Sistema de Monitoreo GPS con Interfaz Web

Introducción a la programación de microcontroladores con Arduino
José Luis Di Biase

Integrantes: Ivan Rosendo,Matias Cionci.

1. Descripción del Proyecto

El proyecto consiste en el desarrollo de un sistema de rastreo y localización. La idea principal se basó en la lectura de datos de posicionamiento global mediante un módulo GPS conectado a un microcontrolador (basado en el tutorial de Naylamp Mechatronics). Para escalar el proyecto y permitir el monitoreo remoto, se incorporó un módulo WiFi que transmite las coordenadas a un servidor Backend desarrollado en Node.js. Finalmente, los datos se visualizan en tiempo real en una interfaz de usuario (Frontend) desarrollada con el framework Angular.

2. Materiales Utilizados

Componentes Electrónicos:

- Placa microcontroladora [Arduino UNO].
- Módulo GPS GY-GPS6MV2.
- Módulo WiFi ESP-01S.
- Cables jumper (M-M, M-H).
- Protoboard.
- Resistencia (1k Ohm).

Herramientas de Taller:

- Soldador (cautín).
- Alambre de estaño.

3. ¿Cómo y dónde se consiguieron los materiales?

1. **Módulo WiFi ESP-01S:** Fue comprado por el precio de $5.000 ARS.
2. **Módulo GPS GY-GPS6MV2:** Fue comprado el precio fue de $7.000 ARS.

4. Código fuente del programa del microcontrolador

C++

‘#include <SoftwareSerial.h>

#include <TinyGPS++.h>

// --- Configuración GPS ---

const int GPS_RX = 4;

const int GPS_TX = 3;

const int GPSBaud = 9600;

SoftwareSerial gpsSerial(GPS_RX, GPS_TX);

TinyGPSPlus gps;

// --- Configuración ESP8266 ---

const int ESP_RX = 10;

const int ESP_TX = 11;

SoftwareSerial esp8266(ESP_RX, ESP_TX);

String serverIP = "10.12.18.219";

String serverPort = "5000";

```
unsigned long ultimoEnvio = 0;
const unsigned long intervaloEnvio = 10000; // Enviar cada 10 segundos (10000 ms)

void setup() {
  Serial.begin(9600);

  gpsSerial.begin(GPSBaud);
  esp8266.begin(9600);

  Serial.println(F("--- Iniciando sistema GPS + WiFi ---"));
  Serial.println(F("Esperando 5 segundos para asegurar la conexion WiFi del ESP..."));
  delay(5000);

  gpsSerial.listen();
}

void loop() {
  while (gpsSerial.available() > 0) {
    gps.encode(gpsSerial.read());
  }

  if (millis() - ultimoEnvio >= intervaloEnvio) {
    ultimoEnvio = millis();
```

```
    if (gps.location.isValid()) {
      String latitud = String(gps.location.lat(), 6);
      String longitud = String(gps.location.lng(), 6);

      Serial.print(F("Ubicación válida: Lat: ")); Serial.print(latitud);
      Serial.print(F(" | Lng: ")); Serial.println(longitud);

      // --- PROCESO DE ENVÍO ---
      esp8266.listen();

      enviarCoordenadas(latitud, longitud);

      gpsSerial.listen();

    } else {
      Serial.println(F("Esperando señal de satélites válida para poder enviar datos..."));
    }
  }
}

void enviarCoordenadas(String lat, String lng) {

  Serial.println(F("1. Abriendo conexion TCP..."));
  String cmd = "AT+CIPSTART=\"TCP\",\"" + serverIP + "\"," + serverPort;
```

```
  esp8266.println(cmd);
  delay(2000);
  imprimirRespuestaESP();

  String peticionHTTP = "GET /api/actualizar?lat=" + lat + "&lng=" + lng + " HTTP/1.1\r\n";
  peticionHTTP += "Host: " + serverIP + "\r\n";
  peticionHTTP += "Connection: close\r\n\r\n";

  Serial.println(F("2. Definiendo tamano del paquete..."));
  cmd = "AT+CIPSEND=" + String(peticionHTTP.length());
  esp8266.println(cmd);
  delay(1000);
  imprimirRespuestaESP();

  Serial.println(F("3. Enviando peticion HTTP..."));
  esp8266.print(peticionHTTP);
  delay(2000);
  imprimirRespuestaESP();

  Serial.println(F("==== Intento de envio terminado ===="));
}

void imprimirRespuestaESP() {
  while (esp8266.available()) {
```

```
    String linea = esp8266.readStringUntil('\n');
    Serial.println(linea);
  }
}
,
```

5. Problemas enfrentados

Problema 1: Conexión inestable de los pines del módulo GPS.

- *Descripción:* Al principio tuvimos problemas para conectar o "pegar" los pines al módulo GPS, lo que generaba falsos contactos y el módulo no lograba encender o no transmitía los datos correctamente al microcontrolador.
- *Solución:* Para asegurar la conductividad eléctrica y la fijación mecánica, resolvimos utilizar un soldador y estaño para soldar de manera definitiva la tira de pines (headers) a los orificios del módulo GPS.

