variable que guarda el ultimo valor registrado por el sonar
```

La primera parte del programa se centra en iniciar los sensores y algunas variables que serán usadas más adelante en el bucle principal.

```
%Bucle principal que termina a los 120 segundos
while( (CurrentTick()-t_ini) <= tiempo)
    %Texto mostrad en la pantalla con los sensores y tiempo

    TextOut(1,LCD_LINE1,strcat('Tiempo: ',num2str(CurrentTick()-t_ini)));
    TextOut(1,LCD_LINE2,strcat('Luz: ',num2str(Sensor(IN_1))));
    TextOut(1,LCD_LINE3,strcat('Sonar: ',num2str(SensorUS(IN_2))));
    TextOut(1,LCD_LINE4,strcat('Botón: ',num2str(0)));



    %Control foraging
    %
    %Entra cuando detecta con el sensor de luz el blanco (70 marcado en el
    %sensor
    % Una vez retirado el papel y pulsado de nuevo el boton reanuda la
    % marcha
    while(Sensor(IN_1) >= 70 && (CurrentTick()-t_ini) <= tiempo)
        %Cambia el centro del panel a verde parpadeante y para el robot
        StatusLight(0,1);
        Off(OUT_AC);


        while(~ButtonPressed(BTNCENTER) && (CurrentTick()-t_ini) <= tiempo)
            TextOut(1,LCD_LINE1,strcat('Tiempo: ',num2str(CurrentTick()-t_ini)));
            TextOut(1,LCD_LINE2,strcat('Luz: ',num2str(Sensor(IN_1))));
            TextOut(1,LCD_LINE3,strcat('Sonar: ',num2str(SensorUS(IN_2))));
        end
        %Cambia la luz a naranja
        StatusLight(1,0);
    end
```

Ya en el bucle principal encontramos el primer comportamiento, que ocurre cuando se encuentra un alimento. En este caso en la jerarquía de comportamientos es el más alto por lo que los otros deben

dejar de funcionar en caso de que este lo esté haciendo. Para ello se comienza comprobando si se detecta un papel, aunque se tiene en cuenta si se han pasado los 2 minutos, pues justo ahí debe terminar el programa, se haya recogido o no el papel. En caso de encontrar un papel se cambia el estado de la luz central y se entra en un bucle hasta que se pulsa el botón y se retira el papel. Una vez hecho esto se cambia el estado de la luz, y se volverá a poner en marcha el robot.

```
while((SensorUS(IN_2) <= 20.5 || SensorUS(IN_2) == 255) ...
        && (CurrentTick()-t_ini) <= tiempo ...
        && Sensor(IN_1) < 70)

    %Guardamos el estado del sensor actual
    if(SensorUS(IN_2) ~= 255)
        marca_sonar_antigua = SensorUS(IN_2);
    end

    %Mostramos los datos de sensores y tiempo
    TextOut(1,LCD_LINE1,strcat('Tiempo: ',num2str(CurrentTick()-t_ini)));
    TextOut(1,LCD_LINE2,strcat('Luz: ',num2str(Sensor(IN_1))));
    TextOut(1,LCD_LINE3,strcat('Sonar: ',num2str(SensorUS(IN_2))));
    TextOut(1,LCD_LINE4,strcat('Botón: ',num2str(0)));

    %Establecemos una potencia a la ruedas aleatoria
    giro_izq = (rand()*10);
    giro_der = (rand()*10);

    %Aplicamos una potencia fuerte a las rueda girando hacia atrás
    %cuando se quede atrancado el robot
    if(Salto_Fuerte1 == 5 || Salto_Fuerte2 == 5 || Salto_Fuerte3 == 5)
        if(rand() >= rand())
            OnFwd(OUT_A, -25);
            OnFwd(OUT_C, -5);
        else
            OnFwd(OUT_C, -25);
            OnFwd(OUT_A, -5);
        end
        Salto_Fuerte1 = 0;
        Salto_Fuerte2 = 0;
        Salto_Fuerte3 = 0;

    %Aplicamo potencia hacia atrás con mayor potencia si el robot está
    %muy cerca de la pared
    elseif (SensorUS(IN_2) <= 10)

