Fase 3. Robot móvil para recorrer un camino con obstáculos

Carlos Fernando Quintero Olaya(2010006),
carlos.fernando.quintero@correounivalle.edu.co,
Joan Esteban Velasco Larrea(1827539)
joan.velasco@correounivalle.edu.co
Jhonier Andrés Vargas (1745647),
jhonier.vargas@correounivalle.edu.co

Resumen. En el siguiente informe se evidencia como fue diseñada las acciones que debe realizar el robot seguidor de línea. De tal modo, que en esta tercera fase del proyecto, se utilizan los módulos implementados en la fase anterior para poder obtener la lectura de los sensores y poder actuar sobre los motores para el movimiento. Ahora le compete al informe explicar cómo debe moverse el robot ante determinados estímulos en sus entradas.

Palabras clave. Robot Seguidor de Línea, Robot seguidor de contornos, Programación AVR, Control de velocidad con PWM.

I. INTRODUCCIÓN

Al momento de llevar a cabo un desarrollo de un proyecto, es importante tener en cuenta las metas u objetivos para alcanzar, así como una metodología clara que permita lograr esas metas u objetivos propuestos. De esta manera se presenta la información correspondiente a la fase final del proyecto cuyo objetivo tiene llevar a cabo un robot móvil capaz de seguir una trayectoria dictada por una línea negra así como también evadir obstáculos que se presenten en medio del camino. Esta entrega final se logra por medio del seguimiento y guía de la metodología RUP, la cual consta de ciertas fases y requerimientos por cada fase llevada a cabo (3 fases) que están en constante evaluación y cambio según el equipo de trabajo vea conveniente o no hacer cambias en pro de mejorar el curso del desarrollo y resultado esperado.

II. DESARROLLO DE ACCIONES INDIVIDUALES

Para la concepción de un robot capaz de seguir una línea de color negro, y que el robot sea capaz de evadir obstáculos (abandonando temporalmente la línea del camino claro está) es necesario definir una serie de agente, u objetos, y las interacciones entre estos: Por ejemplo, el camino delimitado por la línea de color negro es uno de los objetos dentro del proyecto, como lo es también el robot mismo, y la interacción entre estos objetos es que el robot recorre el camino. La Fig. 1 muestra un diagrama conceptual de los objetos e interacciones, de tal manera que se ilustre la idea de lo que hace el robot para seguir el camino y evadir los obstáculos.

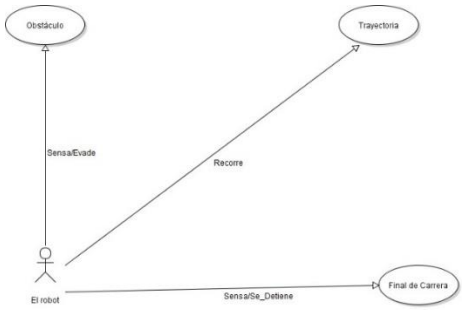


Fig 1. Diagrama Conceptual del Robot Móvil para Recorrer un Camino con Obstáculos.

A continuación, se presenta la estrategia y programación que utilizará el robot para recorrer los caminos y evada los obstáculos.

A. SEGUIR LINEA

La estrategia utilizada para que el robot recorra el camino, es llevada a cabo por sensores IR (emisor - receptor), que permiten constantemente estar evaluando, si el robot está sobre la línea negra y sobre la superficie blanca. De esta manera, se usa luz infrarroja, ya que cuando la luz incide sobre una superficie negra, la luz es absorbida en su totalidad, en cambio, cuando la luz infrarroja es reflejada por una superficie blanca, la luz se refleja casi por completo. Teniendo en cuenta este fenómeno físico, los sensores IR actúan como emisor de la luz infrarroja, y el receptor por un fotodiodo, el cual comprueba si la luz ha sido reflejada [1]. Este tipo de sensor arroja un voltaje de salida que es proporcional a la luz reflejada, en donde obtiene un voltaje alto cuando la superficie es clara, y para una superficie oscura es muy baja. Con esta información, se tienen entonces 4 casos:

1. Cuando el carro está sobre la línea negra los dos receptores reciben en su totalidad el reflejo de la luz infrarroja desde el receptor lo que significa entonces que el robot debe avanzar.
2. Cuando el sensor IR izquierdo está sobre la línea negra y el derecho no, lo que se tendrá es que el sensor izquierdo no detectará la luz reflejada de la superficie blanca, mientras que el derecho si, lo que deberá indicar al robot la corrección de su dirección y por ende

un movimiento hacía la izquierda para volver al caso anterior (1) y seguir avanzando.

3. Cuando el sensor IR derecho está sobre la línea negra y el izquierdo no, lo que se tendrá es que el sensor derecho no detectará la luz reflejada de la superficie blanca, mientras que el izquierdo si, lo que deberá indicar al robot la corrección de su dirección y por ende un movimiento hacía la derecha para volver al caso 1 y seguir avanzando.
4. El último caso es cuando ambos sensores IR están sobre una superficie negra por lo que no detectarán la luz reflejada, que se ha decidido tomar como caso de para o fin de trayecto.

Los casos mencionados anteriormente se ven de manera gráfica en la Fig. 2.

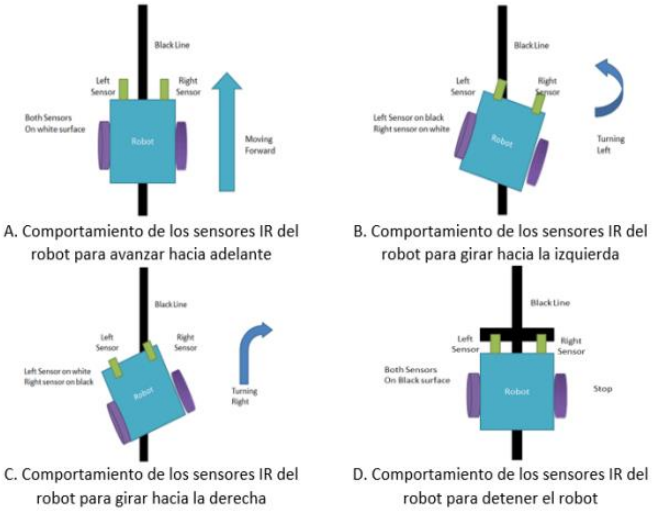


Fig. 2. Estrategia para mantener el robot sobre la línea.

Para la realización e implementación del módulo fue necesario crear dos parámetros que dentro del código sirven como máscaras o argumentos de validación, sobre las lecturas de los sensores encargados de estar informando sobre qué superficie se ubica el robot, para el sensor derecho se usa el pin C2 (ADC2) y para el sensor izquierdo el pin del C3 (ADC3).

```
#define LineaDerecha 0x04
#define LineaIzquierda 0x08
```

Fig. 3. Parámetros para la comunicación sensores IR - Microcontrolador.

Se configura el PWM a usar para controlar la velocidad de los motores, la configuración fue la siguiente:

```
void ConfigPWM(){
    DDRD |= (1<<5) | (1<<6); // Configura el puerto D5 y D6 como salidas de la señal PWM
    TCCR0A |= (1 << WGM01) | (1 << WGM00); // Configuració de Fast PWM
    TCCR0A |= (1 << COM0A1); // Non inverting mode del PWM del puerto OC0A (D6)
    TCCR0A |= (1 << COM0B1); // Non inverting mode del PWM del puerto OC0B (D5)
    TCCR0B |= (1<< CS01) | (1 << CS00); // prescaling 64
}
```

Fig. 4. Configuración PWM.

