



UNIVERSIDAD
DE MÁLAGA



ESCUELA DE INGENIERÍAS INDUSTRIALES

Departamento de Ingeniería de Sistemas y Automática

Área de Conocimiento: Ingeniería de Sistemas y Automática

PROYECTO FIN DE GRADO

**SISTEMA DE ADQUISICIÓN DE IMÁGENES DE PRESIÓN PARA UN
SENSOR TÁCTIL MATRICIAL RESISTIVO**

Grado en Ingeniería en Tecnologías Industriales

Autor: María Fernández Hijano

Tutor: Jesús Manuel Gómez de Gabriel

Cotutor: Juan Manuel Gandarias Palacios

Málaga, septiembre 2018

Declaración de autoría

Yo, **María Fernández Hijano**, estudiante del Grado en Ingeniería en Tecnologías Industriales en la Escuela de Ingenierías Industriales de la Universidad de Málaga, en relación con este Proyecto de Fin de Grado, titulado: “Sistema de adquisición de imágenes de presión para un sensor táctil matricial resistivo”, declaro que asumo la originalidad de dicho trabajo, entendida en el sentido de que no se han utilizado fuentes sin citarlas debidamente.

Málaga, a 10 de septiembre de 2018

A handwritten signature in black ink, appearing to read 'María F. H.' with a stylized flourish at the end.

Fdo.: María Fernández Hijano

Resumen

En este proyecto se presentan el diseño y montaje de un sistema capaz de adquirir datos de un sensor táctil, matricial (32x32), resistivo y convertirlos en una matriz de colores. El objeto de este proyecto es servir como prototipo de un sistema que sea capaz de obtener imágenes de alta resolución y precisión que puedan ser interpretadas, en proyectos futuros, mediante inteligencia artificial. Se han diseñado e implementado el sistema electrónico; el sistema físico: dos placas de circuito impreso y una estructura de conexión interior mediante impresión 3D; y el Firmware: en Arduino y Matlab.

Palabras Clave

Adquisición de datos, adquisición de imágenes, arduino, estructura de conexión, firmware, imágenes de presión, diseño PCB, sensor táctil matricial resistivo, sistema electrónico.

Índice general

| | |
|--|-----------|
| Índice de Figuras | iv |
| 1. Introducción | 1 |
| 1.1. Percepción táctil: importancia y dificultad de su implementación en robots. | 1 |
| 1.2. Sistema de Adquisición de Datos | 2 |
| 1.2.1. Sensores | 2 |
| 1.2.2. Hardware: Dispositivos de Adquisición de Datos | 3 |
| 1.2.3. PC y Software | 4 |
| 1.3. Sensor TekScan | 4 |
| 1.4. Objetivo | 5 |
| 1.5. Estructura del documento | 7 |
| 2. Diseño e implementación del sistema electrónico | 8 |
| 2.1. Diseño del sistema electrónico | 8 |
| 2.1.1. Subsistema de excitación de filas | 10 |
| 2.1.2. Subsistema de lectura de columnas | 10 |
| 2.2. Componentes utilizados | 11 |
| 2.2.1. Arduino UNO | 11 |
| 2.2.2. Shift Registers | 11 |
| 2.2.3. Multiplexores | 12 |
| 2.2.4. Placas de circuito impreso | 12 |
| 2.2.5. Pogo pins | 12 |
| 2.3. Diseño de los circuitos impresos | 13 |
| 3. Diseño del sistema físico | 15 |
| 3.1. Diseño de PCBs | 15 |
| 3.1.1. Contacto con los conectores del sensor | 15 |
| 3.1.2. Colocación de los componentes electrónicos | 16 |
| 3.1.3. Ruteo | 16 |
| 3.1.4. Taladros | 16 |
| 3.1.5. Diseño final | 17 |
| 3.1.6. Montaje y soldadura | 18 |
| 3.2. Estructura de conexión | 19 |
| 3.2.1. Diseño | 20 |

| | |
|---|-----------|
| 3.2.2. Fabricación | 21 |
| 3.3. Dispositivo completo | 22 |
| 3.4. Conclusiones | 23 |
| 4. Firmware | 24 |
| 4.1. Software de adquisición de datos | 24 |
| 4.2. Protocolo de comunicaciones | 25 |
| 4.3. Código Arduino | 26 |
| 4.3.1. Código de prueba | 26 |
| 4.3.2. Bucle principal | 27 |
| 4.3.3. Formato Hexadecimal | 27 |
| 4.4. Código Matlab | 28 |
| 4.4.1. Elección y apertura del puerto serie | 28 |
| 4.4.2. Ordenación de la matriz de datos | 28 |
| 4.4.3. Lectura de datos | 30 |
| 4.4.4. Imágenes de presión | 30 |
| 4.5. Conclusiones | 31 |
| 5. Pruebas y Resultados | 32 |
| 5.1. Puerto serie de Arduino | 32 |
| 5.1.1. Voltajes e Intensidades para un punto del sensor | 32 |
| 5.1.2. Multiplexores | 33 |
| 5.1.3. Montaje completo | 34 |
| 5.2. Imágenes de presión | 36 |
| 5.2.1. Sin ejercer presión | 36 |
| 5.2.2. Lápiz | 37 |
| 5.2.3. Pelota de gomaespuma | 38 |
| 5.2.4. Mano | 39 |
| 5.3. Conclusiones | 40 |
| 6. Conclusiones y trabajos futuros | 41 |
| 6.1. Conclusiones | 41 |
| 6.2. Trabajos futuros | 41 |
| Plano Estructura de Conexión | 44 |
| Bibliografía | 44 |
| A. Código Arduino | 46 |
| B. Código Matlab | 49 |
| C. Hoja de especificaciones Sensor Tekscan 6077 | 51 |

