

AGENDA



Justificación



Objetivos



Selección



Desarrollo



Implementación



Análisis y Resultados



Conclusiones y Recomendaciones

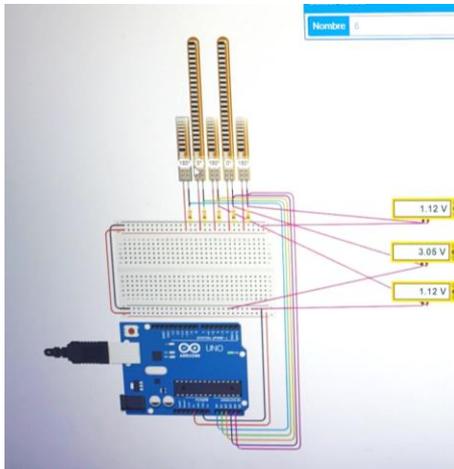


JUSTIFICACIÓN

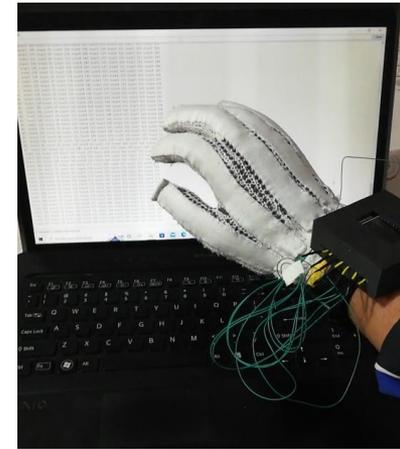
Este proyecto surge de la necesidad de contribuir a los individuos con discapacidad auditiva para lograr comunicarse con lo demás de la sociedad, es por esto que se han desarrollado diferentes tipos de herramientas, en medio de las cuales está una tecnología llamada como estimuladores vibro táctiles, los cuales son capaces de notar la vibraciones emitidas por los diversos tipos de sonidos sirviendo como apoyo a los discapacitados para hacer una lectura un poco más acertada de los labios de una persona hablante



Con el desarrollo e implementación de un sistema de interpretación y traducción del alfabeto dactilográfico del lenguaje de señas, se busca contribuir de manera significativa a mejorar las relaciones interpersonales de las personas que padecen de discapacidades del oído y del habla (sordomudos) en todos los aspectos que para nosotros “personas del común” pueden resultar muy básicas.

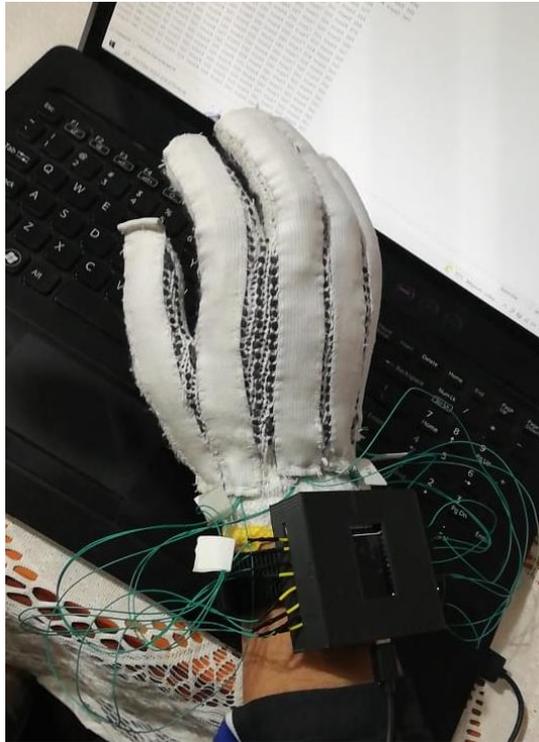


El sistema electrónico desarrollado cuenta con un sistema de reconocimiento de las señas y gestos más utilizados en el Lenguaje de Señas Ecuatoriano, permitiendo elevar la eficiencia del proceso de comunicación de una persona con discapacidad de habla. Como resultado las personas con discapacidad auditiva y del habla van a contar con una herramienta portable, ergonómica y eficaz que les permite comunicarse de manera fluida mejorando su desenvolvimiento dentro de la sociedad.



OBJETIVOS

OBJETIVO GENERAL

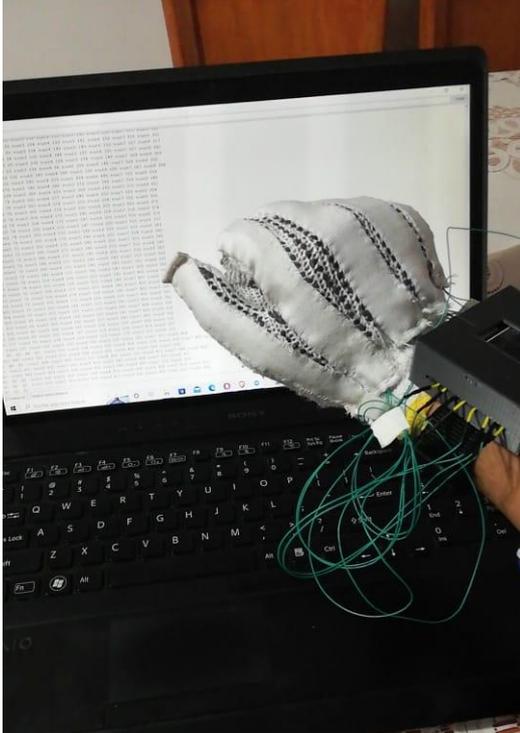


Implementar un dispositivo de interpretación de lenguaje de señas para mejorar la comunicación de personas con discapacidad auditiva y/o del lenguaje.



ESPE
UNIVERSIDAD DE LAS FUERZAS ARMADAS
INNOVACIÓN PARA LA EXCELENCIA

OBJETIVOS ESPECÍFICOS



Analizar el diccionario de Lengua de Señas Ecuatoriano “Gabriel Román”, determinando el abecedario.



