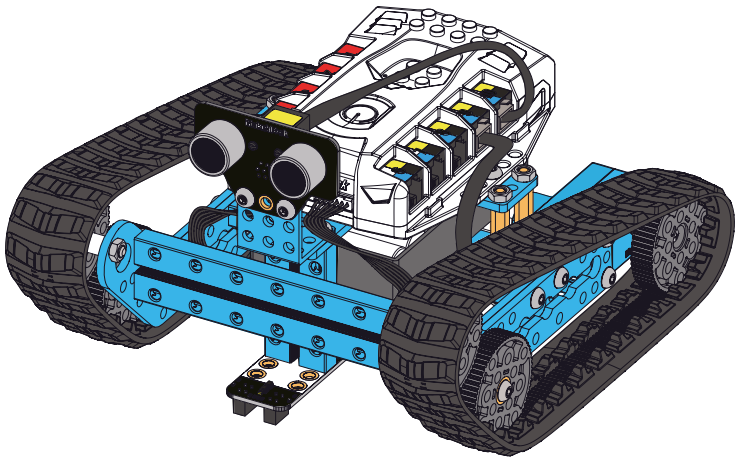
PC Factory
Área de profesionales y especialistas
Equipo de computación y electrónica

Manual de usuario

Makeblock mBot Ranger

Kit robótico educacional 3-en-1



Índice

1. Introducción

mBot Ranger es un robot diseñado por Makeblock con fines educacionales. Este corresponde al modelo intermedio de la compañía con el cual se utilizan las habilidades adquiridas con modelos más básicos, como el mBot. El mBot Ranger amplia la gama de sensores y actuadores con la que cuenta el usuario además de introducir flexibilidad de armado en su estructura, de esta manera el robot se vuelve transformable con al menos 3 modelos diferentes a su disposición. El kit educacional en el cual el mBot Ranger viene incluido cuenta con todo lo necesario para ser autosuficiente, es decir, incluye todo lo necesario para ensamblar y utilizar el robot a excepción de las baterías.

En las próximas secciones se utilizarán nombres en inglés de algunas componentes para evitar ambigüedades

2. Especificaciones técnicas

Categoría	**Especificación**
Software para programaciónn (Mac/Windows/Linux)	mBlock, Arduino IDE
App (iPad/Android)	Makeblock
Microcontrolador	Arduino Mega 2560
Sensores	2 x Light sensor 1 x Sound sensor 1 x Giroscopio/Acelerómetro 1 x Sensor de temperatura 1 x Sensor Ultrasónico 1 x Seguidor de línea
Actuadores	1 x Buzzer 12 x LED RGB 2x Motor DC Encoder
Alimentación	6 x Baterías AA (no incluidas)
Comunicación inalámbrica	Bluetooth/2.4G
Dimensiones	20 x 16,5 x 12 cm (Max)
Peso	1600g

Tabla 1: Especificaciones técnicas mBot Ranger

3. Piezas del kit robótico

Figura 1: Componentes en el kit

4. Microcontrolador

El cerebro del mBot Ranger corresponde a la tarjeta MeAuriga, esta es una tarjeta de desarrollo diseñada por Makeblock en entorno a un microcontrolador ATmega2560, mejor conocido como Arduino Mega. La tarjeta en cuestión se puede ver en la Figura 2, esta cuenta con una conexión USB tipo B para mejor durabilidad y permitir la programación con PC. Con respecto a la alimentación, esta puede trabajar con voltajes desde 6V a 12V los cuales pueden ser suministrados a través de conectar un pack de 6 baterías AA. El resumen de estos datos se puede ver en la Tabla 2.

Para el manejo de sensores y actuadores, cuenta con conexiones RJ25 las cuales están mapeadas a diferentes pines del microcontrolador (ver Figura 2 y Figura 3). Del *pin mapping* se desprende que todos los puertos sirven para el manejo de señales digitales, más no así las analógicas cuyas entradas pueden ser leídas solo en los PORT6-PORT10 y las salidas analógicas generadas en los PORT1-PORT4. Dentro de esto cabe destacar la posibilidad de manejar actuadores de mayor potencia gracias a los drivers integrados en la tarjeta, en especifico, se puede suplir un total de 3A distribuidos entre los puertos PORT1-PORT4 y 1A por motor en los puertos M1 y M2 de motores con encoder. Además de esto, la posibilidad de monitorear la comunicación serial I^2C en cualquiera de los puertos (todos comparten el mismo pin de comunicación) e implementar comunicación UART en el PORT5. Cabe destacar en

esto último que la comunicación I^2C está directamente conectada al integrado IMU 6, por lo que no es de libre uso, de la misma manera la comunicación UART se encuentra conectada al módulo USB de la tarjeta..

En adición a todo esto, cuenta con protección ante altas de corriente y de voltaje para evitar problemas de seguridad ante cortocircuitos o conexiones erróneas. Estas se manifiestan a través del uso de fusibles reiniciables (3.5V para los nodos de motores y 1.5A para todo el resto), diodos Schottky de barrera en ambas alimentaciones y supresores de voltaje transiente (5V para el microcontrolador y 12V para los motores).

Main Characteristic	**Specificaction**		
Main Board			
Microcontroller	Atmega 2560		
Input Voltage (V)	6-12 V_{DC}		
Communication Mode	UART, I2C, GPIO		
Circuit Board Dimension (L x W x H)	100 x 65x 18 mm		
Product Dimension	109 x 88 x 27mm		
PORT1-PORT4	**Min**	**Typ**	**Max**
Max Output Voltage (V)	6	9	12
Maximum instantaneous output current (A)	-	3	5 (peak)
PORT6-PORT10			
Max Output Voltage (V)	-	5	-
Maximum instantaneous output current (A)	-	2.4	3 (peak)
M1-M2			
Max Output Voltage (V)	6	9	12
Maximum continous output current (A)	-	750	1000 (peak)

