Universidad Autónoma Metropolitana Unidad Azcapotzalco
División de Ciencias Básicas e Ingeniería
Licenciatura en Ingeniería en Computación

Reporte del Proyecto Tecnológico

Simulación de tráfico en una intersección a partir de vehículos autónomos

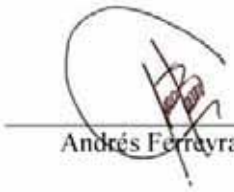
2014 Primavera
29/08/2014

Ramírez Olvera Efraín Jonathan
208332720

Andrés Ferreyra Ramírez
Profesor Titular
Departamento de Electrónica

Eduardo Rodríguez Martínez
Profesor Asociado
Departamento de Electrónica

Yo, Andrés Ferreyra Ramírez, declaro que aprobé el contenido del presente Reporte de Proyecto de Integración y doy mi autorización para su publicación en la Biblioteca Digital, así como en el Repositorio Institucional de UAM Azcapotzalco.



Andrés Ferreyra Ramírez

Yo, Eduardo Rodriguez Martinez, declaro que aprobé el contenido del presente Reporte de Proyecto de Integración y doy mi autorización para su publicación en la Biblioteca Digital, así como en el Repositorio Institucional de UAM Azcapotzalco.

Rodríguez Martínez E.

Eduardo Rodriguez Martinez

Yo, Efraín Jonathan Ramírez Olvera, doy mi autorización a la Coordinación de Servicios de Información de la Universidad Autónoma Metropolitana, Unidad Azcapotzalco, para publicar el presente documento en la Biblioteca Digital, así como en el Repositorio Institucional de UAM Azcapotzalco.

Efraín Jonathan Ramírez Olvera

Tabla de contenido

Índice de imágenes

1. Resumen

El presente documento es el resultado del desarrollo e implementación del proyecto “Simulación de tráfico en una intersección a partir de vehículos autónomos” en el cual se lleva acabo el desarrollo e implementación del protocolo de comunicación a través del cual vehículos autónomos pueden transitar en una intersección de forma ordenada y evitando colisionar entre ellos.

La simulación de tráfico se lleva a cabo en dos escenarios; en el primero de ellos, se tiene a un vehículo intersectando a uno o más vehículos (cadena) en una determinada sección del camino por lo que los vehículos se ven obligados a agruparse en una nueva cadena de vehículos; en el segundo escenario, se tiene una cadena de vehículos recorriendo una determinada ruta, en cierto punto, el camino se separa en dos rutas diferentes y solo un único elemento abandona la cadena para tomar una ruta diferente; por lo tanto, los tres robots que continúan en la misma ruta de ser posible deberán de reagruparse en una sola cadena.

Cuando se crea una cadena, el vehículo que se encuentra al inicio (H) se encarga de dirigir a los demás por lo que el vehículo que le precede (L_1) debe comenzar a seguirlo y administrar su velocidad de acuerdo a la distancia que existe entre ellos con el fin de no impactar; si a este segundo vehículo le precede un tercer vehículo (L_2), este tercer vehículo deberá comenzar a seguir al segundo vehículo (L_1) y así sucesivamente.

En ambos escenarios todos y cada uno de los vehículos llevan un registro de su desplazamiento y recorren las rutas establecidas a una velocidad determinada excepto cuando un vehículo L_j se encuentra siguiendo al vehículo L_{j+1}, en cuyo caso, L_j puede aumentar su velocidad o incluso detenerse por completo dependiendo del avance del vehículo L_{j+1}.

Los caminos están formados por un solo carril que cuenta con una dirección única de navegación. En el camino se hace uso de un tipo de señalización que indica a los robots cuando estos se encuentren próximos a una intersección.

2. Introducción

De acuerdo al Instituto Mexicano del Transporte (IMT) [1], uno de los principales problemas que aqueja a la sociedad mexicana es el constante incremento en los niveles de congestionamiento vial. Esta situación no solo ha provocado fuertes pérdidas económicas en el sector productivo, debido al tiempo perdido por los viajeros a causa de las demoras en el flujo vial, también ha causado el deterioro cada vez más visible a nuestro medio ambiente.

Dentro del reporte técnico ISSN 0188-7297 [2], presentado a la Secretaría de Comunicaciones y Transportes (SCT), el IMT se pronuncia por la adopción de tecnologías conocidas como sistemas inteligentes de transporte (SIT o ITS, por sus siglas en inglés) para aliviar los problemas de congestionamientos, mejorar las condiciones de seguridad de operación del autotransporte terrestre, así como hacer más eficiente la infraestructura vial o la operación de los vehículos comerciales.

La Asociación Mundial de Caminos (WRA por sus siglas en inglés) define el término SIT como un término genérico para la aplicación integrada de tecnologías de comunicación, de control y de procesamiento de la información a sistemas de transporte terrestre [3]. Los diferentes elementos que conforman un SIT son: vehículos, infraestructura, y usuarios (o conductores). En su conjunto, estos elementos forman una red de comunicación, en donde el manejo y distribución de información adquieren un rol central.

Los beneficios que ofrecen los SIT son el mejoramiento en la operación y seguridad de los sistemas de transporte al proveer rutas más eficientes a los viajeros, agilizar los procesos de cobro de cuotas o inspección de vehículos comerciales, proveer mecanismos de advertencia de colisiones o salida de caminos, e incluso la posibilidad de equipar a los vehículos con tecnología capaz de tomar el control en caso necesario. Este último punto ha sido particularmente atractivo para las grandes empresas automovilísticas, quienes han creado vehículos con la habilidad de buscar un lugar libre para estacionarse, dentro de un estacionamiento determinado, y estacionarse sin intervención del conductor.

3. Antecedentes y justificación

Una de las principales áreas de investigación dentro de los SIT es la forma en que la información se transmite y comparte entre sus diferentes elementos. Actualmente existen dos vertientes que estudian la comunicación en los SIT: la primera estudia la interacción vehículo-infraestructura, donde la comunicación se da entre bases recolectoras de información, ubicadas a lo largo de la red carretera, y los vehículos, mientras que la segunda atiende la interacción vehículo-vehículo, donde el flujo de información asemeja más a una red de computadoras. La comunicación vehículo-infraestructura ha demostrado mayor dificultad en adaptarse a la infraestructura carretera existente debido al costo que implica su instalación y puesta en marcha. Por otro lado, en la comunicación vehículo-vehículo existen diversos protocolos de comunicación inalámbrica (e.g. Wi-Fi, WAVE, WiMAX, Bluetooth, IRA y ZigBee), los cuales son modificados por las diversas empresas automovilísticas para crear una solución acorde a sus necesidades. Esto genera incompatibilidad de comunicación entre vehículos de diferentes fabricantes cuando se encuentran interactuando dentro de la infraestructura carretera.

El objetivo principal de este proyecto terminal es el diseño de un protocolo de comunicación vehículo-vehículo que sea flexible y fácil de implementar. Dado que el manejo y distribución de la información en una comunicación vehículo-vehículo asemeja una red de computadoras, el presente proyecto terminal propone que el protocolo de comunicación se base en el protocolo TCP/IP para las capaz de enlace de datos, de red y de transporte del modelo de interconexión de sistemas abiertos. El protocolo de comunicación propuesto, en lo subsecuente referido como protocolo de control de transmisión para ambientes vehiculares (PCTAV), además de modelar la red vehicular, será independiente del sistema operativo y arquitectura que cada fabricante maneje para la implementación de la unidad de control de motor y el sistema de entretenimiento a bordo del automóvil.

Como se desea que PCTAV forme parte de un SIT, este debe demostrar su funcionalidad en situaciones que requieran el flujo de información entre distintos vehículos, es por ello que se eligió simular mediante mini robots autónomos el proceso de incorporación y separación de un vehículo a una vialidad transitada. De esta forma, el vehículo (mini robot) funcionará como unidad recolectora de información a través de sus diferentes sensores, y tomará decisiones basándose no solo en información local, sino a través de la comunicación con otros vehículos.

4. Objetivo

Simular el tráfico vehicular en una intersección o separación de dos caminos, cada uno de ellos con un único carril y una sola dirección, haciendo uso de robots autónomos capaces de comunicarse entre sí.

4.1. Objetivos específicos

- ✓ Diseño del protocolo de comunicación a través del cual se comunican los robots.
- ✓ Control del avance de los robots para que estos no impacten.
- ✓ Implementación del protocolo de intersección.
- ✓ Implementación del protocolo de separación.

5. Marco teórico

Hoy en día grandes empresas automovilísticas han comenzado a dirigir su atención a la creación de carros con la capacidad de estacionarse por sí solos sin la necesidad de que el conductor intervenga en un determinado momento, un ejemplo de esto es la empresa Volvo que ha logrado desarrollar un carro que aparte de ser capaz de estacionarse solo también tiene la capacidad de buscar dentro de un estacionamiento determinado un lugar libre para estacionarse, todo esto sin la necesidad de tener un conductor dentro.

La idea de prescindir del conductor no es una tarea fácil, uno de los mayores retos que esto implica es la sustitución de uno de los sentidos más importantes del ser humano, la visión; Un medio a través del cual se puede intentar remediar esta problemática es haciendo uso de sensores. Pero el uso de estos presenta una gran limitante puesto que no bastaría con solo proveer a todos los vehículos de sensores para que puedan detectarse entre sino que también necesitarían de retroalimentación entre ellos.

Para lograr que un vehículo autónomo sea capaz de transitar a través de diversos caminos, necesitaría estar preparado para llevar a cabo las distintas tareas que debe realizar un conductor como lo es el reconocer y respetar las señales de tránsito, rebasar o cambiar de carril sin afectar a terceros, en resumen conducir de forma prudente. Debido a la amplitud que conlleva el control y manejo de tráfico entre vehículos autónomos, éste proyecto se limitará al análisis y simulación del proceso de incorporación o salida de una vía transitada.

6. Los vehículos y la Pista

6.1 Los vehículos

Figura 1. Robot mOway.

