

Universidad Tecnica Del Norte  
Fica-Ciercom  
Sistemas Embebidos  
Examen Bimestral I

Kevin A. Enriquez-Fuel, Kevin G. Oñate-Pozo, Fredy D. Chicaiza-Chuquilla

13 de diciembre de 2019

## 1. Objetivos

- Realizar una revisión bibliográfica sobre el funcionamiento de los sensores: MQ-7, MQ-135, Sensor de Rayos UV ML8511, Sensor de temperatura TH y el microprocesador MCU.
- Calibrar los sensores para obtener datos precisos
- Documentar el funcionamiento y parámetros de los sensores.
- Realizar el armado de la placa.

## 2. Introducción

### 2.1. Primera Lectura: Calidad del aire y efectos en la salud-¿Como pueden contribuir las redes de sensores inalámbricas?

Uno de los principales problemas que afectan a todos los países, no solo en aquellos que están dentro de grandes metrópolis o en zonas industriales: La contaminación del aire, es un gran problema que desemboca grandes daños, riesgos a la salud y un fuerte prejuicio económico para todas las naciones del mundo. El principal objetivo de este artículo, es plantear diferentes alternativas para brindar una posible solución frente a este problema. Con la introducción de la tecnología y las capacidades de la Internet de los objetos (IO) se pretender disparar nuevos casos de uso y aplicarla a diferentes entornos.

Para el análisis de los parámetros anteriormente nombrados, se utilizaran los sensores: Quimioluminiscencia, difusión pasiva (DP) y Los sensores electroquímicos.

#### 2.1.1. Detectores de quimioluminiscencia

El principio de medición se basa en sobre la reacción radiactiva del ozono con el óxido de nitrógeno (NO). Esta radiación puede ser detectada para medir la cantidad de NO. El NO<sub>2</sub> puede de medirse separando el NO<sub>2</sub> para obtener el NO por medio de los rayos UV.

#### 2.1.2. Muestreadores de difusión pasiva

Es un método de medición integral, es decir, mide todo el NO<sub>2</sub> desde el momento en que fue monitado, hasta la hora en que se recoge.

#### 2.1.3. Sensores electroquímicos

Los sensores electroquímicos consisten ende al menos dos electrodos. El gas se difunde en una cámara, que normalmente está cubierta por una membrana, para proteger de partículas que entran en la cámara.

#### 2.1.4. Sensores Ópticos

Los sensores ópticos se basan en la absorción de luz por gases. Del ultravioleta al infrarrojo, cada molécula tiene un espectro de absorción específico.

### 2.2. Segunda Lectura: Impacto de las emisiones de NOx en el clima y la monitorización mediante la tecnología de sensores inteligentes

En este articulo se intento diseñar un sistema de Flame monitoring, pero con un sistema muy simple de procesamiento de imágenes para controlar las emisiones de CO. Por lo tanto, la investigación en este campo encuentran muchas posibilidades que permitirán a los investigadores desarrollar una técnica aborigen para el sistema de

ame monitoring. Brown et al (2008)desarrollo un chip de fotodiodo de carburo de silicio doble para determinar la temperatura de un producto natural. El concepto que aquí se discute utiliza el cambio en la forma de la banda OH (260-350 nm) con temperatura. La mitad del chip estaba cubierta con un filtro dieléctrico de capas múltiples con una longitud de onda corta de alrededor de 315 nm. Despues de la amplificación, las dos señales produjeron por las partes filtradas y no filtradas de la viruta, que se dividen para producir una proporción, que es muy sensible a los cambios en la temperatura de la llama. La sensibilidad es de aproximadamente 0.35en la temperatura de la llama para temperaturas entre 2700 y 3000 grados Farenheit. La temperatura medida es la medida específica temperatura comprendida por el campo de visión del sensor. El objetivo es desarrollar sensores de avance para reducir el NO, emisiones de CO y CO<sub>2</sub> que mejoran la combustión efectiva. En este método propuesto por Ronald Hanson et al. (2004), un laser de diodo de infrarrojo cercano sintonizable y de absorción se utiliza la espectroscopia. Estrategias de control para la combustión , implica una temperatura del gas robusta y de respuesta rápida sensor, sensor de fibra acoplado para la temperatura del gas, O<sub>2</sub>, CO y una nueva técnica para la detección de hidrocarburos no quemados.

### **2.3. Tercer Lectura: Polluino: una gestión e ciente basada en la nube de Dispositivos IoT para monitoreo de calidad del aire**

La idea fundamental del Internet de las cosas es la distribución de los objetos “ubicos o cosa”, que recogen e intercambiar datos con el fin de alcanzar un objetivo común a través de interacciones mutuas [1].Polluino, es un dispositivo de sensor basado en Arduino que recoge y transmite los datos recogidos por múltiples módulos de sensor. En detalle, los parámetros ambientales se recogen con el objetivo de medir los niveles de contaminación del aire. El encendido / apagado de conmutación de los sensores se controla a distancia de acuerdo a los datos basados en sensores que se almacenan y administran directamente en la infraestructura de Cloud. El Arduino recoge todos los datos de los sensores y los envía a la plataforma en la nube mediante el uso de la ESP8266 modulo Wi-Fi, que está conectado al Arduino a través de un puerto serial de a bordo. Ciudad inteligente guion. Las características típicas de Ciudades inteligentes son la movilidad y la variedad de nodos (dispositivos móviles de usuario, sensores, y vehículos), que puede ser equipado con al menos una interfaz de red, y la amplia

disponibilidad de puntos de acceso inalámbrico. El enfoque propuesto podrá utilizarse para hallar una manera optima de controlar luces trafico c, a fin de mejorar trafico c minimizando al mismo tiempo los niveles de contaminación de aire. Por otra parte, este trabajo futuro se centrara en la comparación entre los protocolos MQTT y COAP.