Tal como se puede ver en la Fig. 4, se configuran los pines 5 y 6 para tener la salida de la señal PWM, se configura el registro TCCR0A en el modo “Fast PWM” y los modos de no inversión a la salida de la señal PWM en los pines 5 y 6 del puerto D. Además se configura prescaler 64 del registro TCCR0B. Ahora bien, ya que sea configurado que en los pines 5 y 6, la salida sea una señal PWM, para configurar el duty cycle o ancho de pulso, para ajustar la velocidad a la que gira el motor se usa la siguiente función:

```
void VelocidadGiro(uint8_t VelIzq, uint8_t VelDer){
    OCR0A = VelIzq;
    OCR0B = VelDer;
}
```

Fig. 5. Ajuste del ancho de pulso y duty cycle del PWM

Como se observa, en la Fig. 5 dado a que se esta utilizando el Timer0, la máxima cuenta va desde 0 hasta 255, en donde, el 0 es una señal en estado bajo con duty cycle del 0%, y el valor máximo de 255, significa una señal en estado alto con un duty cycle del 100 %. De esta manera a través de estas funciones, sirve para modificar el ancho de pulso de la señal que sale de los pines 5 y 6, los cuales se conectan a la entrada ENA y ENB del módulo L298N para ajustar la velocidad de giro de cada motor.

Una vez configurado el PWM y los parámetros sobre los 2 sensores, el código consta de 4 filtros o condicionales if, en donde se evalúa qué caso se tiene de los presentes en la Fig. 1, en base a la información suministrada a partir de las 2 máscaras, como se puede ver en la Fig. 6.

```
int main(void)
{
    DDRC &= ~(1<<0) & ~(1<<1) & ~(1<<2) & ~(1<<3);
    DDRD |= (1<<3) | (1<<4) | (1<<2) | (1<<7);
    ConfigPWM();
    while (1)
    {
        VelocidadGiro(240,240);

        if (!DeteccionLineaDER(LineaDerecha) && !DeteccionLineaIZQ(LineaIzquierda)){
            TForwards();
        }
        if (!DeteccionLineaDER(LineaDerecha) && DeteccionLineaIZQ(LineaIzquierda)){
            TLeft();
        }
        if (DeteccionLineaDER(LineaDerecha) && !DeteccionLineaIZQ(LineaIzquierda)){
            TRight();
        }
        else{
            stop();
        }
    }
```

Fig. 6. Código principal modulo seguidor línea.

Como se ve en la Fig. 6, dependiendo de qué caso se tiene se hace el llamado primero a una función encargada de la configuración de la velocidad de los motores, una función que le indica hacía donde debe moverse dependiendo del caso en el que se encuentre. El primer filtro corresponde al caso A. de la Fig. 2, el segundo al B, el tercero al C y el else del final corresponde al D. (Ver la sección de Pruebas dentro de los Anexos).

Las diferentes funciones que se llaman dependiendo del caso son configuraciones del puerto que se comunica con el puente H para modificar el sentido de giro de las ruedas, y así cambiar el sentido de la trayectoria del robot.

```
#include <avr/io.h>
#include <stdio.h>
#define dir PIND
#define I   PORTD
#define I1  3
#define I2  4
#define I3  2
#define I4  7
void TRight();
void TLeft();
void TForwards();
void TBackwards();
void stop();
```

Fig. 7. Configuración módulo .h funciones movimiento del robot.

En la Fig. 7, se observa la definición de las funciones para cambiar la dirección del robot, así como una configuración del puerto D para hacer más fácil la implementación de las funciones de dirección del robot.

```
#include "Motores.h"
void TRight(){
    I &= ~(1<<I1) & ~(1<<I2) & ~(1<<I3);
    I|=(1<<I4);
}
////////////////
void TLeft(){
    I &= ~(1<<I1) & ~(1<<I3) & ~(1<<I4);
    I|=(1<<I2);
}
///////////////
void TForwards(){
    I &= ~(1<<I1) & ~(1<<I3);
    I|= (1<<I2) | (1<<I4);
}
////////////////
void TBackwards(){
    I &= ~(1<<I2) & ~(1<<I4);
    I |= (1<<I1) | (1<<I3);
}
////////////////
void stop(){
    I &=  ~(1<<I1) & ~(1<<I2) & ~(1<<I3) & ~(1<<I4);
}
```

Fig. 8. Implementación funciones movimiento robot.

La configuración de los diferentes movimientos se hace en base a la comunicación de las combinaciones correspondientes y correctas a través de los pines del puerto D hacía el puente H, para que los motores cambien su posición según lo que se requiere en el momento tal cual como está en la Fig. 8.

B. SEGUIR CONTORNO DEL OBSTACULO

Para el desarrollo de la acción de seguir el contorno (o evasión de obstáculo) con ayuda de los sensores infrarrojos FC-51, para la detección de obstáculos, se consideraron distintas situaciones: Ambos sensores no detectan, el sensor delantero detecta el lateral no, el lateral detecta el de enfrente no y ambos sensores detectan.

Si ambos sensores no detectan es que el robot se encuentra en una esquina, del obstáculo, y dada la forma en que esquiva, éste debe girar a la derecha desde la perspectiva de éste hasta que el sensor lateral vuelva a encontrar el obstáculo. La Fig. 9.A ilustra dicha situación. Además de esto, el robot tiene a desviarse del camino cuando sigue un muro, para corregir dicho desvío también se gira a la derecha desde la perspectiva del robot hasta que vuelva a encontrar un muro en el lateral, la Fig. 9.B ilustra dicho caso.

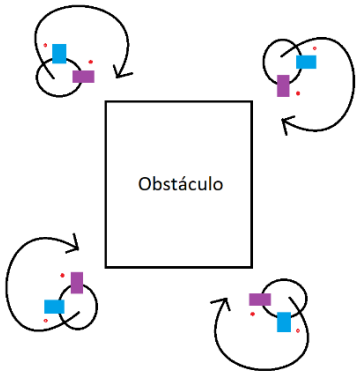


Fig. 9.A . Ambos sensores no detectan el obstáculo en una esquina de este último.

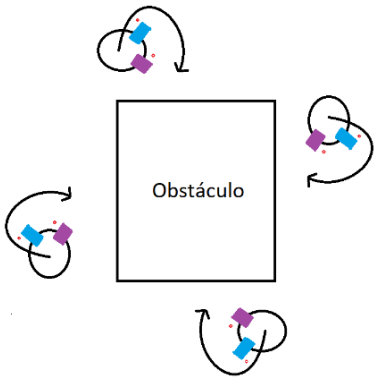


Fig. 9.B . Ambos sensores no detectan el obstáculo desviándose del camino.

Si el sensor de enfrente detecta y el lateral no puede deberse a dos situaciones, la primera es que el robot seguía la línea y justo se encontró con un obstáculo, en cuyo caso debe girar a la izquierda, el otro obstáculo es que justo en la esquina del obstáculo se halle un muro, en cuyo caso el robot debería de ignorar el muro y girar a la derecha. Dado que ambas situaciones tienen una lectura de los sensores iguales, el primer caso se trata en la parte IV, representa la transición entre el estado de seguimiento de línea y el estado de evasión de obstáculos. Por lo que para la acción de evasión de obstáculos se trató el segundo caso. La Fig. 10 muestra el caso de referencia.

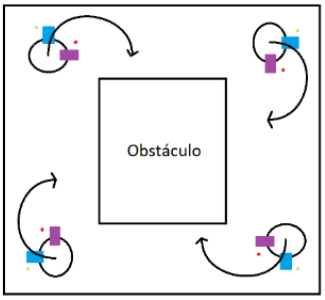


Fig. 10. El sensor delantero detecta y el lateral no, el robot encontró un muro a parte del obstáculo.