Implementar un guante sensorizado que permita identificar las principales señas utilizadas por las personas con discapacidad auditiva y de lenguaje.



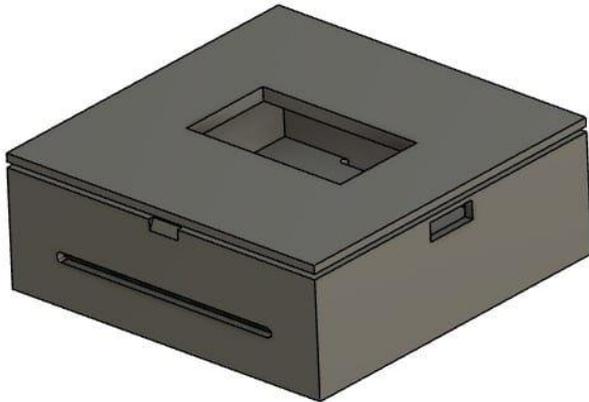
Validar la funcionalidad del dispositivo a través de pruebas para asegurar el correcto desempeño de los sistemas.



DESARROLLO

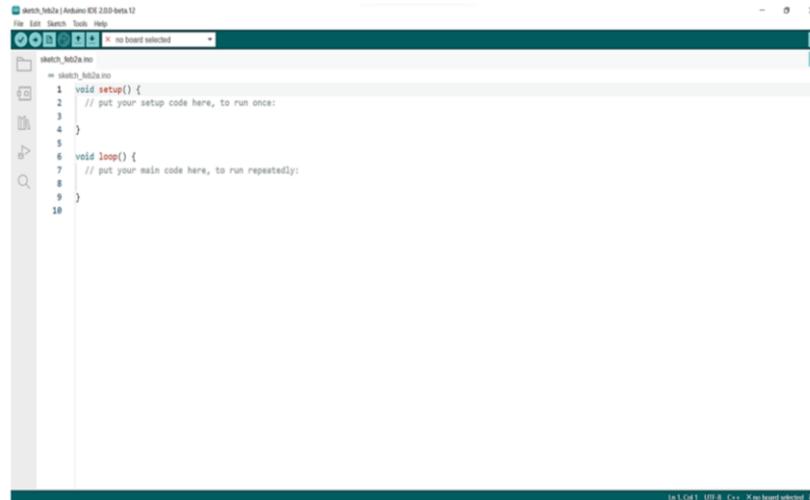
Diseño del prototipo

Para el desarrollo del presente proyecto se necesitó realizar el diseño de una carcasa que se va a utilizar para la implementación de nuestro dispositivo, para lo que se necesitó de un software de dibujo 3D. Se requirió del uso del software de SolidWorks, la cual servirá de soporte para la placa de Arduino y la placa del circuito.



Desarrollo de la programación

Para el desarrollo del programa se utilizó la interfaz de programación de Arduino. Se utilizó este software, ya que es flexible y fácil de usar permitiendo realizar cualquier tipo de aplicación. La programación fue realizada tomando en cuenta las condiciones de funcionamiento requeridas para el guante traductor de señas. Dicha programación en la parte inicial consta de la declaración de librerías, así como las variables a ser utilizadas.

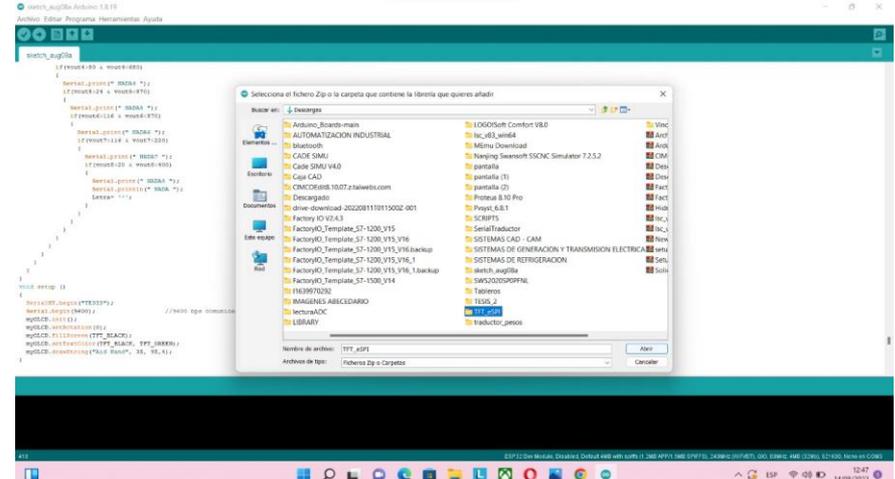
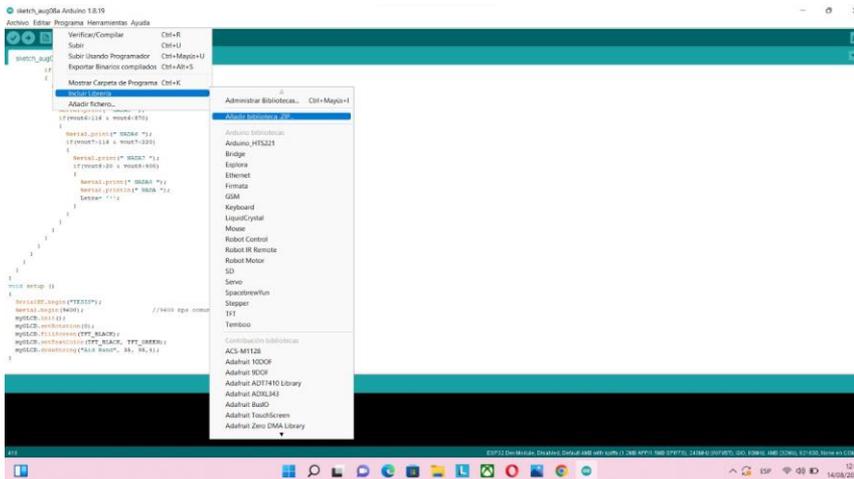


```
sketch_hb2a.no | Arduino IDE 2.0.9-beta.12
File Edit Sketch Tools Help
no board selected
sketch_hb2a.no
1 void setup() {
2 // put your setup code here, to run once:
3
4 }
5
6 void loop() {
7 // put your main code here, to run repeatedly:
8
9 }
10
```