Los robots mOway son herramientas educativas que cuentan con suficientes sensores e indicadores (véase apéndice B) que les permiten interactuar con el entorno, así como también cuentan con un modulo de radio frecuencia a través del cual pueden realizar trabajo colaborativo. Estas características les permiten funcionar como vehículos autónomos.

Figura 2. Equipo de robots mOway.

Para este proyecto se eligieron cuatro de seis mOway (el mOway 5 fue descartado por su incompatibilidad en base al análisis de alcance de los sensores de línea (véase apéndice D); el mOway 2 fue descartado por la diferente velocidad de avance respecto a los demás (mas adelante se trata este tema)). Como se puede observar en la figura 2 en la parte trasera de los robots se les coloco un fragmento de hoja blanca ya que esta amplifica el alcance de los sensores de obstáculos. El color de la hoja no produce un efecto relevante sobre los sensores de objetos por lo que bien pudo ser de otro color, es el material del que esta conformada la hoja lo que altera considerablemente las lecturas de los sensores, es decir, entre mas brillante es el material (por ejemplo, la cinta de aislar) menor es el alcance del sensor; por el contrario un material opaco como lo es el cartón aumenta el alcance del sensor. Es importante mencionar que el fondo que precede al objeto que se está detectando también juega un papel importante, ya que si el fondo está a una distancia no mayor a 15 cm del objeto, puede incrementar las lecturas por parte de los sensores de objetos; entre más cerca se encuentre el fondo al objeto sensado y entre más claro sea el color de este, mayor es el incremento de las lecturas llegando a incrementar en hasta aproximadamente 6%.

6.2 La pista

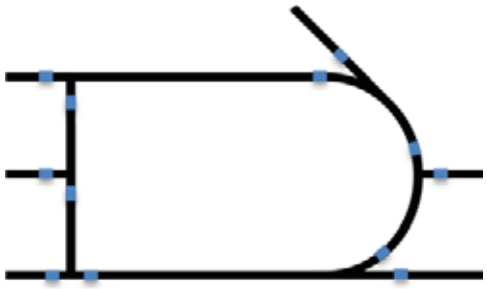


Figura 3. Pista (primera versión).

En un principio los caminos fueron trazados con cinta negra y las señales de intersección con cinta azul como se muestra en la figura 3, sin embargo, los sensores de línea de los robots cuando leen un color, como por ejemplo el blanco con un rango de lectura de 0 a 16, y después leen un color negro con un rango entre 200 y 255(que es el valor máximo), sufren un incremento de 16 a 200 que no es instantáneo ya que para llegar al valor de 200 pasan por todos los valores intermedios entre 16 y 200; el color azul es medido en un rango aproximado de 28 a 120, por lo que cada vez que el sensor pasaba del color blanco al negro y viceversa, se reconocía al color azul sin estar presente, por lo tanto para solucionar este problema los caminos fueron trazados con cinta azul y las señalizaciones con cinta negra. De esta forma las lecturas de los sensores de línea se mantienen entre 10 y 120 cuando el mOway se encuentra recorriendo la pista; cuando las lecturas del sensor de línea se encuentran entre 10 y 255 esto indica que el mOway se encuentra en una intersección.

Las esquinas de la pista fueron modificadas ya que si estas forman un ángulo cercano a 90 grados o menos, cuando un mOway se encuentra siguiendo a otro al atravesar por la intersección, el mOway al que va siguiendo sale del campo de sensado como se muestra en la figura 4, por lo cual las esquinas fueron redondeadas.

Figura 4. Cadena de robots mOway recorriendo esquina de 90°.

La versión final de la pista se muestra en la figura 5. (Por uniformidad todas las líneas inclinadas forman un ángulo de 45°)

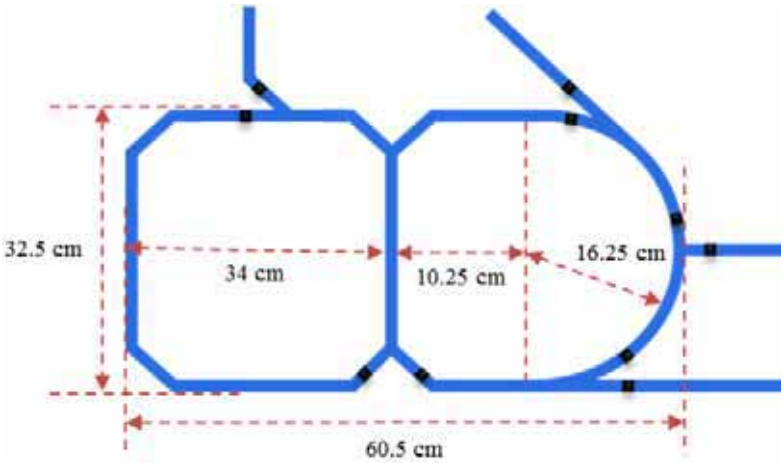


Figura 5. Pista (versión final).

El diseño de la versión final de la pista permite dar lugar a las siguientes secciones donde se llevaron acabo las intersecciones y/o separaciones:

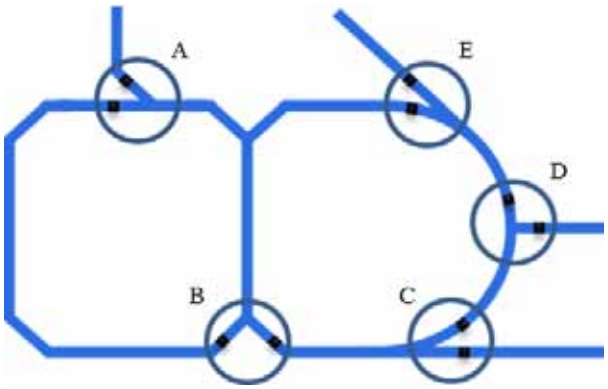


Figura 6. Secciones de intersección y/o separación.

Las señalizaciones de intersección fueron iniciadas a ciertas distancias antes del punto donde se unen los caminos, de tal forma que los mOway tuvieran suficiente tiempo para frenar y evitar colisionar, así como también dejar el suficiente espacio para permitir el libre tránsito del vehículo contrario. En las intersecciones A, B, D las señalizaciones fueron colocadas a aproximadamente 6 cm; En el caso de las intersecciones C, E fueron colocadas a una distancia aproximada de 8 cm.

7. El avance de los vehículos mOway

El rango de velocidad de avance que se le puede establecer a un mOway varía entre cero y 100, pero debido a que existe una cierta diferencia entre los motores de cada uno de los mOway, a pesar de que se les establezca la misma velocidad, el avance de estos es irregular. Debido a lo reducido de la pista se optó por establecer que los mOway avanzaran a la velocidad mínima de 10 cuando estos no se encuentran siguiendo a otro vehículo; en el caso de que sí se encuentren siguiendo a otro vehículo su velocidad puede aumentar hasta 80 y disminuir hasta 0 (detenerse por completo).

Como ya se menciono anteriormente las intersecciones o separaciones sólo se llevaron a cabo de dos caminos a uno o de un camino a dos respectivamente, por lo que la ruta se guarda en un arreglo unidimensional el cual sólo puede contener ceros que denotan vuelta a la izquierda o unos que denotan vuelta a la derecha.

En todo momento el avance de los vehículos está regido por los sensores tanto de seguimiento de línea como de reconocimiento de obstáculos. Los sensores utilizan la reflexión de luz infrarroja que permite no sólo detectar contrastes fuertes sino también discernir entre diferentes tonos. Cuando se detecta una superficie clara, la superficie blanca hace que toda la luz infrarroja se refleje por otro lado cuando se detecta unas superficie de color sólo una parte de la luz emitida se refleja, de esta manera es como se realiza la identificación de colores.

7.1 El seguimiento de línea

El seguimiento de línea se realiza a través de los dos sensores de línea, esto es, cuando el mOway debe avanzar por la izquierda el sensor izquierdo debe mantenerse en el color blanco y el sensor derecho debe mantenerse en el color azul como se muestra en la siguiente figura:

Figura 7. Seguimiento por la izquierda.

Para lograr que el mOway avanzara por la izquierda se establecieron las siguientes reglas:

- Si el sensor izquierdo esta sensando blanco y el sensor derecho esta sensando azul el mOway debe avanzar derecho.
- Si ambos sensores están sensando blanco el mOway debe girar a la derecha hasta que el sensor derecho cense el color azul.
- Si ambos sensores están sensando azul, o si el sensor izquierdo esta sensando azul y el sensor derecho esta sensando blanco, el mOway debe girar a la izquierda hasta que el sensor izquierdo cense el color blanco.

Figura 8. Ambos sensores de línea sensando blanco.

Figura 9. Ambos sensores de línea sensando azul.

El rango numérico que se definió para cada color se realizo en base al análisis del alcance de los sensores (véase apéndice D) quedando de la siguiente manera:

Rango numérico establecido para censo de colores

Color	Sensor izquierdo	Sensor derecho
Blanco	0 – 14	0 – 16
Azul	15 – 120	17 – 140
Negro	121 – 255	141 – 255

El caso del avance por la derecha es similar al avance por la izquierda.

7.2 La obtención de la velocidad

La obtención de la velocidad sólo tiene lugar cuando un mOway se encuentra siguiendo a otro y como ya se menciono anteriormente la velocidad se obtiene a partir de los sensores de obstáculos, esto es, entre más lejos detecta el mOway L_j al mOway L_{j+1}, mas aumenta la velocidad del mOway L_j; por el contrario entre mas cerca se cense el mOway L_{j+1}, el mOway L_j reduce su velocidad incluso hasta detenerse por completo para evitar colisionar.