Índice de figuras

| | |
|--|----|
| 1.1. Partes de un Sistema de Adquisición de Datos | 2 |
| 1.2. Partes del sensor TekScan | 5 |
| 1.3. Sensor TekScan modelo 6077 | 5 |
| 1.4. “I-Scan System” de TekScan | 6 |
| 2.1. Esquema eléctrico simplificado de la electrónica de 8 bits | 9 |
| 2.2. Esquema de funcionamiento del subsistema de excitación de filas | 10 |
| 2.3. Esquema de funcionamiento del subsistema de lectura de columnas | 10 |
| 2.4. Arduino UNO | 11 |
| 2.5. Pogo Pins | 13 |
| 2.6. Funcionamiento Pogo Pin | 13 |
| 2.7. Esquema de la conexión hardware | 14 |
| 3.1. Correspondencia de Pads con canales Shift Registers | 16 |
| 3.2. Correspondencia de Pads con canales Multiplexores | 16 |
| 3.3. Diseño PCB columnas Cara Superior | 17 |
| 3.4. Diseño PCB columnas Cara Inferior | 17 |
| 3.5. Diseño PCB filas Cara Superior | 17 |
| 3.6. PCB filas Cara Inferior | 17 |
| 3.7. PCB columnas Cara Superior | 19 |
| 3.8. PCB columnas Cara Inferior | 19 |
| 3.9. PCB filas Cara Superior | 19 |
| 3.10. PCB filas Cara Inferior | 19 |
| 3.11. Diseño 3D realizado mediante el software SketchUp | 20 |
| 3.12. Pieza para PCB columnas. Cara Superior | 21 |
| 3.13. Pieza para PCB columnas. Cara Inferior | 21 |
| 3.14. Pieza para PCB filas. Cara Superior | 21 |
| 3.15. Pieza para PCB filas. Cara Inferior | 21 |
| 3.16. Montaje del dispositivo visto desde arriba | 22 |
| 3.17. Montaje del dispositivo visto desde abajo | 22 |
| 3.18. Montaje del dispositivo con el sensor acoplado | 22 |
| 4.1. Esquema de actuación del Firmware | 25 |
| 4.2. Array de colores de tipo JET | 31 |

| | |
|---|----|
| 5.1. Resultados obtenidos al ejercer fuerza en un punto del sensor, conectado a una resistencia de 5kOhmios. | 33 |
| 5.2. Resultados obtenidos al ejercer fuerza en un punto del sensor, conectado a una resistencia de 30kOhmios. | 33 |
| 5.3. Resultados obtenidos al ejercer fuerza solo en el punto conectado al Canal 12 de un multiplexor. | 34 |
| 5.4. Resultados en formato hexadecimal obtenidos al ejercer fuerza sobre el sensor. | 35 |
| 5.5. Imágenes de presión del sensor sin ejercer presión | 36 |
| 5.6. Imagen de presión de un lápiz | 37 |
| 5.7. Imagen de presión de una pelota | 38 |
| 5.8. Imagen de presión de una mano | 39 |

Capítulo 1

Introducción

1.1. Percepción táctil: importancia y dificultad de su implementación en robots.

El sentido del tacto es muy importante para los humanos, ya que se ocupa de la percepción espacial de estímulos externos a través de un gran número de receptores de diferentes tipos (mecanorreceptores, termorreceptores y nociceptores) que se distribuyen por todo el cuerpo. Las últimas tendencias en el campo de la robótica tratan de conseguir robots con características cada vez más parecidas a las humanas. No obstante, la complejidad del sentido del tacto hace que este campo todavía no se haya resuelto de forma satisfactoria [8].

Diversos estudios han descubierto que la naturaleza de la descarga eléctrica de varios receptores, en respuesta a los estímulos externos estudiados en muestras de epidermis, es piezoeléctrica. Por ello, los sensores táctiles comerciales suelen ser de esta misma naturaleza [7]. La respuesta humana al estímulo mecánico se consigue mediante mecanorreceptores que están incrustados en la piel a diferentes profundidades. Su número se estima en 241 en la punta de los dedos y 58 en la palma de la mano de un adulto humano, por centímetro cuadrado [9]. Debido, entre otras cosas, a este elevado número de receptores, simular el sentido del tacto en robots es una tarea muy complicada.

Desde el momento en que se estimula la piel hasta la percepción resultante, tienen lugar gran número de fenómenos mecánicos, perceptivos y cognitivos. En contacto con un objeto, la piel se adapta a su superficie y proyecta la deformación a un gran número de mecanorreceptores. Cada mecanorreceptor representa una pequeña porción del objeto y codifica la información táctil espaciotemporal como picos de voltaje [14]. La amplitud del estímulo se transforma en un tren de potenciales de acción -un paso similar a la digitalización y codificación de señales analógicas por un convertidor analógico/digital.

La información relacionada con el evento de contacto se transmite al sistema nervioso central para un mayor nivel de procesamiento e interpretación a través de múltiples

nervios hasta la médula espinal [14].

Por lo tanto, el modo en que los humanos perciben los estímulos mecánicos y procesan esa información puede resultar análogo a un sistema de adquisición de datos.

1.2. Sistema de Adquisición de Datos

La Adquisición de Datos (DAQ - Data Acquisition) [12] es el proceso de medir con un ordenador un fenómeno eléctrico o físico: voltaje, corriente, temperatura, presión, sonido, etc. Un sistema DAQ está formado por sensores, hardware de medidas DAQ y un ordenador con software programable, como se observa en la Figura 1.1.



Figura 1.1: Partes de un Sistema de Adquisición de Datos

1.2.1. Sensores

Un sensor es un dispositivo capaz de transformar un fenómeno físico en una señal eléctrica medible. Dependiendo del tipo de sensor, su salida eléctrica puede ser un voltaje, corriente, resistencia u otro atributo eléctrico que varía con el tiempo [4, 12, 18].

Características técnicas de los sensores:

- Exactitud: diferencia entre el valor medido y el valor real.
- Repetitividad: grado con el que las mismas mediciones varían unas con respecto a otras.
- Resolución: mínima variación de la magnitud de entrada que es medible.
- Precisión: error de medida máximo esperado.
- Sensibilidad: mínimo valor de la variable medida que produce un cambio en la salida.
- Offset: valor de la variable de salida cuando la variable de entrada es nula.
- Linealidad.