### **2.4. Cuarta Lectura: Desarrollo de plataforma de detección para mediciones y análisis de calidad del aire**

AAB College, han desarrollado e implementado una plataforma de detección en el laboratorio, que mide la calidad del aire, que puede ser utilizado para monitorear y puede proporcionar datos para su análisis. Esta plataforma se basa principalmente en la tecnología Arduino. Se han calculado y el valor AQIs de diferentes contaminante, tales como SPM (partículas suspendidas), NRMF (respirable partículas suspendidas), SO<sub>2</sub>, NO<sub>2</sub>, etc, en la India, Delhi. Monitorearon tres sitios diferentes. Recibieron muestras dos veces por semana, por lo que durante un año se recogieron 104 muestras. Las muestras recogidas se analizaron mediante diferentes métodos prescritos por APHA1977 (American Public Health Asociación). Los sensores electroquímicos son los tipos más rápido crecimiento, para este proyecto se ha utilizado sensor Shiney PPD42, que se utiliza para el cálculo de la calidad del aire, principalmente para la concentración de PM10 en burbujas de polvo, pero también es compatible con las mediciones de PM. El Índice de Calidad del Aire (AQI) es muy importante con el fin de identificar la calidad del aire en ambientes diferentes. Hay diferentes sistemas que pueden calcular el índice AQI. Estos sistemas deben ser capaces de monitorizar diferentes contaminantes del aire en tiempo real, informando a la gente sobre el estado actual de la calidad del aire y el envío de la información a un servidor centralizado, que a su vez puede generar estadísticas e informes. Se ha desarrollado un algoritmo llamado “Algoritmo para el cálculo del índice”. El objetivo de este algoritmo es calcular la calidad del aire, sobre la base de los valores tomados de AQI estándares (Índice de Calidad del Aire), y para mostrar resultados en formato básico en la pantalla LCD y (pero no limitados a) muestran el color correspondiente o combinación de colores de las luces LED.

## 2.5. Quinta Lectura: Sistema de monitoreo de la contaminación del aire urbano con modelos de pronostico

Se cree ampliamente que la contaminación del aire urbano tiene un impacto directo en la salud humana, especialmente en los países en desarrollo e industriales, donde las medidas de calidad del aire no están disponibles o se implementan o aplican de manera mínima. Se presenta una red basada en sensores inteligentes que se comunican mediante una red de área local inalámbrica para aplicaciones de monitoreo de la calidad del aire. Esta red utiliza redes neuronales artificiales (ANN) para estudiar los efectos de la temperatura y la humedad en las concentraciones de contaminantes. Principalmente se centra en presentar un NGAM piloto y en el desarrollo de modelos de pronostico precisos para predecir las concentraciones promedio futuras de algunos contaminantes del aire urbano, a saber: O<sub>3</sub>, NO<sub>2</sub> y SO<sub>2</sub>, todos los cuales se mencionan como perjudiciales en Las directrices de la OMS. La calidad del aire es un problema importante que afecta directamente la salud humana. Los datos de calidad del aire se recopilan de forma inalámbrica a partir de motos de monitoreo que están equipados con una variedad de sensores gaseosos y meteorológicos. Estos datos se analizan y utilizan para pronosticar valores de concentración de contaminantes utilizando una plataforma inteligente de maquina a máquina. La plataforma utiliza algoritmos basados en ML para construir los modelos de pronostico aprendiendo de los datos recopilados. Estos modelos predicen 1, 8, 12 y 24 horas antes de los valores de concentración.

## 3. Marco Teórico

### 3.1. Filtro Medio

El filtro medio es uno de los métodos de suavizado más simples. La versión modificada de este algoritmo se discute en [1], [2]. El valor de salida se calcula calculando el promedio de todas las muestras de la ventana. Su fórmula tiene la siguiente forma:

$$z_j = \frac{\sum_{i=-n}^n x_{j+i}}{2n+1}, \quad (1)$$

donde  $x_i$  es una muestra de la señal de entrada,  $2n+1$  es la longitud de la ventana,  $z_j$  representa el valor de salida y  $j$  es el índice actual del valor de salida. Se conocen diversas formas de promediación. Uno de ellos es el uso de la máscara de filtro que consiste en números del mismo valor.

Por ejemplo, cuando la longitud de la ventana es igual a 5, se usa la siguiente máscara:

$$M_1 = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 5 & 5 & 5 & 5 & 5 \end{bmatrix}, \quad (2)$$

En este caso, el valor de salida se calcula convolucionando la máscara  $M_1$  con las muestras en la ventana. La efectividad de este filtro depende estrictamente de la longitud de la ventana.

### 3.2. Filtro Mediano

Es un filtro no lineal que suaviza el ruido de impulso, pero sin difuminar los cambios de señal. A menudo se usa para eliminar ruido de imágenes porque al eliminar el ruido se conserva la nitidez de los bordes. El uso de este filtro, así como su modificación, se han considerado en numerosos trabajos [3] - [6]. El filtro de mediana tradicional funciona eligiendo en la ventana el valor del punto medio llamado mediana. Su fórmula tiene la siguiente forma:

$$z_j = median(x_{j-n}..x_{j-2}, x_{j-1}, x_j, x_{j+1}, x_{j+2},.., x_{j+n}) \quad (3)$$

Para encontrar la mediana, las muestras de la ventana N-muestra se ordenan y la mediana se determina como:

$$\left\{ \begin{array}{ll} s(N+1)/2 & , if N mod 2 == 1 \\ \frac{s(N/2 - 1) + s(N/2 + 1)}{2} & , if N mod 2 == 0 \end{array} \right. \quad (4)$$

### 3.3. Filtro de Kalman discreto

El algoritmo de filtro discreto de Kalman intenta estimar el estado  $x$  de un sistema de tiempo discreto. Se llama un filtro lineal óptimo porque incorpora toda la información del proceso  $y$ , en consecuencia, produce un error que es estadísticamente mínimo. El algoritmo supone que la predicción y la medición del proceso se realizan con la presencia del ruido gaussiano blanco. Típicamente, el algoritmo requiere un procedimiento de cálculo de dos pasos: paso de actualización de tiempo, llamado paso de actualización de predicción y medición, llamado corrección. La predicción proporciona la estimación del estado actual por adelantado y la corrección ajusta la estimación proyectada utilizando la medición actual. En el paso de predicción, la estimación del estado se calcula a partir de la siguiente ecuación:

