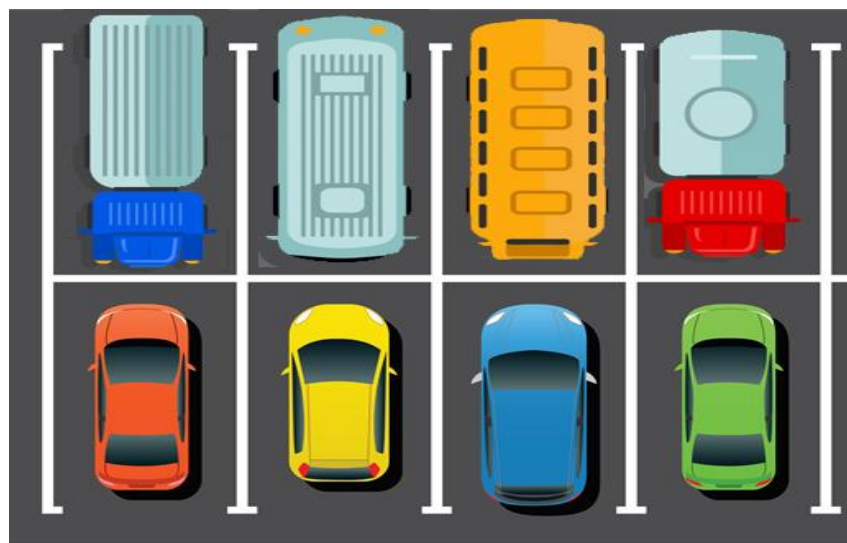


Ingeniería en Sistemas Computacionales

Sistemas Programables

Tema: Estacionamiento Inteligente



Semestre: Octavo

Grupo: 8F71

Catedrático: D. en C. A. C. C. German Cuaya-Simbro

Realizado por:

Matricula	Nombre
14030365	García Lara José Antonio
14030371	Gómez Pereyra Luis Raúl
14030361	Trejo Hernández David
14030595	Zavala Torres Alberto

1 CONTENIDO

1	Contenido	3
2	Tabla de Ilustraciones	4
3	Resumen General del objetivo del sistema	5
4	Descripción del vehículo	5
5	Descripción del funcionamiento del sistema	5
6	Ficha técnica de los componentes de Hardware.....	6
7	Ficha técnica de los componentes del Software	9
8	Vehículo Familiar	9
8.1	Conexiones en la Placa Infiduoino	9
8.1.1	Conexión Microcontrolador – Arduino.....	10
8.2	Desarrollo de la aplicación móvil	11
8.3	Programación en Arduino	13
8.3.1	Tabla de desplazamiento.....	14
8.3.2	Rutinas Establecidas	15
9	Funcionamiento del Vehículo Familiar (Pruebas)	21
10	Discusión de los resultados	23
	Referencias	23

2 TABLA DE ILUSTRACIONES

Ilustración 1 Placa Arduino Uno	6
Ilustración 2 Motor Reductor 6 V 200 RPM	6
Ilustración 3 Protoboard 800 puntos	7
Ilustración 4 Módulo Bluetooth HC-06 Serial Rs232 (TTL)	7
Ilustración 5 Batería 9V	8
Ilustración 6 Puente H L293D	8
Ilustración 7 Logo Arduino.....	9
Ilustración 8 Logo DRAZ.....	9
Ilustración 9 Conexión placa Infiduino	10
Ilustración 10 Layout Principal	11
Ilustración 11 Nombre de los Botones	12
Ilustración 12 Sincronización con el modulo Bluetooth.....	12
Ilustración 13 Funcionamiento de los botones	13
Ilustración 14 Crear Variables.....	13
Ilustración 15 Puerto serial del Bluetooth.....	13
Ilustración 16 Tabla de desplazamiento.....	14
Ilustración 17 Movimiento básico Adelante.....	14
Ilustración 18 Movimiento básico Atrás.....	14
Ilustración 19 Movimiento básico Derecha.....	15
Ilustración 20 Movimiento básico Izquierdo	15
Ilustración 21 Movimiento básico Detener	15
Ilustración 22 Vehículo llega a la entrada	16
Ilustración 23 Vehículo se estaciona en el lugar número 1.....	16
Ilustración 24 Vehículo se estaciona en el lugar número 2.....	17
Ilustración 25 Vehículo se estaciona en el lugar número 3.....	18
Ilustración 26 Vehículo sale del lugar número 1	18
Ilustración 27 Vehículo sale del lugar número 2	19
Ilustración 28 Vehículo sale del lugar número 3	20
Ilustración 29 Vehículo se retira lugar no disponible	20
Ilustración 30 Aplicación desde el dispositivo	21
Ilustración 31 Vehículo Familiar Superior.....	22
Ilustración 32 Vehículo Familiar Lateral	22

3 RESUMEN GENERAL DEL OBJETIVO DEL SISTEMA

Diseñar un estacionamiento automatizado que pueda verificar la capacidad, el acceso de los vehículos se dividirá en 2 grupos, vehículos familiares y de carga. Los vehículos deberán de ser capaces de ser autónomos, es decir, tendrá la capacidad de moverse por sí solo hasta llegar al lugar adecuado e indicado por el sistema.