- Rango: dominio en la magnitud medida en el que puede aplicarse el sensor.
- Rapidez de respuesta: capacidad del sistema para seguir las variaciones de la entrada.

1.2.2. Hardware: Dispositivos de Adquisición de Datos

El hardware de adquisición de datos actúa como la interfaz entre las señales del mundo exterior y un ordenador - digitaliza señales analógicas entrantes para que este último pueda interpretarlas [12, 19].

Algunos dispositivos de adquisición de datos pueden incluir funciones como, por ejemplo: convertidores digitales-analógicos (DACs); líneas de Entrada/Salida digitales (E/S); temporizadores para la generación de pulsos digitales, etc. No obstante, todos tienen tres partes diferenciadas:

Acondicionamiento de Señales

Su objetivo es manipular una señal para que sea apropiada para la entrada a un convertidor analógico-digital (ADC). Este circuito puede incluir:

- Amplificación o atenuación: para conseguir la mayor precisión posible se incrementa la señal de entrada para que coincida con el rango del convertidor ADC.
- Excitación: es necesaria una excitación externa de corriente o voltaje para muchos tipos de sensores; bien por su composición interna (termistor), o bien por su configuración (galgas extensométricas, que se suelen montar en puente de Wheatstone).
- Filtrado: limpian la señal, eliminando ruidos no deseados a una cierta frecuencia.
- Aislamiento: es necesario en muchos casos un aislamiento eléctrico entre el sensor y el ordenador, para proteger al mismo de altas tensiones puntuales que puedan dañarlo.
- Linealización: es necesaria cuando los sensores producen respuestas que no están relacionadas linealmente con las magnitudes físicas que se pretenden medir.

Convertidor Analógico Digital (ADC)

Las señales analógicas de los sensores deben ser convertidas en digitales antes de ser manipuladas por el ordenador.

Un ADC es un dispositivo electrónico que proporciona una representación digital de una señal analógica en un instante de tiempo. Realmente, en la práctica, las señales

analógicas varían continuamente con el tiempo, por lo que, un ADC realiza "muestras" de la señal a intervalos periódicos. Estas son transferidas por un bus al ordenador, donde la señal es reconstruida a través de un software [12]. Existen diferentes formas de implementar un ADC. Las más comunes son: Flash o Conversión directa, Aproximaciones Sucesivas, Codificación Delta, Doble rampa, Pipeline y Sigma-Delta.

Bus de datos

Un bus es la interfaz de comunicación entre el dispositivo de adquisición de datos y el ordenador. Tiene la función de transmitir instrucciones y datos. Existen varios tipos de buses (PCI, PXI, USB, Ethernet e inalámbricos) y cada uno es más apropiado para un uso concreto. A la hora de elegir uno u otro hay que tener en cuenta: los requerimientos de Entrada/Salida; la cantidad de datos que se van a transmitir a través de él; si es necesario sincronizar varios dispositivos y la portabilidad del sistema.

1.2.3. PC y Software

El software de aplicación facilita la interacción entre el ordenador y el usuario para adquirir, analizar y presentar datos de medidas. Puede ser una aplicación pre-construida con funcionalidad predefinida o un entorno de programación para construir aplicaciones con funcionalidad personalizada que, generalmente, son usadas para automatizar múltiples funciones de un dispositivo DAQ, realizar algoritmos de procesamiento de señales y mostrar interfaces de usuario personalizadas. Se puede utilizar además un software controlador, que ofrece al software de aplicación la habilidad de interactuar con un dispositivo DAQ. Simplifica la comunicación con él al abstraer comandos de hardware de bajo nivel y programación a nivel de registro. Generalmente, el software controlador DAQ expone una interfaz de programación de aplicaciones (API) que es usada en un entorno de programación para construir softwares de aplicación.

1.3. Sensor TekScan

Se trata de un sensor táctil matricial resistivo (Figura 1.3) formado por una matriz de electrodos y sus correspondientes conectores. Se asemeja a una matriz de sensores FSR. El sensor es flexible y, por tanto, se adapta a las superficies que se quieren medir. Está disponible en muchos tamaños.

Los sensores TekScan [17] están formados por dos láminas finas y flexibles de poliéster que llevan impresas conductores eléctricos en forma de rayas: en una de ellas, en forma de filas y en la otra, en forma de columnas. Así, cuando las dos láminas se colocan una encima de la otra, se obtiene una cuadrícula, en la que las intersecciones de filas y columnas forman pequeños sensores individuales. Cuando se aplica una fuerza a estos

sensores la resistencia eléctrica cambia de forma proporcionalmente inversa a la fuerza normal aplicada. El esquema de las partes del sensor se observa en la Figura 1.2.

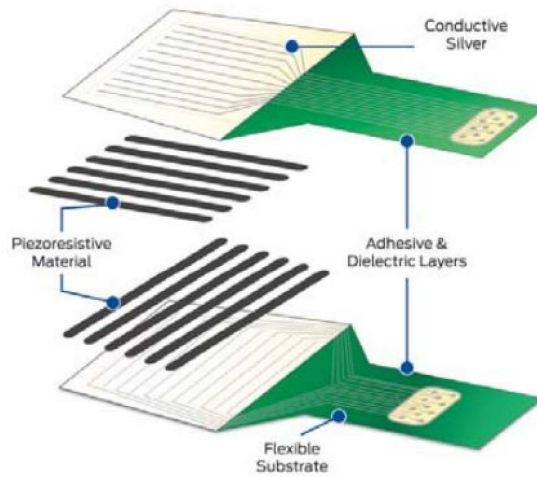


Figura 1.2: Partes del sensor TekScan

En este proyecto se ha utilizado un sensor TekScan, modelo 6077. Sus dimensiones son de 53.3mm x 95.3mm y una densidad de 27.6 sensels/ cm^2 , lo que hacen un total de 1400 sensels resistivos. Estos sensels están distribuidas en una matriz de 28 filas x 50 columnas. La hoja de especificaciones se encuentra en el Apéndice C.