Tabla 2: Tabla de especificaciones MeAuriga

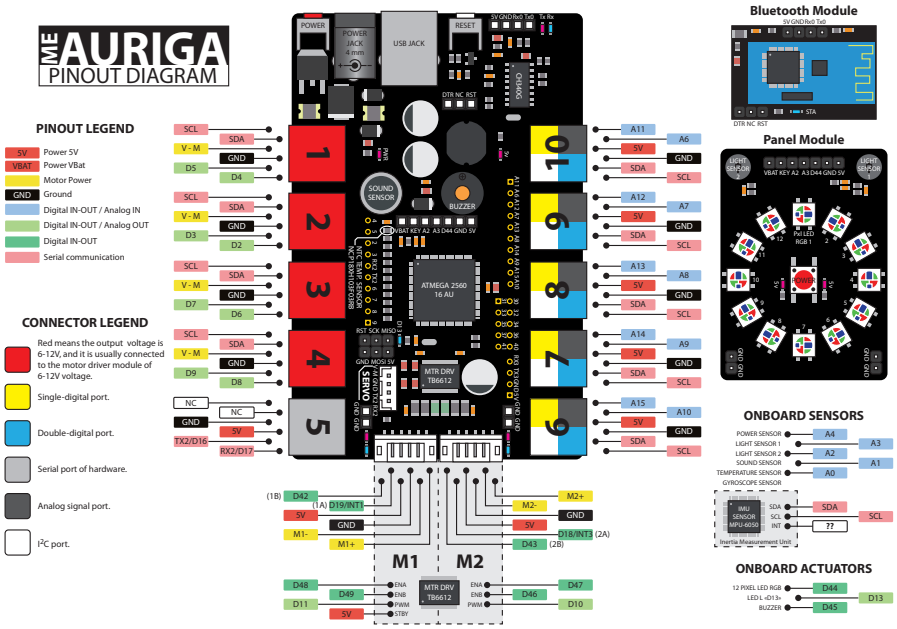


Figura 2: Tarjeta MeAuriga y diagrama de conexiones

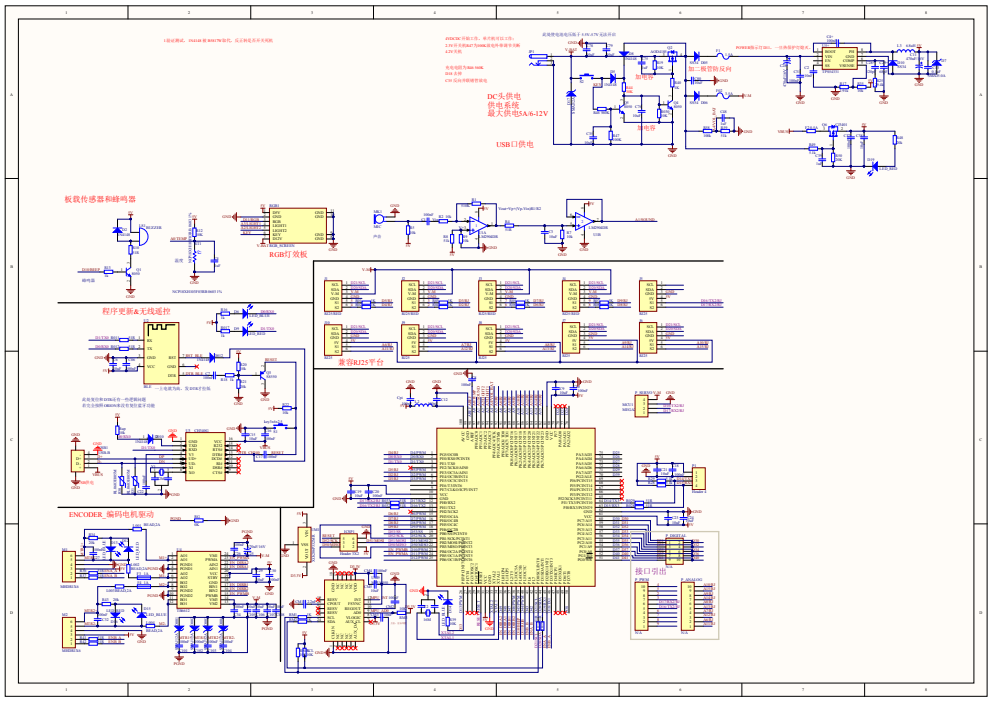


Figura 3: Esquemático MeAuriga

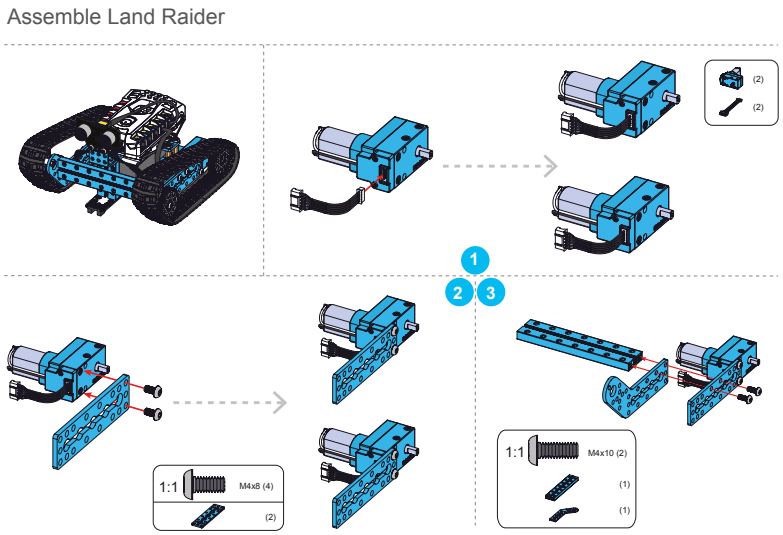
5. Ensamblaje y diseño

El empacado del kit esta altamente organizado, en este las piezas tienen un espacio a su medida en el cual ser guardadas o vienen dentro de bolsas herméticas como se muestra en la Figura 4 (a). Para el ensamblaje, el kit cuenta con un manual que muestra paso a paso lo que es necesario realizar para armar los robots. En la Figura 4 (b) se ve un ejemplo de las instrucciones de ensamblado, el diseño de estas es similar al utilizado por la empresa LEGO y tiene como foco principal su fácil entendimiento y replicación. Es a través de estas instrucciones que los robots puede ser armado con facilidad por niños y jóvenes en menos de 1 hora sin experiencia previa.

Las piezas de los robots y sus estructuras generales tienen múltiples consideraciones para mejorar la experiencia del usuario, varias de estas están pensadas para aumentar su durabilidad o facilidad de uso. Es gracias a estas consideraciones es que el mBot Ranger puede ejercer su propósito de ser un robot para niños y jóvenes y no fallar en el proceso. En particular, las consideraciones más importantes son:

- Tamaño de perforaciones estándar a lo largo de las piezas mecánicas, de manera que con un mismo tipo de tornillo/herramienta sea posible construir la mayoría de las partes de los robots.
- La conexión de cables y posición de los elementos del robot es bastante intuitiva, para las conexiones además se tiene un código de colores que clasifica a los sensores con sus puertos compatibles en la tarjeta. Además se utilizan cables recubiertos con goma o similares, para aumentar su resistencia.
- Detalles visuales para conseguir apariencia adorable, entre ellos, la simulación de un rostro y manos. Dentro de estos se encuentra: la posición del ultrasónico, el cual en todos los modelos juega el papel de ojos para simular un rostro.