4 DESCRIPCIÓN DEL VEHÍCULO

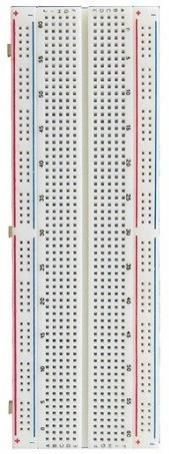

La clasificación del vehículo autónomo que está en desarrollo es de tipo familiar, para su correcto funcionamiento tiene que comunicarse con el sistema para poder estacionarse en un lugar específico. Para lograrlo se requiere la conexión de diferentes dispositivos para poder hacer funcionar un circuito, el cual contiene sensores y actuadores.

5 DESCRIPCIÓN DEL FUNCIONAMIENTO DEL SISTEMA

El estacionamiento debe ser capaz de detectar que tipo de vehículo es el que quiere estacionarse, al detectar el tipo de vehículo que es, deberá detectar si está disponible algún lugar de su clasificación, de ser así, le asignara el lugar correspondiente al vehículo, de lo contrario se negara la entrada a dicho vehículo.



6 FICHA TÉCNICA DE LOS COMPONENTES DE HARDWARE

Material		Descripción
 <p><i>Ilustración 1 Placa Arduino Uno</i></p>	<p>✓ Arduino y/o Infiduno UNO</p>	<p>Arduino Uno es una placa electrónica basada en el microcontrolador ATmega328. Cuenta con 14 entradas/salidas digitales, de las cuales 6 se pueden utilizar como salidas PWM (Modulación por ancho de pulsos) y otras 6 son entradas analógicas.</p> <p>Además, incluye un resonador cerámico de 16 MHz, un conector USB, un conector de alimentación, una cabecera ICSP y un botón de reseteo.</p> <p>La placa incluye todo lo necesario para que el microcontrolador haga su trabajo, basta conectarla a un ordenador con un cable USB o a la corriente eléctrica a través de un transformador.</p>
 <p><i>Ilustración 2 Motor Reductor 6 V 200 RPM</i></p>	<p>✓ Motores</p>	<p>Los motorreductores son apropiados para el accionamiento de toda clase de máquinas y aparatos, que necesitan reducir su velocidad en una forma segura y eficiente.</p> <p>Especificaciones</p> <ul style="list-style-type: none"> ▪ Voltaje de operación: 3 - 6 V ▪ Corriente sin carga (3 V): 150 mA ▪ Corriente sin carga (6 V): 200 mA

		<ul style="list-style-type: none"> ▪ Velocidad sin carga (3 V): 90 RPM + / - 10 % ▪ Velocidad sin carga (6 V): 200 RPM + / - 10 %
 <p><i>Ilustración 3 Protoboard 800 puntos</i></p>	<p>✓ Protoboard</p>	<p>Es un tablero con orificios que se encuentran conectados eléctricamente entre sí de manera interna, habitualmente siguiendo patrones de líneas, en el cual se pueden insertar componentes electrónicos y cables para el armado y prototipado de circuitos electrónicos y sistemas similares.</p>
 <p><i>Ilustración 4 Módulo Bluetooth HC-06 Serial Rs232 (TTL)</i></p>	<p>✓ Módulo Bluetooth</p>	<p>El módulo BlueTooth HC-06 es ideal para aplicaciones inalámbricas, fácil de implementar con PC, microcontrolador o módulos Arduinos.</p> <p>La tarjeta incluye un adaptador con 4 pines de fácil acceso para uso en protoboard. Los pines de la board correspondientes son:</p> <ul style="list-style-type: none"> ▪ VCC ▪ GND ▪ RX ▪ TX

 <p><i>Ilustración 5 Batería 9V</i></p>	<p>✓ Batería</p>	<p>Pila o batería cuadrada de 9 volts para dar energía a aparatos electrónicos, juguetes o a proyectos de electrónica.</p> <p>Es alcalina de larga duración, con capacidad nominal de 150 mAh y está libre de cadmio y mercurio.</p>
 <p><i>Ilustración 6 Puente H L293D</i></p>	<p>✓ Puente H</p>	<p>El integrado L293D incluye cuatro circuitos para manejar cargas de potencia media, en especial pequeños motores y cargas inductivas, con la capacidad de controlar corriente hasta 600 mA en cada circuito y una tensión entre 4,5 V a 36 V.</p> <p>En este caso el manejo será bidireccional, con frenado rápido y con posibilidad de implementar fácilmente el control de velocidad.</p>

7 FICHA TÉCNICA DE LOS COMPONENTES DEL SOFTWARE

Material		Descripción
 <i>Ilustración 7 Logo Arduino</i>	✓ Programa Arduino	<p>Arduino es una plataforma de prototipos electrónica de código abierto (open-source) basada en hardware y software flexibles y fáciles de usar.</p> <p>Características</p> <ul style="list-style-type: none">▪ Barato▪ Multiplataforma▪ Entorno de programación simple y claro▪ Código abierto y software extensible▪ Código abierto y hardware extensible
 <i>Ilustración 8 Logo DRAZ</i>	✓ Aplicación Control Remoto (aplicación Móvil)	<p>Es una aplicación que permite controlar el vehículo remotamente, desarrollado en la plataforma App Inventor</p>

8 VEHÍCULO FAMILIAR

8.1 CONEXIONES EN LA PLACA INFIDUINO

Se usaron los siguientes diagramas para poder conectar los módulos y sensores en Arduino Uno.

