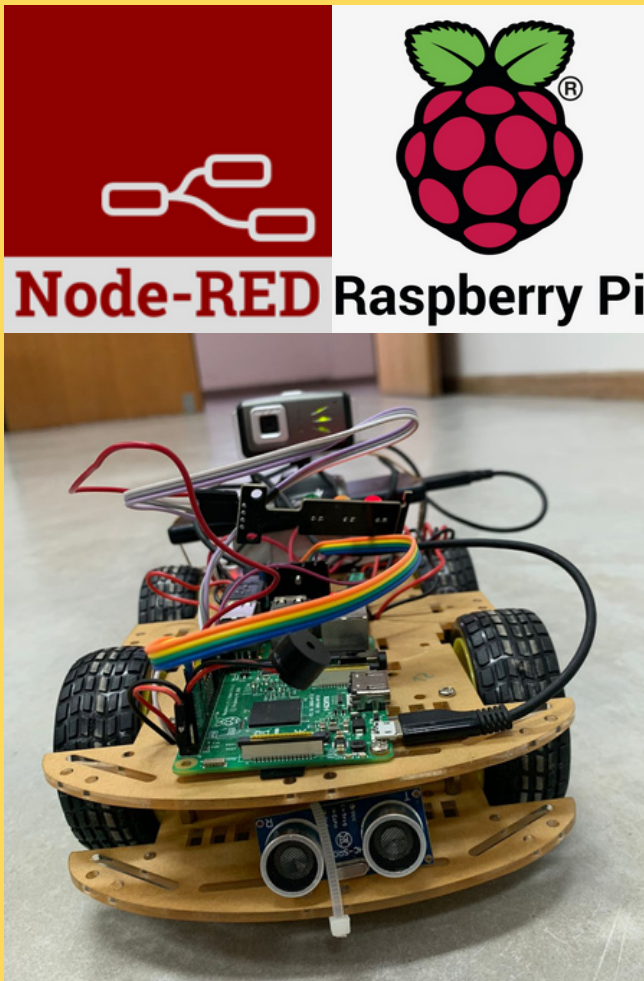


DICIEMBRE 2022

CLOUD COMPUTING. CLOUD ROBOTICS.

ACTIVIDAD INTEGRADORA

Raspberry Pi – Node-Red



PRESENTADO A

Rodríguez Ismael

Rodríguez Uguren Sebastián

PRESENTADO POR

Grupos 3 y 4

Guadalupe Evequoz

Joaquín Spinelli

Juan Cruz Verdolotti

Pablo Gagliardi

Tabla de contenido

	<u>página</u>
Consigna.....	3
Flujo de node-red.....	4
Circuito.....	5
Ambiente de desarrollo.....	6
Interfaz de dashboard.....	10
Servidor de cámara web.....	20

Enlace al video del funcionamiento:

https://www.canva.com/design/DAFS4RoVubA/rSDDAI_OI8uWxv8rloJRBQ/watch?utm_content=DAFS4RoVubA&utm_campaign=designshare&utm_medium=link&utm_source=publishsharelink

Actividad Integradora

INFORME DE ACTIVIDAD

- Se deberá entregar un documento en formato PDF, en el que se explique detalladamente, los pasos que se han seguido para resolver las actividades propuestas.

Dicho documento, debe incluir las capturas de pantalla necesarias en las que se pueda ver el trabajo del alumno.

- Deberá mostrar los flujos realizados en Node-Red (con capturas es suficiente) con una breve explicación de estos y su funcionamiento. Que nodos se utilizaron y porqué.

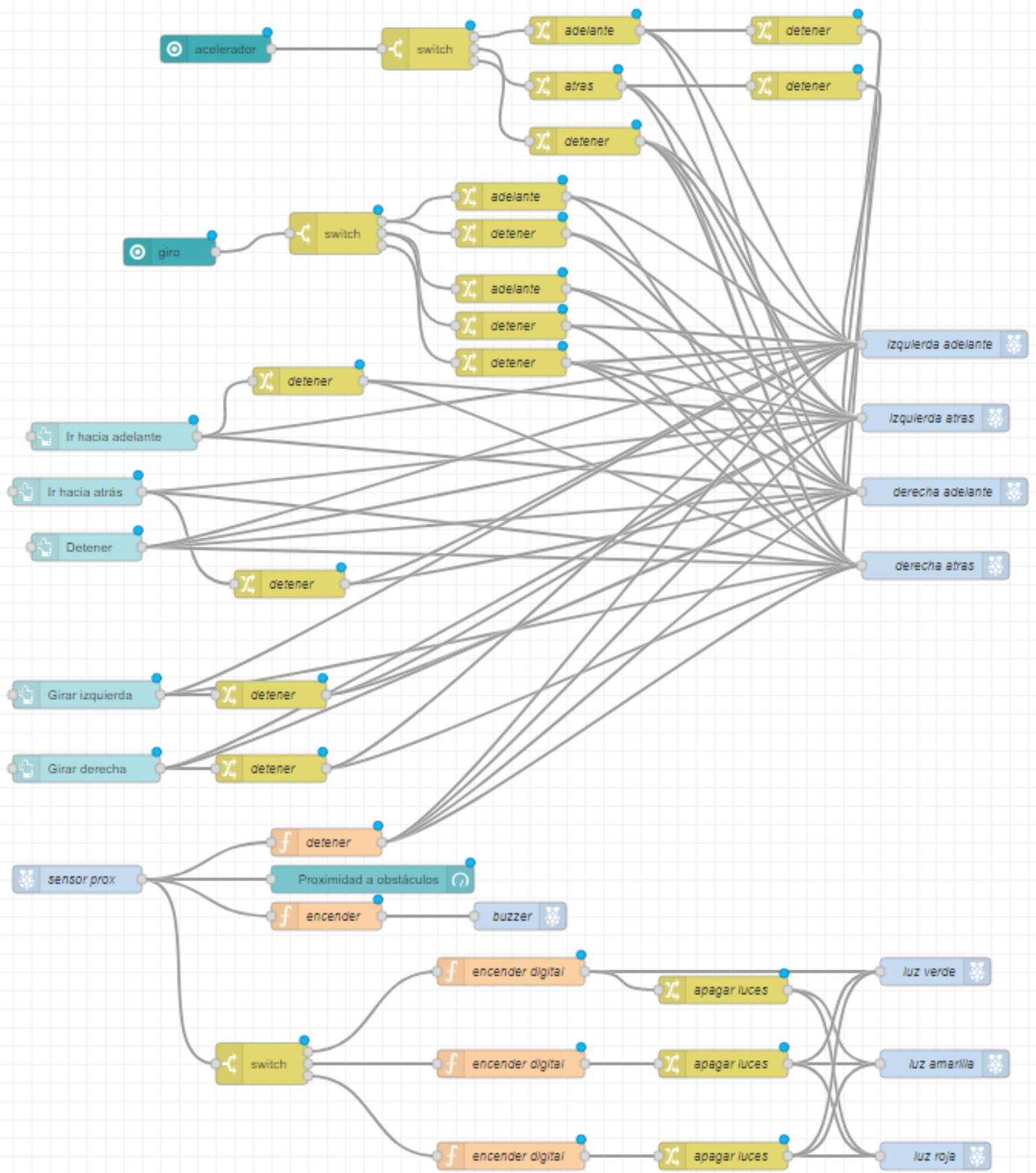
- Puede contener imágenes (jpg, gif, etc) o videos (subir videos a drive o youtube) donde se vea claramente los cambios de estado de los actuadores controlados, y del Dashboard con los valores registrados por los sensores y sus variaciones.

- La fecha límite de entrega del trabajo es hasta el día 03/12 a las 00.00Hs.