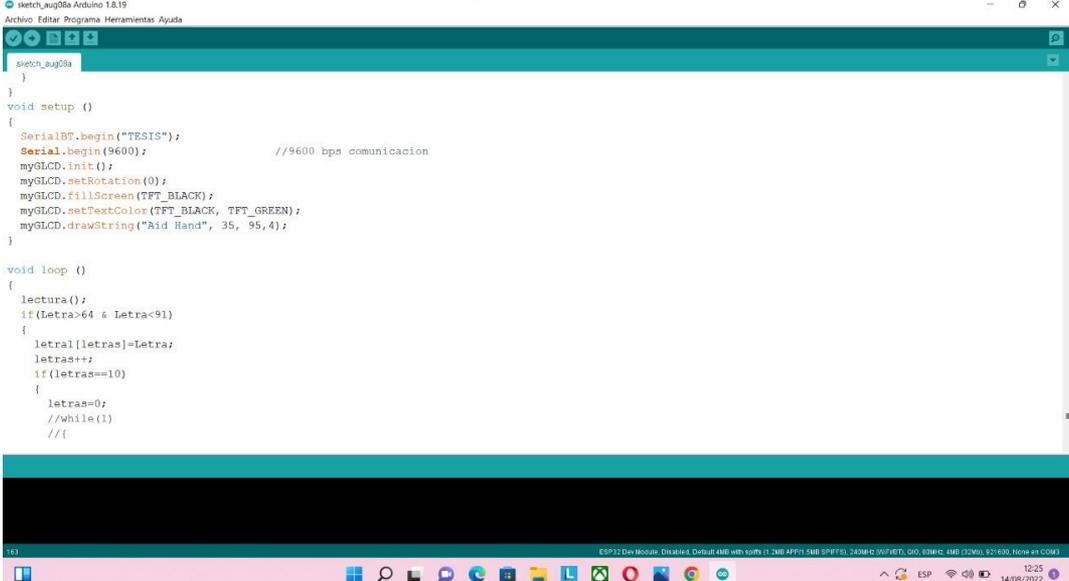
A. Codificación de las librerías

Las librerías a utilizar para el desarrollo de la programación son las siguientes: TFT_eSPI, estas librerías son las que trabajan con los sensores, pantalla. En algunas ocasiones no es necesario la instalación de las mismas porque el programa ya viene con estas, si no será necesario instalarlas desde el mismo software.



B. Desarrollo de la programación para la pantalla

Se muestra el código de programación empleado para la visualización de cada seña del abecedario dactilográfico realizada, ya sea esta de letras y números. Para la programación se utilizó las condiciones int, if y volatile boolean, las mismas que servirán para declarar las variables, evaluar si cumple la condición y pueda continuar con su funcionamiento.



```
sketch_aug08a
}
}
void setup ()
{
  SerialBT.begin("TESIS");
  Serial.begin(9600); //9600 bps comunicacion
  myGLCD.init();
  myGLCD.setRotation(0);
  myGLCD.fillScreen(TFT_BLACK);
  myGLCD.setTextColor(TFT_BLACK, TFT_GREEN);
  myGLCD.drawString("Aid Hand", 35, 95,4);
}

void loop ()
{
  lectura();
  if (Letra>64 & Letra<91)
  {
    letra[Letra]=Letra;
    letras++;
    if (letras==10)
    {
      letras=0;
      //while(1)
      //{}
    }
  }
}
```



```
sketch_aug08a Arduino 1.8.19
Archivo Editar Programa Herramientas Ayuda
sketch_aug08a

Serial.print(" letras ");
delay(100);
Serial.print(letral[0]);
delay(100);
Serial.print(letral[1]);
delay(100);
Serial.print(letral[2]);
delay(100);
Serial.print(letral[3]);
delay(100);
Serial.print(letral[4]);
delay(100);
Serial.print(letral[5]);
delay(100);
Serial.print(letral[6]);
delay(100);
Serial.print(letral[7]);
delay(100);
Serial.print(letral[8]);
delay(100);
Serial.println(letral[9]);
letral[0]=' ';
letral[1]=' ';
letral[2]=' ';
letral[3]=' ';
letral[4]=' ';
letral[5]=' ';
letral[6]=' ';
letral[7]=' ';
letral[8]=' ';
letral[9]=' ';
delay(100);
myGLCD.setTextSize(4);
myGLCD.fillScreen(TFT_BLACK);
myGLCD.setTextColor(TFT_GREEN, TFT_BLACK);

if(Letra=='A') myGLCD.drawString("A", 35, 95,4);
```