El modulo Bluetooth se conecta de manera cruzado, es decir, el pin TX del bluetooth se conecta al pin RX de la placa Infiduinio, y el pin RX del bluetooth se conecta al TX de la placa Infiduinio.

Giro y Reversa de Carros y Robot

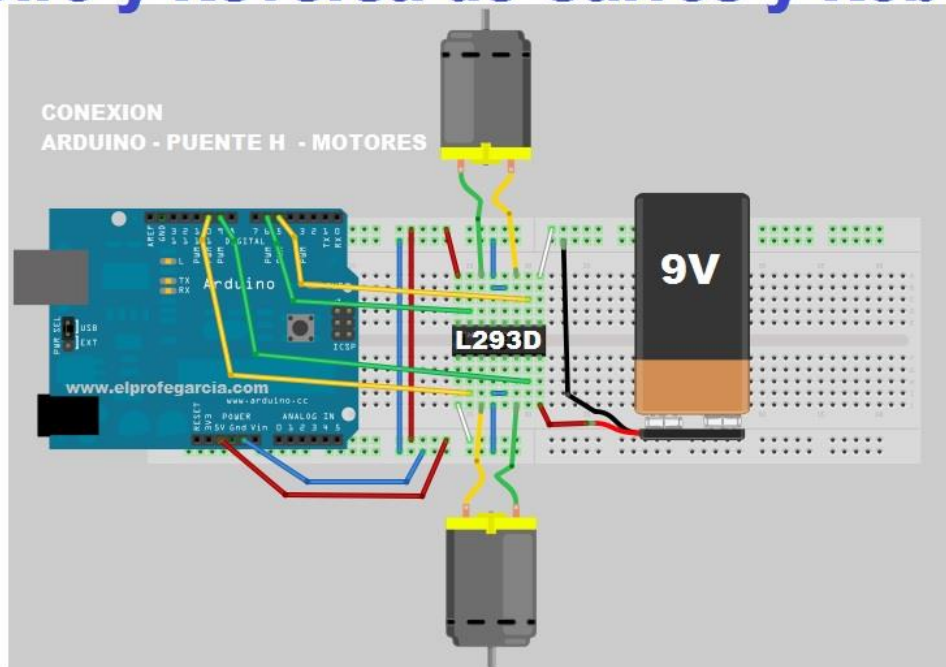


Ilustración 9 Conexión placa Infidduino

En la Ilustración 9 se muestra las conexiones a la placa Arduino que son de la siguiente manera:

8.1.1 Conexión Microcontrolador – Arduino

- 1) Se hace un puente en la protoboard de la parte positiva y la parte negativa.
- 2) Pin 1 del microcontrolador se conecta a Corriente.
- 3) Pin 2 del microcontrolador se conecta al pin 10 del Arduino.
- 4) Pin 3 del microcontrolador se conecta a la parte positiva del Motor.
- 5) Pin 4 y 5 del microcontrolador se hace un puente y se conecta a GND.
- 6) Pin 6 del microcontrolador se conecta a la parte negativa del Motor.
- 7) Pin 7 del microcontrolador se conecta al pin 9 del Arduino.
- 8) Pin 8 del microcontrolador se conecta a la parte positiva de la pila de 9v.
- 9) Pin 9 del microcontrolador se conecta a la parte positiva.
- 10) Pin 10 del microcontrolador se conecta al pin 5 del Arduino.
- 11) Pin 11 del microcontrolador se conecta a la parte positiva del motor.
- 12) Pin 12 y 13 del microcontrolador se hace un puente y se conecta a GND.
- 13) Pin 14 del microcontrolador se conecta a la parte negativa del Motor.
- 14) Pin 15 del microcontrolador se conecta al pin 4 del Arduino.
- 15) Pin 16 del microcontrolador se conecta a Corriente.

8.2 DESARROLLO DE LA APLICACIÓN MÓVIL

La aplicación se desarrolló en App Inventor en el siguiente enlace:

<http://appinventor.mit.edu/explore/>

App Inventor es un entorno de desarrollo de software creado por Google Labs para la elaboración de aplicaciones destinadas al sistema operativo Android.

En la Ilustración 10 se muestra el Layout Principal, donde se crearon los botones que dan funcionalidad a la aplicación.

