SISTEMA DE REALIDAD AUMENTADA CON LENTE MONOCULAR PARA INFORMACIÓN AMBIENTAL BÁSICA Y SIGNOS VITALES BÁSICOS

Brandon Arturo Ruiz García, Ing. *Maria Areli Sanchez García, Ing.*
Magalli Lora Dolores, Ing. *Mauricio Olguín Carbajal, Dr.* *Israel Rivera Zarate, M. en C.*

Instituto Politécnico Nacional
Centro de Innovación y Desarrollo Tecnológico en Cómputo

brandon.ruizgarcia@gmail.com *mariaareli137@hotmail.com*
magalli.lora@outlook.com *molguinc@ipn.mx* *irivera@ipn.mx*

RESUMEN

Se presenta la propuesta de un prototipo de lentes de realidad aumentada, que mostrara diversas mediciones, tales como: temperatura ambiente, ubicación, ritmo cardiaco y temperatura del usuario. Todo esto a través de un prisma y un micro display ubicados en la parte frontal de los lentes, de igual forma se utilizará una tarjeta Arduino nano para el procesamiento de los datos y el desarrollo de una aplicación móvil para el sistema operativo Android. Este dispositivo tiene el propósito de ayudar a las personas a mantenerse informadas de estos parámetros y apoyarlos en sus actividades físicas.

Introducción

La realidad aumentada actualmente está experimentando un fuerte impacto en el mercado tecnológico y esto es debido a la amplia gama de usos que puede tener, desde el sector educativo hasta el área médica. Esta tecnología se ayuda principalmente de dispositivos móviles como son

teléfonos inteligentes y/o tabletas, sin embargo, gracias a los avances tecnológicos se han comenzado a desarrollar otro tipo de dispositivos como son visores o lentes, los cales brindan una mejor integración con el mundo real.
Por este motivo se presenta la propuesta de un prototipo de lentes de realidad aumentada que ayude a los usuarios a monitorear sus signos vitales y las condiciones climáticas del punto donde se encuentre, mientras realiza alguna actividad física.

Los signos vitales serán obtenidos por sensores montados en el armazón de los lentes y que deberá colocarse la persona, esta información será procesada por una tarjeta Arduino y mostrada en un micro display. Las mediciones de temperatura y ubicación se conseguirán de una aplicación desarrollada para el sistema operativo Android, la cual se conectará a un API de servicio web la cual recibe como parámetros la latitud y longitud de donde se desea conocer el clima.

Desarrollo

En primera instancia se desarrolló un módulo en Arduino para obtener los datos de interés que se envían directamente a la placa (temperatura y frecuencia cardiaca). Para esto se utilizó un sensor LM35 con la salida calibrada a 1°C por cada 10 mV y su rango abarca desde los -55°C hasta los 150°C además que requiere de poca corriente para funcionar por lo que es muy útil para el propósito del proyecto. Para hacer el muestreo de frecuencia cardiaca se utilizó un sensor de pulso amplificado ADA-1093, que contiene internamente circuitos de amplificación y cancelación de ruido en un tamaño muy reducido y como lo que se busca es utilizar dispositivos pequeños para no afectar la ergonomía de los lentes, lo convierte en un elemento bastante funcional.

Dentro del código de Arduino para poder trabajar la temperatura se tiene que considerar la salida del LM35, donde los datos obtenidos por el sensor son analógicos y tendrán valores que oscilaran entre 0 y 1023. El voltaje de alimentación del pin puede variar entre los 0v y 5v (5000mV), por lo que si el valor en el pin es 0 entonces se tendrán 0v y si es 1023 se tendrán 5v. de forma que es posible hacer una relación de la siguiente manera:

$$C = \frac{5000mV}{1024}$$

Conociendo esta relación se puede calcular el voltaje que nos da el sensor, multiplicando el valor obtenido por la constante de relación quedando de la siguiente forma:

$$E = C * V$$

Sin embargo, para conocer la temperatura equivalente se tiene que realizar una operación más, anteriormente se mencionó que cada 10mV representaba un aumento de 1°C entonces se tiene que:

$$T = \frac{E}{10mV}$$

Dónde:

C = Constante de relación de valor analógico

E = voltaje censado por LM35

V = valor analógico censado por LM35

T = temperatura

Posteriormente para emplear el sensor ADA-1093 se requiere el uso de la librería ***PulseSensorPlayground.h***, la cual tiene los métodos necesarios para poder conocer la frecuencia cardiaca del usuario. Solo se tiene que definir un valor de umbral, el cual servirá para determinar cuál señal se tomara como un pulso y cual se ignora, por defecto el fabricante sugiere que sea de 550. Una vez definido esto se procede a revisar que el sensor esté funcionando y enviando información, mediante el uso del método begin(),si todo es correcto, se pasara a la sección para comenzar a muestrear, estos datos son obtenidos haciendo una llamada a la función getBeatsPerMinute(), el cual entrega un resultado tipo entero que representa los latidos por minuto del usuario.