163 ESP32 Dev Module, Disabled, Default 4MB with spiiffs (1.2MB APP/1.5MB SPIFFS), 240MHz (WiFi/BT), QIO, 80MHz, 4MB (32Mb), 921600, None en COM3

```
sketch_aug08a Arduino 1.8.19
Archivo Editar Programa Herramientas Ayuda
sketch_aug08a

letral[0]=' ';
letral[8]=' ';
letral[9]=' ';
delay(100);
myGLCD.setTextSize(4);
myGLCD.fillScreen(TFT_BLACK);
myGLCD.setTextColor(TFT_GREEN, TFT_BLACK);

if(Letra=='A') myGLCD.drawString("A", 35, 95,4);
if(Letra=='B') myGLCD.drawString("B", 35, 95,4);
if(Letra=='C') myGLCD.drawString("C", 35, 95,4);
if(Letra=='D') myGLCD.drawString("D", 35, 95,4);
if(Letra=='E') myGLCD.drawString("E", 35, 95,4);
if(Letra=='F') myGLCD.drawString("F", 35, 95,4);
if(Letra=='G') myGLCD.drawString("G", 35, 95,4);
if(Letra=='H') myGLCD.drawString("H", 35, 95,4);
if(Letra=='I') myGLCD.drawString("I", 35, 95,4);
if(Letra=='J') myGLCD.drawString("J", 35, 95,4);
if(Letra=='K') myGLCD.drawString("K", 35, 95,4);
if(Letra=='L') myGLCD.drawString("L", 35, 95,4);
if(Letra=='M') myGLCD.drawString("M", 35, 95,4);
if(Letra=='N') myGLCD.drawString("N", 35, 95,4);
if(Letra=='O') myGLCD.drawString("O", 35, 95,4);
if(Letra=='P') myGLCD.drawString("P", 35, 95,4);
if(Letra=='Q') myGLCD.drawString("Q", 35, 95,4);
if(Letra=='R') myGLCD.drawString("R", 35, 95,4);
if(Letra=='S') myGLCD.drawString("S", 35, 95,4);
if(Letra=='T') myGLCD.drawString("T", 35, 95,4);
if(Letra=='U') myGLCD.drawString("U", 35, 95,4);
if(Letra=='V') myGLCD.drawString("V", 35, 95,4);
if(Letra=='W') myGLCD.drawString("W", 35, 95,4);
if(Letra=='X') myGLCD.drawString("X", 35, 95,4);
if(Letra=='Y') myGLCD.drawString("Y", 35, 95,4);
if(Letra=='Z') myGLCD.drawString("Z", 35, 95,4);

myGLCD.setTextColor(TFT_YELLOW, TFT_BLACK);
if(Letra=='1') myGLCD.drawString("1", 35, 95,4);
```

418 ESP32 Dev Module, Disabled, Default 4MB with spiiffs (1.2MB APP/1.5MB SPIFFS), 240MHz (WiFi/BT), QIO, 80MHz, 4MB (32Mb), 921600, None en COM3



```
sketch_aug08a Arduino 1.8.19
Archivo Editar Programa Herramientas Ayuda
sketch_aug08a
if (Letra=='N') myGLCD.drawString("N", 35, 95, 4);
if (Letra=='O') myGLCD.drawString("O", 35, 95, 4);
if (Letra=='P') myGLCD.drawString("P", 35, 95, 4);
if (Letra=='Q') myGLCD.drawString("Q", 35, 95, 4);
if (Letra=='R') myGLCD.drawString("R", 35, 95, 4);
if (Letra=='S') myGLCD.drawString("S", 35, 95, 4);
if (Letra=='T') myGLCD.drawString("T", 35, 95, 4);
if (Letra=='U') myGLCD.drawString("U", 35, 95, 4);
if (Letra=='V') myGLCD.drawString("V", 35, 95, 4);
if (Letra=='W') myGLCD.drawString("W", 35, 95, 4);
if (Letra=='X') myGLCD.drawString("X", 35, 95, 4);
if (Letra=='Y') myGLCD.drawString("Y", 35, 95, 4);
if (Letra=='Z') myGLCD.drawString("Z", 35, 95, 4);

myGLCD.setTextColor(TFT_YELLOW, TFT_BLACK);
if (Letra=='1') myGLCD.drawString("1", 35, 95, 4);
if (Letra=='2') myGLCD.drawString("2", 35, 95, 4);
if (Letra=='3') myGLCD.drawString("3", 35, 95, 4);
if (Letra=='4') myGLCD.drawString("4", 35, 95, 4);
if (Letra=='5') myGLCD.drawString("5", 35, 95, 4);
if (Letra=='6') myGLCD.drawString("6", 35, 95, 4);
if (Letra=='7') myGLCD.drawString("7", 35, 95, 4);
if (Letra=='8') myGLCD.drawString("8", 35, 95, 4);
if (Letra=='9') myGLCD.drawString("9", 35, 95, 4);
SerialBT.write(Letra);

//letra en el display
while (Letra!='*')
{
  lectura();
}
Letra=' ';
Serial.println("salio del lazo");
myGLCD.fillRect(TFT_BLACK);
}
}
```

418 ESP32 Dev Module, Disabled, Default 4MB with spiffs (1.2MB APP/1.5MB SPIFFS), 240MHz (WiFi/BT), QIO, 80MHz, 4MB (32Mb), 921600, None en COM3



C. Desarrollo para la programación de los sensores flexibles

Se observa el código de programación empleado para el funcionamiento de los sensores flexibles, cuando este elemento empiece a trabajar, enviará los datos obtenidos al dispositivo móvil mediante la aplicación.