Describe la selección de la mediana de la lista ordenada S. Si N es impar, la mediana es la muestra  $s(N+1)/2$ , de lo contrario, la mediana es el promedio de dos elementos  $s(N/2 - 1)$  y  $s(N/2 + 1)$

$$P_k^- = A \cdot P_{k-1} \cdot A^T + Q, \quad (5)$$

Donde  $P_k^-$  es la covarianza de error que se estima a priori,  $P_{k-1}$  también es covarianza de error pero a posterior se estima y  $Q_n * n$  es el error de proceso (covarianza de ruido). A la salida de cada paso de predicción particular, obtenemos las estimaciones consecutivas de  $P_k^-$  y  $x_k^-$ . En el paso de corrección se debe de calcular la ganancia del filtro de Kalman y usar esta ganancia para actualizar la estimación del estado usando la medición actual. La ganancia controla la influencia de la medición en la estimación del estado a posterior en el paso de tiempo  $k$ . La ganancia de Kalman se calcula utilizando la siguiente fórmula:

$$K_k = P_k^- \cdot H^T \cdot (H \cdot P_k^- + R)^{-1}, \quad (6)$$

donde la matriz  $H$  relaciona la medida con el estado  $x$ . La matriz  $R$  es la imprecisión de la medición (covarianza de ruido). Finalmente, la fórmula para la actualización de la estimación del estado con mediciones tiene la siguiente forma:

$$\hat{x}_k = \hat{x}_k^- \cdot K_k \cdot (z_k - H \cdot \hat{x}_k^-), \quad (7)$$

donde  $\hat{x}_k$  es una estimación del estado posterior,  $\hat{x}_k^-$  es el estado observado en el paso de tiempo  $k$ ,  $K_k$  la ganancia que controla la influencia de la medición en  $\hat{x}_k$ ,  $z_k$  es la medición en el paso de tiempo  $k$  y la matriz  $H$  da la relación de la medición a la estado.

Al final del procedimiento de actualización, se requiere la actualización de la covarianza de error. Su fórmula tiene la siguiente forma:

$$P_k = (I - K_k \cdot H) \cdot P_k^-, \quad (8)$$

El algoritmo de filtro discreto de Kalman presentado debe implementarse como una función recursiva, pero en la práctica se implementa más bien en forma iterativa para reducir el tiempo de cálculo. El principal parámetro responsable de la efectividad del suavizado es la covarianza del ruido del proceso.

### 3.4. Node MCU ESP 8266

El firmware NodeMCU fue creado poco después de aparecer el ESP8266, el 30 de diciembre de 2013. Unos meses después, en octubre de 2014 se publicó la primera versión del firmware NodeMCU en Github. Dos meses más tarde se publicaba la primera placa de desarrollo NodeMCU, denominada devkit v0.9, siendo también Open Hardware.

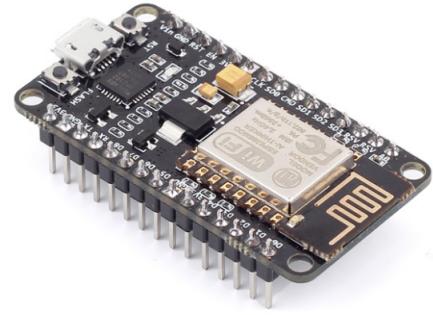


Figura 1: MCU ESP 8266

### 3.5. MUX 74HC4051

El 74HC4051 es un dispositivo útil que puede multiplexar (mux) o demultiplexar (demux) hasta 8 señales analógicas en una sola señal analógica.

Tiene un primo de 16 bits, el 74HC4067, que puede usarse para mux / demux hasta 16 señales.

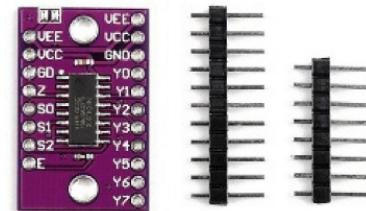


Figura 2: MUX 74HC4051

### 3.6. Sensor de calidad del Aire MQ135

Este sensor se encarga de la detección de concentración de gas en diversos porcentajes, tal y como los son sus análogos MQ-3/4/5. La señal de salida que proporciona el MQ-135 es dual, de carácter analógico y digital.



Figura 3: Sensor de calidad del aire MQ135

MQ135	Amoniaco	100-300
	Alcohol	10-300
	Benceno	10-1000
	NOX, Humo y Dióxido de Carbono	No especificado

Tabla 1: Valores de Gases.

### 3.7. Sensor de Rayos UV GYML8511

El sensor ML8511 UV (ultravioleta) funciona al emitir una señal analógica en relación con la cantidad de luz UV detectada. Esta ruptura puede ser muy útil para crear dispositivos que adviertan al usuario de quemaduras solares o detecten el índice UV en relación con las condiciones climáticas.

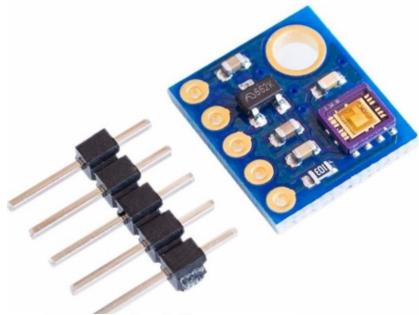


Figura 4: Sensor de Rayos UV

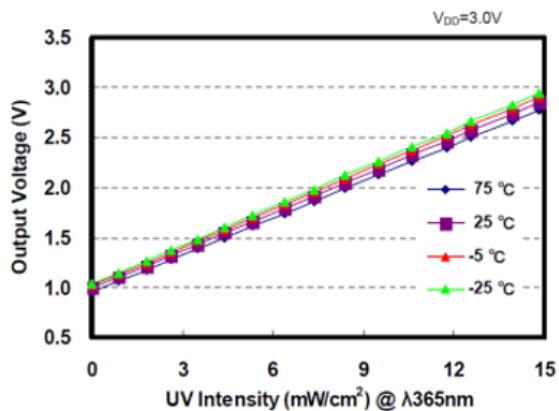


Figura 5: Comportamiento del Sensor



Figura 6: Índice de rayos UV

### 3.8. Sensor de CO (monóxido de carbono) MQ7

Está compuesto por un sensor electro-químico que varía su resistencia al estar en contacto con las sustancias. Los sensores de gases son dispositivos con alta inercia, es decir, la respuesta necesita tiempos largos para estabilizarse tras un cambio de concentración de los gases medidos.



Figura 7: Sensor de CO MQ7

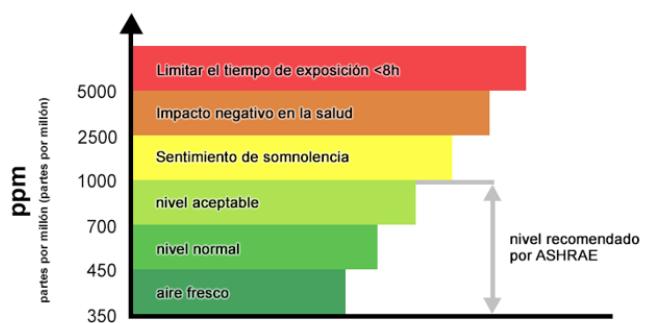


Figura 8: Escala de monóxido de Carbono

#### 3.8.1. Sensor de Temperatura y Humedad DHT11

El DHT11 es un sensor digital de temperatura y humedad relativa de bajo costo y fácil uso. Integra un sensor capacitivo de humedad y un ter-

mistor para medir el aire circundante, y muestra los datos mediante una señal digital en el pin de datos (no posee salida analógica).