Una vez obtenida la fórmula de temperatura e implementado el código de frecuencia se despliega en una micro pantalla, la cual trabaja con la librería de Adafruit SSD1306, que permite mostrar la información de manera sencilla.

A continuación se muestra un diagrama de flujo del funcionamiento del código Arduino.

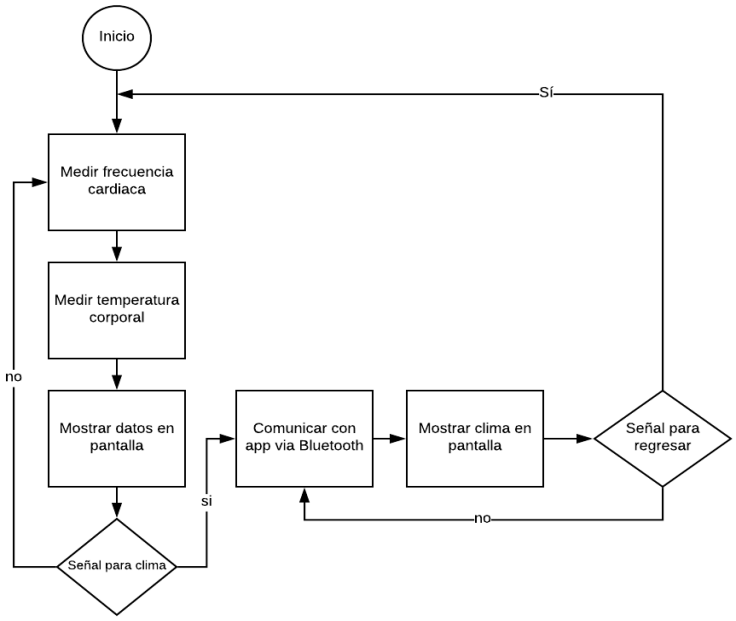

Figura 1. Diagrama de flujo del programa en Arduino.

Para el desarrollo de la aplicación en Android, lo primero es indicar en el archivo ***AndroidManifest.xml*** los permisos necesarios para el correcto funcionamiento, siendo estos los siguientes: Internet, GPS y Bluetooth.

Estos son agregados utilizando la siguiente sitaxis:

<uses-permission Android:name=”Android.permission.***recurso***”/>

Una vez realizado esto se puede proceder a obtener la latitud y la longitud del dispositivo, estos dos datos son los más importantes ya que mostraran el nombre de la calle en la que se encuentra el usuario y posteriormente como parámetros para consumir la API que brindara la información del clima. Para poder conocer las coordenadas de latitud y longitud dentro la aplicación es necesario hacer uso de un LocationManager, que es la clase que dará acceso al servicio de localización del dispositivo móvil. Asi mismo agregar el proveedor que utilizará el servicio, que en este caso será el GPS. Sin embargo esta información no es muy explícita para el usuario por lo que para poder ubicar el nombre de la calle en la que está situado es necesario implementar la clase Geocoder, la cual en su método getFromLocation() que recibe como parámetros los dos datos que ya se generaron, muestra los nombres de las calles. A continuación, se muestra el código necesario para realizar estas operaciones.

```
public void miDireccion(Location loc) throws IOException {
   double longg, latt;
   latt = loc.getLatitude();
   longg = loc.getLongitude();
```

```
    if(latt != 0.0 && longg != 0.0){
        Geocoder geocoder = new Geocoder(this, Locale.getDefault());
        List<Address> lista = geocoder.getFromLocation(loc.getLatitude(), loc.getLongitude(), 1);
        if(!lista.isEmpty()){
            Address address = lista.get(0);
            direccion = "Direccion : " + address.getAddressLine(0);
            tDir = direccion;
        }
    }
}
```

Una vez realizado esto se puede desplegar la ubicación del usuario en su dispositivo móvil, Este es el primer parámetro que se planteó obtener desde la aplicación, como segundo se tiene el clima respecto a las coordenadas en que se ubique el celular inteligente. Para esto es necesario hacer uso de una API de servicio web, ya que no se cuenta con un sensor que nos pueda medir este dato. El recurso a consumir es el que brinda la compañía OpenWeather, la cual es gratis y es de tipo REST (Transferencia de Estado Representacional), por lo cual son necesarios archivos JSON para poder conocer la información.