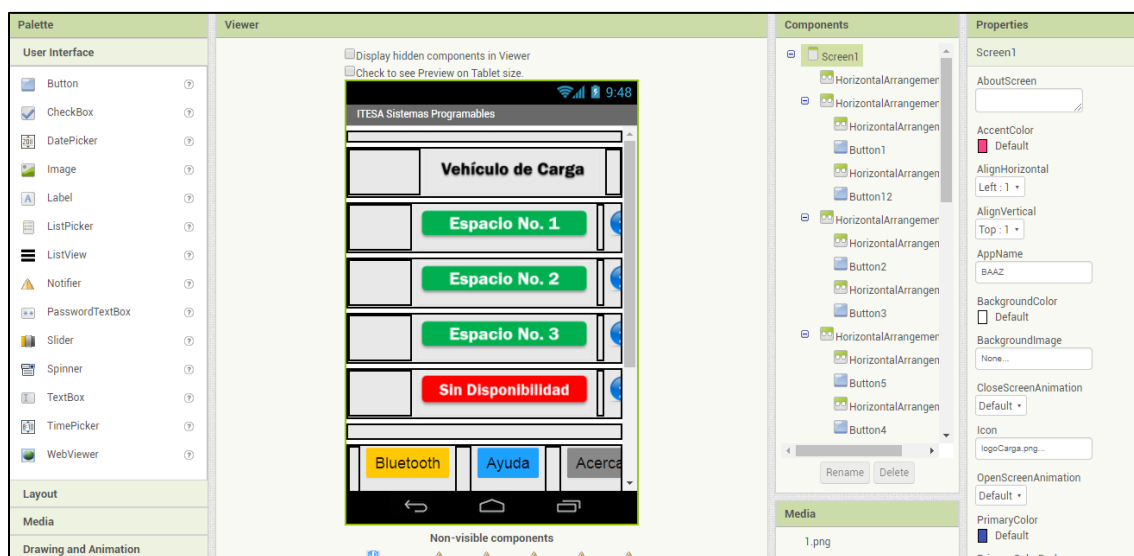


Ilustración 10 Layout Principal

En la ilustración 11 se muestra el nombre de referencia de cada botón.

Vehículo Familiar



Ilustración 11 Nombre de los Botones

A continuación se describirá el código que se estableció para realizar la conexión con el modulo bluetooth y permitir la comunicación entre el vehículo y la aplicación.

En la ilustración 12 permite la sincronización del módulo bluetooth y la aplicación móvil.

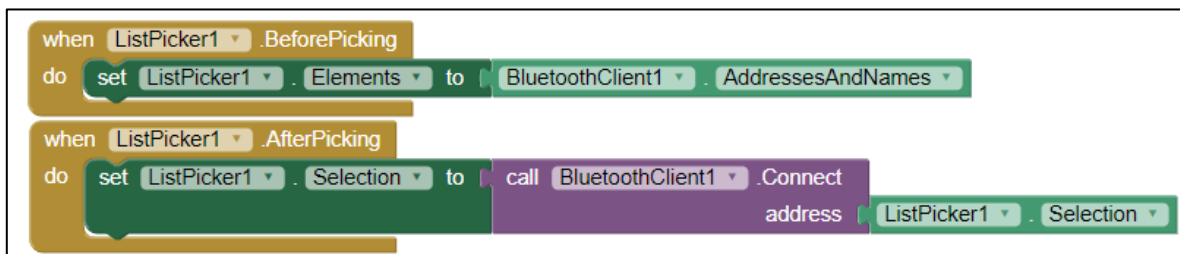


Ilustración 12 Sincronización con el modulo Bluetooth

En la ilustración 13 se muestra el funcionamiento de cada botón.

Al presionar algun boton de la aplicación, esta enviara un carácter al modulo Bluetooth dependiendo el boton que se precione, este la interpretara y la enviara al programa Arduino,que la procesara de acuerdo a las instrucciones que se establecieron.

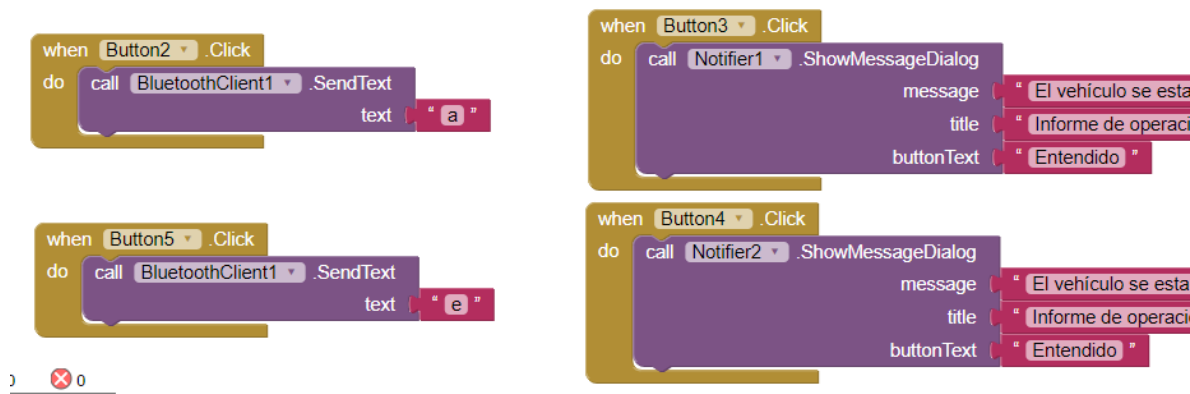


Ilustración 13 Funcionamiento de los botones

8.3 PROGRAMACIÓN EN ARDUINO

El código que se desarrolló para para el funcionamiento del vehículo es el siguiente:

Se crean variables que seran igual a los pines de entrada del Infiduino, se asignara la velocidad y el estado en detenido.

```
//Pines de conexión del driver
int Pin_Motor_Der_A = 9;
int Pin_Motor_Der_B = 10;
int Pin_Motor_Izq_A = 5;
int Pin_Motor_Izq_B = 6;
int vel=255;
int tiempo=0;
```

Ilustración 14 Crear Variables

Se inicia el puesto serial para la comunicación con el bluetooth y se define el ancho de pulso.

```
void setup() {
  // inicializar la comunicación serial a 9600 bits por segundo:
  Serial.begin(9600);
  // configuramos los pines como salida
  pinMode(Pin_Motor_Der_A, OUTPUT);
  pinMode(Pin_Motor_Der_B, OUTPUT);
  pinMode(Pin_Motor_Izq_A, OUTPUT);
  pinMode(Pin_Motor_Izq_B, OUTPUT);
}
```

Ilustración 15 Puerto serial del Bluetooth

8.3.1 Tabla de desplazamiento

Rutina	Der_A	Izq_A	Der_B	Izq_B
Parar	0	0	0	0
Avanzar Adelante	0	0	Vel	Vel
Avanzar Atrás	Vel	Vel	0	0
Girar Izquierda	0	0	0	Vel
Girar Derecha	0	0	Vel	0

Ilustración 16 Tabla de desplazamiento

Para el funcionamiento de las rutinas, es necesario establecer movimientos básicos del desplazamiento del vehículo mediante métodos en Arduino es decir, establecer el movimiento 'Adelante', 'Izquierda', 'Derecha', 'Atrás'.