Sin embargo, este proyecto alcanza hasta un sensor de 32 filas x 32 columnas, por lo que se ha utilizado solo un área del sensor.



Figura 1.3: Sensor TekScan modelo 6077

1.4. Objetivo

El objetivo principal de este proyecto consiste en diseñar e implementar el primer prototipo de un sistema de adquisición de datos para un sensor táctil, matricial -de

32x32 puntos- y resistivo; que, acoplado a una pinza moldeable, sea capaz de reproducir mediante imágenes de colores la presión ejercida en cada uno de los puntos que componen el sensor matricial.

Existen modelos comerciales, como el “I-Scan System” de TekScan [17], que puede verse en la Figura 1.4. Este dispositivo presenta grandes inconvenientes: es de muy elevado coste; su Software solo es compatible con el sistema operativo “Windows”; y, para la toma de imágenes offline, es necesario disponer de una aplicación específica y librerías, que aumentan aún más el coste del dispositivo.

Por tanto, debido a que el modelo comercial del que se dispone no resulta adecuado, en este proyecto se pretende:

- Diseñar un sistema electrónico de adquisición de datos.
- Implementar un soporte físico compacto con una estructura de conexión adecuada.
- Crear un protocolo de comunicación para la lectura de los datos.
- Programar el Firmware del dispositivo para que sea capaz de transformar los datos leídos por el sensor en imágenes de presión.
- Realizar pruebas con diferentes objetos cotidianos y discutir los resultados.

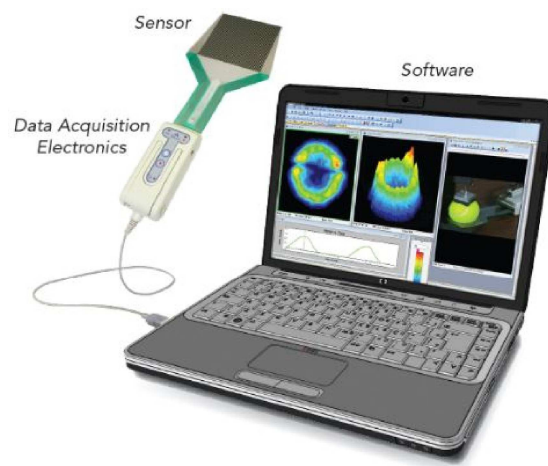


Figura 1.4: “I-Scan System” de TekScan

Las aplicaciones desarrolladas en este proyecto tienen la finalidad de resultar de utilidad en la robótica de rescate: en el reconocimiento de víctimas y de distintas partes del cuerpo para poder realizar mediciones posteriormente.

1.5. Estructura del documento

La memoria se encuentra estructurada de la siguiente forma: en el capítulo 2, se presentan el diseño y la implementación del sistema electrónico. En el capítulo 3 se describe el diseño y montaje del sistema físico: las placas de circuito impreso y la estructura de conexión de ambas. Por otro lado, en el capítulo 4, se explica el Firmware desarrollado para el proyecto, en Arduino y Matlab, así como el protocolo de comunicación entre ambos. En el capítulo 5, se muestran tanto las pruebas iniciales de los códigos programados como los resultados de las imágenes de presión obtenidas. Por último, en el capítulo 6 se exponen las conclusiones finales del proyecto y se detallan posibles objetivos de mejora para trabajos futuros.

Capítulo 2

Diseño e implementación del sistema electrónico

2.1. Diseño del sistema electrónico

El diseño del sistema se basa en la excitación de filas y lectura de columnas. Para ello se ha diseñado un sistema de acondicionamiento de señales y conversión analógico/digital que establece contacto mecánico con los terminales de la lámina del sensor matricial piezorresistivo.

La Figura 2.1 muestra el sistema de detección y un esquema eléctrico simplificado de la electrónica de 8 bits que escanea los puntos de intersección de las filas y columnas del sensor, midiendo la resistencia en cada terminal piezorresistivo del mismo (representados por resistencias variables), cuyo valor decrece de manera inversamente proporcional a la presión ejercida [17, 13].

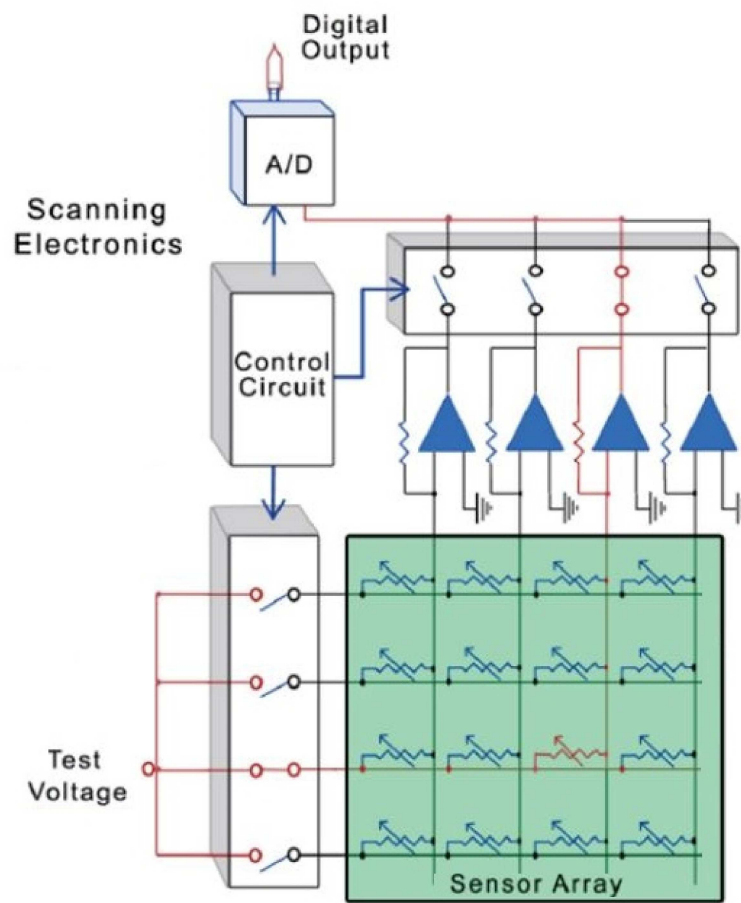


Figura 2.1: Esquema eléctrico simplificado de la electrónica de 8 bits

2.1.1. Subsistema de excitación de filas

En el esquema 2.2 se explica el funcionamiento que debe tener el subsistema de excitación de filas del sensor.