La comunicación de la aplicación con Arduino se hizo a través de Bluetooth con el módulo HC-06, para lograr esto se tuvo usó la librería SoftwareSerial.h de Arduino, la cual permite configurar el módulo para poder conectarlo con la aplicación. En android se creó un socket con el cual se envían flujos con los datos que se mostraran en el display, a continuación, se muestran los métodos empleados.

```
private class ConnectAsyncTask extends AsyncTask<BluetoothDevice, Integer, BluetoothSocket>{
    private BluetoothSocket mmSocket;
    private BluetoothDevice mmDevice;
    @Override
    protected BluetoothSocket doInBackground(BluetoothDevice... device) {
        mmDevice = device[0];
        try {
            String mmUUID = "00001101-0000-1000-8000-00805F9B34FB";
            mmSocket =
mmDevice.createInsecureRfcommSocketToServiceRecord(UUID.fromString(mmUUID));
            mmSocket.connect();
        } catch (Exception e) { }
        return mmSocket;
    }
    @Override
    protected void onPostExecute(BluetoothSocket result) {
        btSocket = result;
        //Enable Button
```

```
    //btToggle.setEnabled(true);
}
```

Para el diseño del armazón se utilizó el programa de modelado 3D Blender, ya que es software libre y brinda muchas herramientas para que se pueda generar un prototipo bastante atractivo, además que brinda las funcionalidades necesarias para poder exportar el proyecto en un archivo aceptado por otros software para imprimir en 3D.

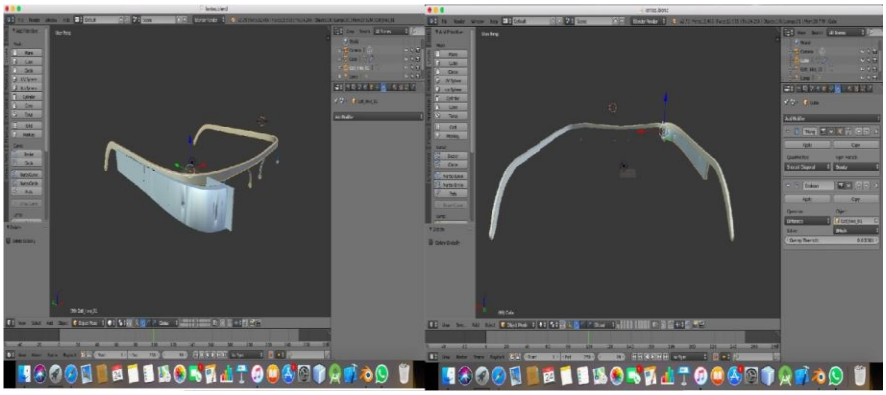

Figura 2. Diseño en Blender de los lentes.

Las siguientes imágenes muestran el circuito utilizado para montar la pantalla, sensores y Arduino, además de una captura de pantalla del diseño preliminar de la aplicación en Android.

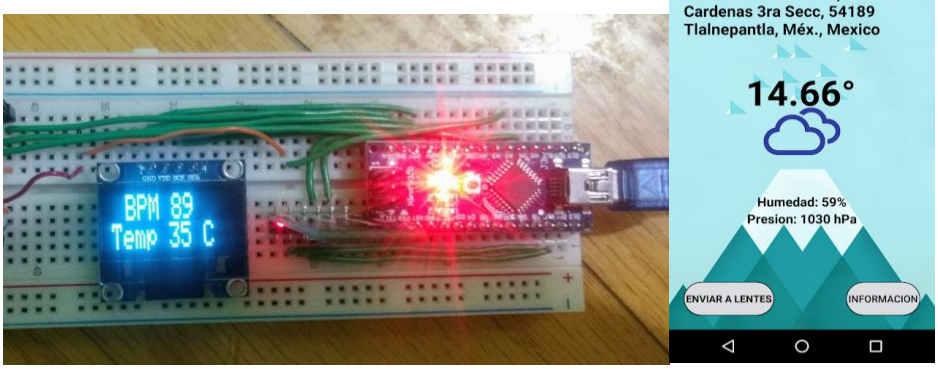

Figura 3. Circuito utilizado e interfaz de la aplicación.

Conclusiones

Arduino es una tarjeta basada en un microcontrolador ATmega328, y gracias a esto permite la elaboración de proyectos a un bajo costo y con buen rendimiento, además de que el tamaño de este

permite crear dispositivos de volúmenes reducidos, lo cual es una parte importante para el desarrollo de este proyecto.

En este proyecto se utilizaron distintas tecnologías que ayudaron al correcto funcionamiento, hasta el momento se tiene el circuito y la aplicación funcionando ya en conjunto, restaría montar todo en el armazón y construir el prisma que reflejara la información mostrada en el display

Referencias

S.Quiand, W. Jark. (1998). The penta-prism LTP_ long-trace-profiler with stationary optical fead and moving penta prism, AIP Review of Scientific Instruments

S.Inagaki, Yoshihisa Nishigori. (1998). Head-mounted display US5742264 A, Matsushita Electric Industrial Co. Ltd

Arduino. Arduino Nano Recuperado: Septiembre 17, 2018 from: https://www.arduino.cc/en/Main/arduinoBoardNano

Texas Instruments. (1999). LM35 Precision Centigrade Temperature Sensors Recuperado: Septiembre 17, 2018 from: http://www.ti.com/lit/ds/symlink/lm35.pdf

AG Electronica (2015). Sensor de pulso amplificado Recuperado: Septiembre 18, 2018 from: http://www.agspecinfo.com/pdfs/A/ADA1093.PDF