Si el sensor lateral detecta y el frontal no, significa que el robot tiene el obstáculo a un lado suyo, en este caso el robot debe simplemente avanzar. La Fig. 11 ilustra el caso.

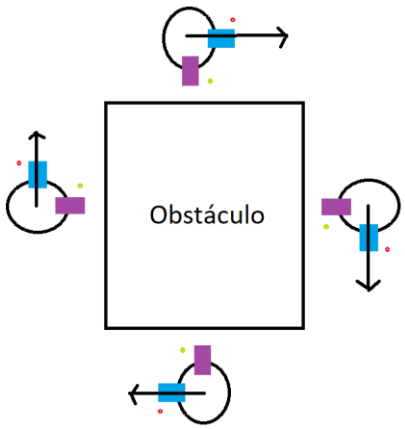


Fig. 11. El sensor lateral detecta y el frontal no, el robot está alineado esquivando el obstáculo.

Si el frontal detecta y el lateral también significa que el robot se encuentra en una esquina que lo rodea, en cuyo caso debe girar a la izquierda hasta que el sensor frontal deje de detectar. La Fig. 12 ilustra la situación

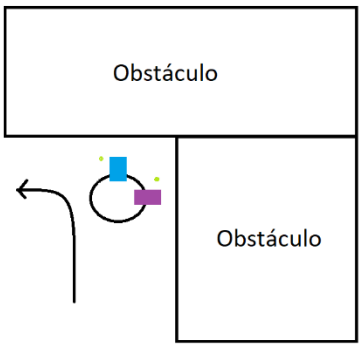


Fig. 12. Ambos sensores, lateral y frontal, detectan, el robot se encuentra en una esquina rodeándolo.

Para la realización e implementación del módulo fue necesario crear dos parámetros que dentro del código sirven como máscaras o argumentos de validación, sobre las lecturas de los sensores encargados de estar informando sobre qué superficie se ubica el robot, para el sensor frontal se usa el pin C0 (ADC0) y para el sensor lateral el pin del C1 (ADC1).

```
#define Ofrente 0x01
#define Olateral 0x02
```

Fig. 13. Parámetros para la comunicación sensores FC-51-junto con el Microcontrolador

Ahora bien, al explicarse las diferentes situaciones junto con los pines a usar, hay que tener en cuenta que el código a programar consta de 4 filtros o condicionales if, en donde se evalúa qué caso corresponde de acuerdo con las Fig 9, 10, 11 y 12, en base a la información suministrada a partir de las 2 máscaras, como se puede ver en la Fig. 14.

```
int main(void)
{
    DDRC &= ~(1<<0) & ~(1<<1) & ~(1<<2) & ~(1<<3);
    DDRD |= (1<<3) | (1<<4) | (1<<2) | (1<<7);
    ConfigPWM();
    while (1)
    {
        VelocidadGiro(240,240);
        if( !ObjetoAlFrente(Ofrente) && !ObjetoAlLado(Olateral) ){
            TRight();
            _delay_ms(50);
        }
        else if ( !ObjetoAlFrente(Ofrente) && ObjetoAlLado(Olateral) ){
            TForwards();
            _delay_ms(50);
        }
        else if ( ObjetoAlFrente(Ofrente) && !ObjetoAlLado(Olateral) ){
            TRight();
            _delay_ms(50);
        }
        else{
            TLeft();
            _delay_ms(50);
        }
    }
}
```

Fig. 14. Código principal modulo seguidor línea.

Como se ve en la Fig. 14, dependiendo de qué caso se tiene se hace el llamado primero a una función encargada de la configuración de la velocidad de los motores, una función que le indica hacía donde debe moverse dependiendo del caso en el que se encuentre. El primer filtro corresponde al caso A. de la Fig. 9, el segundo al de la Fig. 10, el tercero al de la Fig.11 y el else del final corresponde al de la Fig.12. (Ver la sección de Pruebas dentro de los Anexos).

IV. DESARROLLO AGRUPADO Y LÓGICA DE EJECUCIÓN DE LAS ACCIONES EN CONJUNTO

Ahora bien, de acuerdo con las estrategias explicadas y a los diferentes códigos implementados, para realizar estas acciones de seguir línea y evadir obstáculos se ha obtenido el siguiente diagrama de flujo:

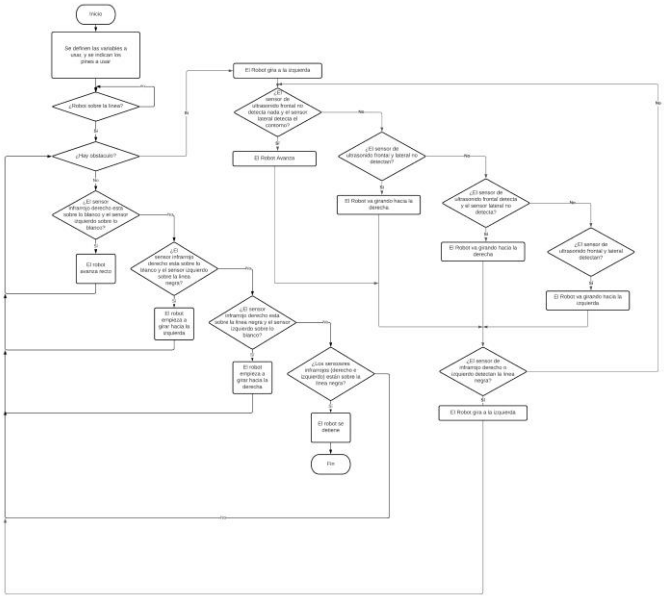


Fig. 15. Diagrama de Flujo del funcionamiento del robot

De esta manera, como se evidencia en la Fig.15, para implementar todo en conjunto primero se definen los puertos y se define la variable SeguirLinea, que sirve como una bandera para determinar las acciones que se va a realizar, es por ello que luego es necesario verificar si hay un objeto en frente del robot, de tal manera que si lo hay la bandera SeguirLinea se pone en 0, o en caso contrario de que no haya un objeto la bandera Seguir Línea se pone en 1, tal y como se observa a continuación:

```
#define LineaDerecha 0x04
#define LineaIzquierda 0x08
#define Ofrente 0x01
#define Olateral 0x02

int main(void)
{
    DDRC &= ~(1<<0) & ~(1<<1) & ~(1<<2) & ~(1<<3);
    DDRD |= (1<<3) | (1<<4) | (1<<2) | (1<<7);
    ConfigPWM();
    int SeguirLinea;
    while (1)
    {
        if(ObjetoAlFrente(Ofrente)) SeguirLinea = 0;
        else SeguirLinea = 1;
```

Fig. 16. Definición de Puertos de entrada y salida, y determinación de acción a través de la bandera SeguirLinea

De esta manera, como se observa en la Fig. 16 ya que la bandera SeguirLinea determina las acciones o ejecución de si seguir la línea, o evadir obstáculo, siendo necesario usar dos while aparte, manejado a través de esta bandera. De tal modo que dentro del primer while si SeguirLinea es igual a 1, luego se comprueba dentro de este mismo si el sensor frontal FC-51 detecta algún obstáculo, ejecutando así la acción de que el robot gire hacia su izquierda, posicionándose paralelamente sobre el objeto, y haciendo que SeguirLinea sea igual a 0 y rompiendo el ciclo con un break, solo si el robot detecta un obstáculo. En caso contrario, si no hay un obstáculo el robot simplemente se ejecuta las acciones de seguir la línea que fue explicado en la Sección 2A.

```
VelocidadGiro(240,240);
while (SeguirLinea == 1 ){
    if(ObjetoAlFrente(Ofrente)){
        SeguirLinea = 0;
        TLeft();
        _delay_ms(450);
        stop();
        break;
    }
    else{
        if (!DeteccionLineaDER(LineaDerecha) && !DeteccionLineaIZQ(LineaIzquierda)){
            TForwards();
        }
        if (!DeteccionLineaDER(LineaDerecha) && DeteccionLineaIZQ(LineaIzquierda)){
            TLeft();
        }
        if (DeteccionLineaDER(LineaDerecha) && !DeteccionLineaIZQ(LineaIzquierda)){
            TRight();
        }
        else{
            stop();
        }
    }
}
```

Fig. 17. Ejecución y comprobación para seguir la línea

Como se observa en la Fig. 17, tal y como se había explicado antes si no hay un objeto enfrente el robot ejecuta las acciones para seguir la línea. En caso contrario, si detecta un obstáculo se ejecuta una acción de que gire hacia la izquierda, ejecutándose en un determinado tiempo para luego parar, y romper el ciclo, haciendo a su vez que SeguirLinea sea igual a 0.