Los mOway cuentan con 4 sensores de obstáculo, por lo que para ponerlos en función de la velocidad se obtiene el promedio sólo de los sensores que se encuentran activos. El rango de cada sensor es de 0 a 255, siendo 255 la lectura obtenida cuando el objeto se encuentra relativamente cerca. Si el sensor esta justo enfrente del objeto y el objeto no permite el paso de luz a pesar de que el sensor debería entregar una lectura de 255 entrega una lectura cercana a 0. En la siguiente grafica se muestra lo anterior:

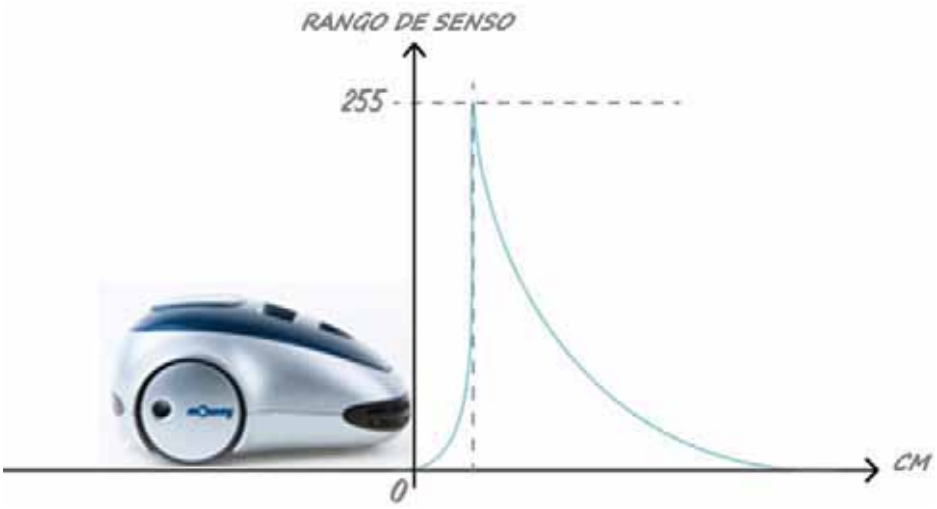


Figura 10. Grafica de sensado de obstáculos.

Algunos de los sensores de objetos son muy sensibles al entorno por lo que a pesar de no existir un objeto pueden entregar lecturas denotando lo contrario, estas lecturas erróneas tienen un rango aproximado de cero a tres por lo cual se estableció que las lecturas menores o iguales a 5 sean omitidas. Para asegurar que los mOway guarden una cierta distancia entre ellos en todo momento a partir de la lectura 210 la velocidad debe ser cero, por lo tanto un sensor de obstáculos se considera activo cuando este entrega lecturas intermedias entre 5 y 210.

8. La intersección

En el momento en que los sensores de línea reconocen la señalización de intersección (cinta negra) una variable que funciona como temporizador es activada, esto sucede pasándola de cero a 120; la variable se va reduciendo de uno en uno y el tiempo que tarda en llegar a cero es el tiempo aproximado que tarda un mOway en atravesar la intersección, este lapso de tiempo se reconoce como tiempo de intersección.

Cuando el mOway se encuentra en tiempo de intersección este debe abrir el canal de comunicaciones e inmediatamente comenzar a enviar así como también de recibir mensajes para averiguar si hay otro mOway en la intersección, de no ser así el mOway no necesita detenerse por lo que sigue avanzando. El mensaje que se envía durante el tiempo de intersección contiene el valor de la variable que funciona como temporizador del tiempo de intersección ya que este valor denota la distancia a la que el mOway se encuentra de la intersección, esto es, si un mOway envía un valor aproximado a 120 entonces el mOway acaba de entrar en la intersección; por el contrario si el mOway envía un valor cercano a cero entonces el mOway esta por salir de la intersección.

En base a lo anterior si un mOway A envía un valor de 80 y recibe el valor 120 de un mOway B, el mOway A debe dejar de enviar mensajes y cerrar el canal de comunicación puesto que llego primero a la intersección y por lo tanto tiene la preferencia de paso; en el caso del mOway B este debe detenerse inmediatamente y esperar a que el mOway A recorra la intersección, durante este tiempo de espera también la variable que funciona como temporizador del tiempo de intersección es detenida, a pesar de esto el mOway B debe seguir enviando mensajes para averiguar si existe un tercer mOway C siguiendo al mOway A y de ser el caso volver a repetir el proceso que realizo con el mOway A, de no suceder lo anterior en cuanto el tiempo de espera termina el tiempo de intersección continua disminuyéndose y el mOway B debe comenzar a seguir al mOway A.

En el caso en que dos mOway lleguen al mismo tiempo a la intersección, el mOway que transita por el camino de la derecha tiene la preferencia de paso.

En el siguiente diagrama de flujo se explica el algoritmo usado para la intersección:

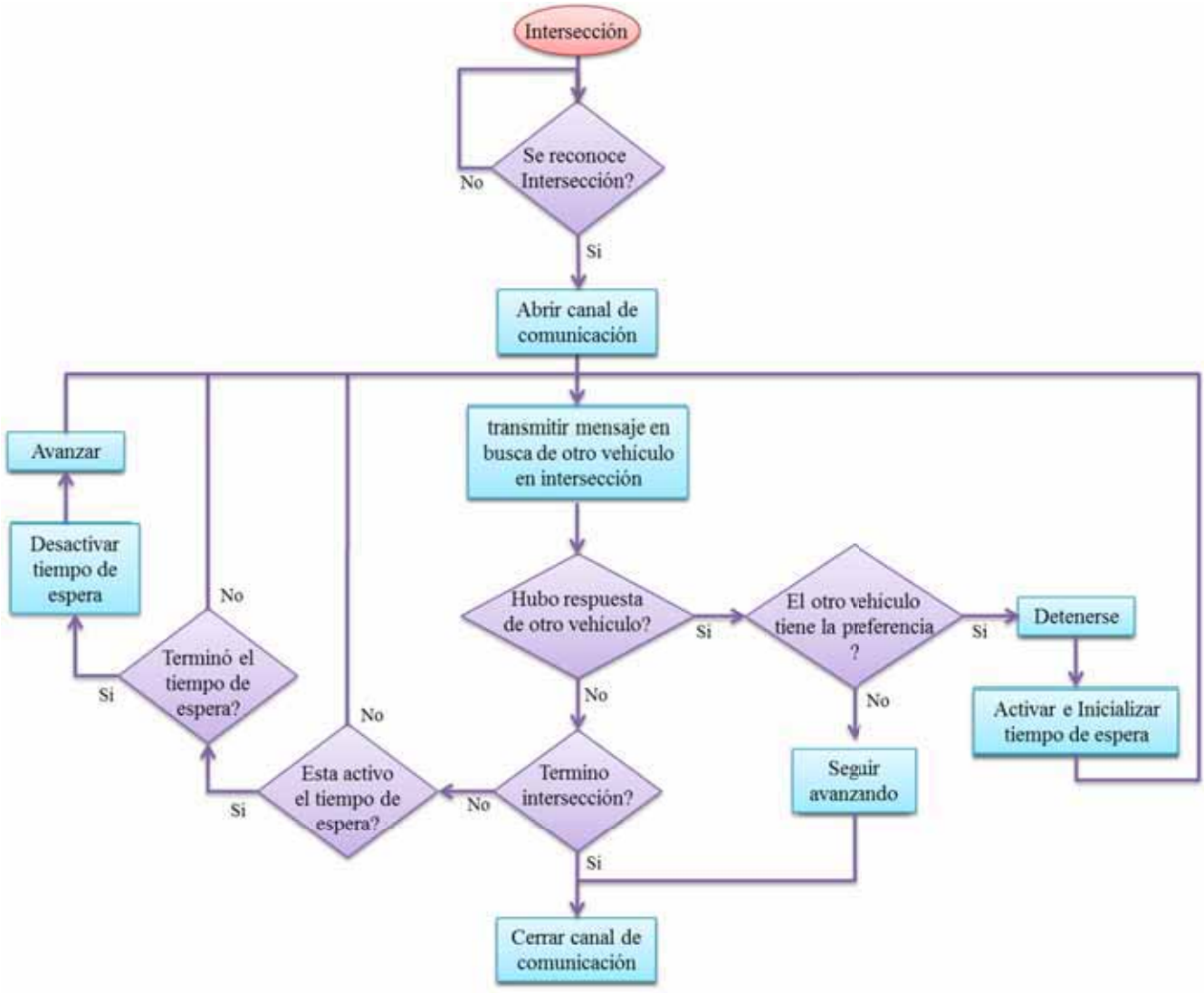


Diagrama de flujo 1. Algoritmo de Intersección

9. La separación

Como ya se mencionó en todo momento los mOway se encuentran pendientes de las lecturas que reciben por parte de los sensores de obstáculos, por lo tanto cuando un mOway A se encuentra siguiendo a un mOway B y el mOway B toma una trayectoria distinta, en el momento en que el mOway A deja de sensar al mOway B, el mOway A inmediatamente deja de obtener la velocidad en base a los sensores de obstáculos y comienza a avanzar a la velocidad constante establecida.

Figura 11. Separación de cadena una cadena.

Cuando se tiene una cadena de tres mOway y el mOway intermedio abandona la cadena, si el último mOway en un determinado momento alcanza al mOway que se encontraba a la cabeza de la cadena, en el instante en que comienza a sensarlo debe dejar de avanzar a velocidad constante y comenzar a obtener la velocidad en base a los sensores de obstáculos formando una nueva cadena.

