Explorador de luz: Raspberry movil enlazada a una base de datos NOSQL

Estuardo José Castillo Monroy - 1019116

Luis Pedro Andrino Menendez - 1198516

Javier Jolon - 1249013

Oscar Winston Lemus Higueros - 1194216

José David Fuentes - 1168315

Universidad Rafael Landívar

Resumen

z. El objetivo de este proyecto es crear un "buscador de luz" el cual funcionará basado en un PIC 16F877 como unidad de control de direcciones y un Raspberry Pi 3b como controlador del dispositivo independiente y conexión a la base de datos NOSQL.

I. INTRODUCCIÓN

Para los proyectos de energía renovable, es de vital importancia el conocer los puntos en donde es mayor la captura de luz. El objetivo de este proyecto es crear un "buscador de luz" el cual funcionará basado en un PIC 16F877 como unidad de control de direcciones y un Raspberry Pi como controlador del dispositivo independiente y conexión a internet. Utilizando el proyecto de buscador de Luz hecho con PIC 16F877, se debe de armar una matriz de 5x5. Se deberán de poner en marcha los buscadores de luz y detectar los 2 puntos con mayor luz (uno para filas y uno para columnas). Estas coordenadas de filas y columnas se guardarán para ser enviado hacia el siguiente dispositivo. El explorador de luz deberá de implementar un transporte controlado por Raspberry, se utilizara un carro a control remoto cuyas funciones de dirección y movimiento sea controlado por la Raspberry Pi. El dispositivo debe de ser autónomo e inalámbrico por lo que debe de tener su propia fuente de alimentación, se utiliza un powerbank como fuente de alimentación. Desde la red de buscadores de luz se deberá de pasar el dato de coordenadas hacia la Raspberry PI. Una vez el explorador de luz tenga las coordenadas hacia donde se debe de mover, deberá de emprender el viaje desde las coordenadas 0,0 hasta las coordenadas proporcionadas por el Buscador de Luz. Una vez haya llegado al punto, deberá de empezar a tomar muestras de su entorno por medio de sensores, se utilizan 3 sensores diferentes, de humedad, de luz y de proximidad. La información de estos sensores debe de ser medida cada 2 segundos durante 2 minutos. Cada vez que se tome una nueva muestra, esta data debe de ser enviada hacia una base de datos cloud. Se debe de implementar una base de datos en donde se almacene cada uno de los datos tomados. La base de datos se realizara en mongo DB. El Explorador de luz deberá de poder encontrar una red de internet, conectarse a ella y empezar a hacer los inserts en tiempo real. Al terminar estos 2 minutos recopilando datos, el Explorador de Luz debe de emprender su viaje hacia el punto 0,0 nuevamente

II. MATRIZ DE BUSCADORES DE LUZ

La matriz de buscadores de luz, es una matriz compuesta por los dispositivos buscadores de luz que estan ubicados 5 en fila que representan el eje x y otros 5 en fila que representan el eje y, de esta manera se forma la matriz de 5x5 que nos sirve para obtener la ubicación con mayor intensidad de luz al obtener los datos de los 10 dispositivos, la posición se calcula obteniendo el dipositivo con mayor luz en x y el dispositvo con mayor luz en y. Los dispositivos suben los datos a una base de datos no relacional.

III. ALGORITMO DE MOVIMIENTO Y MOTORES DC

Los motores empleados para el movimiento del carro son motores DC. Para controlar los motores se optó por usar el integrado LD293D, el integrado funciona tomando voltaje de la raspberry y compartiendo tierra común. Los motores por ser de corriente directa (DC) se mueven dependiendo del voltaje que se les manda. Este voltaje lo manejamos con la raspberry con la libreria GPIO, que nos permite mandar LOW (cero lógico) o HIGH (uno lógico).

El algoritmo de movimiento se basa en mandar salidas a los pines por cierta cantidad de segundos, esto se realizó haciendo varias pruebas poniendo diferentes cantidades de tiempo y viendo cuanto se movía.

IV. SENSORES DE TEMPERATURA, HUMEDAD Y PROXIMIDAD

I. Sensor de Humedad y Temperatura DHT11

No tenemos que confundirnos, el sensor DHT11 es un dispositivo analógico que internamente convierte la señal en formato digital y posteriormente se envía al microcontrolador.

La trama de datos es de 40 bits correspondientes a la información de humedad y temperatura del DHT11.

- 8 bits - Parte entera de la humedad
- 8 bits - Parte decimal de la humedad
- 8 bits - Parte entera de la temperatura
- 8 bits - Parte decimal de la temperatura
- 8 bits - Bits de paridad

Estos bits de paridad nos permiten verificar que la información medida es correcta. Sumando los 4 primero grupos de 8-bit, el resultado debe ser igual a los bit de paridad.