Ahora bien, para la evasión de obstáculos tiene la misma lógica anterior, tal y como se observa a continuación:

```
while (SeguirLinea == 0){
    if( DeteccionLineaDER(LineaDerecha) || DeteccionLineaIZQ(LineaIzquierda) ){
        TForwards();
        _delay_ms(10);
        stop();
        _delay_ms(500);
        VelocidadGiro(210,210);
        while  (!DeteccionLineaIZQ(LineaIzquierda)) TLeft();
        stop();
        TRight();
        _delay_ms(50);
        stop();
        SeguirLinea = 1;
        break;
    }
    else{
        if( !ObjetoAlFrente(Ofrente) && !ObjetoAlLado(Olateral) ){
            TRight();
            _delay_ms(50);
        }
        else if ( !ObjetoAlFrente(Ofrente) && ObjetoAlLado(Olateral) ){
            TForwards();
            _delay_ms(50);
        }
        else if ( ObjetoAlFrente(Ofrente) && !ObjetoAlLado(Olateral) ){
            TRight();
            _delay_ms(50);
        }
        else{
            TLeft();
            _delay_ms(50);
        }
    }
}
```

Fig. 18. Ejecución y comprobación para evadir obstáculos

Como se evidencia en la Figura 18, a través de este segundo while si SeguirLinea es igual a 0, se comprueba dentro de este mismo si el sensor TCRT5000 izquierdo o derecho, detecta la línea negra, ejecutando así la acción de que el robot gire un poquito hacia adelante, y luego este gire hacia su izquierda hasta que nuevamente el sensor izquierdo detecte la línea, para así estar casi encima de línea negra, por lo que es necesario moverse un toque hacia su derecha, estando ahora el robot más centrado y haciendo que SeguirLinea sea igual a 1, y se rompa este ciclo, si ya el robot ha terminado de evadir el obstáculo.. En caso contrario, si todavía no ha evadido el obstáculo, el robot simplemente se ejecuta las acciones de evadir o seguir el contorno del obstáculo que fue explicado en la Sección 2B. (Ver la sección de Pruebas dentro de los Anexos).

V. DISEÑO FINAL DEL DIAGRAMA ESQUEMÁTICO

Ahora bien, después de haber resaltado los puertos que se han utilizado para el desarrollo del robot, se ha facilitado el diagrama de conexión de los módulos sobre el microcontrolador, para su mejor entendimiento:

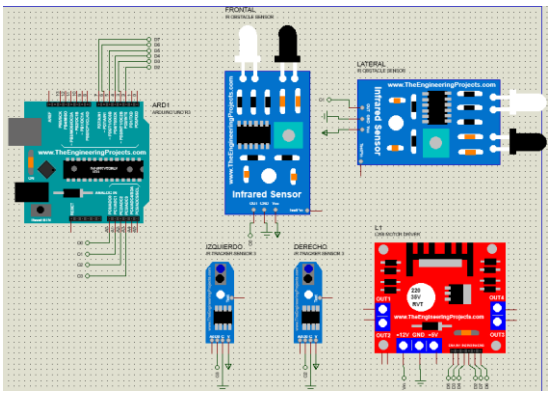


Figura 19. Diagrama de Conexiones

Como se observa en la Figura 19, los módulos se conectan al microcontrolador de la placa de Arduino UNO (Atmega328P), de la siguiente manera:

- Los sensores infrarrojos FC-51 Frontal y Lateral, es necesario que la salida OUT del módulo, se conecte al pin C0 (ADC0) para el sensor frontal, y para el lateral se debe conectar al pin C1 (ADC1) del microcontrolador.

- Los sensores infrarrojos TCRT5000 Izquierdo y Derecho, es necesario que la entrada D0 del módulo, se conecte al pin C3 (ADC3) para el izquierdo, y para el derecho se debe conectar al pin C2 (ADC2) del microcontrolador.

- Finalmente el Motor Drive L298N, las entradas ENA y ENB del módulo se conectan a los pines que puedan generar señales PWM, las cuales son el pin D5 y D6 del microcontrolador, siendo esto útil para controlar las revoluciones de los motores. Por otro lado, las entradas IN1, IN2, IN3, IN4, se deben conectar a los pines digitales del ATMEGA como lo son el D3, D4, D2 Y D7, los cuales sirven para controlar los diferentes modos de operación (Avance, Giro a la derecha, Giro a la izquierda y Detener).

V. CONCLUSIONES

Durante este proyecto se aprendió que:

- El configurar una señal PWM permite ajustar la velocidad a la que gira los motores. De tal manera que permite que el robot no se salga, de la línea si ajusta una velocidad lenta.

- La estrategia para realizar un correcto seguimiento de línea se debe tener en cuenta la lectura de los dos sensores de línea TCRT5000. De tal manera que:

- Cuando el sensor IR de la izquierda y derecha, se encuentran sobre la superficie blanca el robot avanza hacia adelante.

- Cuando el sensor IR de la izquierda esta sobre la línea oscura, y el sensor de la derecha esta sobre la superficie blanca, el robot deberá girar hacia la izquierda.

- Cuando el sensor IR de la izquierda esta sobre la superficie blanca, y el sensor de la derecha esta sobre la línea oscura, el robot deberá girar hacia la derecha.

- Cuando ambos sensores IR, izquierdo y derecho, detectan la línea oscura, indica que el camino ha finalizado, por tanto el robot deberá detenerse.

- La estrategia para realizar un correcto seguimiento de contorno o evasión del obstáculo se debe tener en cuenta la lectura de los dos sensores de obstáculos FC-51. De tal manera que:

- Cuando el sensor de obstáculos Frontal y Lateral no detectan, el robot debe girar a la derecha.

- Cuando solo detecta el sensor Lateral y no el Frontal, el robot debe avanzar hacia adelante.

- Cuando solo detecta el sensor Frontal y no el Lateral, el robot debe girar hacia la derecha.

- Cuando el sensor de obstáculos Frontal y Lateral detectan, el robot debe girar a la izquierda.

- Separar un proyecto en fases que se centren en distintos aspectos del problema a tratar a veces pude resultar útil.

- El trabajar de manera modular permite una análisis y desarrollo más práctico de proyectos relacionados con software o programación, pues se puede operar de manera más organizada y así detectar errores más fácilmente.

- La metodología de desarrollo RUP es una metodología estructurada en la que los desarrolladores deben identificar y planear una estrategia para la resolución de un problema incluso antes de proceso de programación.