ACTIVIDAD PROPUESTA

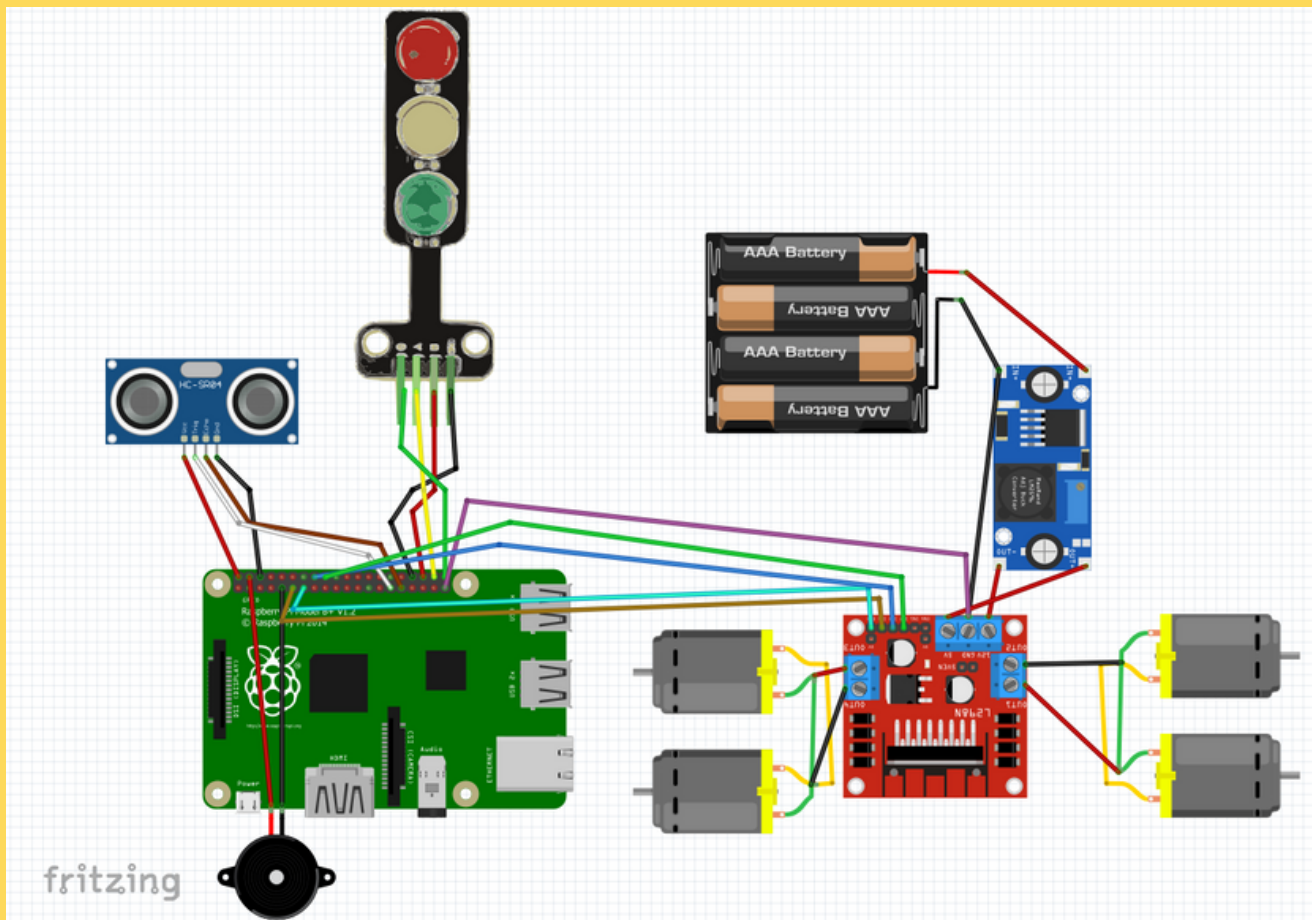
Con los conocimientos adquiridos con las actividades y los talleres realizados, hacer una integración sobre el robot dispuesto por la cátedra, de sensores y actuadores conectados a la Raspberry, con Node-Red y su Dashboard. El objetivo será hacer crecer lo más posible la funcionalidad del robot, logrando que el mismo adquiriera comportamiento "inteligente".

Flujo completo:



Debido a lo complejo que es el flujo, el mismo será explicado en partes (correspondientes a funciones) para facilitar su entendimiento.

Circuito



Las conexiones utilizadas son las mostradas en la imagen correspondiendo a:

- **Sensor de proximidad HC-SR04:** Pin 2 (5v), pin 6 (GND), pin 29 (GPIO 05), pin 31 (GPIO 06).
- **Semáforo:** Pin 34 (GND), 36 (GPIO 16), 38 (GPIO 20) y 40 (GPIO 21).
- **Buzzer:** Pin 3 (GPIO 02) y 9 (GND).
- **Módulo L298N puente H:** Pin 11 (GPIO 17), 13 (GPIO 27), 16 (GPIO 23), 18 (GPIO 24) y 39 (GND).
- **Fuente de alimentación conmutada StepDown D-SUN**
- **Motores de ruedas**
- **Baterías**

Ambiente de desarrollo

Node-RED-dashboard

Utilizamos este paquete para realizar la interfaz de nuestro sistema.

Node-RED-node-pi-gpio

Utilizamos este paquete para poder interactuar con nuestra placa Raspberry PI desde Node-RED, nos permite utilizar nodos acorde a la estructura admitida con Raspberry.

Node-RED-node-pisrf

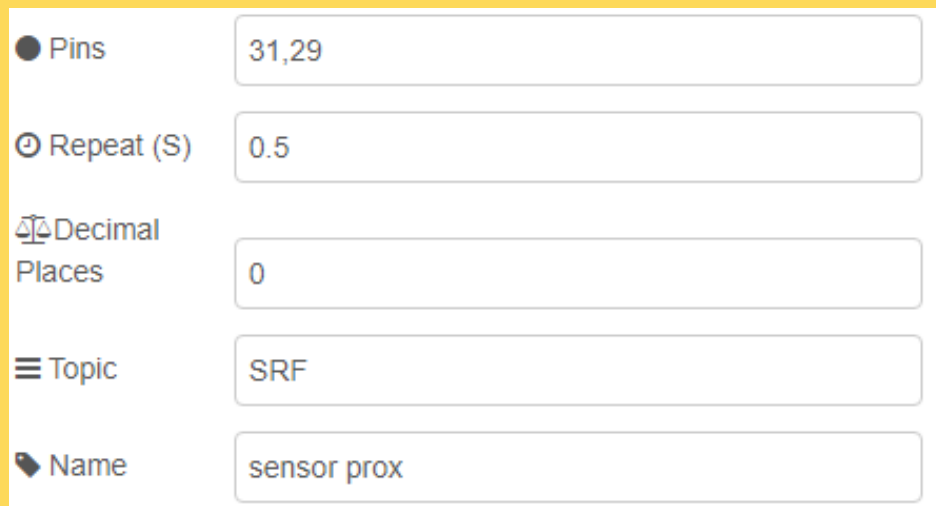
Utilizamos este nodo para poder interactuar con el sensor ultrasónico que conectamos a la Raspberry, puesto que el paquete de GPIO no tiene esta opción.

Node-RED-contrib-ui-joystick

Utilizamos este paquete poder agregar un nodo joystick para manejar la dirección de nuestro autito, dicho nodo se encuentra reflejado en el dashboard.

Para lograr conectar Node-RED con nuestro circuito debemos agregar nodos de entrada u salida para cada uno de los sensores.