       OnFwd(OUT_AC, -15);
       Salto_Fuerte1 = Salto_Fuerte1+1;
    %Si el robot se encuentra cerca de la pared se realizar una potencia
    %moderada
    elseif (SensorUS(IN_2) <= 19)
        OnFwd(OUT_AC, -10);
        Salto_Fuerte2 = Salto_Fuerte2+1;
```

En esta parte del bucle principal controlamos el segundo comportamiento, el sonar. En este caso este bucle estaría por debajo en la jerarquía de comportamientos que debemos seguir. Para que entre en este bucle la distancia que marque el sonar será de 20.5 o que el sensor devuelva 255. Además, tendrá en cuenta si se ha encontrado un alimento por lo que no seguirá en ese caso y tampoco seguirá en caso de que el tiempo haya terminado.
Dentro de este bucle lo primero que realiza el programa es recoger datos del sonar en caso de que no sea 255, con esto se busca que se recojan datos del sonar, para que en caso de que devuelva en algún punto 255, no se le aplique siempre potencia hacia atrás a las ruedas, pues esto haría que en mitad del mapa se de el caso de que llegue un punto que solo se dirige hacia atrás cuando no debería. Casi siempre lo hará cuando está cercano a la pared, de ahí que se registre marcas antiguas del sonar.
Además, el robot aplicará una potencia fuerte hacia atrás, en caso de que se quede en varios intentos estancado en el mismo lugar, ya sea porque se encuentra a una distancia muy cercana a la pared de menos de 10 o también porque se encuentra en una distancia menor de 19 o porque el sonar devuelve todo el rato 255, sin poder salir del mismo lugar en el que lleva atrapado un tiempo. Con esto, no se evitará obviamente que siempre salga de un lugar en el que esté atrancado un rato pues lo debe hacer de manera aleatoria, pero si se consigue que sea el menor tiempo posible, además de que también se evita que se choque con la pared casi toda la ejecución.
Los siguientes casos del if controlan que no se acerque demasiado a la pared el robot aplicando cada vez más potencia hacia atrás en las ruedas, dependiendo de si se encuentra a una distancia de 10 o 19.

```
    elseif (SensorUS(IN_2) == 255 && marca_sonar_antigua < 18)
           marca_sonar_antigua = 30;
           Salto_Fuerte3 = Salto_Fuerte3+1;
           OnFwd(OUT_AC, -12);
    end
    %Bucle que permite aplicar la nueva potencia hacia atrás a las
    %ruedas durante un tiempo
    contador_tiempo2 = CurrentTick() +600;
    while((CurrentTick()-t_ini) <= tiempo && (CurrentTick() <= contador_tiempo2) ...
        && Sensor(IN_1) < 70 )

        TextOut(1,LCD_LINE1,strcat('Tiempo: ',num2str(CurrentTick()-t_ini)));
        TextOut(1,LCD_LINE2,strcat('Luz: ',num2str(Sensor(IN_1))));
        TextOut(1,LCD_LINE3,strcat('Sonar: ',num2str(SensorUS(IN_2))));
        TextOut(1,LCD_LINE4,strcat('Botón: ',num2str(0)));
    end

    %Segun el numero aleatorio sacado para cada rueda lo giraremos a la
    %izquiera o a la derecha
    if (giro_izq > giro_der)

        OnFwd(OUT_A, giro_izq*3);
        OnFwd(OUT_C, -5);
    else

        OnFwd(OUT_C, giro_der*3);
        OnFwd(OUT_A, -5);
    end

    %Bucle que permite aplicar la nueva potencia hacia atrás a las
    %ruedas durante un tiempo
    contador_tiempo2 = CurrentTick()+700;
    while((CurrentTick()-t_ini) <= tiempo && (CurrentTick() <= contador_tiempo2) ...
        && Sensor(IN_1) < 70 && (SensorUS(IN_2) ~= 255 || SensorUS(IN_2) > 10))
        TextOut(1,LCD_LINE1,strcat('Tiempo: ',num2str(CurrentTick()-t_ini)));
        TextOut(1,LCD_LINE2,strcat('Luz: ',num2str(Sensor(IN_1))));
        TextOut(1,LCD_LINE3,strcat('Sonar: ',num2str(SensorUS(IN_2))));
        TextOut(1,LCD_LINE4,strcat('Botón: ',num2str(0)));
    end
end
```

Para esta última parte del bucle del sonar, se aplica una potencia hacia atrás en caso de que el sonar devuelva 255 y la última marca del sonar es menor de 18. Tras esto, un bucle con un determinado tiempo se ejecuta para que los anterior if, puedan aplicar las potencias con un margen de tiempo. Después, y realiza aleatoriamente un giro en las ruedas con una potencia aleatoria durante un determinado tiempo.