VII. REFERENCIAS

[1] S. S. Suryawan, S. M. Musamwar, S.R. Kolhe, S. A. Thengane, S. S. Hanumante, P. H Sahare, “Line Follower & Obstacle Avoider Robot”, IRJET, Volume 06, 2019. Disponible en: https://www.irjet.net/archives/V6/i12/IRJET-V6I1250.pdf

ANEXOS

	Universidad del Valle Robot móvil para recorrer un camino con obstáculos	Rev.: 000
Título: **ESPECIFICACIÓN DE REQUERIMIENTOS FUNCIONALES**	**Documento :** **ERF-001**	Página : **1 de 1**

REVISIÓN HISTÓRICA			
Rev.	**Descripción del Cambio**	**Autor**	**Fecha**
001	Construcción del documento	Carlos Fernando Quintero, Jhonier Andrés Vargas, Joan Velasco	2021-11-21
002	Actualización	Carlos Fernando Quintero, Joan Velasco, Jhonier Andrés Vargas.	2022-02-04

Ref #	Funciones	Categoría
1.0	El robot móvil debe ser capaz de detectar los diferentes obstáculos en el camino, usando dos sensores infrarrojos con referencia FC-51	**E**
2.0	El robot móvil debe ser capaz de detectar la línea negra, usando 2 sensores ópticos infrarrojos TCRT5000	**E**
3.0	El robot móvil debe ser capaz de controlar el sentido de giro de cada uno de los motores por medio del módulo L298N y el microcontrolador ATmega328p.	**E**
4.0	El robot debe ser capaz de mantenerse dentro de la trayectoria y detenerse una vez llega a un final de carrera señalado en la pista.	**E**
5.0	El robot debe ser capaz de evadir los obstáculos rodeándolos.	**E**

Anexo 1. Requerimientos Funcionales del Robot móvil para recorrer un camino con obstáculos

	Universidad del Valle Robot móvil para recorrer un camino con obstáculos	Rev.: 000
Título: **ESPECIFICACIÓN DE REQUERIMIENTOS NO FUNCIONALES**	**Documento :** **ERF-001**	Página : **1 de 1**

REVISIÓN HISTÓRICA			
Rev.	**Descripción del Cambio**	**Autor**	**Fecha**
001	Construcción del documento	Carlos Fernando Quintero, Jhonier Andrés Vargas, Joan Velasco	2021-11-21
002	Actualización	Carlos Fernando Quintero, Joan Velasco, Jhonier Andrés Vargas.	2022-02-04

Ref #	Descripción	Categoría
1.0	Para la realización de este proyecto se usará el Atmel Studio 7.0	**E**
2.0	Se usará un microcontrolador Atmega328 a través del Hardware Arduino	**E**
3.0	Para la realización del esquema físico y conexiones se usará Proteus 8.8	**O**

Anexo 2. Requerimientos No Funcionales del Robot móvil para recorrer un camino con obstáculos

	Universidad del Valle Robot móvil para recorrer un camino con obstáculos	Rev.: 002
Título: **CASO DE USO** **Requerimiento funcional 1.** El robot móvil debe ser capaz de detectar los diferentes obstáculos en el camino, usando dos sensores infrarrojos (uno enfrente y otro a un lado) con referencia FC-51	**Documento :** **CUR-001**	Página : **1 de 1**

REVISIÓN HISTÓRICA			
Rev.	**Descripción del Cambio**	**Autor**	**Fecha**
001	Construcción de caso uso	Carlos Fernando Quintero, Jhonier Andrés Vargas, Joan Velasco	2021-11-21
002	Actualización	Carlos Fernando Quintero, Joan Velasco, Jhonier Andrés Vargas.	2022-02-04

INFORMACIÓN GENERAL	
Actores:	Robot (HW) y el firmware (SW)
Propósito:	El robot debe ser capaz de detectar los obstáculos para evitar choques, a través de dos sensores infrarrojos (uno en frente y otro a un lado) con referencia FC-51
Resumen:	Cuando el robot móvil detecte un obstáculo, por medio de los sensores de infrarrojos, se enviará las señales al microcontrolador para que la procese según sea el caso de detección, es decir, si se detectó objeto al frente, al lado o en ambos lados.
Tipo:	Real

Curso Normal de los Eventos	
Acción del Robot	**Respuesta del Firmware**
	1. Configurar los puertos de entrada (DDRC) y salida (DDRB) para configurar los sensores infrarrojos, tanto para la detección del obstáculo, como de la comunicación del tipo de detección (objeto al frente, al lado o ambos lados).
2. Los dos sensores infrarrojos envían señales eléctricas al puerto DDRC de entrada según la detección hecha.	
	3. Se verifica por medio de 3 filtros qué tipo de detección es por medio de la lectura del puerto C a través de PINC (objeto al frente, al lado o ambos lados).
	4. Dependiendo del tipo de detección (objeto al frente, al lado o ambos lados) se ponen en 1 o 0 los pines del puerto DDRB encargado para la comunicación del tipo de detección (objeto al frente, al lado o ambos lados).ssS

Anexo 3.1. Caso de uso para el Requerimiento funcional 1

	Universidad del Valle Robot móvil para recorrer un camino con obstáculos	Rev.: 000
Título: **CASO DE USO** **Requerimiento funcional 2.** El robot móvil debe ser capaz de detectar la línea negra, usando 2 sensores ópticos infrarrojos TCRT5000	**Documento :** **CUR-001**	Página : **1 de 1**

REVISIÓN HISTÓRICA			
Rev.	**Descripción del Cambio**	**Autor**	**Fecha**
001	Construcción de caso uso	Carlos Fernando Quintero, Jhonier Andrés Vargas, Joan Velasco	2021-11-21
002	Actualización	Carlos Fernando Quintero, Joan Velasco, Jhonier Andrés Vargas.	2022-02-04

INFORMACIÓN GENERAL	
Actores:	Robot, firmware.
Propósito:	El robot móvil debe ser capaz de detectar la línea negra, usando 2 sensores ópticos infrarrojos TCRT5000
Resumen:	Lo que se busca es que el robot móvil sea capaz de reconocer la trayectoria a seguir, la cual es una línea negra por medio de 2 sensores ópticos infrarrojos.
Tipo:	Real

Curso Normal de los Eventos	
Acción del robot	**Respuesta del firmware**
	1. Configurar los puertos de entrada y salida para configurar los dos sensores ópticos infrarrojos, con el comando DDRC.
2. Los dos receptores infrarrojos envían una señal eléctrica al puerto de entrada y salida.	
	3. Se realiza una lectura del puerto C, a través del comando PINC.
	4. Se compara si la mascara correspondiente a cada sensor si es igual a la lectura del PINC
	5. Retorna un 1, si la mascara es igual al PINC, significando que el sensor paso sobre la línea negra. En caso contrario, si la mascara es diferente a la lectura del PINC retorna un 0.

Anexo 3.2. Caso de uso para el Requerimiento funcional 2

	Universidad del Valle Robot móvil para recorrer un camino con obstáculos	Rev.: 000
Título: **CASO DE USO** **Requerimiento funcional 3** El robot móvil debe ser capaz de controlar el sentido de giro de cada uno de los motores por medio del módulo L298N y el microcontrolador ATmega328p.	**Documento :** **CUR-001**	Página : **1 de 1**

REVISIÓN HISTÓRICA			
Rev.	**Descripción del Cambio**	**Autor**	**Fecha**
001	Construcción de caso uso	Carlos Fernando Quintero, Jhonier Andrés Vargas, Joan Velasco	2021-11-21
002	Actualización	Carlos Fernando Quintero, Joan Velasco, Jhonier Andrés Vargas.	2022-02-04
003	Nueva Actualización	Carlos Fernando Quintero, Joan Velasco, Jhonier Andrés Vargas.	2022-03-17

INFORMACIÓN GENERAL	
Actores:	Robot, firmware.
Propósito:	El robot debe ser capaz de controlar el sentido de giro de los motores para moverse.
Resumen:	Se espera que se pueda cambiar el sentido de giro de cada una de las ruedas para poder seguir la trayectoria marcada o esquivar el obstáculo, según el robot lo requiera.
Tipo:	Real

Curso Normal de los Eventos
Respuesta del firmware
1. Se configuran los puertos de salida (Puerto D) para la comunicación unidireccional entre el microcontrolador ATmega328p y el puente H L298N.
2. Se hace el llamado de la dirección a la que se requiere avanzar (Parar, Avanzar, Derecha, Izquierda y Retroceder).