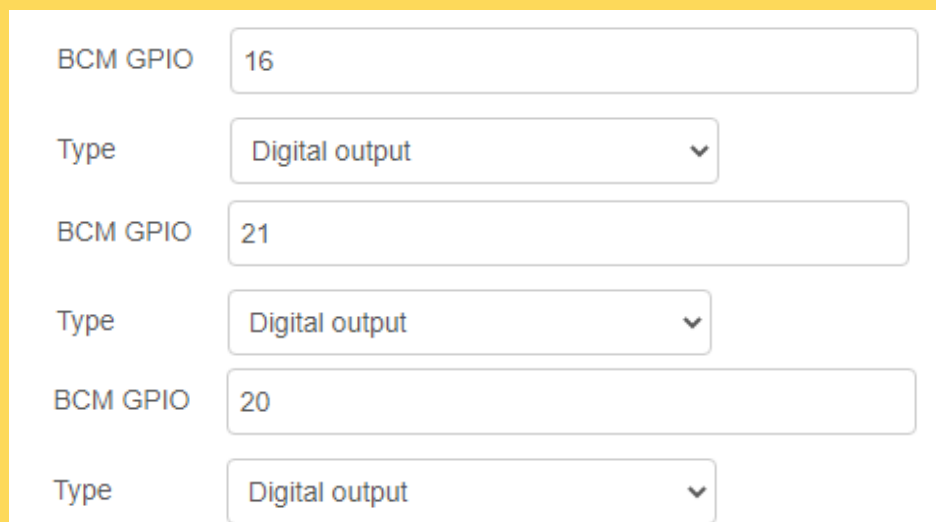
- **Sensor de proximidad:** Lo configuramos en los pines 29 y 31. El mismo captará cada 0,5 segundos las señales. Funciona como un nodo de entrada de datos.



The image shows the configuration interface for a 'Pins' node in Node-RED. It features five input fields with the following values: 'Pins' is set to '31,29'; 'Repeat (S)' is set to '0.5'; 'Decimal Places' is set to '0'; 'Topic' is set to 'SRF'; and 'Name' is set to 'sensor prox'.

Pins	31,29
Repeat (S)	0.5
Decimal Places	0
Topic	SRF
Name	sensor prox

- **Semáforo:** Lo configuramos en los pines 36 (Rojo), 38 (Amarillo) y 40 (Verde) . El mismo funciona como nodo de salida de datos digital.



The image shows the configuration interface for a 'Digital output' node in Node-RED. It displays three separate configurations for different pins. Each configuration consists of a 'BCM GPIO' field and a 'Type' dropdown menu. The first configuration has BCM GPIO 16, the second has BCM GPIO 21, and the third has BCM GPIO 20. All three 'Type' dropdowns are set to 'Digital output'.

BCM GPIO	Type
16	Digital output
21	Digital output
20	Digital output

- **Buzzer:** Lo configuramos en el pin 3. El mismo recibe un valor de frecuencia de sonido dependiendo la proximidad. Funciona como un nodo de salida de datos.

BCM GPIO	<input type="text" value="2"/>
Type	<input type="text" value="PWM output"/>

- **Ruedas izquierdas (hacia adelante):** Se configuro en el pin 11. Funciona como salida de datos y recibe un número que indica la velocidad a la que se moverán las dos ruedas izquierdas hacia adelante.

BCM GPIO	<input type="text" value="17"/>
Type	<input type="text" value="PWM output"/>
Frequency	<input type="text" value="100"/> Hz

- **Ruedas izquierdas (hacia atrás):** Se configuro en el pin 13. Funciona como salida de datos y recibe un número que indica la velocidad a la que se moverán las dos ruedas izquierdas hacia atrás.

BCM GPIO	<input type="text" value="27"/>
Type	<input type="text" value="PWM output"/>
Frequency	<input type="text" value="100"/> Hz

- **Ruedas derechas (hacia adelante):** Se configuro en el pin 16. Funciona como salida de datos y recibe un número que indica la velocidad a la que se moverán las dos ruedas derechas hacia adelante.

The screenshot shows the configuration for a Raspberry Pi GPIO pin. The 'BCM GPIO' field is set to 23. The 'Type' dropdown menu is set to 'PWM output'. The 'Frequency' field is set to 100 Hz.

- **Ruedas derechas (hacia atras):** Se configuro en el pin 18. Funciona como salida de datos y recibe un número que indica la velocidad a la que se moverán las dos ruedas izquierdas hacia atrás.

The screenshot shows the configuration for a Raspberry Pi GPIO pin. The 'BCM GPIO' field is set to 24. The 'Type' dropdown menu is set to 'PWM output'. The 'Frequency' field is set to 100 Hz.

- **Función Detener:** Si la proximidad es menor a 20, envía 0 a las ruedas para detener el auto. Si es mayor a 20, no retorna nada y descarta el mensaje.

The screenshot shows the configuration for a 'Function' node in Node-RED. The 'Name' field is set to 'detener'. The 'On Message' tab is selected. The code in the editor is as follows:

```
1 if (msg.payload < 20) {  
2   msg.payload = 0;  
3   return msg;  
4 }
```

- **Change:** Precisabamos un nodo change porque el joystick devuelve un objeto completo con posición, angulos, etc.

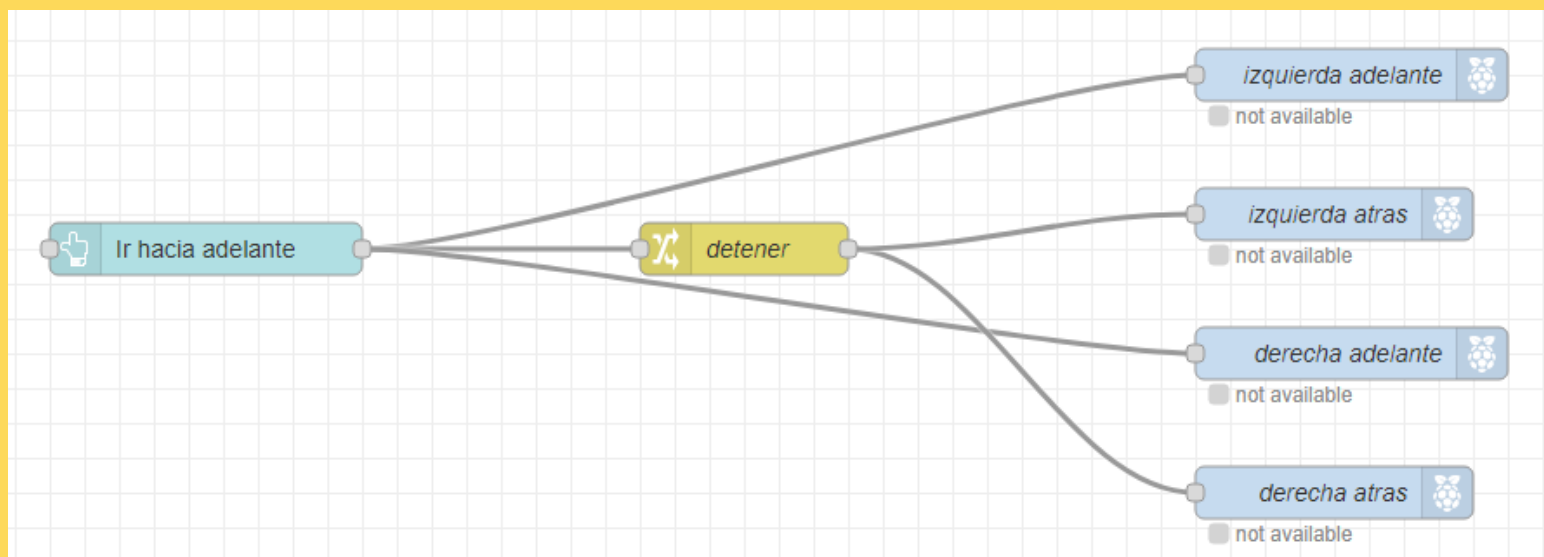
Interfaz de dashboard

Comenzamos agregando nodos dashboard para realizar nuestra interfaz de interacción con el auto. Para el correcto funcionamiento debemos crear y asignar el grupo de los nodos.