El sensor enviará un rango de datos según el movimiento que se realice dando paso a ver si cumplen con las condiciones de cada letra del abecedario dactilográfico que las personas con discapacidad auditiva y de lenguaje lo usan.

```
sketch_aug08a Arduino 1.8.19
sketch_aug08a
void lectura()
{
  //lectura sensor 1
  delayMicroseconds(1);
  vout1=analogRead(sensor1);
  vout1=vout1;
  v1t=v1t+vout1;

  delayMicroseconds(1);

  //lectura sensor 2
  vout2=analogRead(sensor2);
  vout2=vout2;
  v2t=v2t+vout2;
  delayMicroseconds(1);

  //lectura sensor 3
  vout3=analogRead(sensor3);
  vout3=vout3;
  v3t=v3t+vout3;
  delayMicroseconds(1);

  //lectura sensor 4
```

```
sketch_aug08a Arduino 1.8.19
sketch_aug08a
//lectura sensor 4
vout4=analogRead(sensor4);
vout4=vout4;
v4t=v4t+vout4;
delayMicroseconds(1);

//lectura sensor 5
vout5=analogRead(sensor5);
vout5=vout5;
v5t=v5t+vout5;
delayMicroseconds(1);

//lectura sensor 6
vout6=analogRead(sensor6);
vout6=vout6;
v6t=v6t+vout6;
delayMicroseconds(1);

//lectura sensor 7
vout7=analogRead(sensor7);
vout7=vout7;
v7t=v7t+vout7;
delayMicroseconds(1);
```



```
sketch_aug08a Arduino 1.8.19
Archivo Editar Programa Herramientas Ayuda
sketch_aug08a
Serial.println(vout8);
Letra=' ';

if(vout1>525 & vout1<675)
{
  //Serial.print(" S1 ");
  if(vout2>275 & vout2<425)
  {
    //Serial.print(" S2 ");
    if(vout3>535 & vout3<700)
    {
      //Serial.print(" S3 ");
      if(vout4>435 & vout4<585)
      {
        //Serial.print(" S4 ");
        if(vout5>425 & vout5<590)
        {
          //Serial.print(" S5 ");
          if(vout6>400 & vout6<600)
          {
            //Serial.print(" S6 ");
            if(vout7>700 & vout7<900)
            {
              //Serial.print(" S7 ");
              if(vout8>275 & vout8<425)
              {
                //Serial.print(" S8 ");
                Serial.println(" A ");
                Letra=" A ";
              }
            }
          }
        }
      }
    }
  }
}

113
```

```
sketch_aug08a Arduino 1.8.19
Archivo Editar Programa Herramientas Ayuda
sketch_aug08a
}
if(vout1>505 & vout1<660)
{
  //Serial.print(" H1 ");
  if(vout2>255 & vout2<405)
  {
    //Serial.print(" H2 ");
    if(vout3>250 & vout3<400)
    {
      //Serial.print(" H3 ");
      if(vout4>195 & vout4<400)
      {
        //Serial.print(" H4 ");
        if(vout5>28 & vout5<670)
        {
          //Serial.print(" H5 ");
          if(vout6>19 & vout6<670)
          {
            //Serial.print(" H6 ");
            if(vout7>655 & vout7<850)
            {
              //Serial.print(" H7 ");
              if(vout8>125 & vout8<500)
              {
                //Serial.print(" H8 ");
                Serial.println(" H ");
                Letra=" H ";
              }
            }
          }
        }
      }
    }
  }
}

113
```

```
sketch_aug08a Arduino 1.8.19
Archivo Editar Programa Herramientas Ayuda
sketch_aug08a
}
if(vout1>455 & vout1<605)
{
  //Serial.print(" V1 ");
  if(vout2>150 & vout2<300)
  {
    //Serial.print(" V2 ");
    if(vout3>495 & vout3<645)
    {
      //Serial.print(" V3 ");
      if(vout4>95 & vout4<245)
      {
        //Serial.print(" V4 ");
        if(vout5>130 & vout5<280)
        {
          //Serial.print(" V5 ");
          if(vout6>25 & vout6<175)
          {
            //Serial.print(" V6 ");
            if(vout7>700 & vout7<850)
            {
              //Serial.print(" V7 ");
              if(vout8>350 & vout8<500)
              {
                //Serial.print(" V8 ");
                Serial.println(" V ");
                Letra=" V ";
              }
            }
          }
        }
      }
    }
  }
}

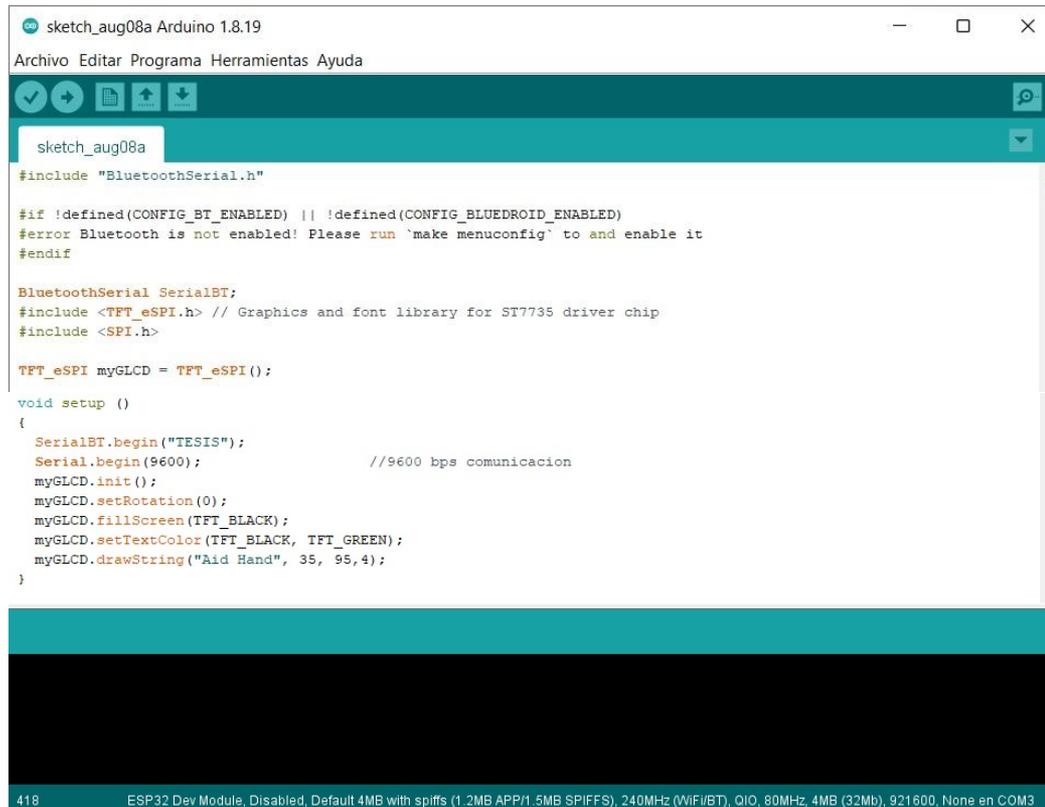
113
```