Si el valor de la variable dirección es Parar, entonces:

Curso Normal de los Eventos
Respuesta del firmware
3. Se escribe en el registro del puerto de salida (PORTD) y en los bits menos significativos el valor de cero, para desactivar todos los pines I1, I2, I3, e I4 del L298N.

Si el valor de la variable dirección es Avanzar, entonces:

Curso Normal de los Eventos
Respuesta del firmware
4. Se escribe en el registro del puerto de salida (PORTD), escribiendo un 1 lógico sobre el pin 4 y 7, para activar respectivamente los pines I2 e I4 del L298N

Si el valor de la variable dirección es Derecha, entonces:

Curso Normal de los Eventos
Respuesta del firmware
5. Se escribe en el registro del puerto de salida (PORTD), escribiendo 1 lógico solamente sobre el pin 7, para activar el pin I4 del L298N.

Si el valor de la variable dirección es izquierda, entonces:

Curso Normal de los Eventos
Respuesta del firmware
6. Se escribe en el registro del puerto de salida (PORTD), escribiendo un 1 lógico sobre el pin 4, para activar el pin I2 del L298N

Si el valor de la variable dirección es Retroceder, entonces:

Curso Normal de los Eventos
Respuesta del firmware
7. Se escribe en el registro del puerto de salida (PORTD), escribiendo 1 lógico sobre el pin 3 y 2, para activar los pínes I1 e I3 del L298N

Anexo 3.3. Caso de uso para el Requerimiento funcional 3

	Universidad del Valle Robot móvil para recorrer un camino con obstáculos	Rev.: 000
Título: **CASO DE USO** **Requerimiento funcional 4.** El robot debe ser capaz de mantenerse dentro de la trayectoria y detenerse una vez llega a un final de carrera señalado en la pista.	**Documento :** **CUR-001**	Página : **1 de 1**

REVISIÓN HISTÓRICA			
Rev.	**Descripción del Cambio**	**Autor**	**Fecha**
001	Construcción de caso uso	Carlos Fernando Quintero, Jhonier Andrés Vargas, Joan Velasco.	2021-11-21
002	Actualización	Carlos Fernando Quintero, Jhonier Andrés Vargas, Joan Velasco.	2022-03-17

INFORMACIÓN GENERAL	
Actores:	firmware.
Propósito:	El robot debe ser capaz de seguir la línea negra detectada y avanzar
Resumen:	Lo que se busca es que el robot móvil sea capaz de reconocer la trayectoria a seguir, la cual es una línea negra por medio de 2 sensores ópticos infrarrojos.
Tipo:	Real

Curso Normal de los Eventos	
Acción del robot	**Respuesta del firmware**
	Se configura el puerto C como entrada (pin 2 y 3) para la lectura de los sensores IR responsables de la detección de la posición del robot y el puerto D como salida (pines 0, 1, 2 y 3) para el manejo de los motores a través del puente H.
Los sensores IR envían la información de la detección de la superficie (blanca o negra) de forma análoga a un valor de voltaje diferente, según sea la detección.	
	La información es recibida en los pines 2 y 3 del puerto C.
	Se leen los valores medidos por los sensores a través de los parámetros definidos para captar la información de la detección "LineaDerecha" (pin 2 puerto C) y "LineaIzquierda" (pin 3 puerto C).

Curso Normal de los Eventos
Respuesta del firmware
1. Se verifica el estado de los parámetros definidos en el software que usan las funciones o filtros para saber qué acción tomarán los sensores izquierdo y derecho: 00, 01, 10, 11.

Si el estado en conjunto de los parámetros es 00, entonces:

Curso Normal de los Eventos
Respuesta del firmware
Se llama la función stop() para detener el robot, pues el estado conjunto 00 significa que el robot ha terminando el recorrido.

Si el estado en conjunto de los parámetros es 01, entonces:

Curso Normal de los Eventos
Respuesta del firmware
Se llama la función TLeft(); pues este estado significa que el sensor izquierdo está sobre la línea negra y el derecho no, por lo que es necesario la correción de la posición del robot y por ende que gire a la izquierda.

Si el estado en conjunto de los parámetros es 10, entonces:

Curso Normal de los Eventos
Respuesta del firmware
Se llama la función TRight(); pues este estado significa que el sensor derecho está sobre la línea negra y el izquierdo no, por lo que es necesario la correción de la posición del robot y por ende que gire a la derecha.

Si el estado en conjunto de los parámetros es 11, entonces:

Curso Normal de los Eventos
Respuesta del firmware
Se llama la función VelocidadGiro(); para configurar la velocidad a la que avanzarán los motores y por ende el robot, además se llama la función TForwards(), pues el estado conjunto 11 signficia que los dos sensores están detectando la superficie blanca y por ende el robot está sobre la línea negra para seguirla y avanzar.

Anexo 3.4. Caso de uso para el Requerimiento funcional 4

	Universidad del Valle Robot móvil para recorrer un camino con obstáculos	Rev.: 000
Título: **CASO DE USO** **Requerimiento funcional 5** El robot móvil debe ser capaz de evadir los obstáculos rodeandolos	**Documento :** **CUR-001**	Página : **1 de 1**

REVISIÓN HISTÓRICA			
Rev.	**Descripción del Cambio**	**Autor**	**Fecha**
001	Construcción de caso uso	Carlos Fernando Quintero, Jhonier Andrés Vargas, Joan Velasco	2021-11-21
002	Actualización	Carlos Fernando Quintero, Jhonier Andrés Vargas, Joan Velasco.	2022-03-16

INFORMACIÓN GENERAL	
Actores:	firmware.
Propósito:	El robot debe girar y avanzar para rodear un obstáculo que detecte
Resumen:	Se espera que a partir de los cambios que puedan existir en la trayectoria marcada y los obstáculos el robot cambie la velocidad en cada rueda para girar.
Tipo:	Real

Curso Normal de los Eventos
Respuesta del firmware
1. Se hace el llamado de las funciones "ObjetoAlFrente" y "ObjetoAlLado" para conocer la posición del obstáculo a evadir.

Si ninguna de las funciones retorna un 1, entonces:

Curso Normal de los Eventos
Respuesta del firmware
2. Se llama a la función girar derecha.

.

Si la función "ObjetoAlFrente" retorna un 1, entonces:

Curso Normal de los Eventos
Respuesta del firmware
3. Se llama la función Girar Derecha.