- **Gauge:** El nodo Gauge configurado como "Donut", indicará en su centro la distancia detectada por el sensor de proximidad en un rango de 0 a 100 y variará su color dependiendo de lo censado.
- **Joystick "Acelerador":** El nodo Joystick "acelerador" hace lo mismo que los botones ir hacia adelante o atrás, agregándole una nueva característica: cuando deja de accionarse el joystick, vuelve al centro y el auto se detiene. Este joystick en particular acepta solo ser accionado de forma vertical.
- **Joystick "Giro":** El nodo Joystick "giro" hace lo mismo que los botones girar derecha o izquierda, agregándole una nueva característica: cuando deja de accionarse el joystick, vuelve al centro y el auto se detiene. Este joystick en particular acepta solo ser accionado de forma horizontal.
- **Botones de dirección.**

Botón ir hacia adelante:



El botón "Ir hacia adelante" envía el valor 35 a los pines 11 y 16, y envía 0 a los pines 13 y 18.

La configuración se repite de igual forma para los otros botones.

Los botones hacen que el robot se mueva hasta que se le indique hacer otra cosa o se acerque demasiado a un obstáculo.

El nodo switch "detener" también está configurado de igual manera para los botones.

Group: [Home] Default

Size: auto

Icon: optional icon

Label: Ir hacia adelante

Tooltip: optional tooltip

Color: optional text/icon color

Background: optional background color

☒ When clicked, send:

Payload: 35

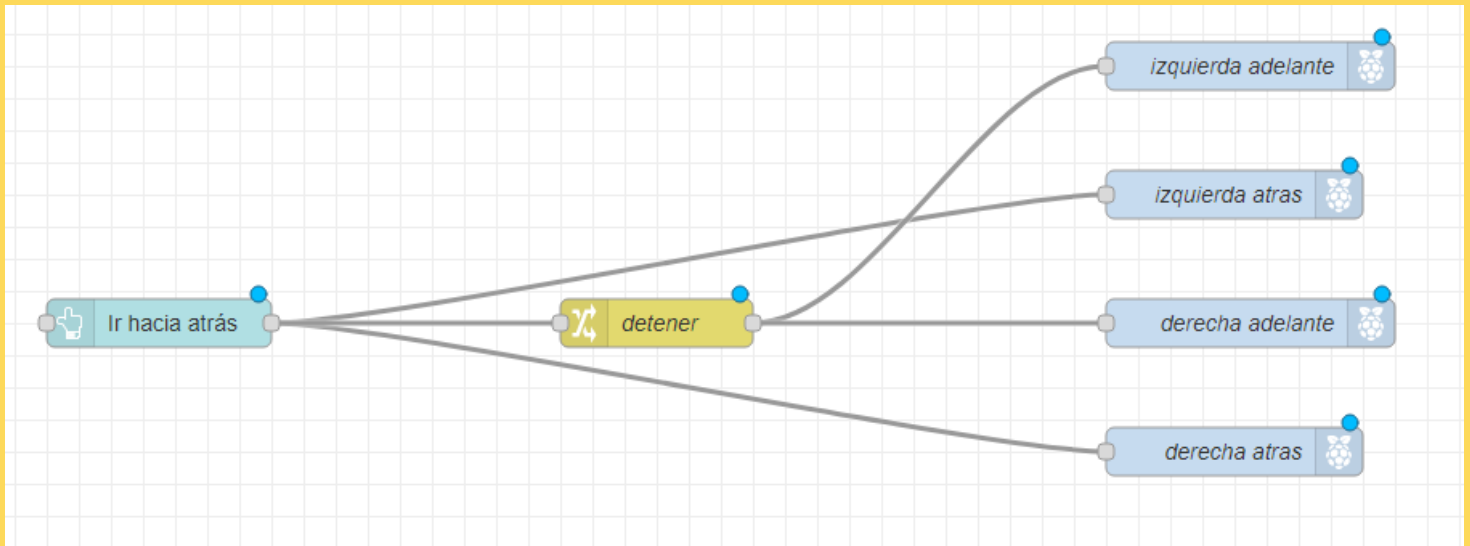
Topic: msg. topic

Name: detener

Rules:

Set msg. payload to the value 0

Botón ir hacia atrás:

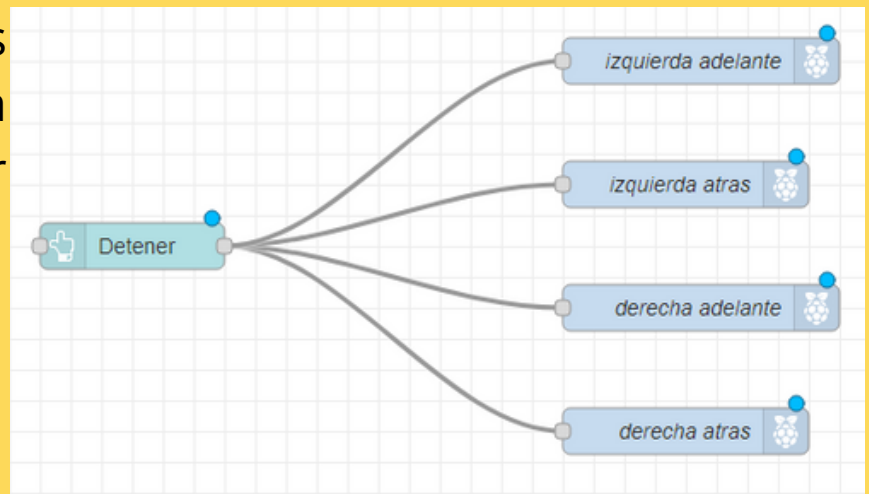


El botón ir hacia atrás funciona de la misma manera que el anterior pero a la inversa.

Botón Detener:

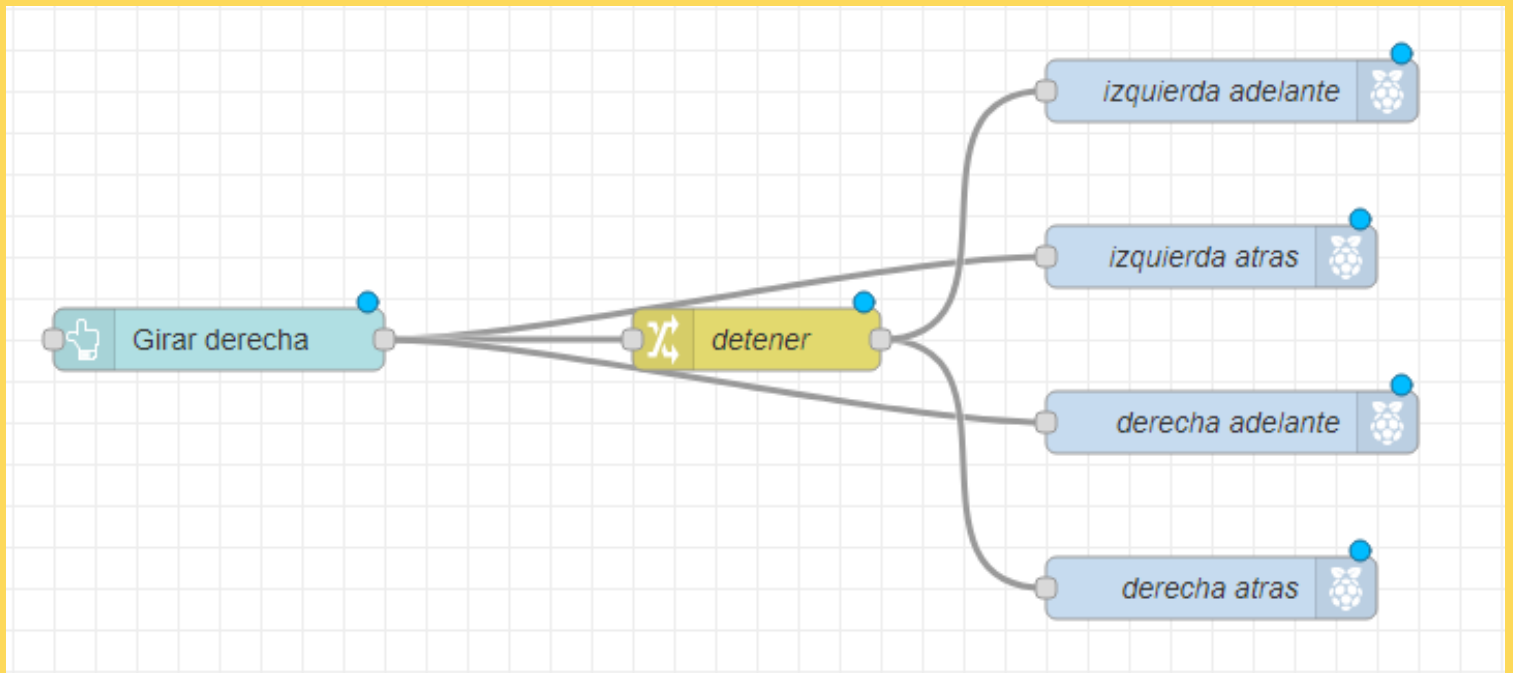
El botón "Detener" envía el valor 0 a todos los pines.