ESPE
UNIVERSIDAD DE LAS FUERZAS ARMADAS
INNOVACIÓN PARA LA EXCELENCIA

D. Desarrollo para la programación de la conexión por bluetooth

Se muestra la programación utilizada para la conexión por medio de bluetooth del dispositivo, el mismo que cumple con el protocolo de comunicación para enviar los datos a la aplicación móvil.



```
sketch_aug08a Arduino 1.8.19
Archivo Editar Programa Herramientas Ayuda
sketch_aug08a
#include "BluetoothSerial.h"

#if !defined(CONFIG_BT_ENABLED) || !defined(CONFIG_BLUEDROID_ENABLED)
#error Bluetooth is not enabled! Please run `make menuconfig` to and enable it
#endif

BluetoothSerial SerialBT;
#include <TFT_eSPI.h> // Graphics and font library for ST7735 driver chip
#include <SPI.h>

TFT_eSPI myGLCD = TFT_eSPI();

void setup ()
{
  SerialBT.begin("TESIS");
  Serial.begin(9600); //9600 bps comunicacion
  myGLCD.init();
  myGLCD.setRotation(0);
  myGLCD.fillScreen(TFT_BLACK);
  myGLCD.setTextColor(TFT_BLACK, TFT_GREEN);
  myGLCD.drawString("Aid Hand", 35, 95,4);
}

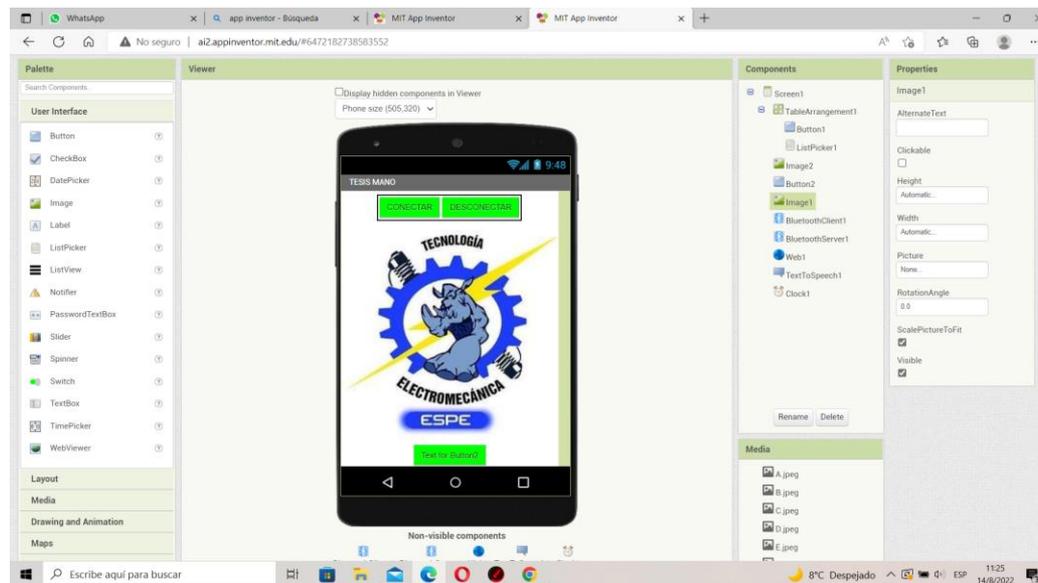
418 ESP32 Dev Module, Disabled, Default 4MB with spiiffs (1.2MB APP/1.5MB SPIFFS), 240MHz (WiFi/BT), QIO, 80MHz, 4MB (32Mb), 921600, None en COM3
```



Desarrollo de la aplicación móvil

Para el desarrollo de la aplicación móvil se utilizó la interfaz de la aplicación App Inventor, donde vamos a tener la inicialización de todas las variables iniciales incluyendo los archivos correspondientes a cada letra.

Después se configura la conexión bluetooth tanto para que se inicie el mismo como para la desconexión que se realiza con los botones en la interfaz y la recepción de datos enviados desde la placa.



MIT APP INVENTOR

MANOF

Screen1

Blocks

Viewer

when Button1.Click

do call BluetoothClient1.Disconnect

when ListPicker1.BeforePicking

do set ListPicker1.Elements to BluetoothClient1.AddressesAndNames

when ListPicker1.AfterPicking

do set ListPicker1.Selection to call BluetoothClient1.Connect address ListPicker1.Selection

when Clock1.Timer

do if call BluetoothClient1.BytesAvailableToReceive >= 0

then set global DATO to call BluetoothClient1.ReceiveText numberOfBytes call BluetoothClient1.BytesAvailableToReceive