De acuerdo a la tabla de desplazamiento se realizaron los movimientos siguientes básicos.

Método para avanzar hacia adelante.

```
void Mover_Adelante()
{
    digitalWrite (Pin_Motor_Der_A, 0);
    digitalWrite (Pin_Motor_Der_B, vel);
    digitalWrite (Pin_Motor_Izq_A, 0);
    digitalWrite (Pin_Motor_Izq_B, vel);
}
```

Ilustración 17 Movimiento básico Adelante

Método para avanzar hacia Atrás.

```
void Mover_Retroceso()
{
    digitalWrite (Pin_Motor_Der_A, vel );
    digitalWrite (Pin_Motor_Der_B, 0 );
    digitalWrite (Pin_Motor_Izq_A, vel );
    digitalWrite (Pin_Motor_Izq_B, 0 );
}
```

Ilustración 18 Movimiento básico Atrás

Método para girar a la Derecha.

```

void Mover_Derecha()
{
    digitalWrite (Pin_Motor_Der_A, 0 );
    digitalWrite (Pin_Motor_Der_B, vel );
    digitalWrite (Pin_Motor_Izq_A, 0);
    digitalWrite (Pin_Motor_Izq_B, 0);
}

```

Ilustración 19 Movimiento básico Derecha

Método para girar a la Izquierda.

```

void Mover_Izquierda()
{
    digitalWrite (Pin_Motor_Der_A, 0);
    digitalWrite (Pin_Motor_Der_B, 0);
    digitalWrite (Pin_Motor_Izq_A, 0);
    digitalWrite (Pin_Motor_Izq_B, vel );
}

```

Ilustración 20 Movimiento básico Izquierdo

Método para detener.

```

void Mover_Stop()
{
    digitalWrite (Pin_Motor_Der_A, 0);
    digitalWrite (Pin_Motor_Der_B, 0);
    digitalWrite (Pin_Motor_Izq_A, 0);
    digitalWrite (Pin_Motor_Izq_B, 0);
}

```

Ilustración 21 Movimiento básico Detener

8.3.2 Rutinas Establecidas

Al presionar en la aplicación el botón **LLEGAR A LA ENTRADA** envía al módulo bluetooth el carácter 'a', el cual el programa Arduino indica al vehículo llegar a la entrada del estacionamiento.

Arduino asigna la rutina establecida cuando recibe el carácter a cada uno de los valores de entrada para que el auto realice la rutina.

```
//-----CARRO LLEGA A ENTRADA-----
if(dato=='a'){
//movemos el carro hacia la entrada
if(tiempo<1000) // 100 cilcos de 1ms
{
Mover_Adelante();
delay(1500);
Mover_Stop();
delay(200);
Mover_Izquierda();
delay(4000);
Mover_Stop();
delay(200);
Mover_Adelante();
delay(4000);
Mover_Stop();
delay(200);
}
else //ya transcurrió 100ms (100ciclos)
{
Mover_Stop();
}
}
```

Ilustración 22 Vehículo llega a la entrada

Al presionar en la aplicación el botón **ENTRA E. 1** envía al módulo bluetooth el carácter 'b', el cual el programa Arduino Indica al vehículo estacionarse en el lugar número 1.

Arduino asigna la rutina establecida cuando recibe el carácter a cada uno de los valores de entrada para que el auto realice la rutina.

```
//-----ESTACIONAMIENTO 1-----
if(dato=='b'){
//Entra el carro hacia el estacionamiento numero 1
if(tiempo<1000) // 100 cilcos de 1ms
{
Mover_Adelante();
delay(750);
Mover_Stop();
delay(200);
Mover_Derecha();
delay(1075);
Mover_Stop();
delay(200);
Mover_Adelante();
delay(900);
Mover_Stop();
delay(200);
}
else //ya transcurrió 100ms (100ciclos)
{
Mover_Stop();
}
}
```

Ilustración 23 Vehículo se estaciona en el lugar número 1

Al presionar en la aplicación el botón **ENTRA E. 2** envía al módulo bluetooth el carácter 'c', el cual el programa Arduino Indica al vehículo estacionarse en el lugar número 2.

Arduino asigna la rutina establecida cuando recibe el carácter a cada uno de los valores de entrada para que el auto realice la rutina.

```
//-----ESTACIONAMIENTO 2-----  
if(dato=='c'){  
  //Entra el carro hacia el estacionamiento numero 2  
  if(tiempo<1000) // 100 cilcos de 1ms  
  {  
    Mover_Adelante();  
    delay(4000);  
    Mover_Stop();  
    delay(200);  
    Mover_Derecha();  
    delay(1500);  
    Mover_Stop();  
    delay(200);  
    Mover_Adelante();  
    delay(4000);  
    Mover_Stop();  
    delay(200);  
  }  
  else //ya transcurrió 100ms (100ciclos)  
  {  
    Mover_Stop();  
  }  
}
```

Ilustración 24 Vehículo se estaciona en el lugar número 2

Al presionar en la aplicación el botón **ENTRA E. 3** envía al módulo bluetooth el carácter 'd', el cual el programa Arduino Indica al vehículo estacionarse en el lugar número 3.