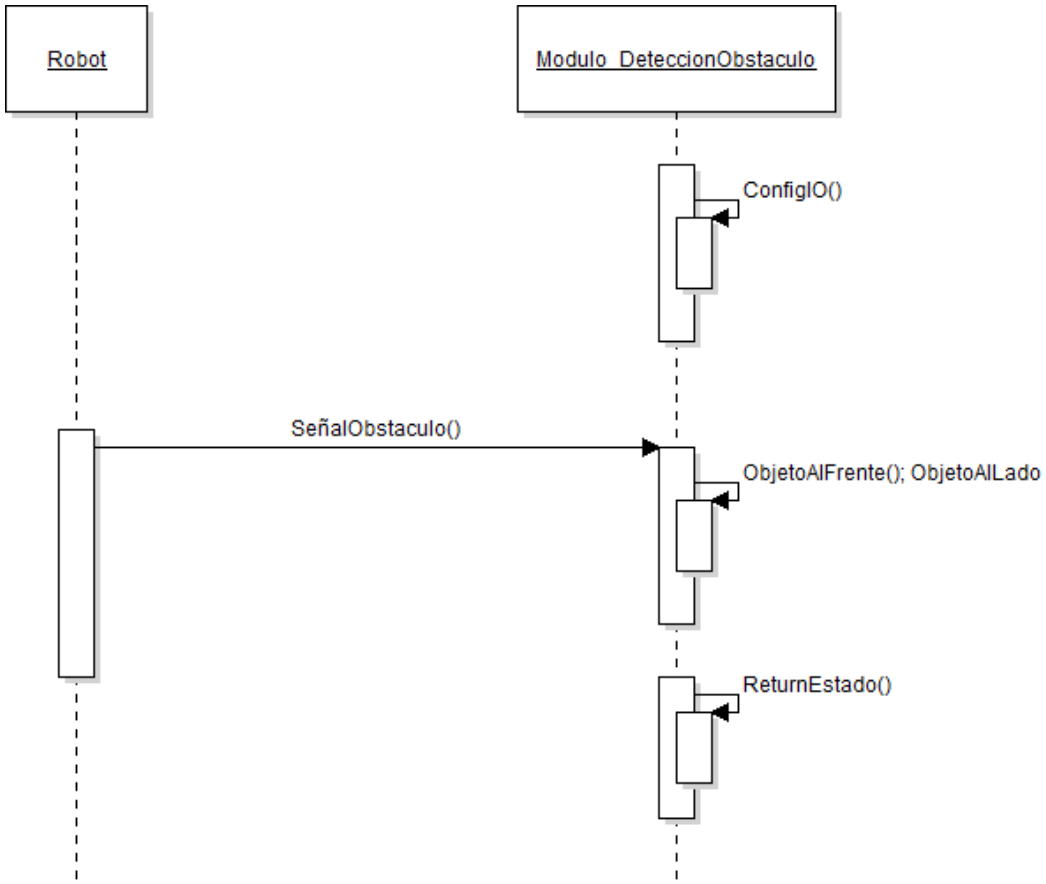
Si la función "ObjetoAlLado", entonces:

Curso Normal de los Eventos
Respuesta del firmware
4. Se llama la función Avanzar.

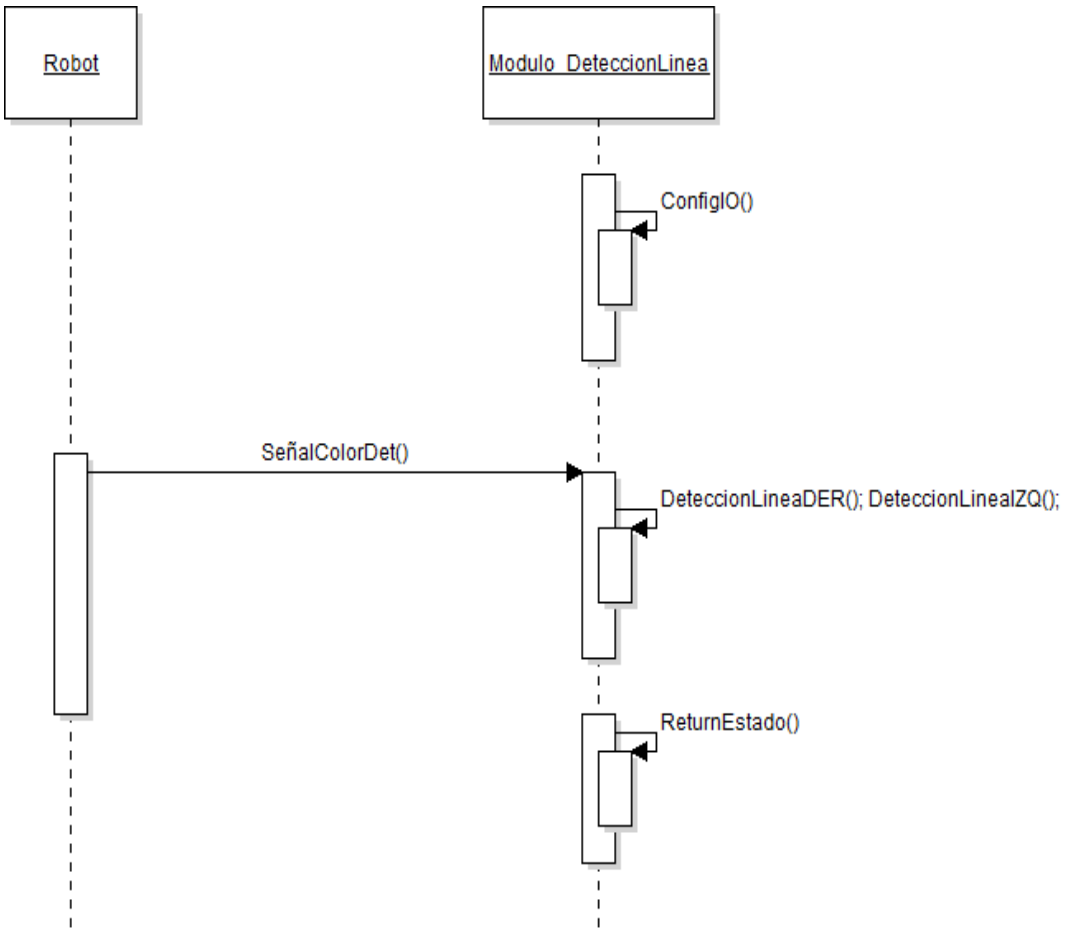
Si ambas funciones retornan un 1, entonces:

Curso Normal de los Eventos
Respuesta del firmware
5. Se llama la función girar izquierda.

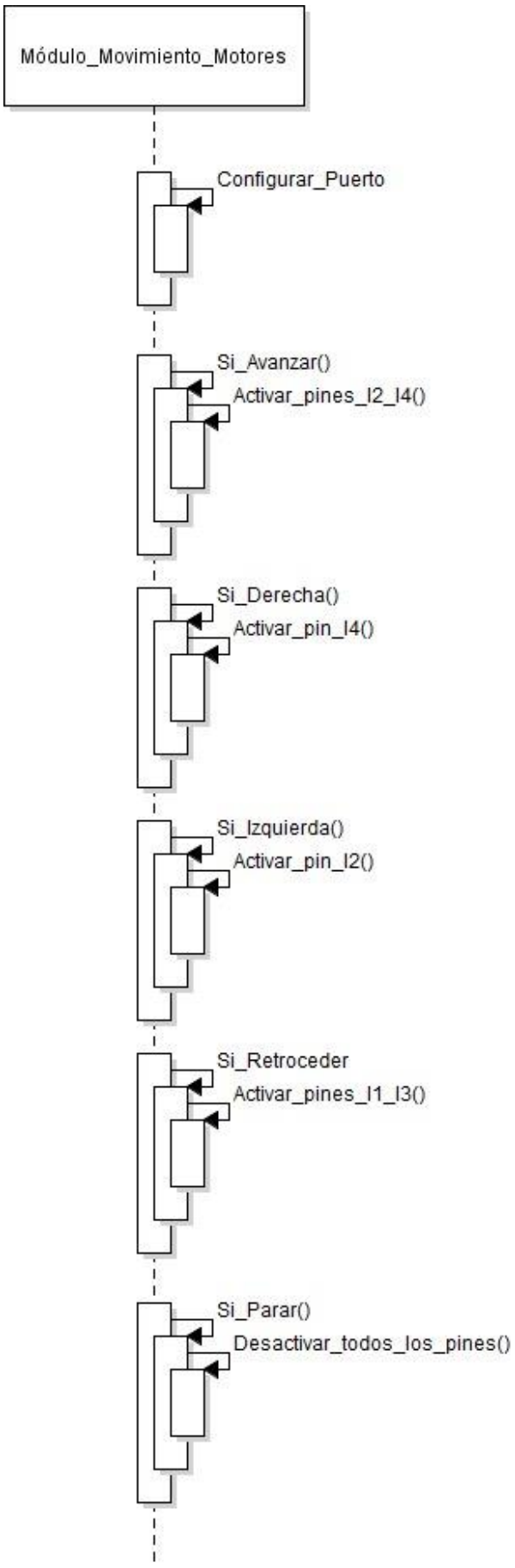
Anexo 3.5. Caso de uso para el Requerimiento funcional 5