En el siguiente diagrama de flujo se explica el algoritmo usado para la separación:

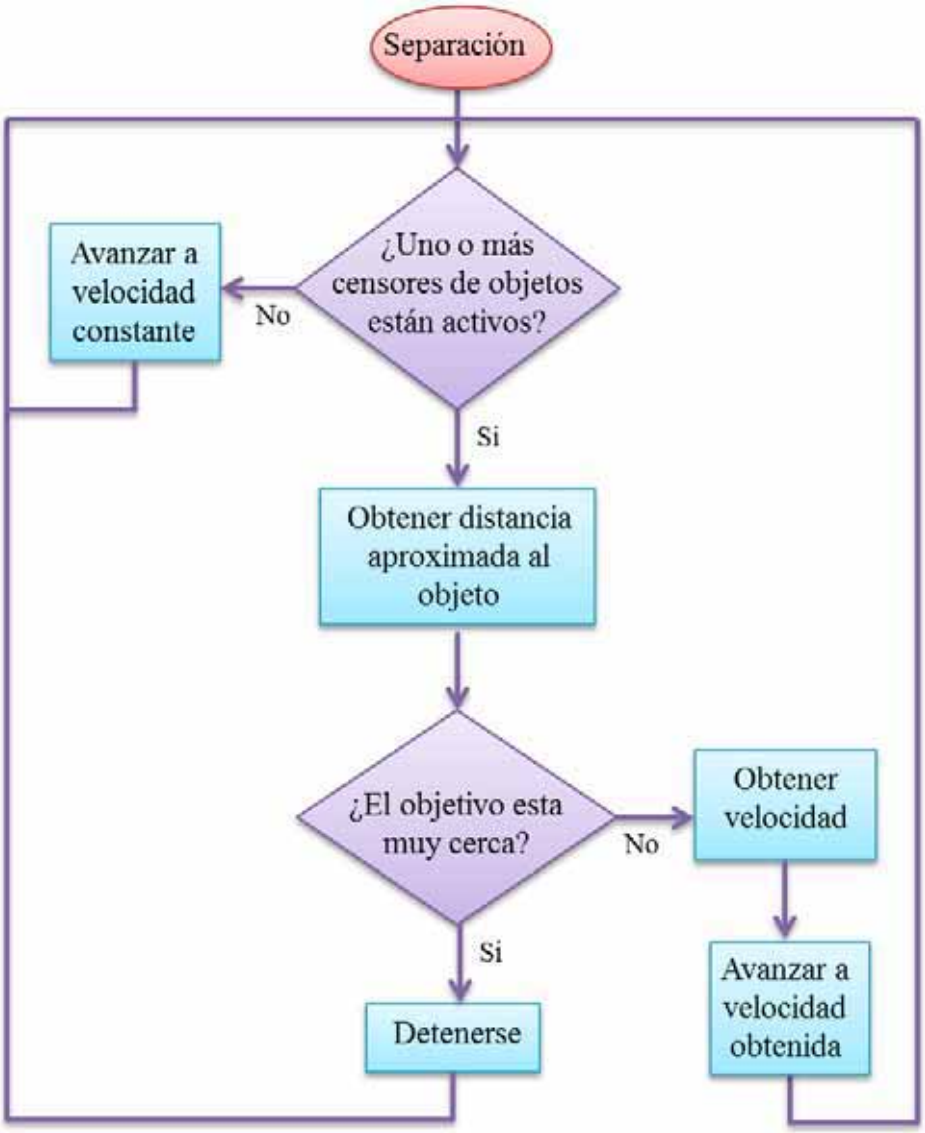


Diagrama de flujo 2. Algoritmo de separación

10. Conclusiones

Debido a la poca similitud en el alcance de los sensores de obstáculos y sensores de línea, así como también de los motores (aunque se les especifique la misma velocidad en un determinado tiempo, el avance de estos es irregular) de cada mOway y a pesar de que el código fue desarrollado en base a las características de los cuatro mOways seleccionados no se logró un óptimo resultado.

La calidad del proyecto puede ser mejorada si el código desarrollado se adecua a cada uno de los mOway, pero esto requiere de un análisis más extenso de las características así como alcances de cada uno de los mOway. Por ejemplo, en este proyecto cuando un vehículo se encuentra siguiendo a otro, este calcula la velocidad de avance obteniendo el promedio de las lecturas recibidas por parte de los sensores de objetos activos, pero este promedio contiene un cierto índice de error ya que el alcance de cada uno de los sensores varía considerablemente, esto es, aunque sólo los dos sensores centrales de un mOway se encuentren activos sensando el mismo objeto a la misma distancia estos entregan lecturas distintas, en base al análisis de alcance de sensores (apéndice D) se puede observar el caso anterior; si se coloca un objeto a 5.2 cm enfrente del mOway 1 el sensor central izquierdo entrega una lectura aproximada de 70 y el sensor central derecho de aproximadamente 5, el promedio obtenido es de 37.5 que es casi la mitad de la lectura del sensor central izquierdo.

11. Bibliografía

[1] Eric Moreno Quintero, “Reducción de congestión vehicular y los principios de Wardrop”, NOTAS del Instituto Mexicano del Transporte, vol. 110, no. 2, Febrero 2008.

[2] Jorge A. Acha Daza, Juan C. Espinosa Rescala, “Hacia una arquitectura nacional para los sistemas inteligentes de transporte”, Reporte técnico ISSN 0188-7297, Instituto Mexicano del Transporte, 2004.

[3] Manual en línea: “ITS Handbook”, http://road-network-operations.piarc.org/.

[4] Misael Romero Delgado, “Reporte de verano científico”.

A. Manual de Usuario

A.1. Instalación del programa mOwayWorld

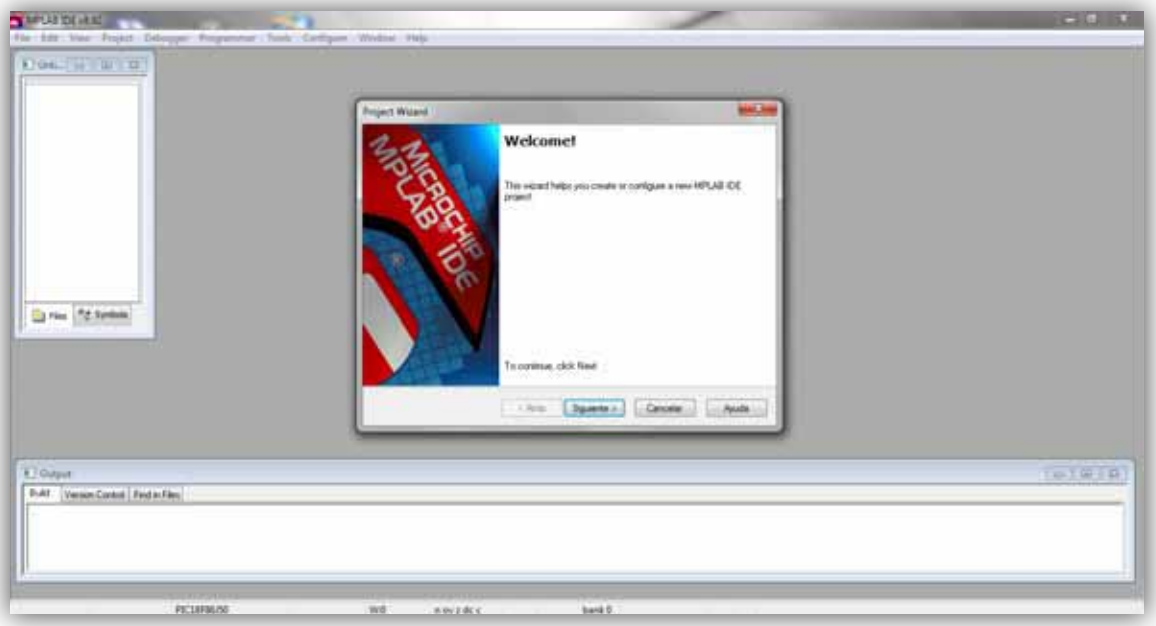
La instalación de mOwayPack se puede llevar acabo por medio del “disco de instalación” incluido en el kit o desde la pagina web principal, en este ultimo caso solo basta con dirigirse a la sección de descargas y descargar el instalador, cabe mencionar que el software solo esta disponible para el sistema operativo Windows y Linux.

A.2. Instalación del programa MPLAB C18

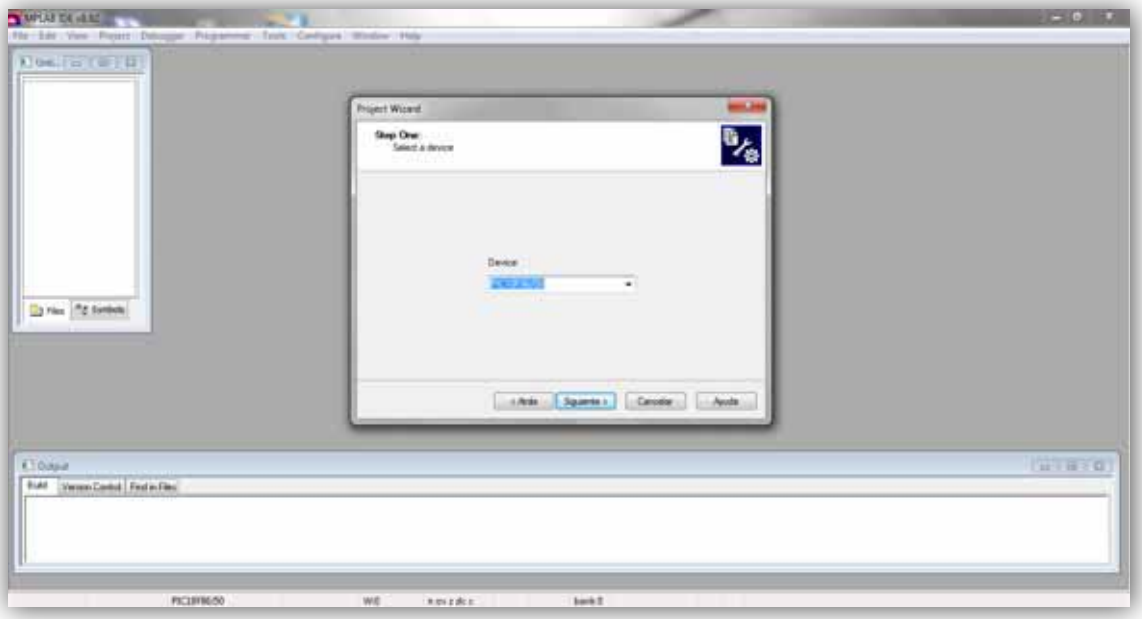
C18 de MPLAB, es un compilador que sirve para la familia de microcontroladores PIC18. Este programa se puede descargar gratuitamente desde la página www.microchip.com/c18. Una vez en la página se selecciona el archivo instalador “MPLAB C for PIC18 V3.46 in LITE mode” (la versión puede variar).