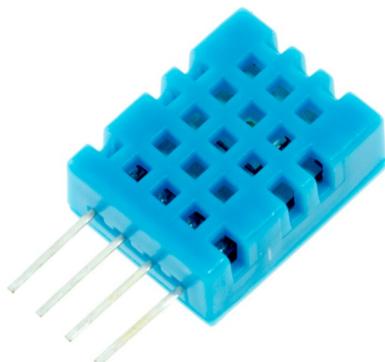


Figura 9: Sensor de Temperatura y Humedad DHT11

Tipo de sensor	Parámetro	Rango de medida	Valores Obtenidos
DHT11	Temperatura Humedad	De 0 a 50 °C	25°C
Sensor MQ-7	Monóxido Carbono (CO)	20 a 2000ppm	657 ppm
Sensor MQ-135	Amoniaco (NH3) Sulfuro Benceno (C6H6) Humo	De 10 ppm a 1000 ppm	300 ppm
Sensor Rayos UV GYM L8511	UV	Índice de 1 a 11 Índice > 12 (extremadamente alto)	Entre un índice de 3 a 4

Figura 10: Escala de Sensores

## 4. Desarrollo

### 4.1. Filtro de suavizado Exponencial-Gaussiano

```

457 int valuesLength = sizeof(values) / sizeof(int);
458
459 int getMeasure()
460 {
461     int static index = 0;
462     index++;
463     return values[index - 1];
464 }
465
466 void setup()
467 {
468     Serial.begin(115200);
469
470     for (int iCount = 0; iCount < valuesLength; iCount++)
471     {
472         int value = getMeasure();
473         int filteredLP = EMALowPassFilter(value);
474         Serial.print(value);
475         Serial.print(",");
476         Serial.println(filteredLP);
477     }
478 }
479
480 void loop()
481 {
482     delay(10000);
483 }
484
485 int EMALowPassFilter(int value)
486 {
487     EMA_LP = EMA_ALPHA * value + (1 - EMA_ALPHA) * EMA_LP;
488     return EMA_LP;
489 }
```

Figura 11: Filtro de suavizado Exponencial-Gaussiano

### 4.2. Filtro de suavizado Medio

```

486 // circularBufferAccessors == circularBuffer + windowSize
487 void setup()
488 {
489     Serial.begin(115200);
490
491     for (int iCount = 0; iCount < valuesLength; iCount++)
492     {
493         float med = AddValue(getMeasure());
494         Serial.print(values[iCount]);
495         Serial.print(",\t");
496         Serial.println(med);
497     }
498 }
499 void loop()
500 {
501 }
```

Figura 12: Filtro de suavizado Medio

```

455 int valuesLength = sizeof(values) / sizeof(int);
456
457 int getMeasure()
458 {
459     int static index = 0;
460     index++;
461     return values[index - 1];
462 }
463
464 int appendToBuffer(int value)
465 {
466     *circularBufferAccessor = value;
467     circularBufferAccessor++;
468     if (circularBufferAccessor >= circularBuffer + windowSize)
469         circularBufferAccessor = circularBuffer;
470 }
471
472 long sum;
473 int elementCount;
474 float mean;
475 float AddValue(int value)
476 {
477     sum -= *circularBufferAccessor;
478     sum += value;
479     appendToBuffer(value);
480
481     if (elementCount < windowSize)
482         ++elementCount;
483     return (float) sum / elementCount;
484 }
```

Figura 13: Filtro de suavizado Medio

### 4.3. Filtro de suavizado Mediano

Filtro\_Mediano

```

1 const int windowSize = 3;
2 int buffer>windowSize];
3 int medianBuffer>windowSize];
4 int* medianBufferAccessor = medianBuffer;
5
6 #define MEDIAN(a, n) a[((n)&1)?((n)/2):(((n)/2)-1))];
7
8 int values[] = {
9 0.20,
10 1.10,
11 0.20,
12 0.80,
```

Figura 14: Filtro de Suavizado Mediano

```

459 int valuesLength = sizeof(values) / sizeof(int);
460
461 int getMeasure()
462 {
463     int static index = 0;
464     index++;
465     return values[index - 1];
466 }
467
468 int appendToBuffer(int value)
469 {
470     *medianBufferAccessor = value;
471     medianBufferAccessor++;
472     if (medianBufferAccessor >= medianBuffer + windowSize)
473         medianBufferAccessor = medianBuffer;
474 }
475
476 int elementCount;
477 float AddValue(int value)
478 {
479     appendToBuffer(value);
480
481     if (elementCount < windowSize)
482         ++elementCount;
483 }
484

```

Figura 15: Filtro de Suavisado Mediano

```

485 void setup()
486 {
487     Serial.begin(115200);
488
489     float timeMean = 0;
490     for (int iCount = 0; iCount < valuesLength; iCount++)
491     {
492         int value = getMeasure();
493         long timeCount = micros();
494
495         AddValue(value);
496         memcpy(buffer, medianBuffer, sizeof(medianBuffer));
497         QuickSortAsc(buffer, 0, elementCount - 1);
498         int med = MEDIAN(medianBuffer, windowSize);
499
500         timeCount = micros() - timeCount;
501         timeMean += timeCount;
502         Serial.print(value);
503         Serial.print(",");
504         Serial.println(med);
505     }
506
507     Serial.println(timeMean / valuesLength);
508 }
509

```

Figura 16: Filtro de Suavisado Mediano

```

510 void loop()
511 {
512 }
513
514 void QuickSortAsc(int* arr, const int left, const int right)
515 {
516     int i = left, j = right;
517     int tmp;
518
519     int pivot = arr[(left + right) / 2];
520     while (i <= j)
521     {
522         while (arr[i] < pivot) i++;
523         while (arr[j] > pivot) j--;
524         if (i <= j)
525         {
526             tmp = arr[i];
527             arr[i] = arr[j];
528             arr[j] = tmp;
529             i++;
530             j--;
531         }
532     };
533
534     if (left < j)
535         QuickSortAsc(arr, left, j);
536     if (i < right)
537         QuickSortAsc(arr, i, right);
538 }