Arduino asigna la rutina establecida cuando recibe el carácter a cada uno de los valores de entrada para que el auto realice la rutina.


```
//-----ESTACIONAMIENTO 3-----
if(dato=='d'){
  //Entra el carro hacia el estacionamiento numero 3
  if(tiempo<1000) // 100 cilcos de 1ms
  {
    Mover_Adelante();
    delay(4000);
    Mover_Stop();
    delay(200);
    Mover_Derecha();
    delay(1500);
    Mover_Stop();
    delay(200);
    Mover_Adelante();
    delay(4000);
    Mover_Stop();
    delay(200);
  }
  else //ya transcurrió 100ms (100ciclos)
  {
    Mover_Stop();
  }
}
```

Ilustración 25 Vehículo se estaciona en el lugar número 3

Al presionar en la aplicación el botón **SALE E. 1** envía al módulo bluetooth el carácter 'e', el cual el programa Arduino Indica al vehículo salir del lugar número 1.

Arduino asigna la rutina establecida cuando recibe el carácter a cada uno de los valores de entrada para que el auto realice la rutina.

```
//-----SALIR ESTACIONAMIENTO 1-----
if(dato=='e'){
  //Sale el carro del estacionamiento numero 1
  if(tiempo<1000) // 100 cilcos de 1ms
  {
    Mover_Retroceso();
    delay(4000);
    Mover_Stop();
    delay(200);
    Mover_Izquierda();
    delay(1500);
    Mover_Stop();
    delay(200);
    Mover_Adelante();
    delay(4000);
    Mover_Stop();
    delay(200);
    Mover_Derecha();
    delay(1500);
    Mover_Stop();
    delay(200);
  }
  else //ya transcurrió 100ms (100ciclos)
  {
    Mover_Stop();
  }
}
```

Ilustración 26 Vehículo sale del lugar número 1

Al presionar en la aplicación el botón **SALE E. 2** envía al módulo bluetooth el carácter 'f', el cual el programa Arduino Indica al vehículo salir del lugar número 2.

Arduino asigna la rutina establecida cuando recibe el carácter a cada uno de los valores de entrada para que el auto realice la rutina.

```
//-----SALIR ESTACIONAMIENTO 2-----  
if(dato=='f'){  
  //Sale el carro del estacionamiento numero 2  
  if(tiempo<1000) // 100 cilcos de 1ms  
  {  
    Mover_Retroceso();  
    delay(4000);  
    Mover_Stop();  
    delay(200);  
    Mover_Izquierda();  
    delay(1500);  
    Mover_Stop();  
    delay(200);  
    Mover_Adelante();  
    delay(4000);  
    Mover_Stop();  
    delay(200);  
    Mover_Derecha();  
    delay(1500);  
    Mover_Stop();  
    delay(200);  
  }  
  else //ya transcurrió 100ms (100ciclos)  
  {  
    Mover_Stop();  
  }  
}
```

Ilustración 27 Vehículo sale del lugar número 2

Al presionar en la aplicación el botón **SALE E. 3** envía al módulo bluetooth el carácter 'g', el cual el programa Arduino Indica al vehículo salir del lugar número 2.

Arduino asigna la rutina establecida cuando recibe el carácter a cada uno de los valores de entrada para que el auto realice la rutina.

```
//-----SALIR ESTACIONAMIENTO 3-----
if(dato=='g'){
  //Sale el carro del estacionamiento numero 3
  if(tiempo<1000) // 100 cilcos de 1ms
  {
    Mover_Retroceso();
    delay(4000);
    Mover_Stop();
    delay(200);
    Mover_Izquierda();
    delay(1500);
    Mover_Stop();
    delay(200);
    Mover_Adelante();
    delay(4000);
    Mover_Stop();
    delay(200);
    Mover_Derecha();
    delay(1500);
    Mover_Stop();
    delay(200);
  }
  else //ya transcurrió 100ms (100ciclos)
  {
    Mover_Stop();
  }
}
```

Ilustración 28 Vehículo sale del lugar número 3

Al presionar en la aplicación el botón **SIN DISPONIBILIDAD** envía al módulo bluetooth el carácter 'h', el cual el programa Arduino Indica al vehículo salir del lugar número 2.

Arduino asigna la rutina establecida cuando recibe el carácter a cada uno de los valores de entrada para que el auto realice la rutina.

```
//-----ESTACIONAMIENTO NO DISPONIBLE-----
if(dato=='h'){
  //El auto no entra al estacionamiento
  if(tiempo<1000) // 100 cilcos de 1ms
  {
    Mover_Derecha();
    delay(1500);
    Mover_Stop();
    delay(200);
    Mover_Adelante();
    delay(4000);
    Mover_Stop();
    delay(200);
    Mover_Derecha();
    delay(1500);
    Mover_Stop();
    delay(200);
  }
  else //ya transcurrió 100ms (100ciclos)
  {
    Mover_Stop();
  }
}
```

Ilustración 29 Vehículo se retira lugar no disponible

9 FUNCIONAMIENTO DEL VEHÍCULO FAMILIAR (PRUEBAS)

En la Ilustración 24 se visualiza la aplicación desarrollada desde el dispositivo móvil.