(a) Piezas del empaque selladas



(b) Ejemplo de instrucción de armado

Figura 4: Empaque y ensamblaje del mBot Ranger

6. Sensores

El robot cuenta con una serie de sensores que pueden ser usados a través de la tarjeta MeAuriga para crear diferentes algoritmos y proyectos. Estos se separan en dos bloques principales, aquellos que vienen integrados en la tarjeta y los que se conectan externamente en forma de módulos electrónicos.

- *Light Sensor*/Sensor de luminosidad: Foto-transistor cuya señal corresponde a una medición del nivel de luminosidad en el entorno, está conectado a entradas analógicas `A2` y `A3` en el Arduino. Sus mediciones son directamente proporcionales al nivel de luz, donde una alta iluminación corresponde a un alto valor de voltaje. En condiciones normales, con la iluminación de una oficina, se obtienen valores cercanos a 300. Los datos de este se encuentran en la Tabla 3. Un ejemplo de código para la lectura de mediciones es el siguiente:

```
MeLightSensor lightsensor_12(12);      //Sensor de luz en puerto 12
int lux = lightsensor_12.read();       //Valor luminico
```

Figura 5: Light Sensor

Características primarias		
Categoría	Min	Max
Voltajes de operación (V)	0	5
Lectura MCU[(1)]	0	1023
Librería asociada	MeLightSensor	
mBlock	light sensor (portN)	
Ubicación	Integrado	

Tabla 3: Especificaciones sensor de luminosidad

[(1)]Valores medidos con la configuración predeterminada del ADC (Analog-to-Digital Converter)

- *Sound Sensor*/Sensor de sonido: Transductor capaz de transformar sonido en señales eléctricas, estas señales son de naturaleza analógica y son accesibles a través de la lectura del pin `A1` del Arduino. Las mediciones de este sensor no son lo suficientemente precisas como para ser utilizado como micrófono de voz, pero si permite el detectar niveles de volumen en el ambiente en donde un sonido fuerte es asociado a una alta medición de voltaje. En condiciones normales, con el ruido propio de una oficina, se obtienen valores cercanos a 150. Los datos de este se encuentran en la Tabla 4. Un ejemplo de código para la lectura de mediciones es el siguiente:

```
MeSoundSensor soundsensor_14(14);      //Sensor en el puerto 14
int soundLevel = soundsensor_14.strength();       //Valor sonoro
```

Figura 6: Sound Sensor

Características primarias		
Categoría	Min	Max
Voltaje de operación (V)	0	5
Lectura MCU[(1)]	0	1023
Librería asociada	MeSoundSensor	
mBlock	sound sensor (PortN)	
Ubicación	Integrado	

Tabla 4: Especificaciones sensor de sonido

[(1)]Valores medidos con la configuración predeterminada del ADC (Analog-to-Digital Converter)

- *NTC Thermoresistor*/Sensor de temperatura: Componente electrónico capaz de ser usado para mediciones de temperatura, su implementación con el termoresistor

NCP18XH103F03RB produce señales de naturaleza analógicas y se encuentra conectado al pin `A0` del Arduino. Este corresponde a un resistor cuyas propiedades de resistividad cambian con la temperatura, en especifico, a medida que aumenta la temperatura disminuye su resistencia. Para ser utilizado como sensor de temperatura esta conectado como parte de un divisor resistivo del voltaje de alimentación del microcontrolador, de esta manera a medida que aumenta la temperatura disminuye el voltaje medido por el Arduino. El resumen de las especificaciones de este sensor se encuentra en la Tabla 5. Un código de prueba para utilizar este termoresistor es el siguiente:

```
MeOnBoardTemp temperature_onboard(PORT_13); //Sensor en el PORT_13
temperature_onboard.readValue();            //temperatura en
   Celcius (C)
```

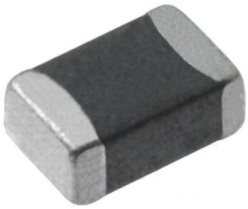


Figura 7: NTC Thermistor

Características primarias		
Categoría	Min	Max
Voltaje de operación (V)	0.01	4.55
Lectura MCU[(1)]	2	930
Rango de temperaturas[(2)] (°C)	-20	120
Librería asociada	MeOnBoardTemp	
mBlock	temperature on board °C	
Ubicación	Integrado	

Tabla 5: Especificaciones sensor de temperatura

[(1)]Valores medidos con la configuración predeterminada del ADC (Analog-to-Digital Converter)

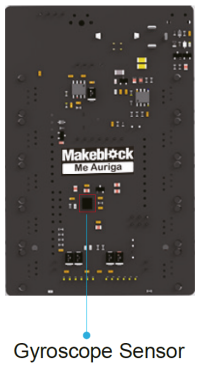
[(2)]Valores extrapolados a través de las curvas entregadas por el fabricante. El robot NO resiste todo este rango de temperaturas.

- *IMU Inertia Measurement Unit*/Unidad de medición de inercia: Circuito integrado que combina las funciones de un acelerómetro y un giroscopio, entrega su información a partir de un vínculo de comunicación serial I^2C el que está conectado a los pines `D21` y `D22` del Arduino (los mismos que se encuentran en los PORT1-PORT10 con RJ25). Su funcionamiento se basa en medir las proyecciones de la aceleración de gravedad g en los ejes del integrado y medir las velocidades angulares mediante un giroscopio, y así generar la información que es comunicada a través de I^2C al Arduino para su procesamiento. Los ángulos son calculados en las librerías para Arduino de mBlock a través de funciones y filtros respectivos al modelo de movimiento. El resumen de las especificaciones de este sensor se encuentra en la Tabla 6. Por último, un código de prueba para este sensor se muestra a continuación:

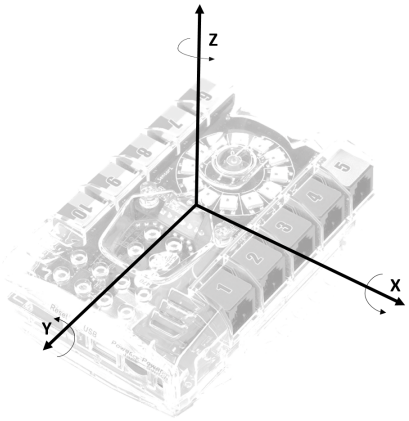
```
MeGyro gyro_0(0, 0x69); //PORT0, Address 0x69
gyro_0.begin();         //Comunicacion I2C con gyro
anguloX = gyro_0.getAngle(1);  //X: 1, Y:2, Z:3
```

Características primarias		
Giroscopio	**Min**	**Max**
Rango total (°/s)	± 250	± 2000
Factor de sensitividad de escala (LSB /(°/s))	16.4	131
Sensitividad de aceleración lineal	0.1 (°/s)/g	
Densidad espectral de ruido nominal	0.005 (°/s)/$\sqrt{Hz}$ @10Hz	
Acelerómetro	**Min**	**Max**
Rango total (g)	± 2	± 6
Factor de sensitividad de escala (LSB /g)	2048	16384
Densidad espectral de ruido nominal	400μ g/$\sqrt{Hz}$ @10Hz	
Resolución ADC	16 bits	
Mediciones de angulos	**Min**	**Max**
Ángulo X	-90°	90°
Ángulo Y	-90°	90°
Angulo Z	-180°	180°
Librería asociada	MeGyro	
mBlock	3-axis gyro on board (Coord-Axis) angle	
Ubicación	Integrado	

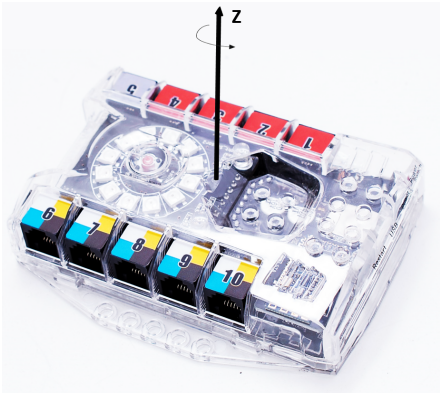
Tabla 6: Especificaciones giroscopio



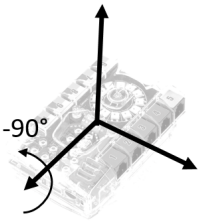
(a) Ubicación del integrado



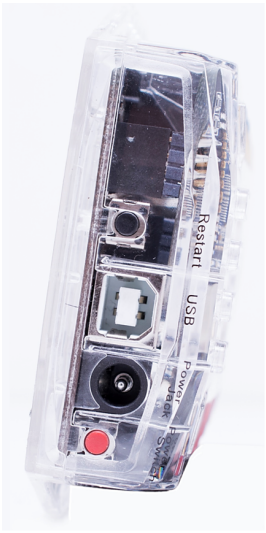
(b) Ejes de giro referenciados

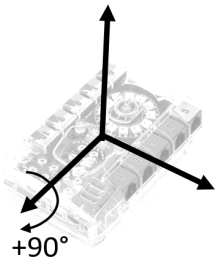


(c) Eje de giro Z, de -180° a 180°

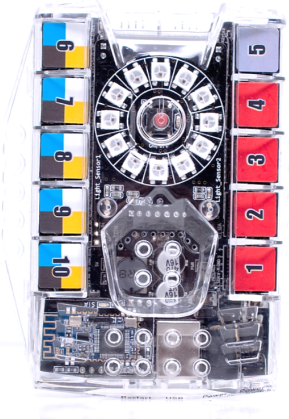


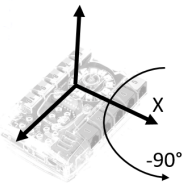
(d) Giro de -90°, eje Y



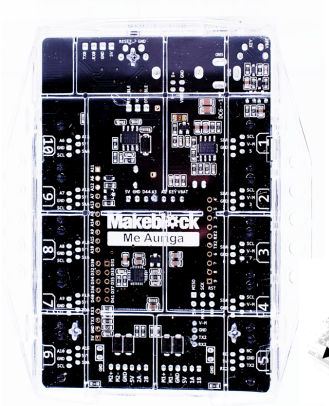


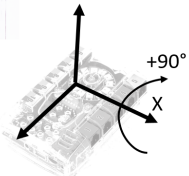
(e) Giro de +90°, eje Y





(f) Giro de -90°, eje X





(g) Giro +90°, eje X

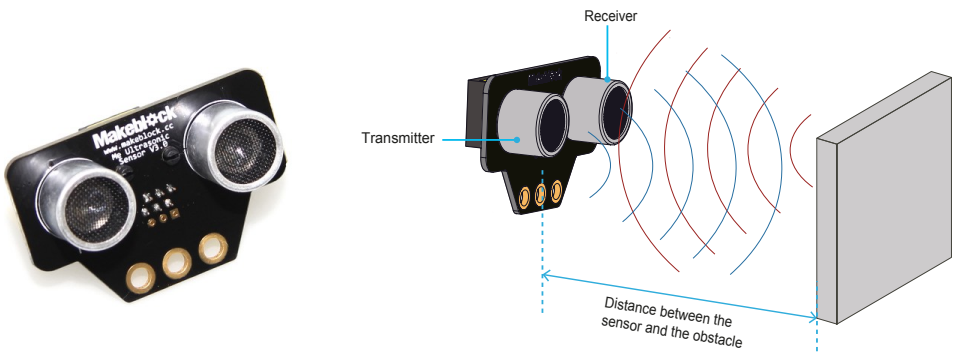
Figura 8: Giroscopio integrado en MeAuriga

- *Ultrasonic Sensor*/Sensor Ultrasónico: Dispositivo capaz de enviar y recibir ondas ultrasónicas, la información proveniente de este es de naturaleza digital y se obtiene del módulo electrónico externo `Me Ultrasonic Sensor`. En el programa por defecto del mBot Ranger, este requiere ser conectado al `PORT10` a través de un cable RJ25. Su método de funcionamiento corresponde al enviar una onda ultrasónica y luego recibir el eco de esta, con el tiempo transcurrido entre enviar el sonido y recibirlo se calcula la distancia hacia el objeto más cercano (se asumen 340m/s como velocidad del sonido para cálculos internos). Las especificaciones de este se encuentran en al Tabla 7. Las especificaciones de este se encuentran en la Tabla 7. Un ejemplo de código corresponde al siguiente, donde se guarda la medición de distancia en la variable "dist":

```
MeUltrasonicSensor ultrasonic_10(10);     //Ultra sonico en PORT10
int dist = ultrasonic_10.distanceCm();   //distancia medida en cm
```

Características primarias		
Categoría	Min	Max
Voltajes digitales (V)	0	5
Rango distancias(3) (cm)	3	4000
Frecuencia portadora	40 kHz	
Ángulo de medición(4)	30°	
Librería asociada	MeUltrasonicSensor	
mBlock	ultrasonic sensor (PORTn) distance	
Ubicación	Externo	

Tabla 7: Especificaciones módulo `Me Ultrasonic Sensor`



(a) Sensor externo (b) Modo de funcionamiento

Figura 9: Módulo `Me Ultrasonic Sensor`

[3]Los valores cercanos y superiores a los 100cm son poco confiables debido a la alta probabilidad de ecos adicionales causados por el entorno.