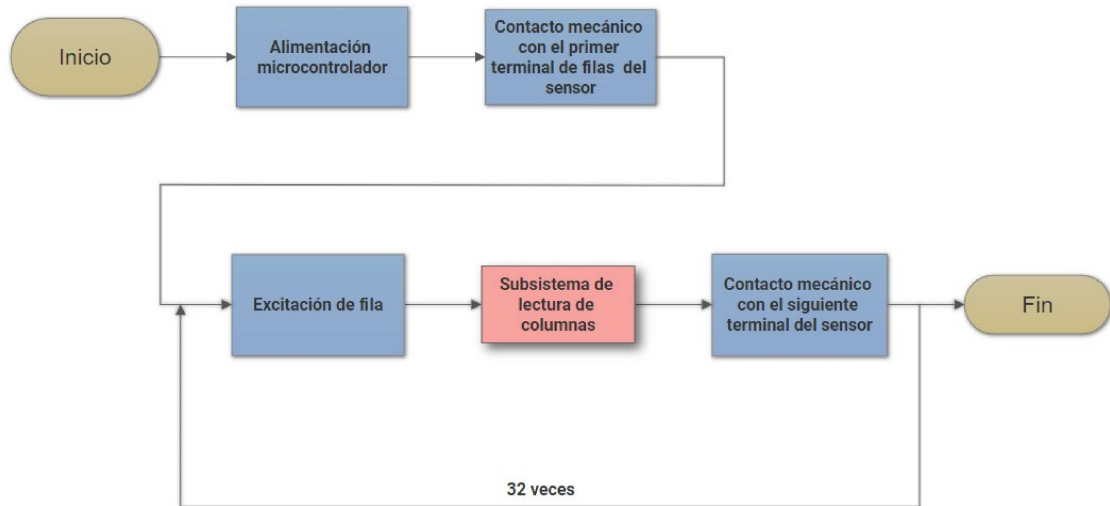


Figura 2.2: Esquema de funcionamiento del subsistema de excitación de filas

2.1.2. Subsistema de lectura de columnas

En el esquema 2.3 se explica el funcionamiento que debe tener el subsistema de lectura de columnas del sensor.

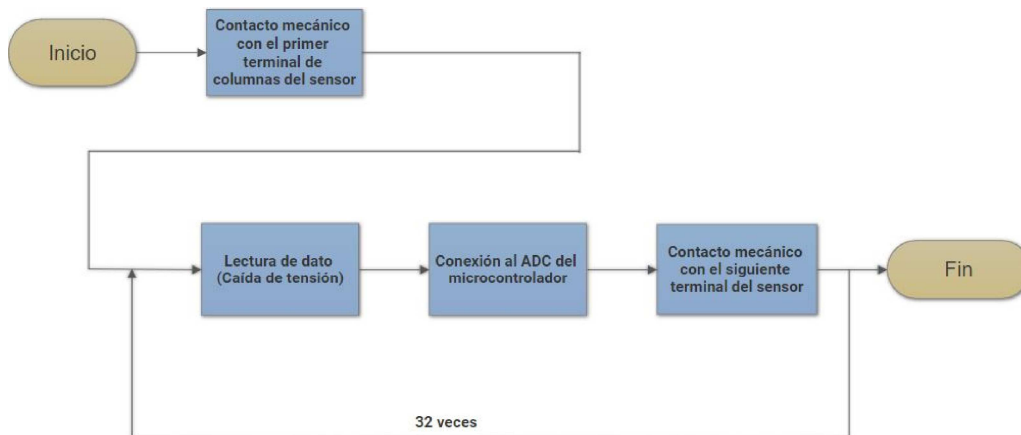


Figura 2.3: Esquema de funcionamiento del subsistema de lectura de columnas

2.2. Componentes utilizados

2.2.1. Arduino UNO

Arduino UNO [1] es una placa de microcontrolador basada en el ATmega328P, que puede verse en la Figura 2.4. Tiene 14 pines digitales de entrada/salida, 6 entradas analógicas, un cristal de cuarzo de 16 MHz, una conexión USB, un conector de alimentación, un encabezado ICSP y un botón de reset. Simplemente hay que conectarlo a un ordenador con un cable USB o con un adaptador de AC/CC o a una batería.

Se ha decidido utilizar este microcontrolador dado que cumple con los requisitos necesarios para el proyecto, además de por su facilidad de uso, la gran cantidad de contenidos e información de ayuda disponible en la red y su bajo coste.

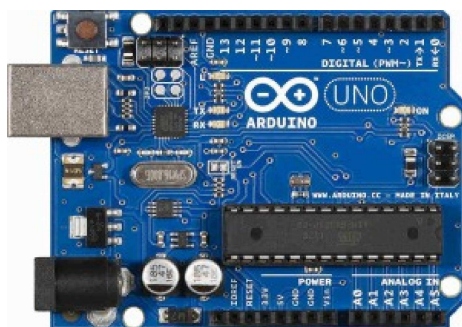


Figura 2.4: Arduino UNO

2.2.2. Shift Registers

Un Shift Register o Registro de Desplazamiento es un circuito digital secuencial (los valores de sus salidas dependen de sus entradas y de los valores anteriores) consistente en una serie de biestables conectados en cascada, que comparten la misma señal de reloj.

Para la **excitación de cada fila**, se han utilizado **cuatro Shift Registers modelo: 74HC595D** [22] con entrada serie y salida paralelo (SIPO).

Se han estudiado otras posibilidades en lugar de la utilización de Shift Registers.