```
con_Tiempo = CurrentTick()+1200;
    while((CurrentTick()-t_ini) <= tiempo && ...
        CurrentTick() <= con_Tiempo && Sensor(IN_1) < 70 ...
        && SensorUS(IN_2) >= 32 && SensorUS(IN_2) ~= 255)
        TextOut(1,LCD_LINE1,strcat('Tiempo: ',num2str(CurrentTick()-t_ini)));
        TextOut(1,LCD_LINE2,strcat('Luz: ',num2str(Sensor(IN_1))));
        TextOut(1,LCD_LINE3,strcat('Sonar: ',num2str(SensorUS(IN_2))));

        giro_izq = (rand()*10);
        giro_der = (rand()*10);

        if (N < 3)
            N=N+1;
            con_Tiempo2 = CurrentTick()+900;
            while((CurrentTick()-t_ini) <= tiempo && ...
                CurrentTick() <= con_Tiempo2 && Sensor(IN_1) < 70 ...
                && SensorUS(IN_2) >= 30 && SensorUS(IN_2) ~= 255)
                TextOut(1,LCD_LINE1,strcat('Tiempo: ',num2str(CurrentTick()-t_ini)));
                TextOut(1,LCD_LINE2,strcat('Luz: ',num2str(Sensor(IN_1))));
                TextOut(1,LCD_LINE3,strcat('Sonar: ',num2str(SensorUS(IN_2))));

                OnFwd(OUT_A, 15);
            end
        else


            if (giro_izq > giro_der)
                OnFwd(OUT_A, giro_izq);
                OnFwd(OUT_C, giro_der/2);
            else
                OnFwd(OUT_A, giro_izq/2);
                OnFwd(OUT_C, giro_der);
            end

            N = 0;
            con_Tiempo2 = CurrentTick()+450;
            while((CurrentTick()-t_ini) <= tiempo && ...
                CurrentTick() <= con_Tiempo2 && Sensor(IN_1) < 70 ...
                && SensorUS(IN_2) >= 30 && SensorUS(IN_2) ~= 255)

                TextOut(1,LCD_LINE1,strcat('Tiempo: ',num2str(CurrentTick()-t_ini)));
                TextOut(1,LCD_LINE2,strcat('Luz: ',num2str(Sensor(IN_1))));
                TextOut(1,LCD_LINE3,strcat('Sonar: ',num2str(SensorUS(IN_2))));


            end
        end
```

El último bucle, es para controlar el comportamiento con menor importancia en la jerarquía de comportamientos del robot. Este comportamiento solo ocurre durante un tiempo de 1,2 segundos. Además, para que se produzca el sensor del sonar no puede dar valores menores a 32 ya que estaríamos aproximándonos a la pared y pronto entraría en acción el sonar, también evitando en parte que haga un movimiento que perjudique el funcionamiento del robot. En caso de encontrar algún alimento también se sale del bucle.
Dentro bucle se le ha indicado al robot que cada tres iteraciones en las que el robot deambula de forma recta, luego lo hace girando aleatoriamente hacia algún lado. En cada una de las decisiones se da un tiempo para que pueda realizar la acción.

Conclusión

En esta práctica se ha realizado muchas pruebas de ejecución del robot, siempre buscando que el robot lo haga de la mejor forma posible. Aunque en la práctica había un esquema general con una serie de subrutinas, se ha aplicado según los conocimientos que se tienen de Matlab, pero aun así el robot sigue todas las pautas correctamente. En nuestro caso, se subirá el mejor intento realizado el robot donde ha conseguido recoger todos los alimentos, pero tras muchos cambios en el código y muchas pruebas, en concreto en las últimas 10, ha recogido siempre 4 alimentos y sin hacer movimientos extraños o que puedan hacer sospechar su aleatoriedad. Por tanto, aunque el robot no es perfecto, si realiza de la mejor manera posible todas sus tareas.