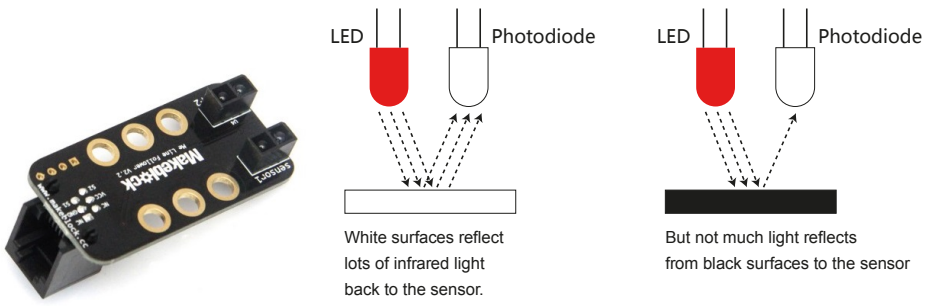
[4]El ángulo de incidencia máximo se ve afectado por al distancia hacia el objeto.

- *Line-follower sensor*/Seguidor de línea: Corresponde a un arreglo emisor-receptor de señales infrarojas, la información de este son señales de naturaleza digital y viene incluido en el módulo externo `Me Line Follower`. Este funciona a través de medir la diferencia de luz reflejada por superficies de diferentes colores, debido a la configuración digital del módulo de Makeblock, este solo puede diferenciar entre blanco y negro. El umbral utilizado en el módulo es tal que la tolerancia con respecto al negro es bastante baja, es decir, solo superficies bien opacas serán interpretadas como negro. La medición del sensor es entregada a través de una variable `uint_8` cuyos 2 bits menos significativos representan el estado actual de los receptores, por ejemplo, el estado IZQ = Blanco y DER = Negro se representa como $state = S_1S_2 = (10)_2 = 4$. Un código de prueba corresponde al siguiente:

```
MeLineFollower linefollower_9(9);              //sensor en PORT9
uint_8 state = linefollower_9.readSensors();   //estados: (S1S2)
```

Características primarias		
Categoría	Min	Max
Voltajes digitales (V)	0	5
Color detectado[5]	Negro	Blanco
Librería asociada	MeLineFollower	
mBlock	line follower (PORTn) (Side) is (color)	
Ubicación	Externo	

Tabla 8: Especificaciones módulo `Me Line Follower`



(a) Sensor externo (b) Modo de funcionamiento

Figura 10: Módulo `Me Line Follower`

[5] El color a detectar debe ser opaco para su correcto funcionamiento. Por ejemplo, el negro de un plumón permanente suele ser demasiado brillante como para ser interpretado como negro.

7. Actuadores

- *Buzzer*/Zumbador: Buzzer pasivo, elemento piezo eléctrico que se encarga de producir sonido de acuerdo a la señal que se le envíe y está conectado a la salida digital `D5` del Arduino. Debido a ser del tipo pasivo, para su funcionamiento, se necesita enviar una señal PWM por el pin digital con la frecuencia deseada a sonar en el buzzer. En la Tabla 9 se presentan los datos del Buzzer. Un ejemplo de código es el siguiente, donde se toca una nota <La> durante 250 milisegundos:

```
MeBuzzer buzzer;
int freq = 440;          //en Hz
int duration = 250;      //en ms
buzzer.tone(freq,duration);   //toca la nota <La> durante 250ms
```

Características primarias		
Categoría	Min	Max
Voltajes digitales (V)	0	5
Frecuencias de operación (Hz)	50[6]	2500
Librería asociada	MeBuzzer	
mBlock	play tone on note (Name) beat (Fraction)	
Ubicación	Integrado	

Tabla 9: Especificaciones Buzzer

[6]Las notas con frecuencias inferiores a 200Hz presentan un alto nivel de frecuencias armónicas que distorsionan la señal.

- LED RGB: Diodo emisor de luz, corresponden a un arreglo de 3 diodos de diferente color (R,G,B) cuya suma de colores produce el color final resultante en el LED. En particular, se utiliza un arreglo de LEDs RGB provenientes del componente WS2812 para controlar varios LEDs con un solo pin de información y este se encuentra conectado a la salida `D13/SCK` del Arduino. Debido a esto ultimo se requiere enviar la información a los LEDs de forma serial con el protocolo NZR, con este se deben enviar bloques de 24-bit con la configuración de cada componente de color (en el orden GRB). Las especificaciones se encuentran en la Tabla 10 y mayor detalle sobre la comunicación con estos componentes en `https://cdn-shop.adafruit.com/datasheets/WS2812.pdf`. Por último un código de ejemplo de su uso:

```
MeRGBLed rgbled_7(7, 7==7?2:4);   //LEDs RGB por defecto
rgbled_7.setColor(0,60,15,0);     //R=60, G=15, B=0 (board)
rgbled_7.show();
```

Características primarias	Especificación		
Categoría	Min	Typ	Max
Voltajes alimentación (V)	0	5	7
Corriente directa (Vdd)	0.1uA	-	18.5 mA
Corriente comunicación (D_{OUT})	-	10mA	-
Cantidad de colores	16.777.216		
Frecuencia comunicación	800 kHz		
Dimensiones	50x50mm		
Librería asociada	MeRGBLed		
mBlock	set led on board (Side) red (value) green (value) blue (value)		
Ubicación	Integrado		