- La utilización de **I/O Expanders**. Pero se ha llegado a la conclusión de que la lectura de cada punto de la matriz sería más lenta. Además, la programación de Shift Registers en serie es más fácil de implementar.
- La utilización de una placa Arduino con más salidas digitales, por ejemplo, la **Arduino MEGA** y de esta forma, no haber incorporado los Shift Registers.

No obstante, se han decidido utilizar debido a que, de esta manera, cambiando solamente un par de líneas de código, el proyecto puede ser ampliado para la adquisición de datos de matrices mayores, y ser fácilmente adaptado para su uso en otras placas de microcontrolador.

2.2.3. Multiplexores

Los Multiplexores (MUX) son circuitos combinatoriales que poseen 2^n canales de datos de entrada, un canal de salida y n líneas de control. Estas líneas de control son capaces de seleccionar una, y solo una, de las entradas de datos para permitir su transmisión desde la entrada seleccionada hacia dicha salida. De esta forma, conseguimos seleccionar solo una columna del sensor cada vez para poder leer los datos.

Para la **adquisición de datos de cada columna**, se han utilizado **dos multiplexores modelo: 74HC4067**, en su versión SMD (por compactación) [21], que disponen de 16 canales de entrada cada uno, consiguiendo así 32 canales en total, que cubren las necesidades del proyecto.

Al principio del proyecto, se utilizaron módulos de Multiplexores y se tomaron medidas con ellos. Más adelante, debido al tamaño de los módulos, se optó por utilizar los componentes SMD para realizar un diseño más compacto de las placas de circuito impreso.

2.2.4. Placas de circuito impreso

Una placa de circuito impreso (PCB: Printed Circuit Board) soporta mecánicamente y conecta eléctricamente componentes electrónicos o eléctricos usando pistas conductoras, Pads y otras características grabadas a partir de una o más capas de lámina de cobre laminado sobre o entre capas de lámina de un sustrato no conductor. Los componentes generalmente están soldados en la PCB tanto para conectarlos eléctricamente como para fijarlos mecánicamente.

Se han diseñado dos PCBs para compactar el diseño y cubrir las necesidades del proyecto. El diseño de las mismas se explica en el Capítulo 3, Sección 3.1.

2.2.5. Pogo pins

Un Pogo Pin es un dispositivo utilizado para conectar (temporalmente) dos placas de circuito impreso. Normalmente, tiene la forma de un cilindro delgado que contiene dos pasadores afilados, accionados por un muelle, como se puede ver en las Figuras 2.5 y 2.6. Presionados entre dos circuitos electrónicos, los extremos del Pogo Pin hacen contacto con los dos circuitos y los conectan entre sí.



Figura 2.5: Pogo Pins

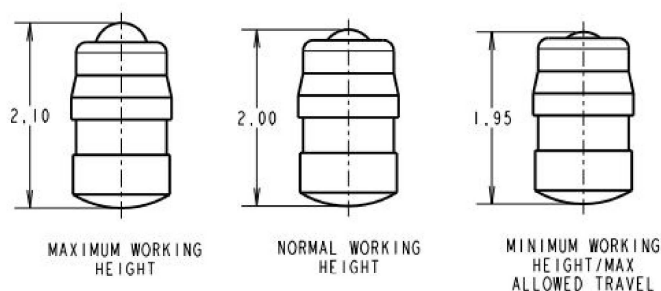


Figura 2.6: Funcionamiento Pogo Pin

Debido a que nuestro sensor está formado por una lámina flexible, se han decidido utilizar Pogo Pins ya que se adaptan mejor a la superficie que un conector Berg normal y garantizan un contacto óptimo.