set Button2.Text to get global DATO

set Image1.Picture to 1.jpeg

if get global DATO = A

then set Image1.Picture to A.jpeg

call TextToSpeech1.Speak message A

Show Warnings

if get global DATO = B

then set Image1.Picture to B.jpeg

call TextToSpeech1.Speak

Media

1.jpeg

2.jpeg

MIT APP INVENTOR

MANOF

Screen1

Blocks

Viewer

if get global DATO = C

then set Image1.Picture to C.jpeg

call TextToSpeech1.Speak message C

Show Warnings

if get global DATO = D

then set Image1.Picture to D.jpeg

call TextToSpeech1.Speak message D

Show Warnings

if get global DATO = E

then set Image1.Picture to E.jpeg

call TextToSpeech1.Speak message E

Show Warnings

Media

A.jpeg

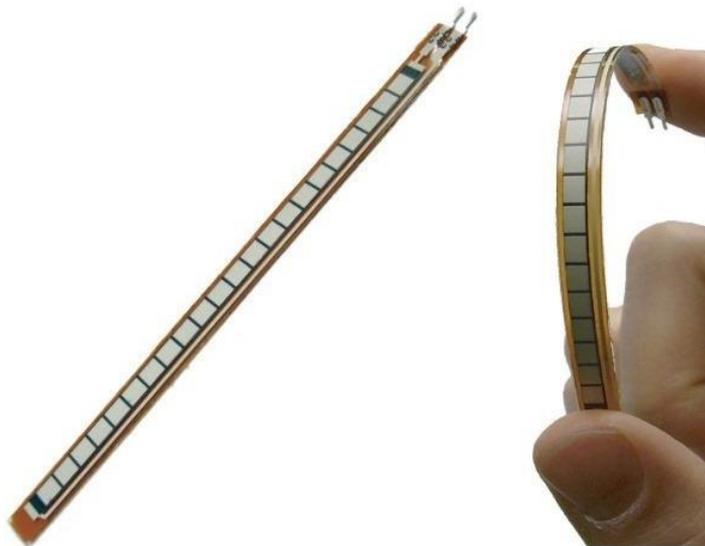
B.jpeg

C.jpeg



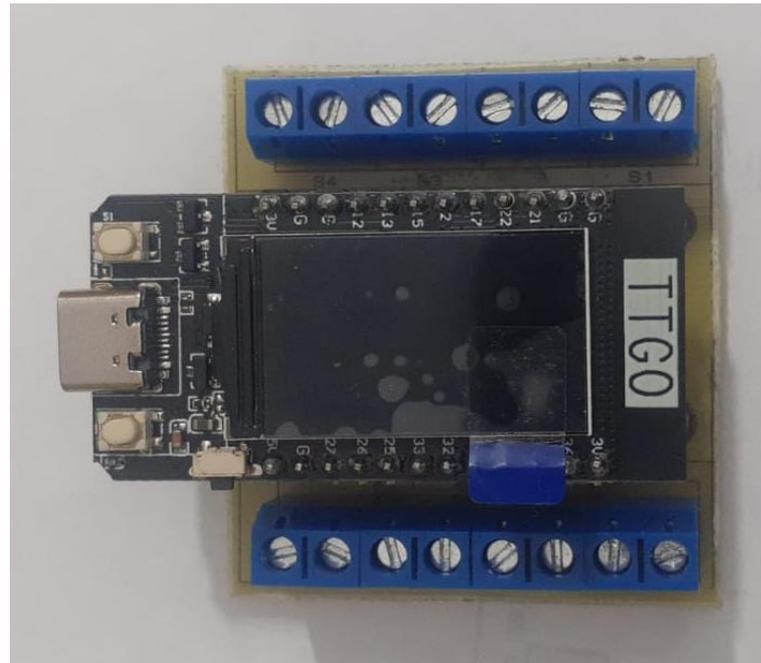
Implementación del prototipo

Para la implementación del prototipo, se consideró primordial la utilización de sensores resistivos, los cuales nos van a contribuir a obtener un rango de datos según el movimiento de cada uno de los dedos conforme corresponda a la señal del lenguaje dactilográfico.



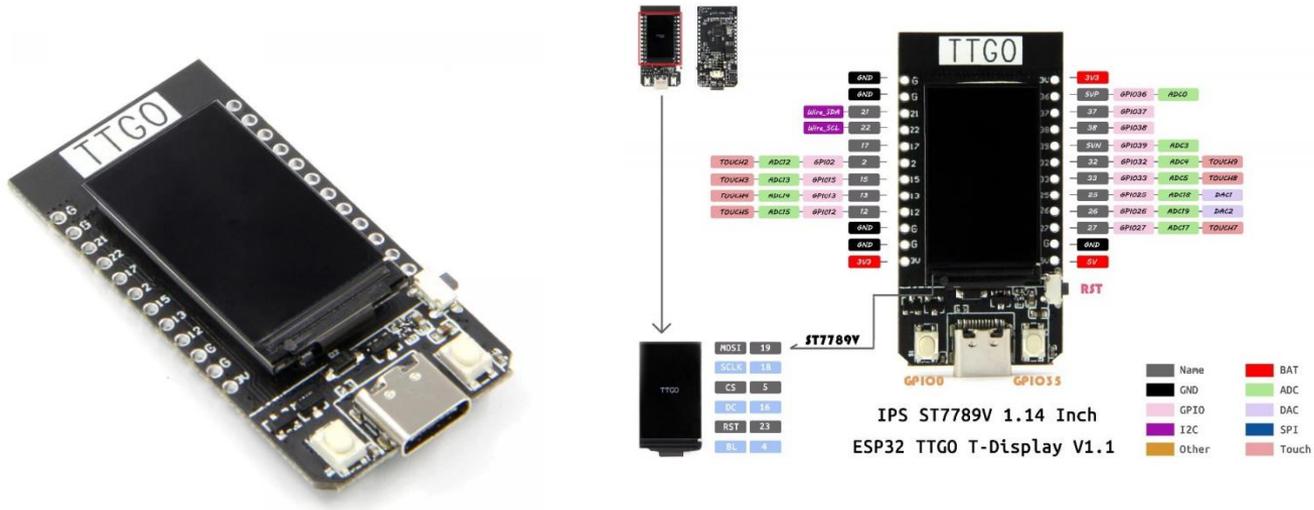
Implementación del sistema electrónico

Para poner en marcha el sistema electrónico se lo hizo con ayuda de cableado, por las dimensiones y tamaños de los dispositivos a ser usados. Se dio paso a la identificación de los pines a usar tanto del Arduino, como de los sensores y pantalla.