Tabla 10: Especificaciones LED RGB

(a) RGB LED (b) Buzzer

Figura 11: Actuadores integrados

- *Encoder Motor*/Motor con Encoder: Motores de corriente continua, funcionan a través de ser alimentados con un determinado valor de voltaje continuo entre sus terminales y producir un torque proporcional a dicha alimentación, además incluyen encoders ópticos conectados a sus ejes principales lo cual permite monitorear la posición actual del eje. La dinámica de los encoder se basa en detectar el paso de luz a través de determinadas ranuras, a medida que el rotor va girando, con esta información se pueden calcular velocidades y posiciones absolutas de los ejes de los motores. Debido a trabajar

con corrientes mayores a las que maneja el microcontrolador requiere de drivers para su uso (integrados), de esta manera su conexión corresponde a salidas PWM del Arduino. La señal PWM *(Pulse Width Modulated)* utilizada debe tener como ciclo de trabajo el porcentaje del voltaje total que se desea utilizar para los motores. Los datos técnicos de estos motores se encuentran en la Tabla 11 y un ejemplo de código para utilizar estos motores es el siguiente:

```
MeEncoderOnBoard Encoder_1(SLOT1);
MeEncoderOnBoard Encoder_2(SLOT2);

void isr_process_encoder1(void) {   //Interrupcion encoder
  if(digitalRead(Encoder_1.getPortB()) == 0){
        Encoder_1.pulsePosMinus();
  }
  else{
        Encoder_1.pulsePosPlus();
  }
}

void isr_process_encoder2(void) {
  if(digitalRead(Encoder_2.getPortB()) == 0){
        Encoder_2.pulsePosMinus();
  }
  else{
        Encoder_2.pulsePosPlus();
  }
}

void moveDegrees(int direction,long degrees, int speed_temp)
{
  speed_temp = abs(speed_temp);
  if(direction == 1){
        Encoder_1.move(-degrees,(float)speed_temp);
        Encoder_2.move(degrees,(float)speed_temp);
  }
  else if(direction == 2){
        Encoder_1.move(degrees,(float)speed_temp);
        Encoder_2.move(-degrees,(float)speed_temp);
  }
  else if(direction == 3){
        Encoder_1.move(-degrees,(float)speed_temp);
        Encoder_2.move(-degrees,(float)speed_temp);
```

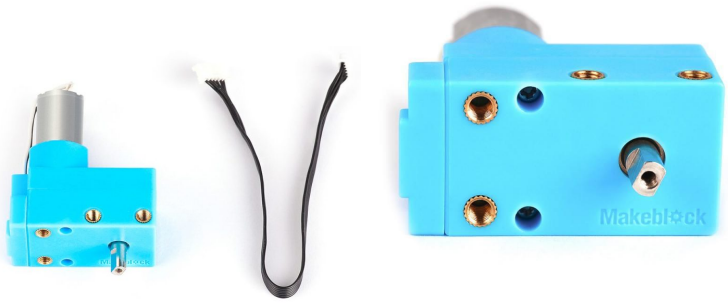
```
  }
  else if(direction == 4){
        Encoder_1.move(degrees,(float)speed_temp);
        Encoder_2.move(degrees,(float)speed_temp);
  }
}

void setup(){
    TCCR1A = _BV(WGM10);
    TCCR1B = _BV(CS11) | _BV(WGM12);
    TCCR2A = _BV(WGM21) | _BV(WGM20);
    TCCR2B = _BV(CS21);
    attachInterrupt(Encoder_1.getIntNum(), isr_process_encoder1,
   RISING);
    attachInterrupt(Encoder_2.getIntNum(), isr_process_encoder2,
   RISING);
    Encoder_1.setPulse(9);
    Encoder_1.setRatio(39.267);
    Encoder_1.setPosPid(1.8,0,1.2);
    Encoder_1.setSpeedPid(0.18,0,0);
    Encoder_2.setPulse(9);
    Encoder_2.setRatio(39.267);
    Encoder_2.setPosPid(1.8,0,1.2);
    Encoder_2.setSpeedPid(0.18,0,0);
}

void loop(){
    moveDegrees(1,1000,100);    //Avanza 1000 grados a 100RPM
}
```

Categoría	Especificación
Ratio de reducción	39,6:1
Tensión nominal	7.4V
Corriente SIN carga	≤240mA
Corriente CON carga	≤750mA
Rapidez máxima sin carga	350 ± 5 % RPM
Velocidad con carga nominal	178 ± 10 % RPM
Velocidad nominal	14000 ± 5 % RPM
Par de carga nominal	800 g.cm
Par de ruptura	5 kg.cm
Eje de salida	9 mm
Potencia	3.7W
Precisión del encoder	360 pasos
Librería asociada	MeEncoderOnBoard
mBlock	(Dir) (Grados) degrees at the speed of (Speed) rpm
Ubicación	Externo

Tabla 11: Especificaciones Encoder Motor



(a) Motor con cable para conexión

(b) Zoom al eje del motor

Figura 12: 180 Encoder Motor

8. Modelos Base

El kit mBot Ranger esta diseñado para poder transformarse en 3 robots con funcionalidades diferentes, cada uno de estos con un programa y/o aplicaciones de prueba.

8.1. Land Raider

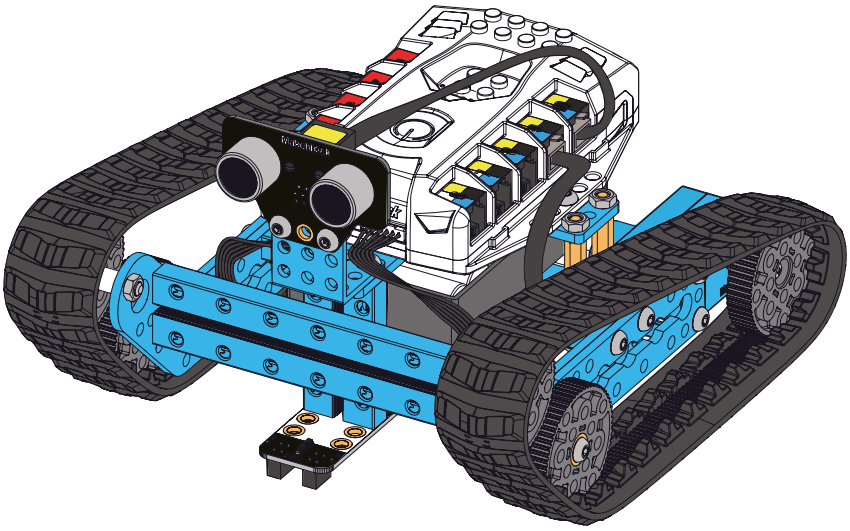


Figura 13: Land Raider

El Land Raider es un robot diseñado para ser todo terreno al utilizar orugas tipo tanque en sus ruedas. Su apariencia, ver Figura 13, hace referencia al robot Wall-E de la famosa película de Disney-Pixar con su nombre y consiste en la de un robot explorador de terrenos inhóspitos, como lo son algunos robots utilizados en misiones a Marte y zonas desérticas.