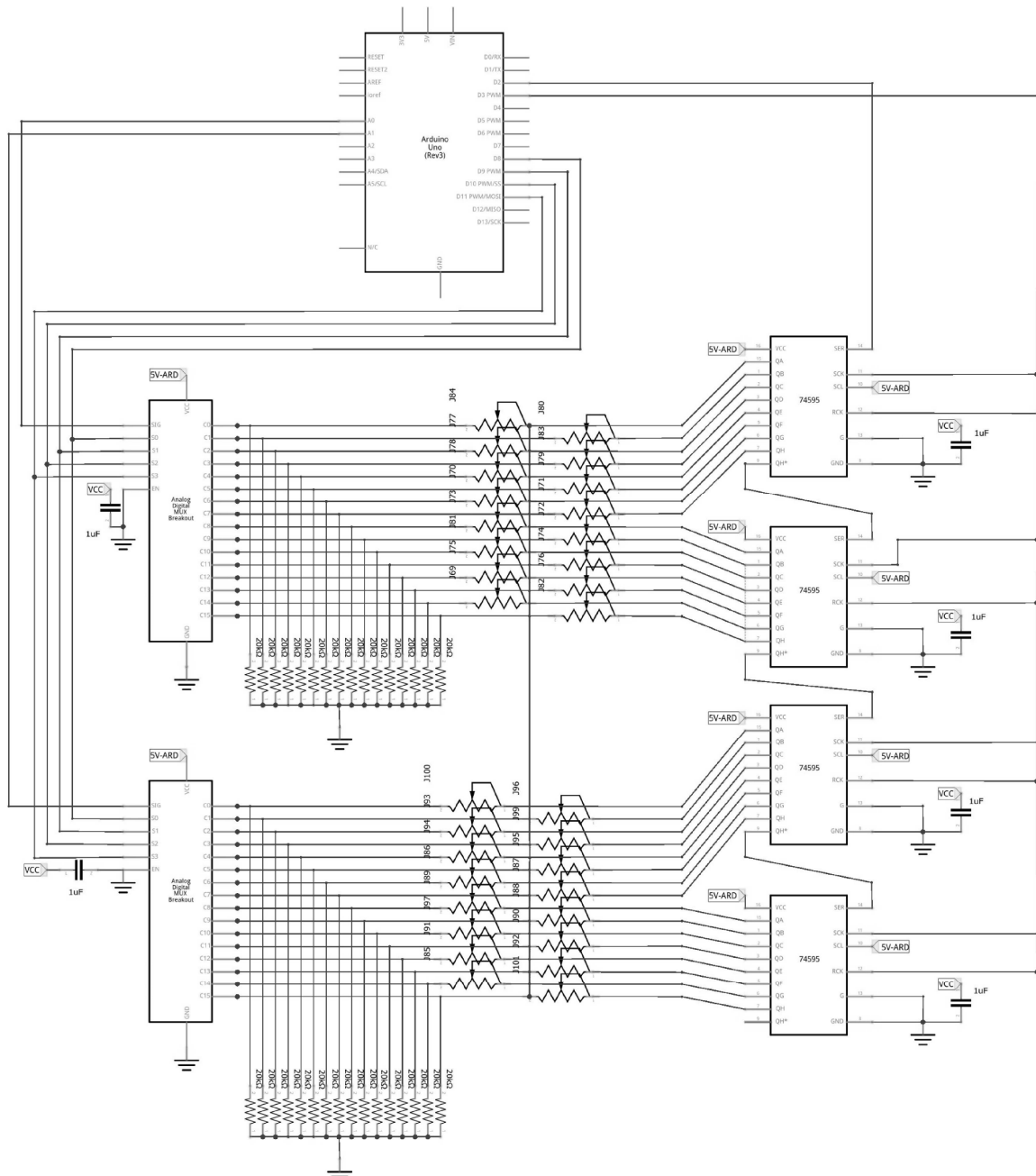
2.3. Diseño de los circuitos impresos

En la Figura 2.7 se muestra el esquema de conexión hardware completo.

- La placa Arduino se conecta al ordenador mediante un cable USB y alimenta a los componentes con una tensión de 5V.
- Cada multiplexor se conecta a una entrada analógica de la placa Arduino (aunque realmente este posee un único ADC). Además, como este componente posee 16 canales de salida se necesitan cuatro salidas digitales del microcontrolador, 4 bytes.
- Para poder medir la resistencia del sensor, en cada salida del multiplexor se ha conectado un divisor de tensión formado por una resistencia de 30 kOhmios y el Pogo Pin correspondiente a cada uno de los terminales de las columnas del sensor, que en el esquema de conexiones 2.7 se han representado como resistencias variables.
- Los Shift Registers se colocan en cascada, aprovechando el canal de Overflow del SR anterior. De este modo, mediante la señal de reloj, se va pasando de un canal a otro. Se necesitan dos salidas analógicas del microcontrolador para el flujo de datos y su señal de reloj.
- Las salidas de los Shift Registers se conectan directamente al Pogo Pin correspondiente a cada uno de los terminales de las filas del sensor.
- Se han conectado, además, condensadores de desacoplo entre la alimentación de cada circuito integrado y la puesta a tierra, con el fin de minimizar el ruido.

CAPÍTULO 2. DISEÑO E IMPLEMENTACIÓN DEL SISTEMA ELECTRÓNICO

Para más información, puede consultarse la asignación de pines en el Apéndice A y en el Capítulo 5 y las Hojas de Especificaciones de cada componente [21], [22].



Capítulo 3

Diseño del sistema físico

3.1. Diseño de PCBs

Se ha decidido diseñar un dispositivo, lo más compacto posible, que recoja los componentes electrónicos necesarios para el sistema de adquisición de imágenes. Para ello, se han diseñado dos placas de circuito impreso. Una que se encargaría de la lectura de las filas y la otra, de las columnas. Estas placas se sitúan una encima de otra y han de respetar los espacios en los que se introduciría el sensor Tekscan y sus respectivos conectores.

Para el diseño de las PCBs se ha utilizado el software “EAGLE” de Autodesk [2], con su licencia para estudiantes. EAGLE (siglas de Easily Applicable Graphical Layout Editor) es un programa de diseño de diagramas y PCBs con autoenrutador.

3.1.1. Contacto con los conectores del sensor

Para garantizar la lectura del sensor, se han utilizado Pogo Pins, como se explica en el Capítulo 2. Para su colocación en la PCB se han utilizado Pads Through-Hole de diámetro de drill = 8mm, ya que el diámetro de los Pogo Pins es de 7,5 mm. Los Pads de una misma fila y columna están separados una distancia de 0,2 inches. Y la separación entre filas y entre columnas es de 0,1 inches. Se ha estudiado qué conector corresponde a cada canal de los Shift Registers y a cada canal de los Multiplexores para poder adquirir los datos correctamente. La correspondencia puede verse en las Figuras 3.1 y 3.2. Ambas placas deben encajar a la perfección para garantizar el contacto de los pogo pins con el sensor y la lectura correcta de filas y columnas.

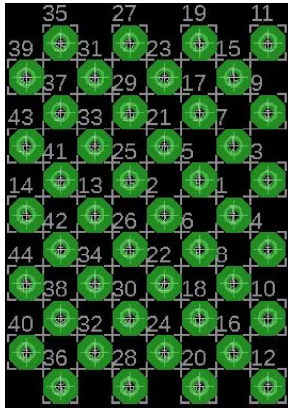


Figura 3.1: Correspondencia de Pads con canales Shift Registers

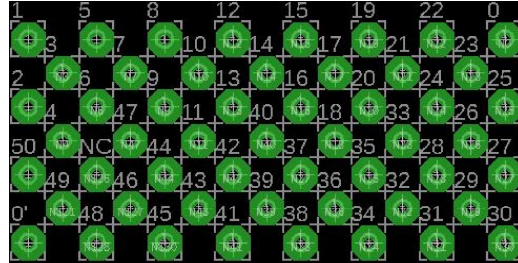


Figura 3.2: Correspondencia de Pads con canales Multiplexores

3.1.2. Colocación de los componentes electrónicos

Tanto los Multiplexores como los Shift Registers y las resistencias se han colocado intentando simplificar al máximo las conexiones entre Pads y Pogo Pins, tratando de realizar un diseño limpio y óptimo.