La configuración del botón "Detener", envía 0 como payload.

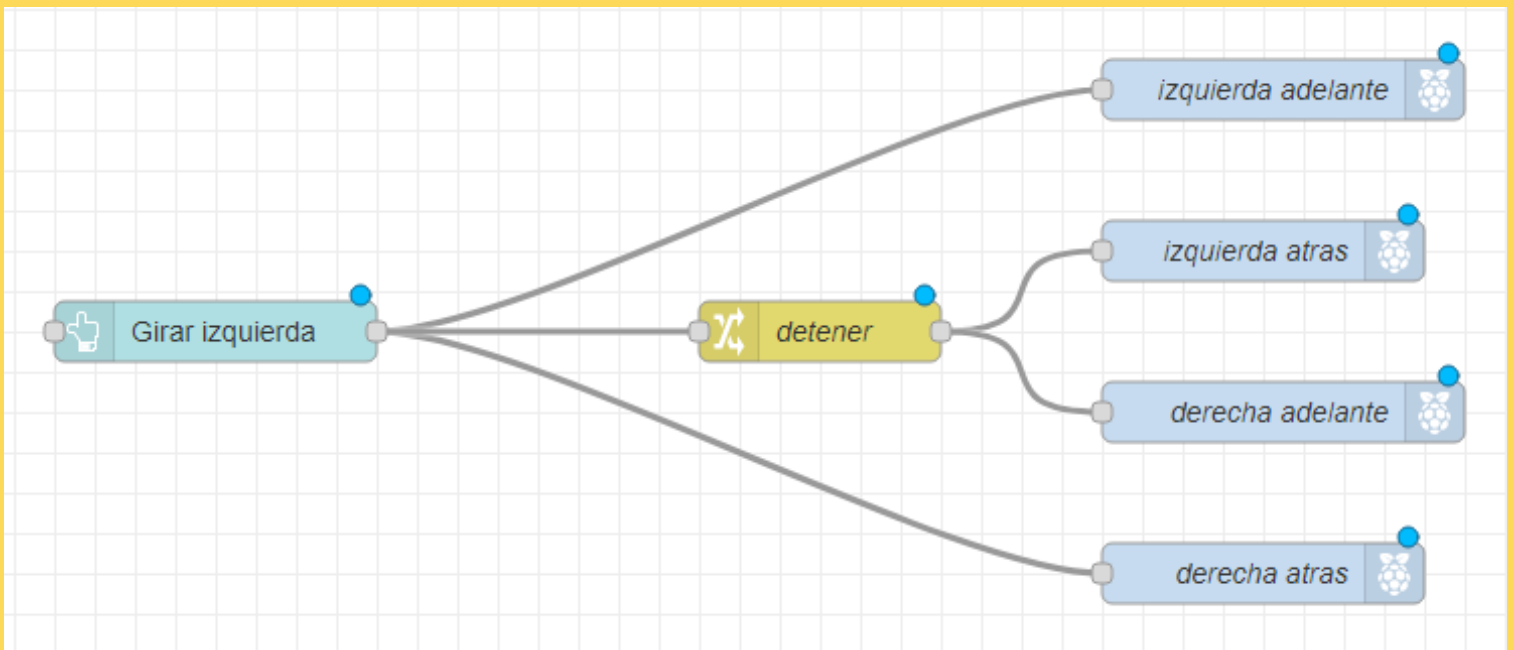


Group	[Home] Default
Size	auto
Icon	optional icon
Label	Detener
Tooltip	optional tooltip
Color	optional text/icon color
Background	optional background color
<input checked="" type="checkbox"/> When clicked, send:	
Payload	0
Topic	msg. topic

Botones de giro:

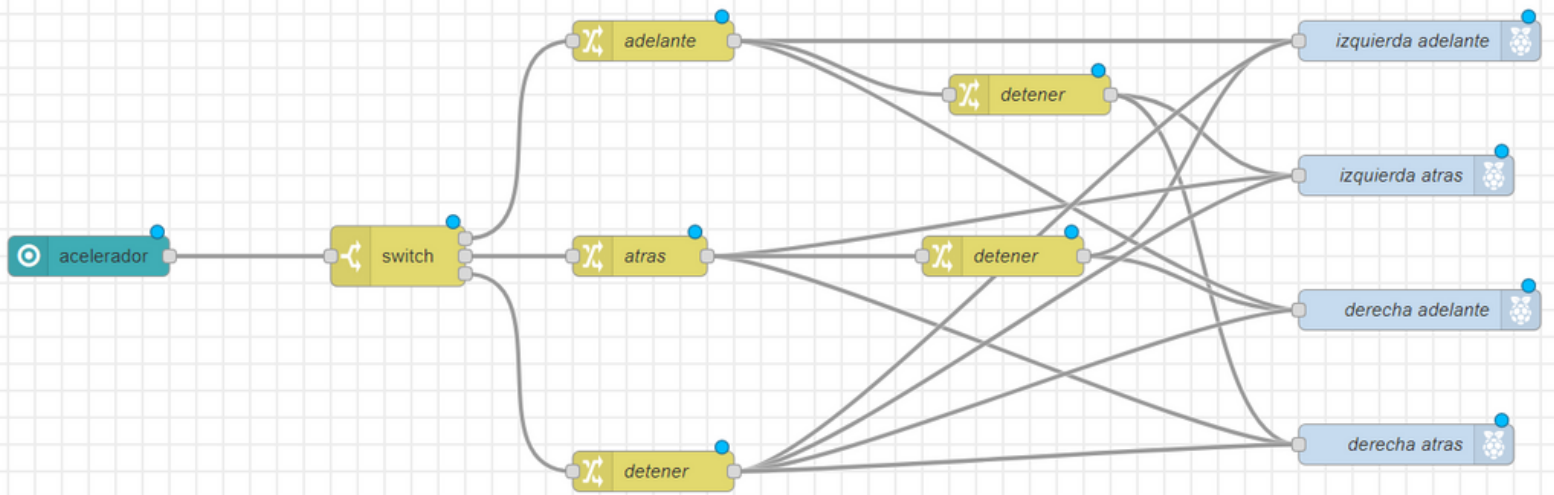


El botón "Girar derecha" envía el valor 35 a los pines 13 y 16, y envía 0 a los pines 11 y 18.



El botón "Girar izquierda" envía el valor 0 a los pines 13 y 16, y envía 35 a los pines 11 y 18. El inverso que el anterior.

Joysticks:



El nodo Joystick "acelerador" hace lo mismo que los botones ir hacia adelante o atrás, agregándole una nueva característica: cuando deja de accionarse el joystick, vuelve al centro y el auto se detiene.