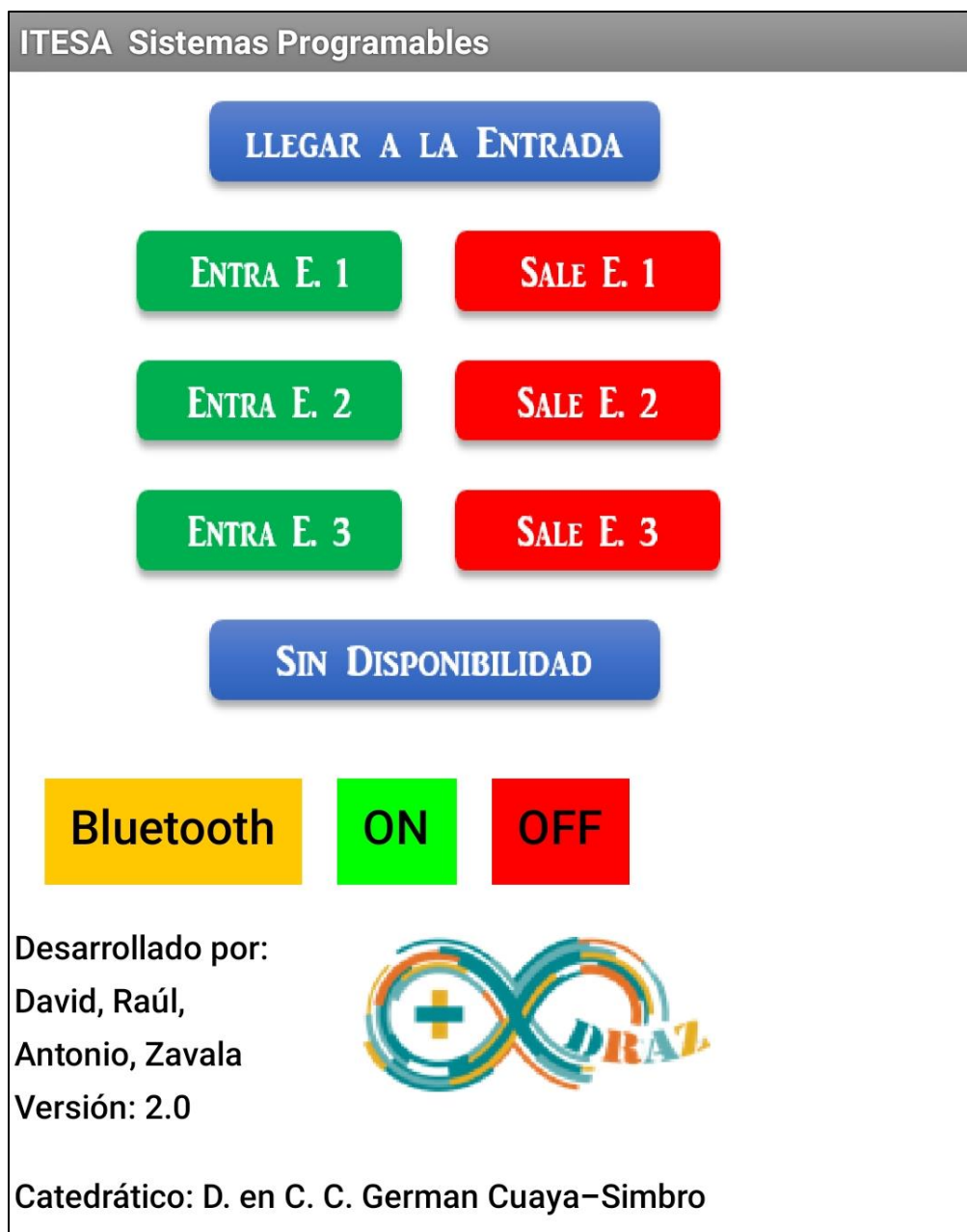


Ilustración 30 Aplicación desde el dispositivo

En las siguientes ilustraciones se muestra el vehículo familiar conectado a la placa Infiduinio
Desde la parte superior y desde el perfil.