A continuación se especifican algunas características técnicas del sensor DHT11

Cuadro 1: *Características técnias del DHT11*

Sensor Humedad & Temperatura	
Modelo	DHT11
Alimentación	3.3-5V
Consumo	2.5mA
Salida	Señal Digital
Temperatura	
Rango	0-50 C
Precisión	+-2 C
Humedad	
Rango	20-90 %
Precisión	+-5 %

II. Sensor de Proximidad HC-SR04

El funcionamiento de los sensores de ultrasonidos es, por lo general, un concepto sencillo. Y el HC-SR04 no es ningún caso especial.

El HC-SR04 posee dos transductores cilindricos de color gris. El sensor envía un pulso de ultrasonidos a través de un transductor cuándo el pin “Trig” está en alto. El pulso avanza hasta que choca con un obstáculo y rebota, volviendo así al sensor. El segundo transductor detecta la señal de este “eco” y mide el tiempo que ha tardado la señal en rebotar. Después, activa el pin “Echo” durante un tiempo proporcional al que ha tardado en llegar el “eco” al segundo transductor. Esta señal después se envía a la Raspberry Pi, que mide la duración del pulso

y la multiplica por una constante para tener la distancia en centímetros aproximada.

Es importante mencionar que el sensor HC-SR04 es alimentado con una tensión de 5 voltios, y la señal que envía a la raspberry pi es de la misma magnitud. Sin embargo, la raspberry pi opera con aproximadamente 3.3 voltios por lo que fue necesario emplear por medio de resistencias de 330 y 680 ohmios un divisor de voltaje para regular la salida del sensor.

V. CONEXIÒN RASPBERRY PI-BASE DE DATOS

La base de datos fue empleada en mongoDB, por lo que es una base de datos no relacional. A partir de eso se utiliza el servicio de mlab para poder manejar la base de datos por medio de una aplicacion diseñada en Node.js. Para la conexion con la base de datos se hace uso de un modulo de mongoDB para python que se llama pymongo, que permite conectarse a la base de datos y realizar tanto consultas como inserts.

VI. CONCLUSIONES

- El proyecto tiene cierto grado de dificultad, debido a programación que hay que realizar en el pic, para hacer el buscador de luz, y la secuencia que hay que programar para hacer mover los motores, lo más difícil es coordinar el tiempo que toma en hacer la respectiva búsqueda.
- En el explorador de luz, uno de los problemas más difíciles fue hacer que el carro caminara lo más recto posible. Inicialmente el explorador era de 3 ruedas, 2 con motor y una solo de soporte, al momento de hacer las respectivas pruebas verificamos que la llanta de soporte hacia que se desviara, por lo tanto se le agregan otras 2 llantas con motor para evitar el problema, a la hora de hacer las pruebas respectivas el inconveniente que surge es que los motores no tienen la suficiente fuerza para mover al explorador completo, por el peso que tenía y por la batería que alimentaba para alimentar la raspberry. Por lo tanto se regresa a la solución anterior, 3 llantas, con la variación de que ahora estará fija, sin moverse. El resultado es bastante satisfactorio y por lo tanto se decide dejarlo con 3 llantas.
- Otro problema que surge con las 3 llantas es que 1 de ellas no tenía la suficiente tracción y hacia que se desviara de su trayectoria, la solución fue colocarle un contrapeso para asegurarse que siempre estuviera tocado el suelo.
- Respecto a la raspberry el problema que surgió fue con la comunicación inalámbrica, utilizamos el programa puty para dicha comunicación, pero a la hora de configurar no lográbamos hacer que se comunicara, la solución era bastante sencilla, ir a la configuración de la raspberry y hacer las modificaciones para habilitar el puerto SSH y con eso se solucionaba el problema.
- La programación de la raspberry se realizó en thony Python. Es relativamente sencillo hacer la programación teniendo algún conocimiento de programación previo.
- Otro de los problemas que surgio fue la base de datos, por ser una base en modo gratuito, se saturaba y se quedaba sin conexion, por lo que se optó pagar una base de datos que cumpliera con los requisistos necesarios para el proyecto.
- Con los sensores solo habia que hacer las programación correctamente para que no tener problemas, el codigo se encuentra en internet y se puede utilizar sin ningún problema.

Referencias

[1] Noticias de seguridad informática (2016). 10 Vulnerabilidades importantes que afectan a la seguridad de bases de datos empresariales. Recuperado de:

http://noticiasseguridad.com/importantes/10-vulnerabilidades-comunes-que-afectan-la-seguridad-de-base-de-datos-empresariales/