A.3. Creación de un proyecto en MPLAB C18

Abrimos el programa MPLAB seleccionamos la opción “Project Wizard” que se encuentra en la barra de herramientas (Project>Project Wizard); y nos aparece la ventana siguiente.

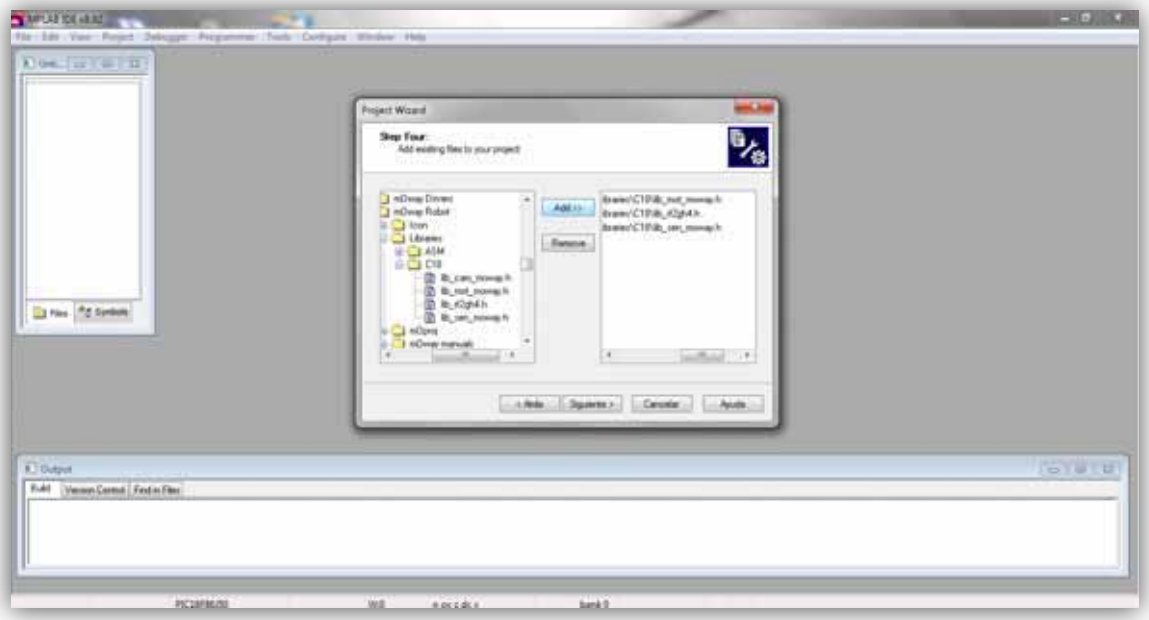


Damos clic en siguiente y a continuación nos pedirá que seleccionemos el PIC que utilizaremos para el proyecto, para ello hay que verificar cuál es el microcontrolador que estamos utilizando, (para este proyecto se usó el PIC18F86J50).



Seleccionamos el PIC y damos clic en siguiente. Ahora se nos solicitará el ámbito de trabajo que vamos a utilizar para programar, como se va a programar en MPLAB C18 pondremos en la pestaña de "Active Toolsuite" la opción de "Microchip C18 Toolsuite" y seleccionaremos la opción "MPLAB C18 C compiler (mcc18.exe) V3.46".

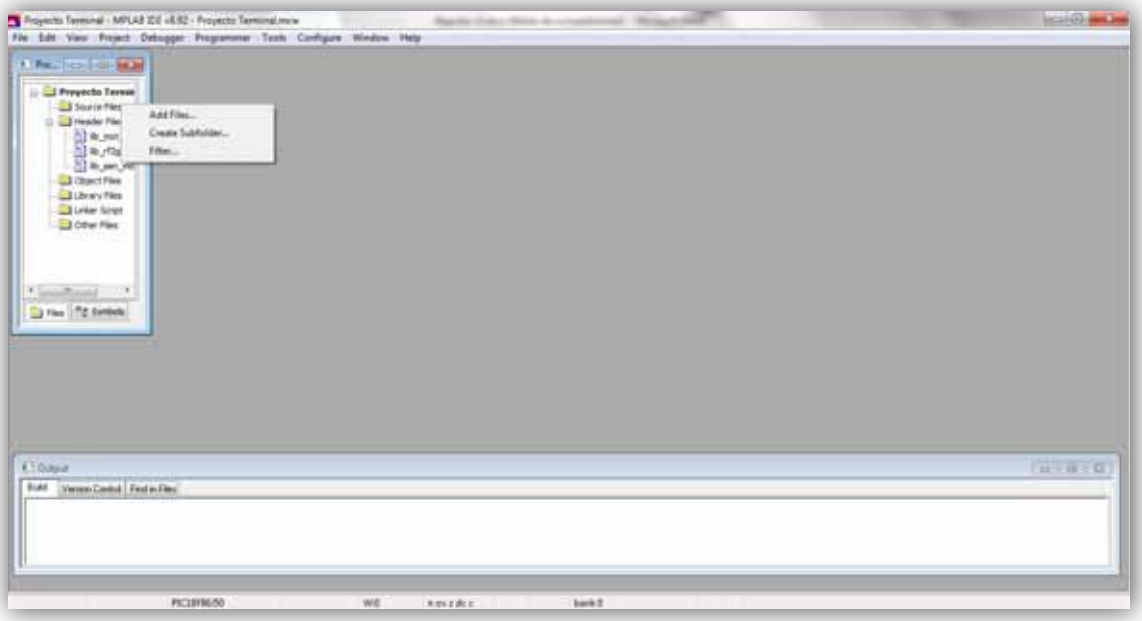
Después de dar clic en siguiente debemos introducir la dirección en la que queremos que se guarde nuestro proyecto así como el nombre para nuestro proyecto (Workspace). Volvemos a dar clic en siguiente y ahora nos piden que agreguemos las librerías que se van a utilizar en nuestro proyecto, para la utilización de los Moway en el lenguaje C se utilizaran tres librerias, la del manejo de los motores, la del manejo de los sensores y la del transceptor de radiofrecuencia, llamadas lib_mot_moway.h, lib_sen_moway.h y lib_rf2gh4.h respectivamente, las cuales se encuentran en los archivos donde se instaló el "MowayPack v3" (Archivos de programa > mOwayPack v3 > Moway Robot > Libraries > C18 > Librería), instalados desde el "CD de instalación" del kit "mOway" o desde la página principal de mOway.



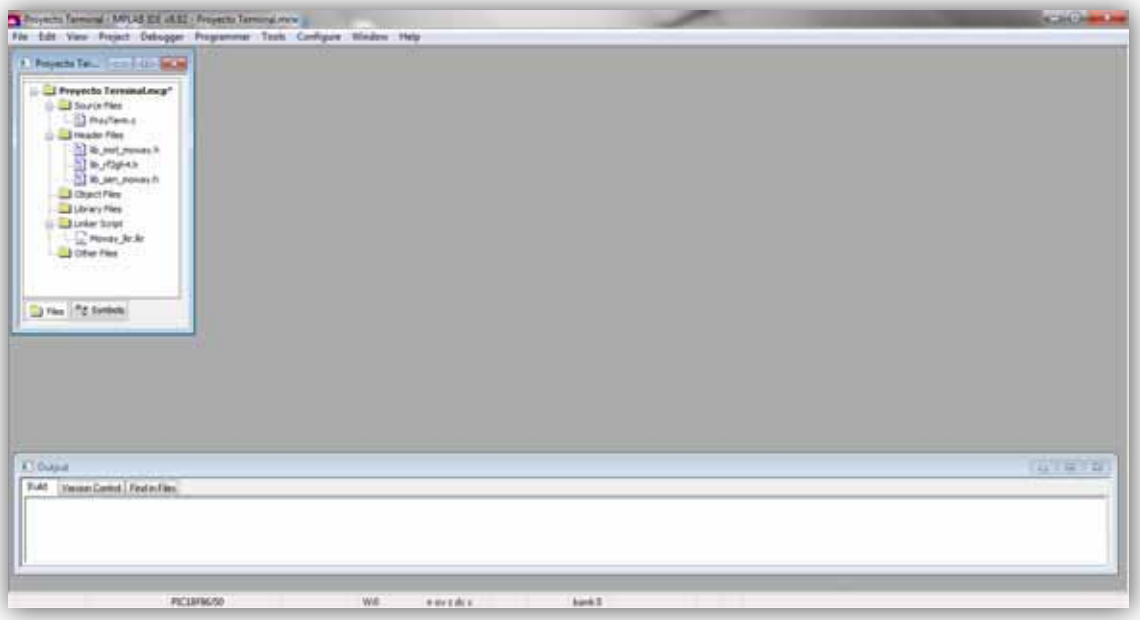
Una vez agregadas las librerías damos clic en aceptar y por último en finalizar. Para crear un documento principal y escribir nuestro código en C, tenemos que habilitar las ventanas de Project y Output, estas se habilitan en la barra de herramientas (View > Output/Project).

Ahora en la barra de herramientas damos clic sobre File y seleccionamos "Add new file to project…". En la misma carpeta donde creamos el proyecto, crearemos el documento de código en C el cual deberá tener una terminación de la siguiente manera "nombre_del_código.c".