Este joystick en particular acepta solo ser accionado de forma vertical.

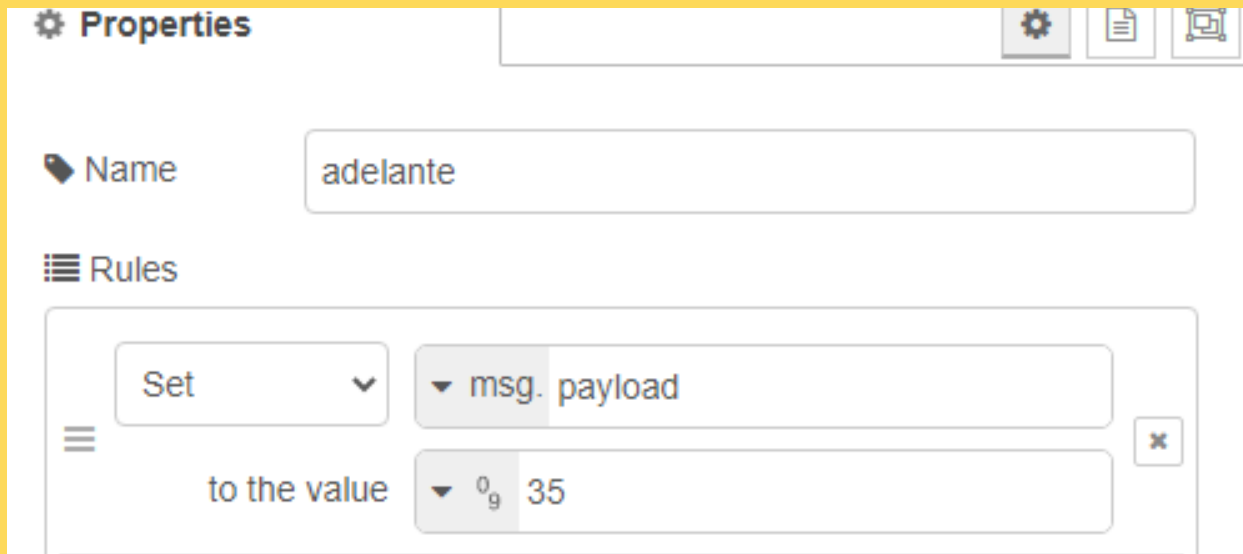
La configuración del switch, el cual tendrá una salida distinta si recibe que se está avanzando (up), retrocediendo (down) o null (joystick vuelve al centro).

Name	acelerador
Group	[Home] Default
Size	3 x 3
Interval	0 MilliSec.
Trigger	90° crossings
Directions	Only vertical
Shape	Circle
Threshold	1

Name	Name
Property	msg.payload.direction.y
Condition	== a _z up → 1
Condition	== a _z down → 2
Condition	is null → 3

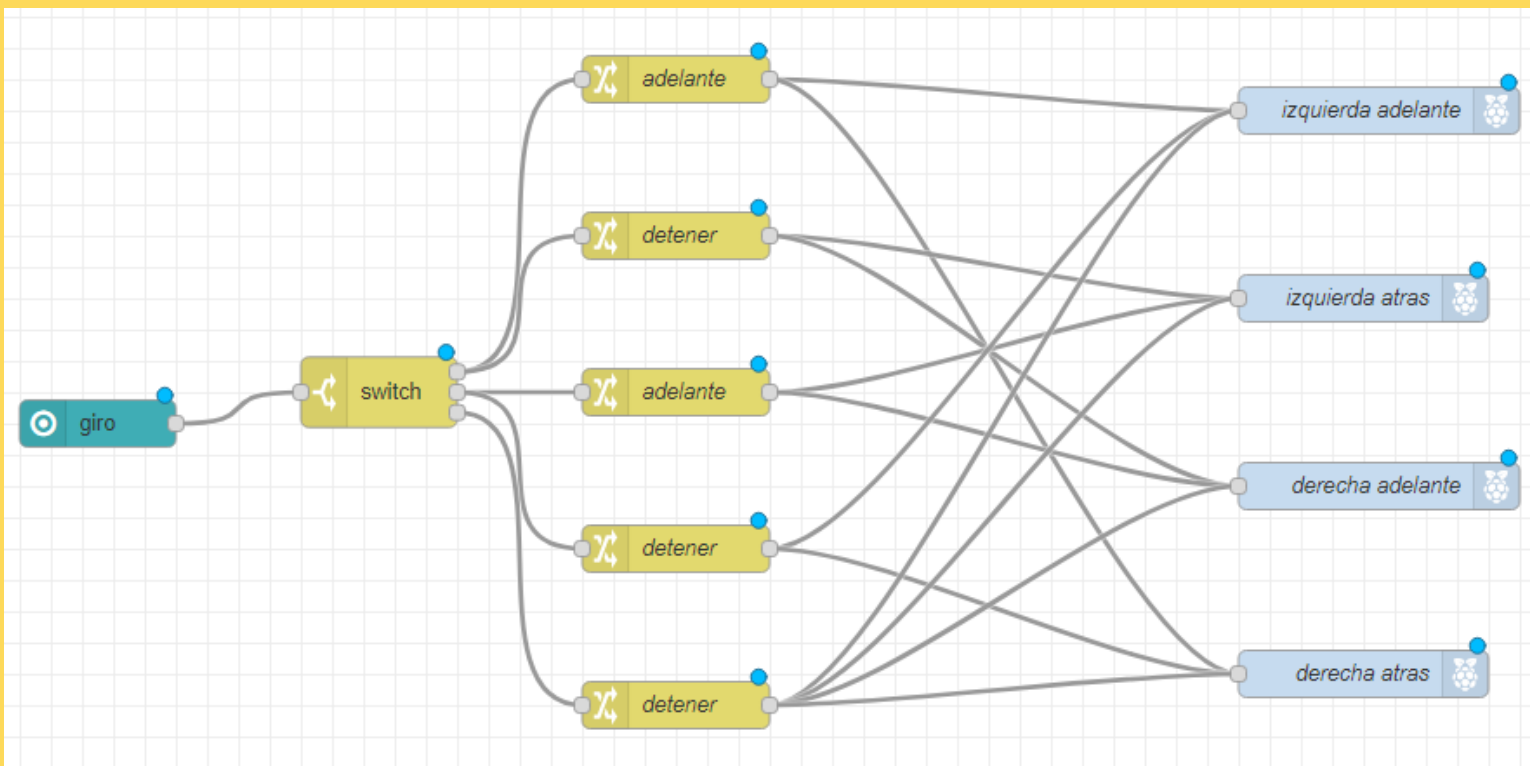
Configuración change adelante/atrás:

Precisabamos un nodo change porque el joystick devuelve un objeto completo con posición, ángulos, etc.



El nodo Joystick "giro" hace lo mismo que los botones girar derecha o izquierda, agregándole una nueva característica: cuando deja de accionarse el joystick, vuelve al centro y el auto se detiene.

Este joystick en particular acepta solo ser accionado de forma horizontal.