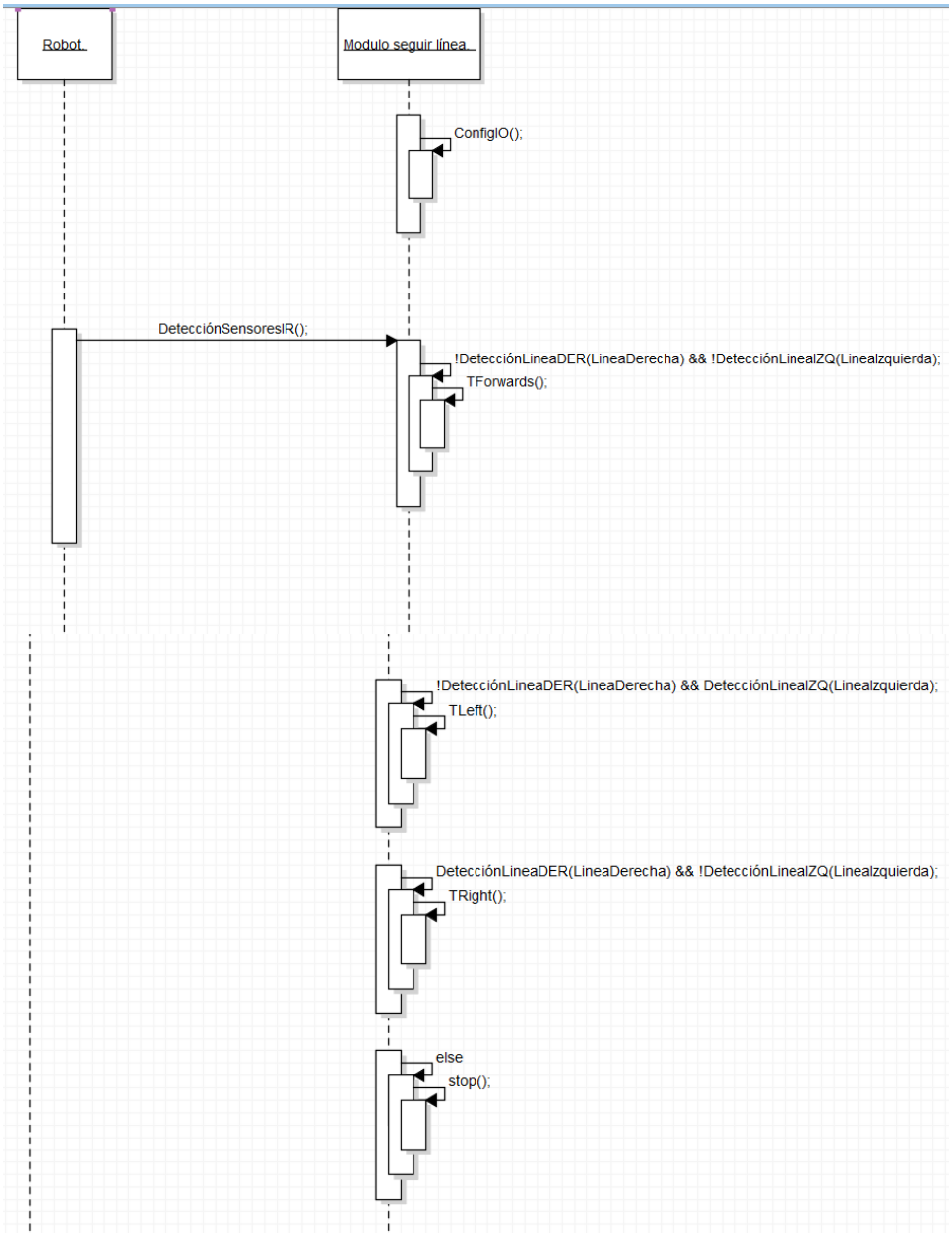
Anexo 4.1. Diagrama secuencial para el Requerimiento funcional 1



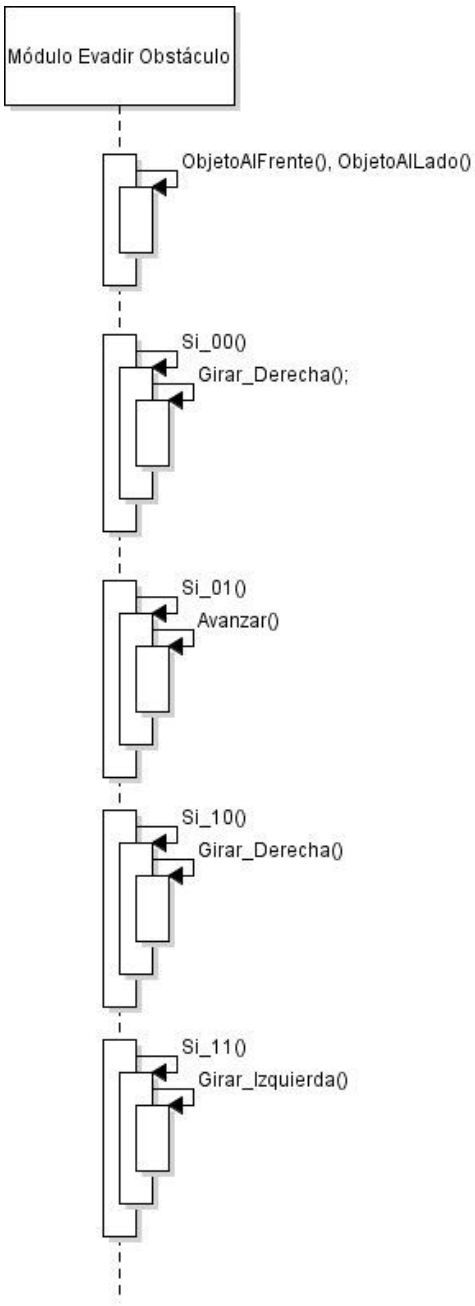
Anexo 4.2. Diagrama secuencial para el Requerimiento funcional 2



Anexo 4.3. Diagrama secuencial para el Requerimiento funcional 3



Anexo 4.4. Diagrama secuencial para el Requerimiento funcional 4



Anexo 4.5. Diagrama secuencial para el Requerimiento funcional 5

ENLACE DE VIDEOS Y PRUEBAS DEL SISTEMA

- Sistema de seguimiento de línea:
https://drive.google.com/file/d/10-Y8Vb6JneLfGJXaWTDSpyYozMRxWazJ/view?usp=sharing
- Sistema de seguimiento del contorno del obstáculo (Evasión de obstáculos):
https://drive.google.com/file/d/104xOeoMsnnEqzaBa_adcC0CzPCWFrlhA/view?usp=sharing
- Sistema de seguimiento de línea junto con evasión de obstáculos:
https://drive.google.com/file/d/10FYSkoYAHA2tWERyuqqnBcqd3jWUDDMp/view?usp=sharing