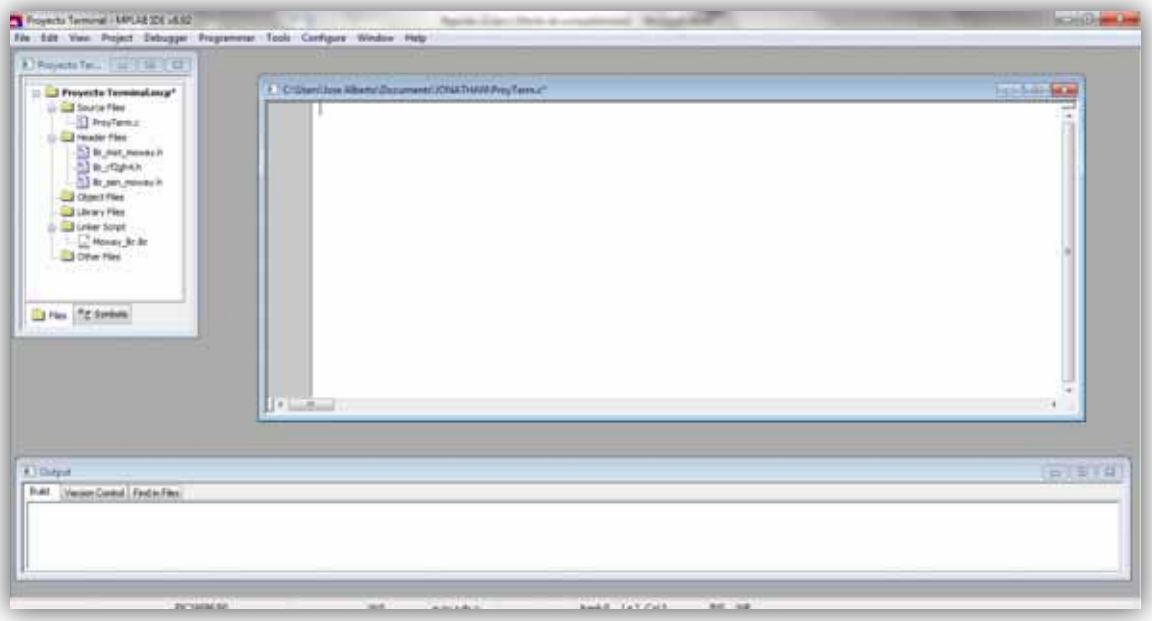
En la ventana de “proyect” damos clic con el botón secundario sobre “Source files”, seleccionamos la opción “Add Files…” y agregamos el archivo que creamos anteriormente con terminación “.c”.



Finalmente debemos agregar el linker que proporciona Moway cuando se instala MowayWorld al igual que las librerías. En la ventana de Project damos clic secundario sobre “Linker Script”, seleccionamos la opción “Add Files…” y agregamos el archivo mOway_lkr.lkr. que lo podemos encontrar en la carpeta Archivos de programa > mOwayPack v3 > Moway Robot > mOproj > C18 > mOway_first_project_C. También lo tenemos que agregar a la carpeta donde creamos nuestro proyecto.



Finalmente en la ventana “proyect” damos doble clic sobre el documento que creamos con terminación .c para comenzar a editarlo.



A.4. De la implementación del código al programado de un mOway

Para realizar la programación en lenguaje C, en el manual de usuario de mOway se incluye un encabezado principal (header) y además se anexa un pequeño código (main), este pequeño código permite desplazar al mOway hacia adelante hasta que este tope con un obstáculo y se vea obligado a realizar un giro de 180º y seguir hacia adelante hasta encontrar otro obstáculo y así sucesivamente, el código de encabezado y main se muestran enseguida:

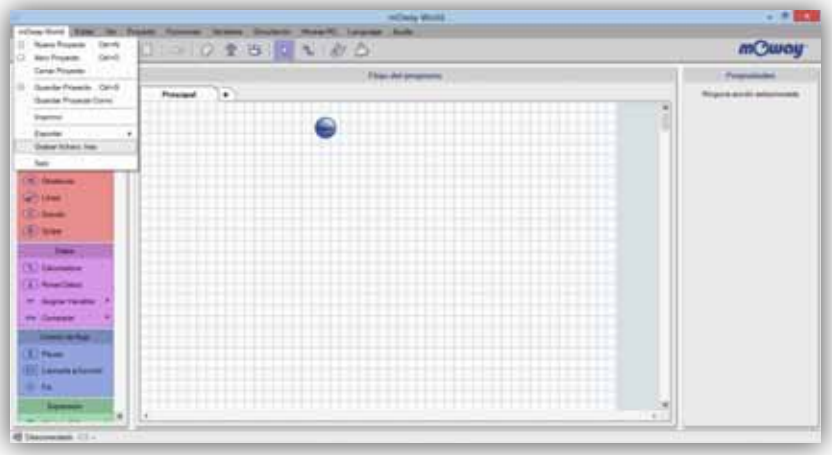
```
//**********************************[HEADER]**********************************//
#include "p18f86j50.h"//Moway microcontroller
#include "lib_mot_moway.h" //Engines library
#include "lib_sen_moway.h"  //Sensor library

// SALTO DEL BOOTLOADER
#define REMAPPED_RESET_VECTOR_ADDRESS                0x1000
  //Dirección de reset para saltar el bootloader.
#define REMAPPED_HIGH_INTERRUPT_VECTOR_ADDRESS 0x1008          //High
priority interruption adress for the correct bootloader jump
#define REMAPPED_LOW_INTERRUPT_VECTOR_ADDRESS  0x1018 //Low     priority
interruption adress for the correct bootloader jump
#pragma config XINST=OFF
```

```
void YourHighPriorityISRCode();    //Function that executes the needed code in case of the high priority interruption jumps.
void YourLowPriorityISRCode();     //Function that executes the needed code in case of the low priority interruption jumps.
//****************Remapeo del reset y las interrupciones de alta y baja prioridad*******************//
extern void _startup (void);
#pragma code REMAPPED_RESET_VECTOR = REMAPPED_RESET_VECTOR_ADDRESS
void _reset (void)
{ _asm goto _startup _endasm }
#pragma code

#pragma code REMAPPED_HIGH_INTERRUPT_VECTOR = REMAPPED_HIGH_INTERRUPT_VECTOR_ADDRESS
void Remapped_High_ISR (void) { _asm goto YourHighPriorityISRCode _endasm }

#pragma code REMAPPED_LOW_INTERRUPT_VECTOR = REMAPPED_LOW_INTERRUPT_VECTOR_ADDRESS
void Remapped_Low_ISR (void) { _asm goto YourLowPriorityISRCode _endasm }

#pragma code

#pragma interrupt YourHighPriorityISRCode
void YourHighPriorityISRCode()
{ //Código para interrupción de alta prioridad
}

#pragma interruptlow YourLowPriorityISRCode
void YourLowPriorityISRCode()
{ //Código para interrupción de baja prioridad
}


//********************************** [MAIN] **********************************//
void main(){
  //Configuración de sensores (utilizar solo si se necesita)
  SEN_CONFIG();
  //Configuración de motores (utilizar solo si se necesita)
  MOT_CONFIG();
```

```
//Función para que el LED verde parpadee
LED_TOP_GREEN_ON_OFF();
//Avanzar hacia adelante
MOT_STR(50, FWD, TIME, 0);
 while(1){
  //Hacer el chequeo del sensor izquierdo central
          if (SEN_OBS_DIG(OBS_CENTER_L))
  {
            LED_FRONT_ON();
   //Rotar el Moway
            MOT_ROT(20, FWD, CENTER, LEFT, ANGLE, 50) ;
            while(!MOT_END){};
           //Seguir hacia adelante
           MOT_STR(50, FWD, TIME, 0);
  }       else{
    LED_FRONT_OFF();
  }
 }
}
```

Ya que tenemos escrito el código en la interfaz de MPLAB C18 ya se puede realizar la construcción de nuestro archivo de lenguaje C a un archivo .Hex que es el que se utiliza para programar los microcontroladores, esto se realiza en la barra de herramientas en Project > Build All y nos debe generar el mensaje que se muestra en la parte inferior "BUILD SUCCEEDED".

Ahora nos dirigimos a mOwayword para gravar el archivo .hex al robot mOway por lo que no debemos olvidar conectarlo a la computadora). En la pestaña Moway World elegimos "Grabar fichero .hex" nos aparecerá una ventana en la cual sólo debemos ingresar la dirección de nuestro archivo .hex, damos aceptar y automáticamente comienza a grabarse el robot mOway.



A.5. Instrucciones básicas para la programación de un mOway

Instrucciones básicas para la programación de mOways

Sección	Instrucción	Descripción
Motor	Delay10KTCYx(i);	Retrasar ´i´ milisegundos el motor.
	MOT_STOP();	Detener completamente el motor.
	Sleep();	Mantener en modo “ahorro de energía” al microcontrolador.
Luces	LED_FRONT_ON();	Encender led frontal blanco.
	LED_FRONT_FF();	Apagar led frontal blanco.
	LED_FRONT_ON_OFF();	Mantener led frontal blanco parpadeando.
	LED_TOP_RED_ON();	Encender indicador rojo.
	LED_TOP_RED_OFF();	Apagar indicador rojo.

B. Características de un robot mOway

Figura 1. Características de un robot mOway

- PIC18F86J50 como microcontrolador principal.
- Grupo motor con control de trayectoria comandado por I2C.
- Sensores infrarrojos anticolisión.
- Sensor de intensidad de luz direccional.
- Sensores optorreflectivos infrarrojos para el suelo.
- Indicador luminoso superior bicolor.
- Led frontal.
- Leds rojos traseros.
- Sensor de temperatura.
- Acelerómetro de 3 ejes.
- Micrófono.
- Altavoz.
- Bus de expansión SPI/I2C para tarjetas electrónicas.
- Módulo de radiofrecuencia para comunicación inalámbrica.
- Batería LI-PO recargable por USB.
- Autonomía de 2 horas.

C. Grupo de sensores e indicadores del robot mOway

Este grupo consta de diferentes sensores e indicadores luminosos conectados al microprocesador de mOway con los que el robot interactúa con el mundo exterior.

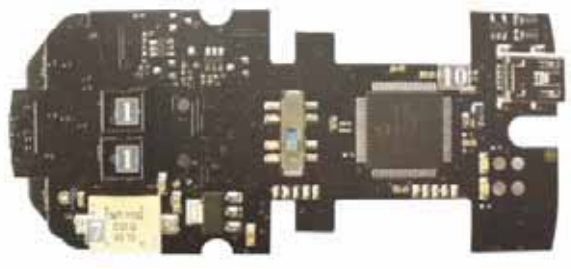


Figura 1. Sensores e indicadores.