```

Figura 17: Filtro de Suavisado Mediano

## 5. Análisis de Resultados

Para la recolección de datos se ha tomado muestras de dos ambientes. El primer ambiente corresponde a un sector urbano y el segundo ambiente corresponde a un ambiente cercano al tubo de escape de un automóvil, se ha escogido estos dos ambientes con el fin de obtener variación en los datos y a la hora de filtrar las señales poder analizar los datos filtrados de una manera muy eficiente y de esta forma empezar a investigar en distintos ambientes que se pueda obtener acceso. A continuación se muestran las gráficas obtenidas de cada filtro implementado en cada uno de los sensores (MCU,DHT11,MQ7,MQ135,UV) demostrando así los datos muestreados y filtrados, tomando en cuenta que la señal muestreada está representada por el color azul y la señal filtrada está representada por el color rojo. De acuerdo con los filtros analizados el mejor filtro de suavizado ha sido el filtro Gaussiano nel cual ha dado mejores valore MSE con respecto a las muestras que hemos aplicado este filtro. Por lo tanto este filtro es el más ideal para emplearlo en nuestro proyecto

### 5.1. Análisis de Resultados con el Filtro Medio

#### 5.1.1. Datos de Humedad-Sensor DHT11

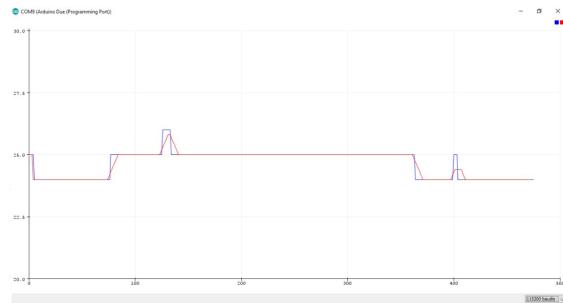


Figura 18: Representación gráfica de Datos Muestreados y Filtrados

### 5.1.2. Datos de Temperatura-Sensor DHT11

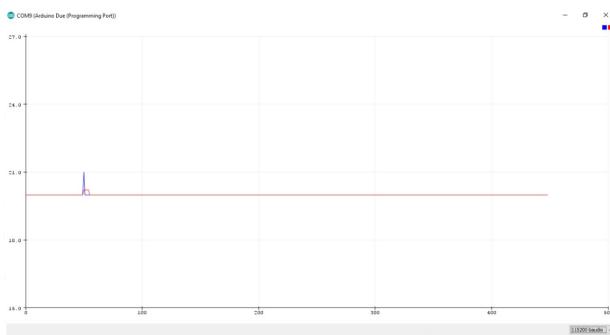


Figura 19: Representación gráfica de Datos Muestreados y Filtrados

### 5.1.3. Datos de CO(Monóxido de Carbono)-Sensor MQ7

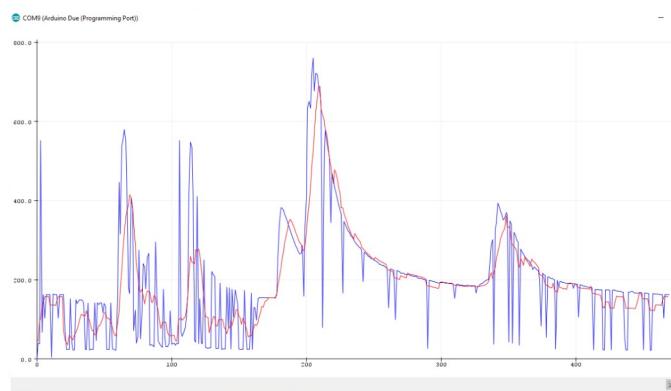


Figura 20: Representación gráfica de Datos Muestreados y Filtrados

### 5.1.4. Datos de la calidad del aire -Sensor MQ135

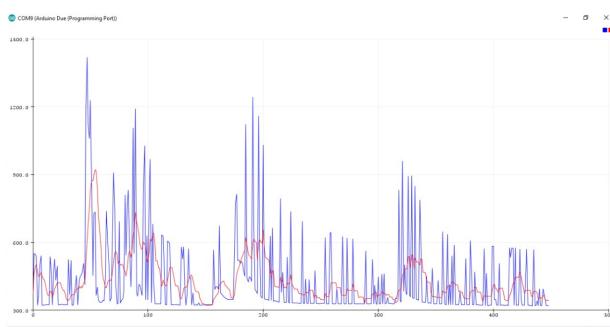


Figura 21: Representación gráfica de Datos Muestreados y Filtrados

### 5.1.5. Datos de rayos UV-Sensor GYML8511

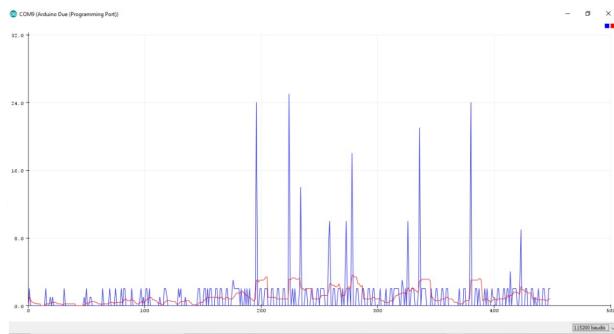


Figura 22: Representación gráfica de Datos Muestreados y Filtrados

## 5.2. Análisis de Resultados con el Filtro Exponencial-Gaussiano

### 5.2.1. Datos de Humedad-Sensor DHT11

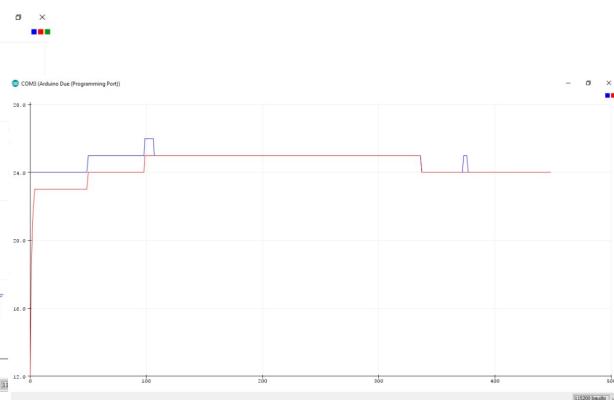


Figura 23: Representación gráfica de Datos Muestreados y Filtrados