Configuración Switch:

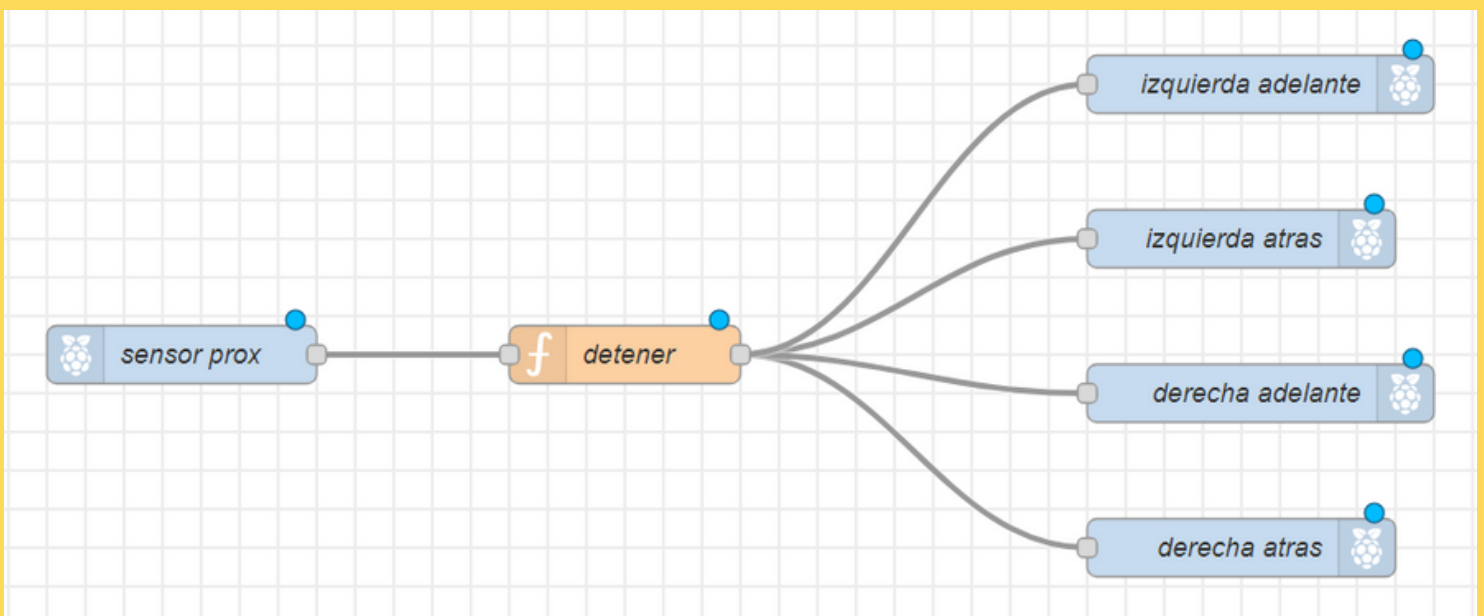
A diferencia del Switch del acelerador que revisaba el eje Y del payload, este revisa el eje X.

The screenshot shows the configuration for a Node-RED Switch node. The 'Name' field is empty. The 'Property' dropdown is set to 'msg.payload.direction.x'. Below this, there are three conditions:

- Condition 1: Operator '==', Variable 'a_z', Value 'left', Output '→ 1'.
- Condition 2: Operator '==', Variable 'a_z', Value 'right', Output '→ 2'.
- Condition 3: Operator 'is null', Output '→ 3'.

Frenos automáticos de emergencia:

Esta sección del flujo es la encargada de detener el auto de acuerdo a lo detectado con el sensor de proximidad.



Configuración del sensor de proximidad: trigger: 31,
echo: 29

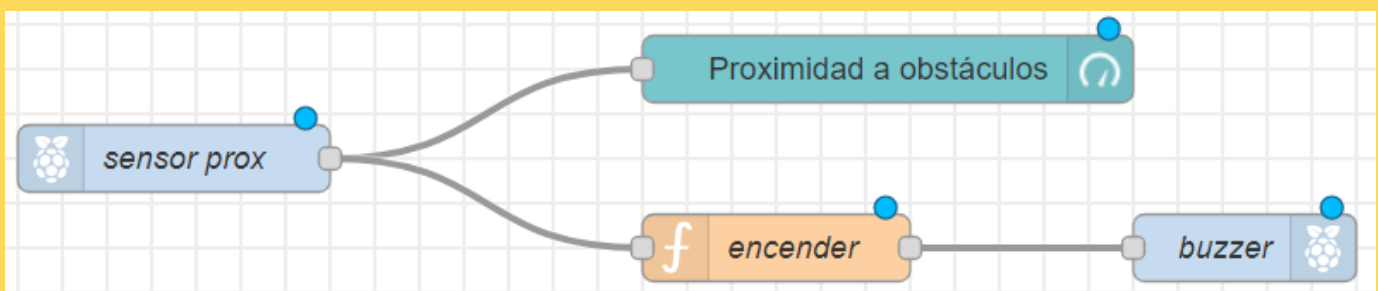
Pins	31,29
Repeat (S)	0.5
Decimal Places	0
Topic	SRF
Name	sensor prox

Función "detener":

Si el payload es menor a 20, envía 0 a las ruedas para detener el auto. Si es mayor a 20, no retorna nada y descarta el mensaje.

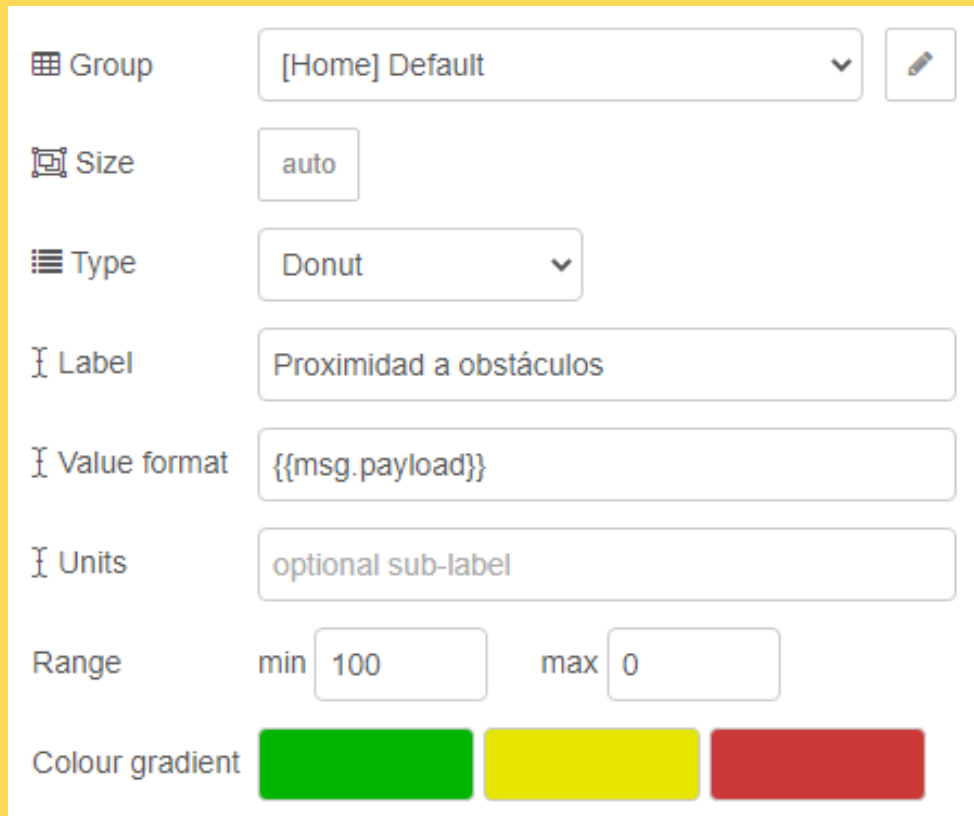
Name	detener		
Setup	On Start	On Message	On Stop
<pre>1 if (msg.payload < 20) { 2 msg.payload = 0; 3 return msg; 4 }</pre>			

Este flujo se encarga de mostrar proximidad en el dashboard con un nodo "gauge", y controlar el encendido/apagado del buzzer.