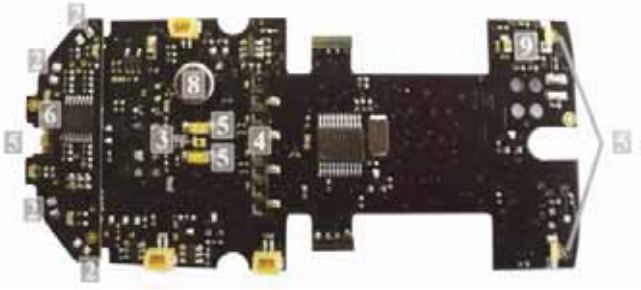


Figura 2. Sensores e indicadores.

[1] Sensores de línea (2).
[2] Sensores detectores de luz (4).
[3] Sensor de luz.
[4] Conector de expansión.
[5] 4 LEDs.
[6] Sensor de temperatura.
[7] Speaker.
[8] Micrófono.
[9] Acelerómetro.
[10] Nivel de Batería.

D. Alcance de los sensores de los robots mOway

Cada uno de los sensores que conforman a un mOway presentan diferentes alcances de cenzado y estos también varían de mOway a mOway por lo que es importante conocer el alcance de cada uno de ellos.

D.1. Sensores de línea

Cada mOway cuenta con dos sensores de línea montados en la parte inferior delantera del robot. Los sensores de línea fueron puestos a prueba en su modo analógico en tres colores diferentes, blanco (fondo de la pista), azul (color del camino) y negro (señal de intersección). El robot 5 fue descartado debido a que las lecturas de sus sensores presentan una gran diferencia respecto a los otros. Los resultados fueron los siguientes:

Sensores de línea

Robot	Blanco		Azul		Negro	
	Izquierdo	Derecho	Izquierdo	Derecho	Izquierdo	Derecho
mOway 1	13 - 14	14 - 16	15 - 100	17 - 100	101 - 235	101 - 240
mOway 2	13 - 14	14 - 16	15 - 85	17 - 120	86 - 240	121 - 245
mOway 3	12 - 14	12 - 14	15 - 80	15 - 70	81 - 240	71 - 235
mOway 4	11 - 12	12 - 13	13 - 30	14 - 55	31 - 235	56 - 235
mOway 5	15 - 22	13 - 14	23 - 130	15 - 95	131 - 245	96 - 240
mOway 6	13 - 14	15 – 16	15 - 85	17 - 115	86 - 245	116 - 240
Rango aproximado	0 - 14	0 – 16	15 - 100	17 - 120	100 - 240	120 - 245

D.2. Sensores de obstáculos

Cada robot mOway cuenta con 4 sensores de obstáculos en la parte frontal, el alcance de cada uno d ellos fue puesto a prueba en su modo analógico a distancias de 2.6 cm y 5.2 cm y en 5 diferentes ángulos como se muestra en la siguiente imagen.

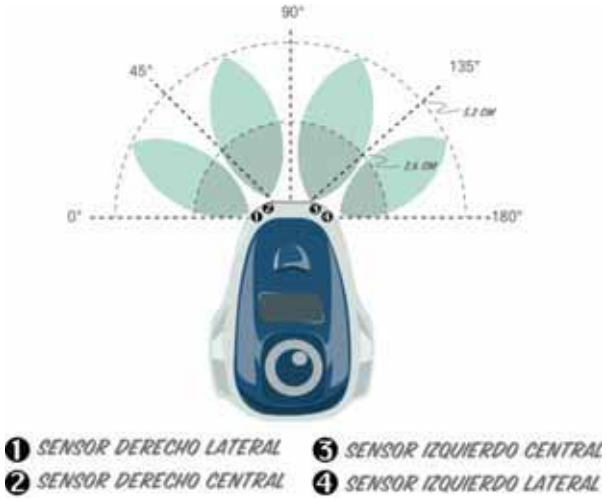


Figura 1. Grafica aproximada de alcance de sensores de objetos.

Censo a 0°

Robot	Sensores activos	2.6 cm. (1/4 de vuelta)	5.2 cm. (1/2 vuelta)
1	Izquierdo lateral	255	50 – 75
2	Izquierdo lateral	195 – 255	0 – 5
3	Izquierdo lateral	255	0 – 10
4	Izquierdo lateral	255	65 – 80
5	Izquierdo lateral	255	0 – 20
6	Izquierdo lateral	255	25 – 40

Censo a 45°

Robot	Sensores activos	2.6 cm. (1/4 de vuelta)	5.2 cm. (1/2 vuelta)
1	Izquierdo lateral	255	120 – 130
	Izquierdo central	255	45 – 60
2	Izquierdo lateral	255	65 – 75
	Izquierdo central	215 – 230	0 – 1
3	Izquierdo lateral	255	95 – 102
	Izquierdo central	215 – 230	30 – 40
4	Izquierdo lateral	255	95 – 110
	Izquierdo central	255	45 – 55
5	Izquierdo lateral	255	255
	Izquierdo central	255	185 – 205
6	Izquierdo lateral	255	80 – 90
	Izquierdo central	255	20 – 40

Censo a 90°

Robot	Sensores activos	2.6 cm. (1/4 de vuelta)	5.2 cm. (1/2 vuelta)
1	Izquierdo central	255	115 – 125
	Derecho central	185- 200	25 – 45
2	Izquierdo central	165 – 175	20-35
	Derecho central	225 – 235	80 – 90
3	Izquierdo central	150 – 155	45 – 50
	Derecho central	175 – 185	35 – 40
4	Izquierdo central	205 – 15	60 – 70
	Derecho central	255	130 – 145
5	Izquierdo central	255	100 – 115
	Derecho central	200 – 225	25 – 45
6	Izquierdo central	255	105 – 125
	Derecho central	205 – 230	30 – 55

Censo a 135°

robot	Sensores activos	2.6 cm. (1/4 de vuelta)	5.2 cm. (1/2 vuelta)
1	Derecho central	255	60 – 70
	Derecho lateral	170 – 190	0 – 10
2	Derecho central	245 – 255	45 – 55
	Derecho lateral	255	20 – 30
3	Derecho central	230 - 250	50 – 70
	Derecho lateral	255	45 – 70
4	Derecho central	255	65 – 80
	Derecho lateral	255	65 – 80
5	Derecho central	255	20 – 40
	Derecho lateral	255	50 – 70
6	Derecho central	255	55 – 70
	Derecho lateral	255	95 – 110

Censo a 180°

robot	sensor	2.6 cm. (1/4 de vuelta)	5.2 cm. (1/2 vuelta)
1	Derecho lateral	200 - 255	55 – 65
2	Derecho lateral	255	115 – 135
3	Derecho lateral	255	125 – 140
4	Derecho lateral	255	70 – 90
5	Derecho lateral	255	30 - 65
6	Derecho lateral	255	45 – 80