### 5.2.2. Datos de Temperatura-Sensor DHT11



Figura 24: Representación gráfica de Datos Muestreados y Filtrados

### 5.2.3. Datos de CO(Monóxido de Carbono)-Sensor MQ7

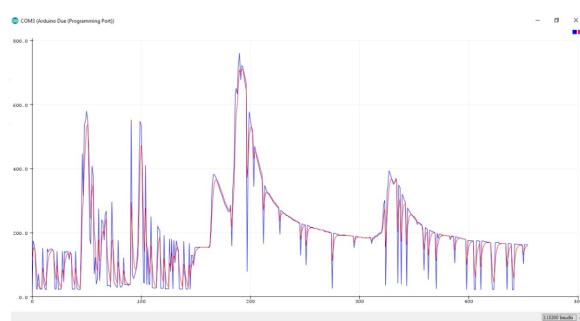


Figura 25: Representación gráfica de Datos Muestreados y Filtrados

### 5.2.4. Datos de la calidad del aire -Sensor MQ135

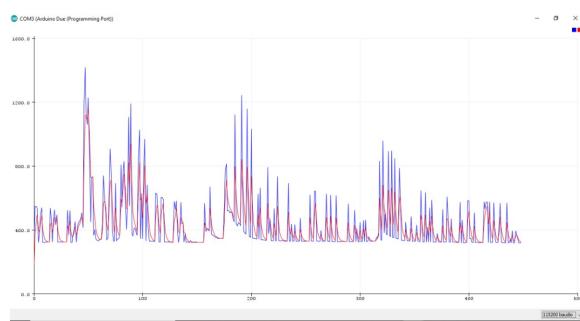


Figura 26: Representación gráfica de Datos Muestreados y Filtrados

### 5.2.5. Datos de rayos UV-Sensor GYML8511

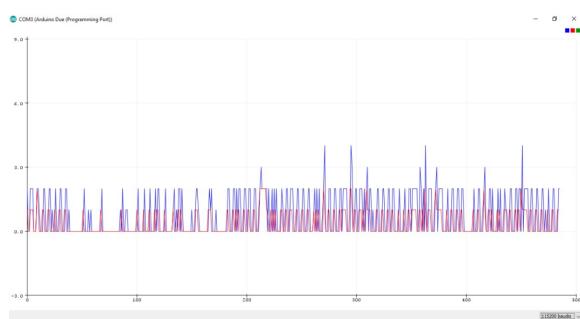


Figura 27: Representación gráfica de Datos Muestreados y Filtrados

### 5.3. Análisis de Resultados con el Filtro Mediano

#### 5.3.1. Datos de Humedad-Sensor DHT11

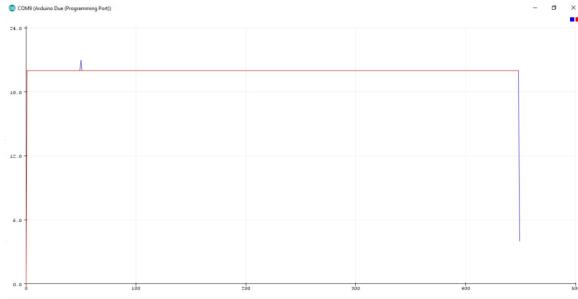


Figura 28: Representación gráfica de Datos Muestreados y Filtrado

#### 5.3.2. Datos de Temperatura-Sensor DHT11

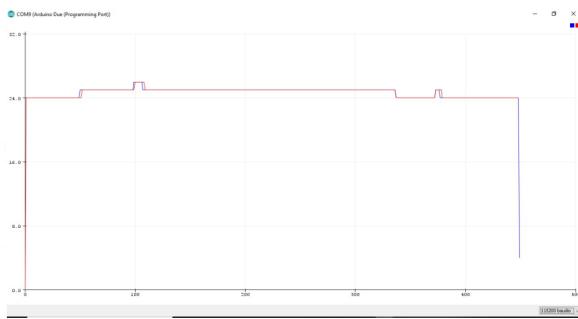


Figura 29: Representación gráfica de Datos Muestreados y Filtrados

#### 5.3.3. Datos de CO(Monóxido de Carbono)-Sensor MQ7

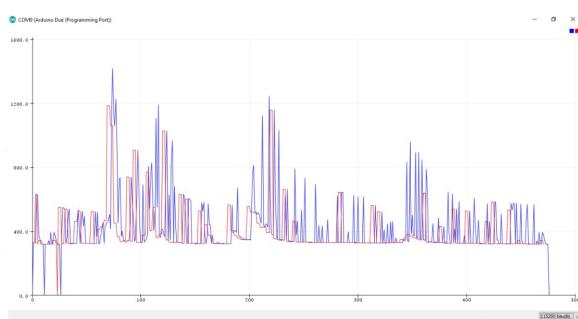


Figura 30: Representación gráfica de Datos Muestreados y Filtrados

### 5.3.4. Datos de la calidad del aire -Sensor MQ135

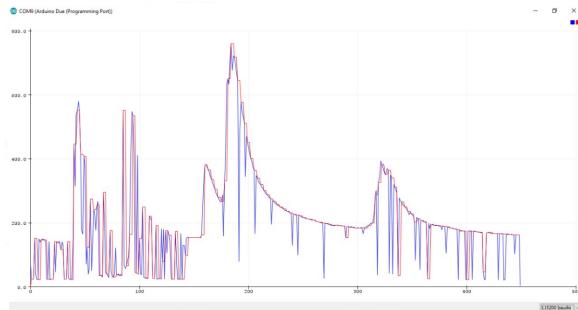


Figura 31: Representación gráfica de Datos Muestreados y Filtrados

### 5.3.5. Datos de rayos UV-Sensor GYML8511

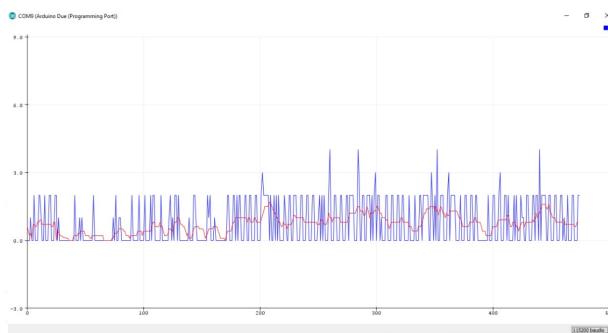


Figura 32: Representación gráfica de Datos Muestreados y Filtrados

### 5.3.6. Tabla de Resultados

	Filtro Medio	Filtro Gaussiano	Filtro Mediano
Humedad	0,032561247	0,668151448	1,29844098
Temperatura	0,001781737	0,412026726	0,893095768
CO	24945,88629	7860,122494	26567,85301
CO2	8577,007947	2002,977728	7541,870824
Raios UV	5,511632071	0,405345212	0,851356125

Figura 33: Tabla de Resultados

- Para el sensor de humedad el filtro que mejor suaviza la señal es el filtro Medio, ya que presenta una menor distancia entre los puntos que se están suavizando, por lo que se tiene un menor valor de MSE.
- Para el sensor de Temperatura el filtro que mejor suaviza la señal es el filtro Medio, ya que presenta una menor distancia entre los puntos que se están suavizando, por lo que

se tiene un menor valor de MSE.