Sus características principales corresponden a la de ser un vehículo todo terreno gracias al uso de orugas de tanque como superficie de contacto. Este modelo presenta una velocidad de movimiento intermedia-alta y el mayor peso de los 3. En el modo todo terreno se recomienda evitar la instalación del modulo `Me Line Follower` como se presenta en el manual debido al riesgo de que sufra daños con el terreno. Dentro de lo que viene programado por defecto para el Land Raider se encuentran un varios modos accesibles a través de las secciones de la Figura 17(a)-(b), entre estos se destacan los modo Manual, modo Esquiva-obstáculos y modo Seguidor de línea.

8.2. Dashing Raptor

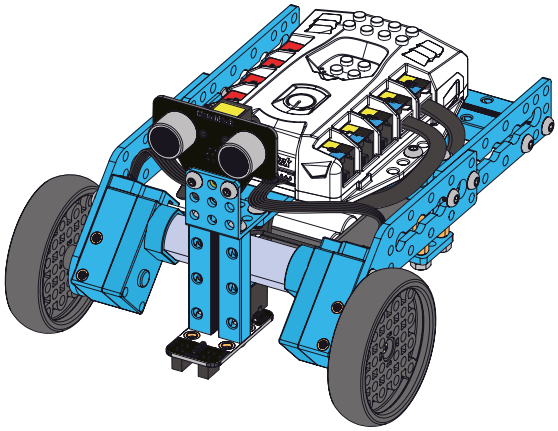


Figura 14: Dashing Raptor

El Dashing Raptor es un robot diseñado para ocupar el máximo potencial de los motores del kit mBot Ranger. En este se reemplazan las orugas por ruedas convencionales en la parte delantera como fuente de tracción, además se utiliza un pivote en la parte trasera en vez de ruedas para conseguir menor roce y así alcanzar mayores velocidades. Su uso principal corresponde al de un vehículo de carreras, con una velocidad alta y un peso intermedio con respecto a los 3 modelos ofrecidos. Por defecto, para el Dashing Raptor se encuentra programado un modo Manual, accesible a través de la sección "Coche Racing" de la App.

8.3. Nervous Bird

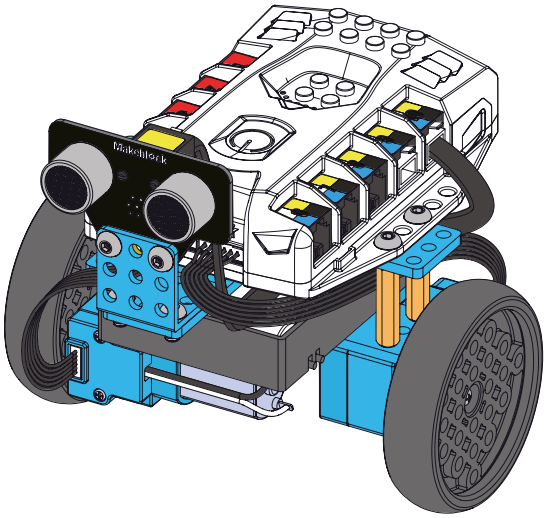
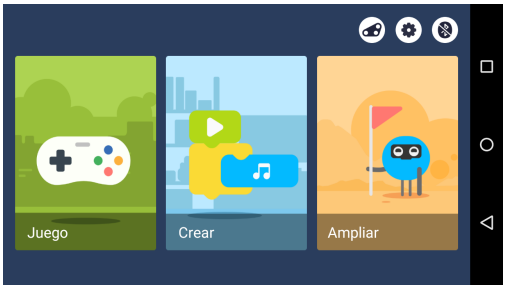


Figura 15: Nervous Bird

El Nervous Bird es un robot diseñado para demostrar las capacidades del IMU [6] integrado en el robot y hacer posible el usar una estructura con solo 2 puntos de apoyo (ruedas). La funcionalidad principal de este modo es el auto-balanceo, este consiste en la mantención del equilibrio del robot a través de cálculos internos para un controlador PID. Esta capacidad de equilibrio es accesible a través de la sección ”Pájaro Loco” de la App.

9. Makeblock App

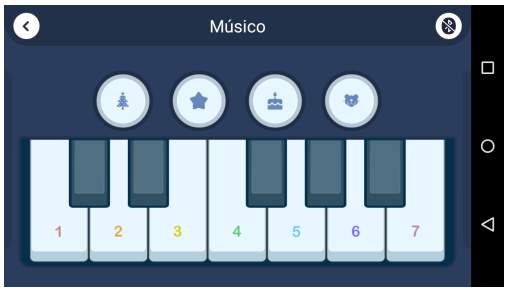
El mBot Ranger viene programado para ser utilizado a través de la Makeblock App. Para el uso demostrativo de sus capacidades, mBot Ranger tiene acceso a varias secciones dentro de la categoría ”Juego”. En estas se presentan ejemplos de lo que puede hacer el mBot Ranger a través de interfaces didácticas e intuitivas, ejemplos de estas se pueden ver en la Figura 16. En (a) se tiene control del movimiento de manera gradual a través de lo que simula ser un mando analógico, además de tener botones con secuencias de movimientos como lo son ”correr”, ”dar vueltas” y ”terremoto”. En (b) se presenta una pantalla para el dibujo de trayectorias, gracias al uso de motores con encoder el mBot Ranger es capaz de recorrer trayectorias similares a las dibujadas en esta sección con ciertos margenes de error. En (c) se muestra un piano musical con el cual se pueden tocar diferentes notas en el mBot Ranger, además de tener canciones de muestra que están pre-programadas en el robot (entre estas están Estrellita, Navidad, Feliz Cumpleaños y Fray Jacobo).