E. Código

```
#include "p18f86j50.h"       //Moway microcontroller
#include "lib_mot_moway.h"         //Engines library
#include "lib_sen_moway.h"         //Sensor library
#include "lib_rf2gh4.h"                  //radio frecuencia

//*****************************************************************************//
// SALTO DEL BOOTLOADER
//*****************************************************************************//
#define REMAPPED_RESET_VECTOR_ADDRESS                0x1000 //Dirección de reset para saltar el bootloader.
#define REMAPPED_HIGH_INTERRUPT_VECTOR_ADDRESS       0x1008 //High priority interruption adress for the correct bootloader jump
#define REMAPPED_LOW_INTERRUPT_VECTOR_ADDRESS        0x1018 //Low priority interruption adress for the correct bootloader jump
#pragma config XINST=OFF
void YourHighPriorityISRCode();   //Function that executes the needed code in case of the high priority interruption jumps.
void YourLowPriorityISRCode();    //Function that executes the needed code in case of the low priority interruption jumps.
//***************Remapeo del reset y las interrupciones de alta y baja prioridad******************//
extern void _startup (void);
#pragma code REMAPPED_RESET_VECTOR = REMAPPED_RESET_VECTOR_ADDRESS
void _reset (void)
{_asm goto _startup _endasm}
#pragma code
#pragma code REMAPPED_HIGH_INTERRUPT_VECTOR = REMAPPED_HIGH_INTERRUPT_VECTOR_ADDRESS
void Remapped_High_ISR (void) { _asm goto YourHighPriorityISRCode _endasm }
#pragma code REMAPPED_LOW_INTERRUPT_VECTOR = REMAPPED_LOW_INTERRUPT_VECTOR_ADDRESS
void Remapped_Low_ISR (void) { _asm goto YourLowPriorityISRCode _endasm }
#pragma code
#pragma interrupt YourHighPriorityISRCode
void YourHighPriorityISRCode() {}
#pragma interruptlow YourLowPriorityISRCode
void YourLowPriorityISRCode() {}


//***************[VARIABLES GLOBALES DE INICIALIZACION]*************************************//
unsigned int avanzar = 1;              //0: detenerse, 1: seguir linea, 2: seguir moway
unsigned int direccion = 0;            //0: izq, 1: der (DIRECCION INICIAL)
unsigned int velocidad = 10;           //0 - 50: velocidad
unsigned int estatus = 0;        //0: cabeza, 2: eslabon, 3:cola
```

```
unsigned int intersec = 0;          //TIEMPO DE INTERSECCION
unsigned int detenerse = 0;               //si es diferente a 0 detenerse
unsigned int comunicar = 1;        //1: COMUNICAR 0: NO COMUNICAR

//unsigned int ruta[5] = {1,1,1,1,1};
//unsigned int ruta[5] = {1,0,1,0,1};
unsigned int ruta[5] = {0,0,0,0,0};
unsigned int r = 0;

unsigned int envmsj = 0;


//VARIABLES PARA RADIOFRECUENCIA
unsigned char data_in[8];
unsigned char dir_in = 1;
static unsigned char data_out[8];
static unsigned char dir_out = 0x00;
char ret;

void comunicarse();


//***************[PROTOCOLO                                                    DE
AGRUPAMIENTO]*************************************************************//
void agruparse(int tmprecibido, int tmpenviado)
{
  int pasar;
  int tmpdedif;
  int i;

          RF_OFF();                          //CERRAR CANAL DE COMUNICACION

  if (tmprecibido==tmpenviado)                                    //SI   MOWAYS   SE
ENCUENTRAN A MISMA DISTANCIA DE INTERSECCION
  {
          if (direccion==1)
  //vehiculo con direc. "1" tiene la preferencia (CABEZA)
                  pasar=1;
          else if (direccion==0)
  //vehiculo con direc. "0" debe ceder el paso (ESLABON)
                  pasar=2;
  }

  if (tmpenviado>tmprecibido ||  pasar==1)                    //CABEZA
   {
     comunicar=0;                          //CERRAR COMUNICACIÓN
```

```
        LED_TOP_RED_OFF();
        LED_TOP_GREEN_ON();
   }

  if (tmpenviado<tmprecibido || pasar==2)                    //ESLABON
   {
        if (tmpenviado==tmprecibido)
  //OBTENER TIEMPO DE ESPERA DE ACUERDO A DISTANCIA DE AMBOS VEHICULOS A LA
INTERSECCION
        {
              tmpenviado=0;
        }
        tmpdedif=(210-(tmprecibido-tmpenviado))+15 ;

        for (i=tmpdedif; i>=0; i--)
  //SENSAR ESLABON EN CADENA CONTRARIA MIENTRAS ESPERA
     {
              if (i%15==0)
        {
                    envmsj=1;
                    comunicarse();
                    Delay10KTCYx(5);
              }
     }

        envmsj=0;
        avanzar = 2;                          //MODO DE AVANZAR: COMENZAR A SEGUIR
        LED_TOP_GREEN_OFF();
     LED_TOP_RED_ON();
   }
}


//***************[COMUNICACION]*****************************************************//
void comunicarse()
{
  unsigned int tmpin;
  unsigned int tmpout;

  tmpout = (210-intersec);

  RF_ON();                                    //INICIAR CANAL DE COMUNICACION

   if (RF_RECEIVE(&dir_in,&data_in[0]) != 2)
   {
        MOT_STOP();
```

```
      tmpin = data_in[0];

      tmpout = tmpout/15; tmpout = tmpout*15;
      data_out[0] = (tmpout);
      ret = RF_SEND(dir_out,data_out);

      agruparse(tmpin, tmpout);
   }
   else if (tmpout%15==0 || envmsj==1)
   {
            tmpout = tmpout/15; tmpout = tmpout*15;
      data_out[0] = (tmpout);
      ret = RF_SEND(dir_out,data_out);
   }
}



//***************[VELOCIDAD]*****************************************************************//
void vel(int soax)
{
   if (soax > 210)
      detenerse = 1;
   else
   {
      if (velocidad != 0)
         velocidad = (velocidad + soax/2.62)/2;
      else
      {
         velocidad = soax/2.62;
      }
            velocidad = 80 - velocidad;
   }
}

//***************[OBTENER
VELOCIDAD]*****************************************************************//
void obtenervel()
{
   unsigned int sol_i;                    //sensor de obstaculos lateral izquierdo
  unsigned int soc_i;                     //sensor de obstaculos central izquierdo
  unsigned int soc_d;                     //sensor de obstaculos central izquierdo
  unsigned int sol_d;                     //sensor de obstaculos laterlal derecho
  int div=0;

  sol_i = SEN_OBS_ANALOG(OBS_SIDE_L);
```

```
   soc_i = SEN_OBS_ANALOG(OBS_CENTER_L);
   soc_d = SEN_OBS_ANALOG(OBS_CENTER_R);
   sol_d = SEN_OBS_ANALOG(OBS_SIDE_R);
  velocidad = 0;
  detenerse = 0;

  if (intersec>0 && direccion==1)
        velocidad=velocidad;        //DESACTIVAR SENSORES IZQUIERDOS EN
INTERSECCION SI SE AVANZA POR LA DERECHA
  else
  {
        if (sol_i > 5 && (detenerse == 0))
      vel(sol_i);

        if (soc_i > 5 && (detenerse == 0))
      vel(soc_i);
  }

  if (intersec>0 && direccion==0)
        velocidad=velocidad;        //DESACTIVAR SENSORES IZQUIERDOS EN
INTERSECCION SI SE AVANZA POR LA IZQUIERDA
  else
  {
        if (soc_d > 5 && (detenerse == 0))
      vel(soc_d);

        if (sol_d > 5 && (detenerse == 0))
      vel(sol_d);
  }

  if (detenerse==1)
        velocidad = 0;
  else if (velocidad==0)
  {
        velocidad = 10;
        avanzar = 1;
        LED_TOP_RED_OFF();
        LED_TOP_GREEN_ON();
  }

}


//**************[SEGUIR LINEA]*****************************************************************//
void seguirlinea()
{
```

```
unsigned int sla_i;         //sensor linea analogo izquierdo
unsigned int sla_d;         //sensor linea analogo derecho
unsigned int sol_i;                     //sensor de obstaculos lateral izquierdo
unsigned int soc_i;                     //sensor de obstaculos central izquierdo
unsigned int soc_d;                     //sensor de obstaculos central izquierdo
unsigned int sol_d;                     //sensor de obstaculos laterlal derecho

sla_i = SEN_LINE_ANALOG(LINE_L);
 sla_d = SEN_LINE_ANALOG(LINE_R);
sol_i = SEN_OBS_ANALOG(OBS_SIDE_L);
 soc_i = SEN_OBS_ANALOG(OBS_CENTER_L);
 soc_d = SEN_OBS_ANALOG(OBS_CENTER_R);
 sol_d = SEN_OBS_ANALOG(OBS_SIDE_R);

if (sol_i>5 || soc_i>5 || soc_d>5 || sol_d>5)
{
          avanzar=2;
          LED_TOP_GREEN_OFF();
   LED_TOP_RED_ON();
}

 if (avanzar == 2)                  //ADMINISTRAR VELOCIDAD (SOLO SI ESTA SIGUIENDO A OTRO
MOWAY)
    obtenervel();

 if (velocidad > 0)             //Si VELOCIDAD ES DIFERENTE DE CERO AVANZAR
 {
   if (direccion == 0)
   {
     if ((sla_i <= 18) && (sla_d >= 19))
     {
       MOT_STR(velocidad,FWD,TIME,1);
     }
     else if ((sla_i <= 18) && (sla_d <= 18))
     {
       MOT_ROT(velocidad,FWD,WHEEL,RIGHT,ANGLE,1);
     }
     else if ((sla_i >= 19) && (sla_d >= 19))
     {
       MOT_ROT(velocidad,FWD,WHEEL,LEFT,ANGLE,1);
     }
     else if ((sla_i >= 19) && (sla_d <= 18))
     {
       MOT_ROT(velocidad,FWD,WHEEL,LEFT,ANGLE,1);
     }
   }
```

```
      else if (direccion == 1)
      {
         if ((sla_i >= 19) && (sla_d <= 18))
         {
            MOT_STR(velocidad,FWD,TIME,1);
         }
         else if ((sla_i <= 18) && (sla_d <= 18))
         {
            MOT_ROT(velocidad,FWD,WHEEL,LEFT,ANGLE,1);
         }
         else if ((sla_i >= 19) && (sla_d >= 19))
         {
            MOT_ROT(velocidad,FWD,WHEEL,RIGHT,ANGLE,1);
         }
         else if ((sla_i <= 18) && (sla_d >= 19))
         {
            MOT_ROT(velocidad,FWD,WHEEL,RIGHT,ANGLE,1);
         }
      }

      if (intersec==0 && (sla_i>=180 || sla_d>=180))   //SI SE RECONOCE SENAL DE INTERSECCION
      {
         if (avanzar == 1)                  //SI ES CABEZA
         {
            intersec=210;                    //INICIAR TIEMPO DE INTERSECCION
         }
         else if (avanzar == 2)              //SI ES ESLABON
         {
            intersec=210;                    //INICIAR TIEMPO DE INTERSECCION
         }
      }
    }
}


//***************[MAIN]************************************************//
void main()
{
  unsigned int i = 200;
  Delay10KTCYx(200); //Función de retraso, 2 segundos = (200*(4/4MHz)*10000).

  //**********Configuración de sensores (utilizar solo si se necesita)************//
  SEN_CONFIG();
  //********** Configuración de motores (utilizar solo si se necesita)***********//
  MOT_CONFIG();
  //********** Configuración de modulo RF (utilizar solo si se necesita)*********//
```

```
  RF_CONFIG_SPI();
  RF_CONFIG(0x00,0x01); // RF_CONFIG(Canal de comunicación (debe ser el mismo para todos los mOway), Identificador (cada mOway debe tener un único identificador))

  LED_FRONT_ON();

  while(avanzar != 0)
  {
    seguirlinea();              //SEGUIR LINEA

        if(intersec > 0)            //SI TIEMPO DE INTERSECCION ESTA ACTIVO
    {
            if (comunicar==1)                                   //SI COMUNICARSE ESTA ACTIVADO
        comunicarse();              //LLAMAR A COMUNICARSE

            if (intersec==1)                                    //SI TIEMPO DE INTERSEC ES IGUAL A 1
            {
                if (direccion!=ruta[r])                         //SI RUTA ES DIFERENTE DE DIRECCION CAMBIA
                    direccion=ruta[r];
                r++;
                comunicar=1;
            }

      intersec--;               //RESTAR CONTADOR DE TIEMPO DE INTERSECCION
    }
    else                                //TERMINO TIEMPO DE INTERSECCION
      RF_OFF();                 //CERRAR CANAL DE COMUNICACION
  }
}
```