Configuración del gauge:

Recibe el payload del sensor de proximidad y el color varía según el valor del payload entre 100 y 0



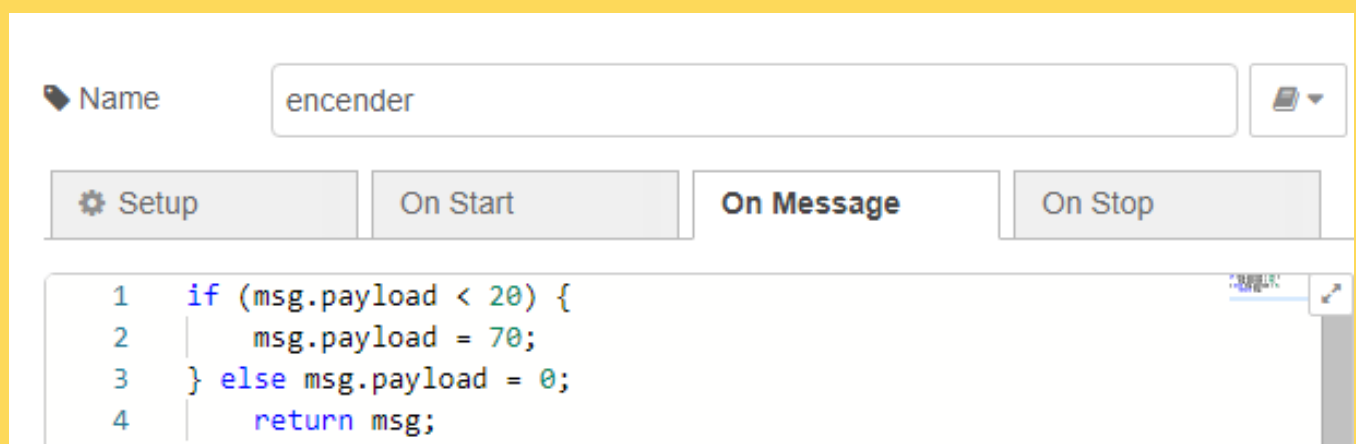
The screenshot shows the configuration for a Gauge widget. The settings are as follows:

- Group:** [Home] Default
- Size:** auto
- Type:** Donut
- Label:** Proximidad a obstáculos
- Value format:** {{msg.payload}}
- Units:** optional sub-label
- Range:** min 100, max 0
- Colour gradient:** A gradient bar with three segments: green, yellow, and red.

Función "encender":

Si la proximidad es menor a 20, le envía 70 al buzzer para que suene a esa frecuencia.

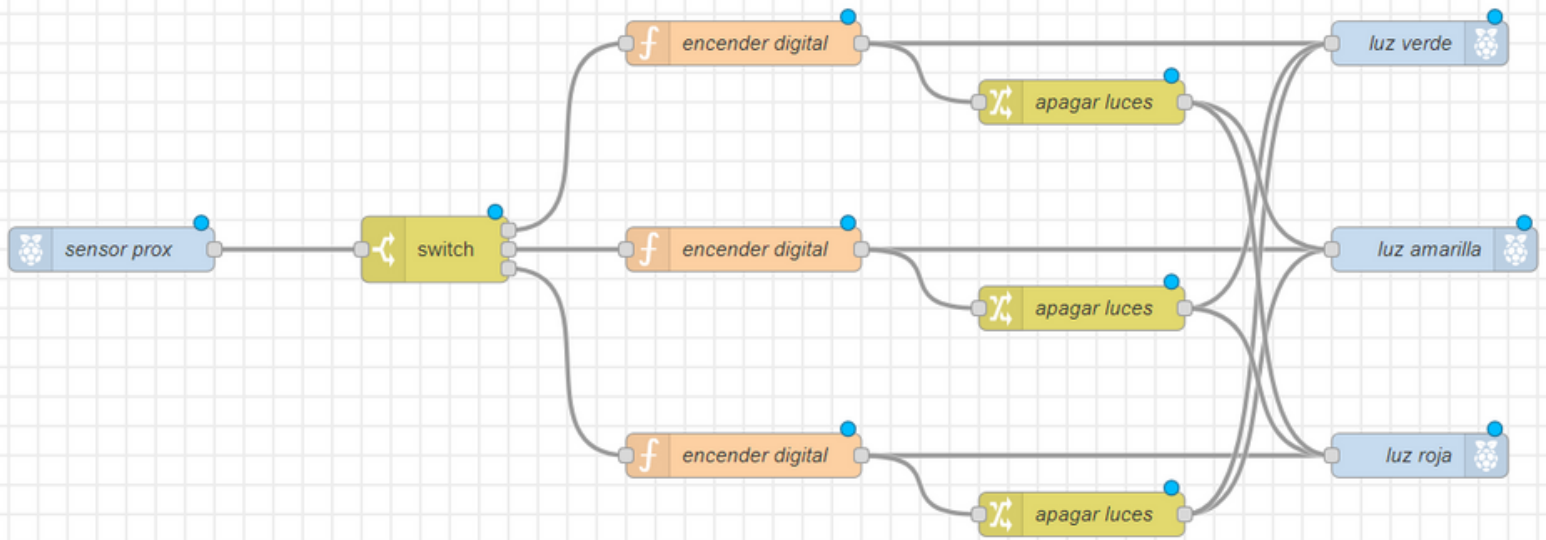
Si la proximidad es mayor a 20, lo apaga.



The screenshot shows the configuration for a function named 'encender'. The function is triggered by the 'On Message' event. The code is as follows:

```
1 if (msg.payload < 20) {  
2   msg.payload = 70;  
3 } else msg.payload = 0;  
4 return msg;
```

Esta parte del flujo es la encargada de encender/apagar las luces del semáforo según la proximidad detectada



Configuración de luces: verde= pin 40, amarilla= pin 38, roja= pin 36

BCM GPIO	16
Type	Digital output

BCM GPIO	20
Type	Digital output

BCM GPIO	21
Type	Digital output

Función "encender digital":

Cambia el payload a 1 para que los pines digitales funcionen

Name

Setup

On Start

On Message

On Stop

```
1 msg.payload = 1;
2 return msg;
```

Configuración switch:

Dependiendo la proximidad se enciende una luz:

- Luz verde si proximidad > 50 .
- Luz amarilla si proximidad ≥ 30 .
- Luz roja si proximidad < 30 .

The screenshot shows a configuration interface for a switch. It has a 'Name' field and a 'Property' dropdown set to 'msg. payload'. Below these are three rules, each with a menu icon on the left, a condition, and a result on the right.

- Rule 1: Condition is $>$ with a value of 50. Result is $\rightarrow 1$.
- Rule 2: Condition is 'is between' with values 30 and 49. Result is $\rightarrow 2$.
- Rule 3: Condition is $<$ with a value of 30. Result is $\rightarrow 3$.

Nodo change "apagar luces". Cambia el payload a 0 para apagar las otras 2 luces

The screenshot shows a configuration interface for a 'change' node. It has a 'Name' field set to 'apagar luces'. Below it is a 'Rules' section with one rule. The rule has a 'Set' action, a 'msg. payload' property, and a value of 0.

Rule: Set msg. payload to the value 0

Servidor Webcam



Buscando formas de mejorar nuestro proyecto, nos encontramos con que tenemos todo lo necesario para crear un servidor de cámara Web en la Raspberry para poder ver en tiempo real el recorrido del robot.

Paso 1: reunir los componentes

- Raspberry Pi 3 Model B (con raspbian y conectada a la red)
- Cámara web USB (Usaremos una antigua cámara web).
- Fuente de alimentación 5V 2A (Usaremos un Power Bank)

Paso 2: Conéctese a su Pi por conexión SSH (PUTTY)

Paso 3: instale el software y realice las configuraciones instalamos ' `sudo apt-get install motion` ' y editamos los archivos de configuración.

Paso 4: Inicie el servidor.

Primero se debe reiniciar el software con el comando '`sudo service motion restart`' y presionamos Enter. Después iniciamos el servicio con '`sudo motion`'.

ENLACE AL TUTORIAL DEL SERVIDOR PARA LA WEBCAM

<https://www.instructables.com/How-to-Make-Raspberry-Pi-Webcam-Server-and-Stream-/>