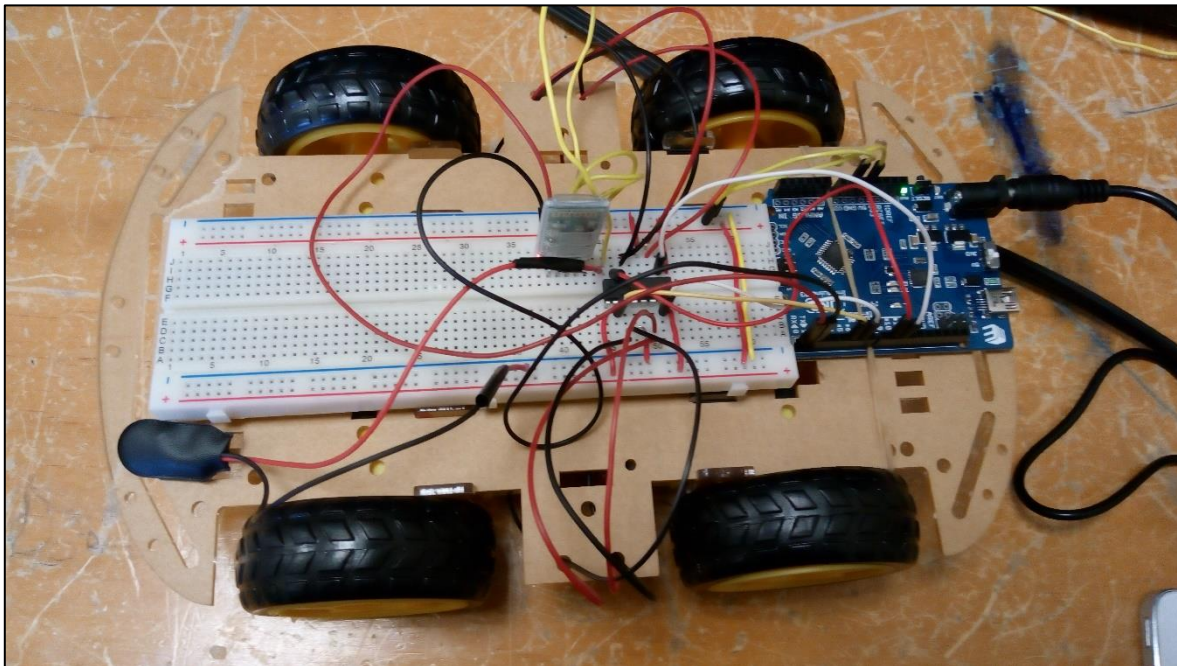


Ilustración 31 Vehículo Familiar Superior

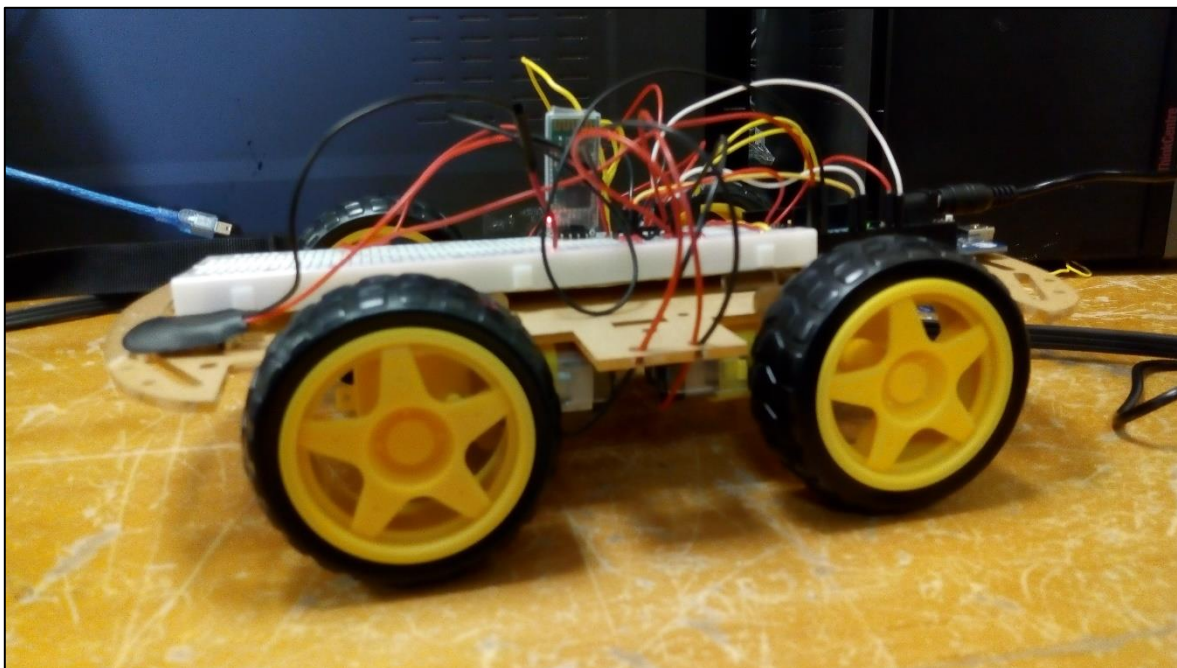


Ilustración 32 Vehículo Familiar Lateral

10 DISCUSIÓN DE LOS RESULTADOS

El funcionamiento del carro con Arduino resulto satisfactorio, aunque se encontraron algunas dificultades, en la cual la principal fue la duración de la batería, ya que no se incluyó un driver 1289D “Manejador de Motores” (el cual nos ayudaría a que tenga una mayor duración la batería). Otro problema que resulto importante fue el ciclo de las rutinas (no tenía una condición de paro) la cual nos generó el problema de que nuestra rutina no termine.

REFERENCIAS

Módulo de bluetooth. Visitado 31 de enero del 2018. ELECTRONILAB. Sitio Web:

<https://electronilab.co/tienda/modulo-bluetooth-hc-06-serial-rs232ttl/>

Sensor ultrasónico. Visitado 31 de enero del 2018. ELECTRONILAB. Sitio Web:

<https://electronilab.co/tienda/sensor-de-distancia-de-ultrasonido-hc-sr04/>

Arduino, LiquidCrystal Library. Revisado el 2 de marzo de 2018. Sitio web:

<https://www.arduino.cc/en/Reference/LiquidCrystal>