- Para el sensor de CO el filtro que mejor suaviza la señal es el filtro Gaussiano, ya que presenta una menor distancia entre los puntos que se están suavizando, por lo que se tiene un menor valor de MSE .
- Para el sensor de CO2 el filtro que mejor suaviza la señal es el filtro Gaussiano, ya que presenta una menor distancia entre los puntos que se están suavizando, por lo que se tiene un menor valor de MSE.
- Para el sensor de Rayos UV el filtro que mejor suaviza la señal es el filtro Gaussiano, ya que presenta una menor distancia entre los puntos que se están suavizando, por lo que se tiene un menor valor de MSE.

### 5.4. Diagrama de Bloques

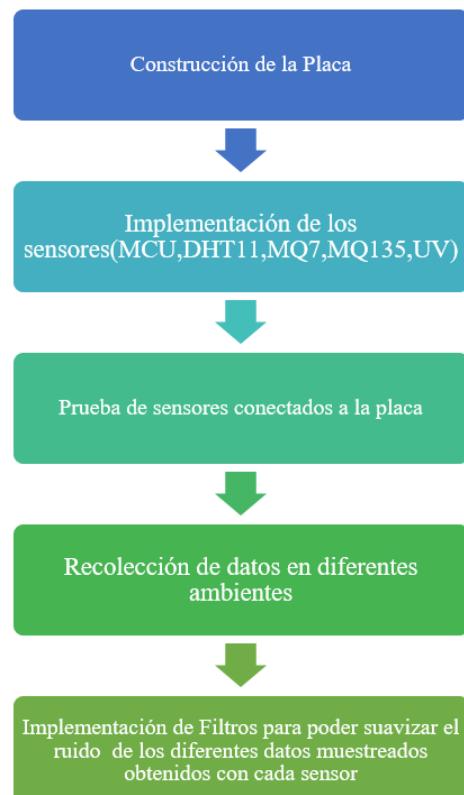


Figura 34: Diagrama de Bloques

## 5.5. Flujograma de Actividades

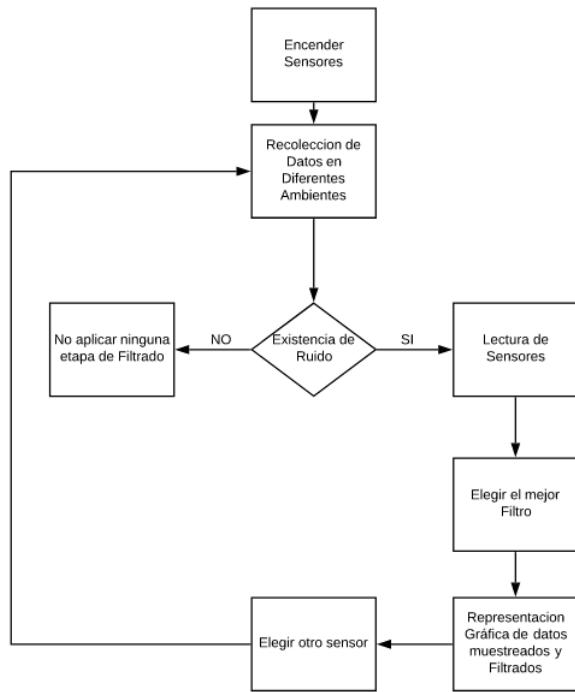


Figura 35: Flujograma de Actividaes

## 5.7. Circuito Esquemático

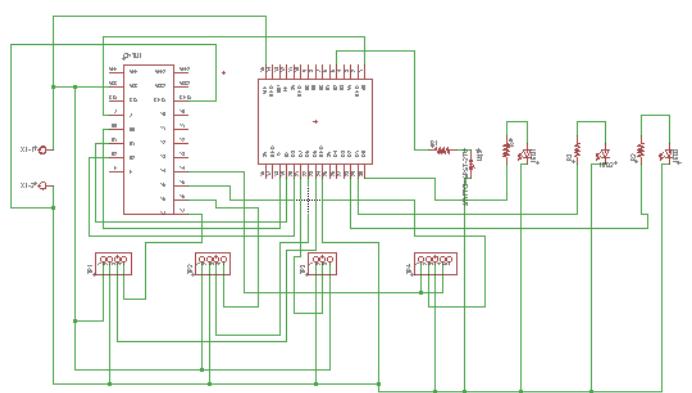


Figura 37: Circuito Esquemático

## 5.6. Simulación en Fritzing

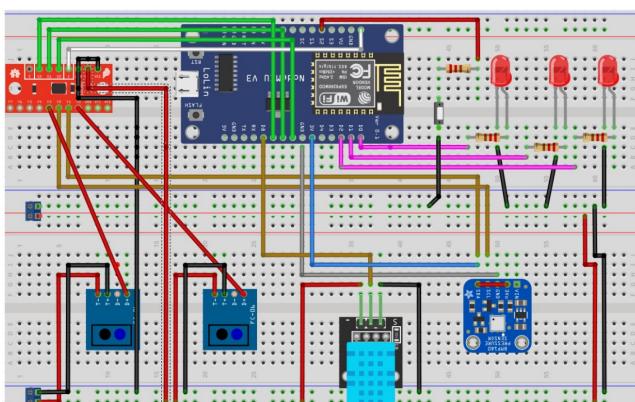


Figura 36: Simulación en Fritzing

## 5.8. Diseño e Impresión del Circuito

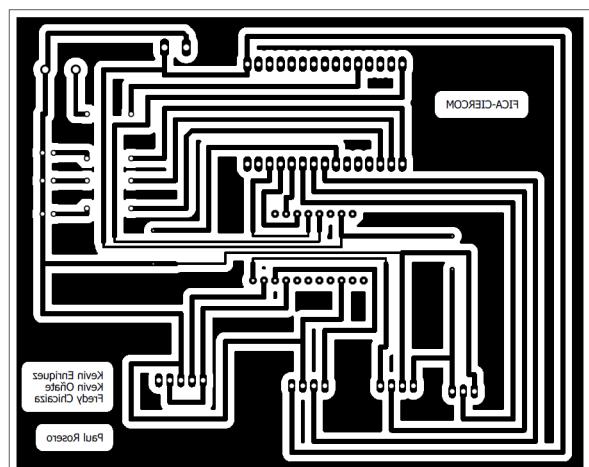


Figura 38: Diseño e Impresión del Circuito

## 5.9. Circuito implementado en Baquelita

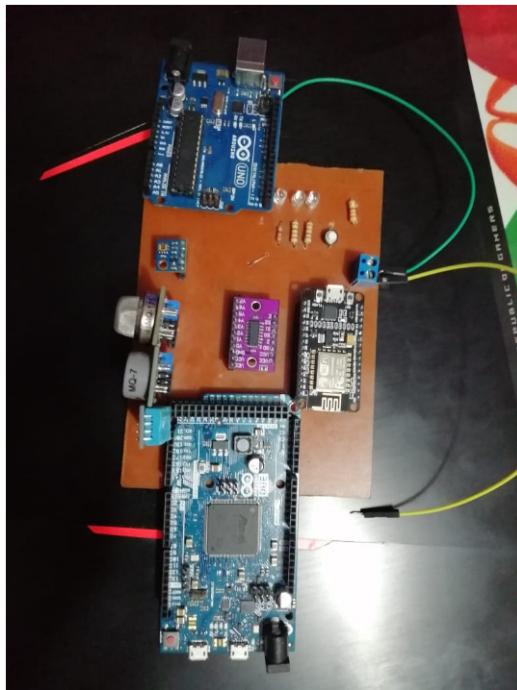


Figura 39: Circuito implementado en Baquelita

## 6. Conclusiones y Recomendaciones

### Conclusiones:

- El Filtro de suavizado Gaussiano nos ha dado mejores valore MSE con respecto a las muestras que hemos aplicado este filtro. Por lo tanto este filtro es el más ideal para emplearlo en nuestro proyecto.
- Los sensores MQ, permiten detectar el nivel de presencia de ciertos gases toxicos, por lo cual es necesario regularlos para detectar datos reales en el ambiente.
- Para determinar el umbral de funcionamiento de cada uno de los sensores, es necesario revisar sus datasheets .
- Dentro de la estructura del microprocesador ESP8266, el botón Flash permite guardar el programa dentro de la memoria del microprocesador.
- Se ha elegido como umbral mínimo 300 porque es difícil muestrear valores más bajos. Luego se ha verificado si el valor de ppm es mayor que el de maxx, si es así, maxx