6. Paso a paso del armado

Soldadura del GPS: Se soldaron los pines de conexión al módulo GY-GPS6MV2 utilizando soldador y estaño.

Conexión de Hardware: Se conectó el módulo GPS al microcontrolador especificando los pines TX y RX. Luego, se conectó el módulo WiFi ESP-01S asegurando su correcta alimentación [a 3.3V].

Programación del Microcontrolador: Se escribió y cargó el código para leer las sentencias NMEA del GPS y enviarlas vía WiFi al servidor.

Desarrollo del Servidor (Backend): Se levantó un servidor con Node.js [y Express] escuchando en un puerto específico para recibir la ubicación.

Desarrollo de la Interfaz (Frontend): Se creó la aplicación en Angular que consume la API del backend para mostrar los datos de localización al usuario.

6. Software necesario y sus versiones

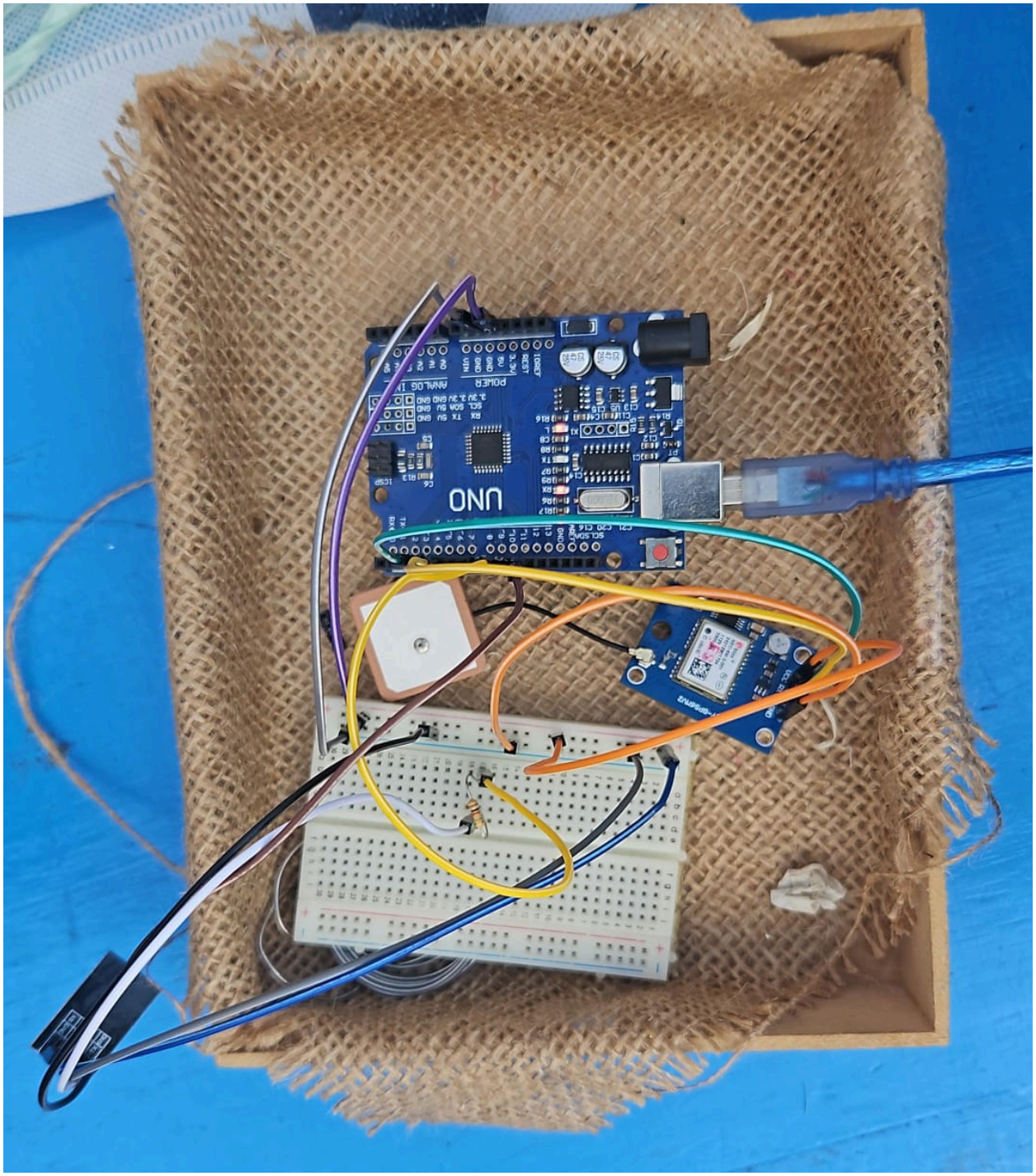
Arduino IDE: Versión [1.8.19]

Node.js: Versión [24.16.0]

Angular CLI: Versión [21]

Librerías externas (Arduino): * `TinyGPS++`.

- `SoftwareSerial` (incluida en el IDE).



Repositorio : https://github.com/MatiasCionci/TPArduinoRastreadorUnq#