A. Acoplamiento de la Tarjeta ESP32 TTGO

Es tarjeta basada en ESP32, a diferencia de otras placas es que cuenta con algunas características de hardware adicionales como: pantalla a color de 1.14 pulgadas, interfaz para carga de baterías litio y 2 botones. Al igual que cuenta con Wifi y Bluetooth.



B. Acoplamiento de los sensores flexibles

Para el acoplamiento de los sensores flexibles se utilizó cables de red flexible, el mismo que nos ayudará a la correcta conexión de cada uno de estos a su respectiva entrada de la placa.



ANÁLISIS Y RESULTADOS



¿El dispositivo cumple con su función?

¿Existen avances en la comunicación de las personas con discapacidad?

El dispositivo de interprete de señas es una herramienta útil para la enseñanza?

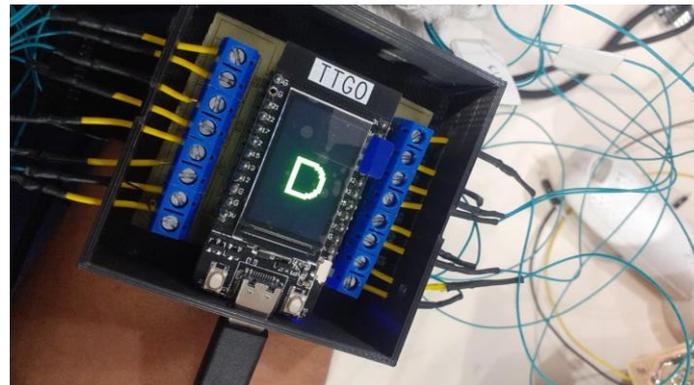


ESPE
UNIVERSIDAD DE LAS FUERZAS ARMADAS
INNOVACIÓN PARA LA EXCELENCIA

PRUEBAS DE FUNCIONAMIENTO

Las pruebas de funcionamiento se realizaron de manera independiente, esto quiere decir que se fue obteniendo los datos de manera que se iba realizando la programación para cada sensor.

Algo muy importante que hay que recalcar en esto es que los sensores son dispositivos análogos, por lo que sus valores se van a reflejar en datos altos. Esto varía de acuerdo al tipo de actividad que se vaya a realizar ya que en este tipo de dispositivos se podría decir que se omite el valor de los rangos estipulados para las mediciones físicas.



CONCLUSIONES



Se logró elaborar un guante robótico que gracias a sensores es capaz de replicar los movimientos que la mano realiza según el lenguaje dactilográfico y que por medio de la aplicación móvil que por sonido replica lo que se hace al realizar el movimiento de la mano, esta llega a entablar una comunicación entre las personas sordomudas y la sociedad.



La investigación realizada acerca del lenguaje dactilográfico nos acercó más a la realidad que viven las personas sordomudas al momento de querer manifestarse con alguna necesidad, de igual forma la importancia de proyectos que faciliten al discapacitado entablar una comunicación con la sociedad o personas cercanas a ella.



Se logró los mayores grados de libertad previstos en el guante con el afán de que los movimientos que se realizarán sean fluidos y logren la posición necesaria según las letras del abecedario y de igual forma las frases más importantes para una persona sordomuda.



Se pudo observar en las pruebas que la interfaz de la aplicación móvil es muy intuitiva y similar a otras aplicaciones de mensajería con el fin de que el usuario final no tenga problemas al momento de recibir la señal del guante.



Se tomó como referencia un guante sensorizado del tamaño de una persona adulta con el fin de poner a prueba la flexibilidad de los sensores al gesticular una letra o frase y que de forma efectiva sea convertido en señales que lleguen en forma de texto a la aplicación móvil y luego sea reproducida.



RECOMENDACIONES



Una vez concluido el proyecto, se recomienda que si se desea tener una mejor precisión en el funcionamiento del guante se debe usar más sensores flex en los dedos representando cada falange, o a su vez se puede incluir un módulo giroscopio.

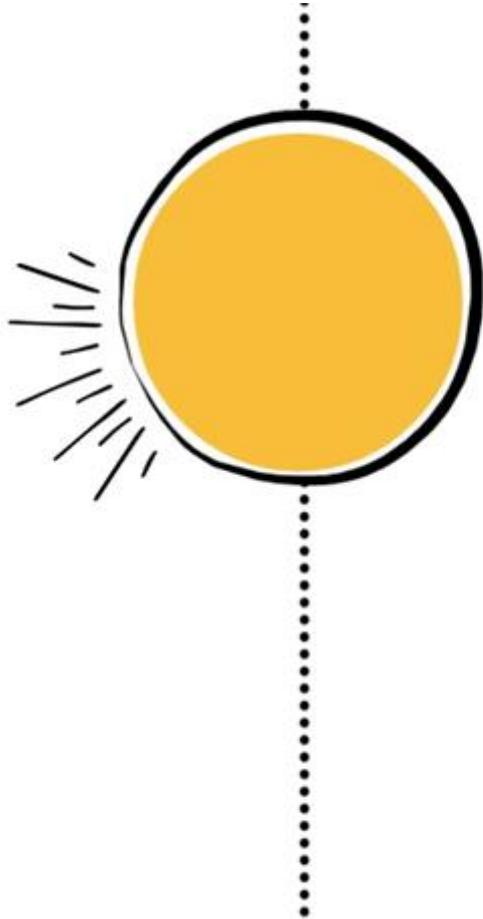


Se recomienda usar módulos wireless para establecer una mejor comunicación inalámbrica, puesto que ofrecen mejores ventajas en comparación a los módulos bluetooth tales como: mayor alcance, velocidad y manejo de datos.



Al usar la mano robótica se la debe colocar sobre una superficie plana, ya que al hacer algún movimiento brusco puede caerse o sufrir algún accidente.





GRACIAS POR
SU ATENCIÓN



ESPE
UNIVERSIDAD DE LAS FUERZAS ARMADAS
INNOVACIÓN PARA LA EXCELENCIA