tomará ese valor.

### Recomendaciones:

- Para obtener una mejor calibración se debe analizar las tablas de escala de referencia y valores, tanto del sensor de rayos UV como los sensores MQ.
- Para no tener errores en la ejecución del código, se debe incluir las librerías que no estén actualizadas ya que las versiones anteriores ejecutan un buen proceso de recolección de datos referente al sensor.
- Quemar los sensores de 11 a 48 horas ya que esto mejora su funcionamiento a la hora de recolectar los datos.

## 7. Bibliografía

- [1] B. Fu, X. Xiong, and G. Sun, “An efficient mean filter algorithm,” in The 2011 IEEE/ICME International Conference on Complex Medical Engineering, May 2011, pp. 466–470.
- [2] P. Shrivastava and U. P. Singh, “Noise removal using first order neighborhood mean filter,” in 2014 Conference on IT in Business, Industry and Government (CSIBIG), March 2014, pp. 1–6.
- [3] P. K. Sinha and Q. H. Hong, “An improved median filter,” IEEE Transactions on Medical Imaging, vol. 9, no. 3, pp. 345–346, Sep 1990.
- [4] N. C. Gallagher, “Median filters: a tutorial,” in 1988., IEEE International Symposium on Circuits and Systems, June 1988, pp. 1737–1744 vol.2.
- [5] Z. Wang and D. Zhang, “Progressive switching median filter for the removal of impulse noise from highly corrupted images,” IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing, vol. 46, no. 1, pp. 78–80, Jan 1999.
- [6] T. Chen, K.-K. Ma, and L.-H. Chen, “Tri-state median filter for image denoising,” IEEE Transactions on Image Processing, vol. 8, no. 12, pp. 1834–1838, Dec 1999.
- Joan Ribas Lequerica.,(2015).Arduino para jóvenes y no tan jóvenes,España:ANAYA MULTIMEDIA.
- Germán Tojeiro Calaza.,(2014).Taller de Arduino: un enfoque práctico para principiantes,Colombia:ALFAOMEGA.

## 8. Anexos



Figura 40: Planchado del circuito en Baquelita



Figura 42: Equipo de Trabajo



Figura 41: Perforación de Baquelita

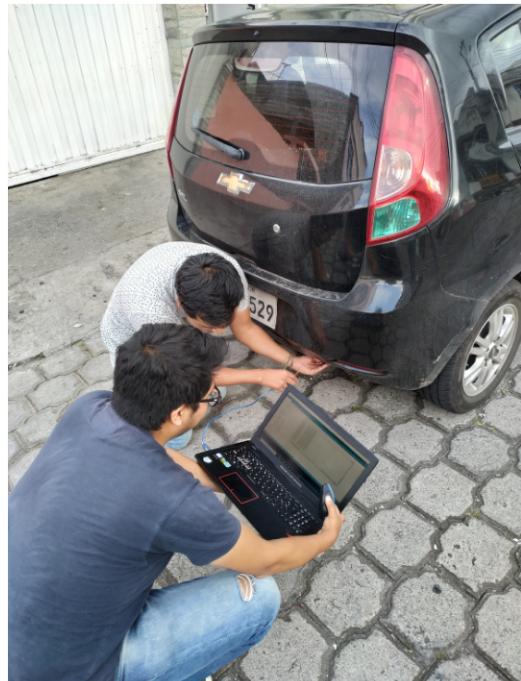


Figura 43: Recolección de datos en el tubo de escape de un automóvil