(a) Menú Principal

(b) Movimiento y secuencias

(c) Dibujo de trayectorias



(d) Piano musical

Figura 16: Zona de juegos

El listado de las secciones a las que se tiene acceso en el menú "Juego" se pueden ver en la Figura 17 (a)-(c). Además, en la sección "Ampliar" se encuentran programas de prueba para lo modelos armables con los packs de complementos (*Add-on*), como se muestra en Figura 17 (d). Los diferentes modos de operación disponibles a través de estas secciones, son los siguientes:

- Manual: En esta modalidad se puede manejar el robot a través de los bloques de movimiento ofrecidos en la interfaz de la Figura 17 (d). Con esto se puede manejar el robot armado con un relativo control sobre la velocidad de movimiento.
- Esquiva-obstáculos: En esta el robot comienza a moverse de forma automática y esquiva obstáculos en frente suyo haciendo uso del sensor ultrasónico. El algoritmo de evasión corresponde a retroceder cuando se detecta un objeto a menos de 10cm de distancia.
- Seguidor de línea: En este modo el mBot Ranger empieza a avanzar en línea recta hasta que encuentra una línea negra (o similar) en el suelo y la comienza a seguir. La naturaleza de este modo requiere que haya un buen nivel de contraste entre el camino (linea negra) y el fondo (preferentemente blanco). Con este modo el robot puede seguir caminos arbitrarios dibujados por el usuario, por ejemplo, con plumón negro sobre una cartulina.
- Autobalanceo: En este el robot procede a mantener el equilibrio en torno a la referencia $\theta_x = 0$. El algoritmo utilizado para este objetivo es calcular la velocidad de los motores con a través de un controlador PID. Debido a la naturaleza de este modo, la velocidad máxima de movimiento es relativamente baja para permitir al controlador un mayor rango de acción.

Finalmente, está el acceso al menú "Crear" en el cual se puede entrar a lo que es la programación por bloques, junto al probar los sensores y actuadores a mayor cabalidad. Esto se puede ver en la Figura 18.

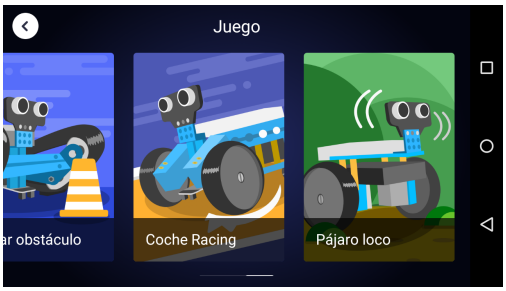
10. Consideraciones

A continuación se presentan algunas consideraciones a tener en mente al momento de utilizar el robot mBot:

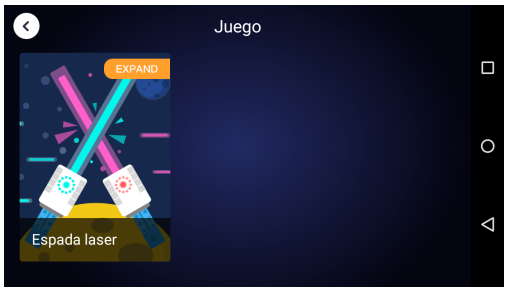
- Ensamblado: Seguir las instrucciones con cuidado, no es difícil equivocarse en la posición en que se debe colocar alguno de los tornillos. Lo cual puede que no presente un cambio
- Estructura: Las baterías estan ubicadas de manera que su impacto sea lo menos posible sobre la estética y el centro de gravedad del robot, esto trae una desventaja al momento de que estas se descarguen, pues para poder acceder a ellas y cambiarlas es necesario desarmar el robot hasta cierto nivel.

(a) Secciones dedicadas al Land Raider 1

(b) Secciones dedicadas al Land Raider 2



(c) Dashing Raptor y Nervous Bird



(d) Aplicaciones con Add-on

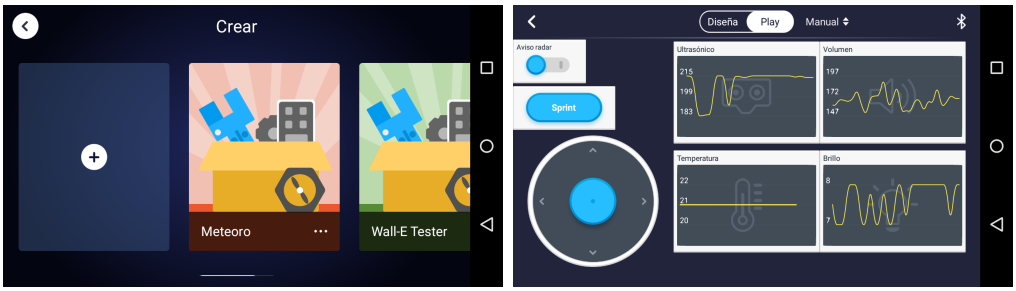
Figura 17: Secciones dedicadas al mBot Ranger

- Alimentación: Principalmente debido a los motores del kit, el mBot Ranger tiene consumo de energía superior a su predecesor mBot tanto así que NO se recomienda el uso de baterías alcalinas, pues estas tendrían una corta duración y al largo plazo esto implicaría un mayor gasto en reposiciones.

- Modo manual: Las instrucciones de ($\rightarrow$) y ($\leftarrow$) a veces son confundidas.

- Modo esquiva-obstáculos: Debido al algoritmo de esquive, el robot tiende a quedar estancado en un ciclo de retroceso-avance al momento de encontrar un obstáculo estático (como una pared). Además mientras el mBot Ranger esta girando en este modo no es posible enviarle otras instrucciones, esto puede provocar que quede atascado entre un conjunto de obstáculos y sea necesario extraerlo de manera manual del sitio. Requiere de que el obstáculo tenga una altura y ancho mínimos para poder ser detectados, debido a la forma y posición del sensor ultrasónico (alrededor de unos 6$\sim$7cm sobre el suelo).

- Seguidor de línea: Requiere de un grosor mínimo para el camino utilizado ($\sim$3cm), que corresponde al ancho del módulo `Me Line Follower` utilizado. Además requiere de que el camino tenga una buena curvatura para ser capaz de seguirla correctamente, el dibujo de un ∞ que viene incluido esta cerca del borde de las limitaciones del modelo Land Raider [8.1]. El modelo Dashing Raptor [8.2] es demasiado rápido para que el algoritmo de seguimiento de línea que viene programado por defecto sea capaz de



(a) Menú Crear (b) Interfaz de creación

Figura 18: Secciones dedicadas a la programación

mantener al robot dentro de una curva como la del ∞ disponible en los ejemplos.

- Auto-balanceo: Es en este modo en que se nota más el efecto de uso de *delays* y los parámetros utilizados en el controlador base, esto se debe a que en general el robot tiene problemas para mantener el equilibrio de manera estática por periodos prolongados. Si bien este comportamiento pareciere ser deseado, y por el nombre de "pájaro loco" para el modelo de equilibrio, las oscilaciones que presenta son incrementales con lo cual tiende a diverger. A pesar de esto, presenta una estabilidad bastante buena cuando el robot está en movimiento.