Para la colocación de los condensadores de desacoplo se ha tenido en cuenta que estos deben estar situados lo más cerca posible de la alimentación del circuito integrado.

3.1.3. Ruteo

Para una primera aproximación de las trazas se ha utilizado la función de autoruteo tanto por la cara superior como por la inferior, con la incorporación de vías para poder pasar una traza de una capa a otra. Este ruteo no ha resultado ser el óptimo. Por tanto, se ha procedido a separar las vías que se encontraban demasiado próximas a los pads de los componentes electrónicos que posteriormente van a ser soldados; sobre todo, aquellas muy próximas a Pads SMD, debido a la complejidad de su soldadura.

3.1.4. Taladros

Se han realizado cuatro taladros de 4mm de diámetro para poder colocar correctamente ambas placas y que la lectura de datos del sensor sea correcta.

3.1.6. Montaje y soldadura

Una vez se tienen las placas de circuito impreso, se procede al soldado de los componentes electrónicos. Para ello, se ha utilizado un soldador de estaño, y debido al reducido tamaño de los componentes, ha sido necesario soldar con microscopio.

Hay que diferenciar los dos tipos de montaje de componentes electrónicos en placas de circuito impreso para poder soldar cada componente de forma adecuada.

- **Through-Hole:** es un tipo de montaje en el que el componente atraviesa la placa de una cara a otra. Utiliza los agujeros que se practican en las placas para crear puentes eléctricos entre una de las caras de la placa de montaje a la otra. Este montaje suele ser bastante delicado, ya que si se calienta demasiado la placa, se puede dañar el contacto entre las pistas de una de las caras del circuito y la otra, resultando inoperante y dejando inútil la placa.

En nuestro caso, los únicos componentes que se han soldado utilizando este método han sido los Pogo Pins y conectores Berg.

- **SMD:** es el tipo de montaje más utilizado en la actualidad. Se basa en el montaje de los componentes sobre la superficie del circuito impreso. De este modo, los componentes SMD que suelen ser más pequeños que sus análogos Through-Hole no la atraviesan la placa. Las conexiones se realizan mediante contactos planos, o terminaciones metálicas en los bordes del componente.

Así se han soldado las resistencias, condensadores de desacoplo, multiplexores y shift registers.

En las Figuras 3.7, 3.8, 3.9 y 3.10 pueden verse las placas de circuito impreso con los componentes ya soldados.

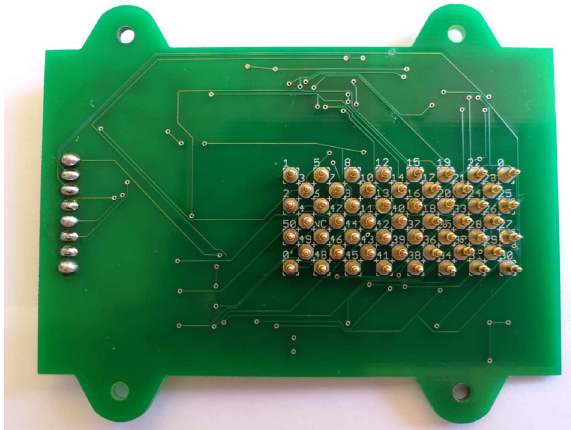


Figura 3.7: PCB columnas Cara Superior

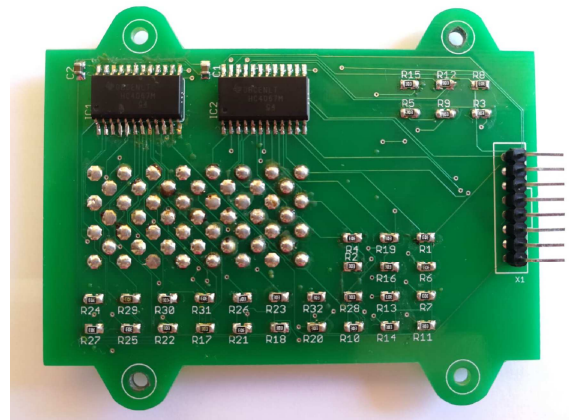


Figura 3.8: PCB columnas Cara Inferior

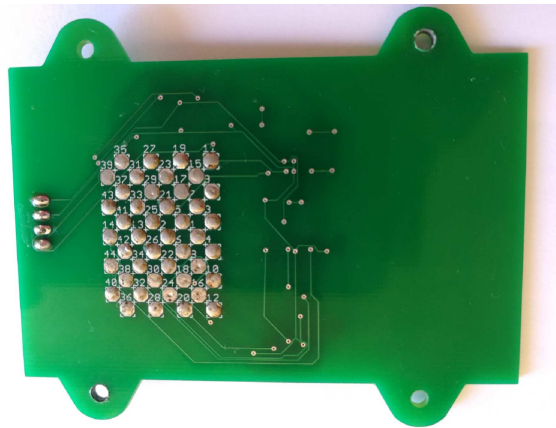


Figura 3.9: PCB filas Cara Superior

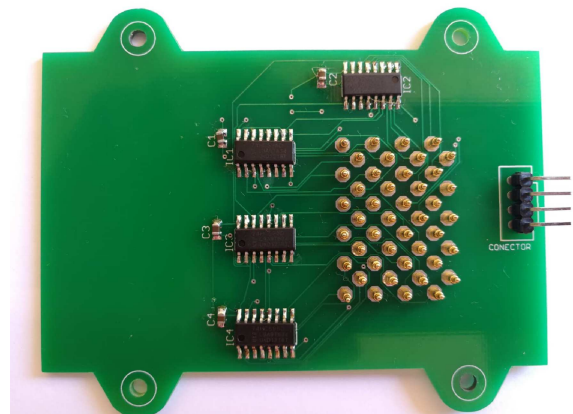


Figura 3.10: PCB filas Cara Inferior

3.2. Estructura de conexión

Las placas de circuito impreso deben encajar entre sí para garantizar el contacto de los conectores con el sensor. Sin embargo, si se introdujera el sensor directamente entre ambas placas, debido a los componentes soldados, a los circuitos impresos y a que el sensor es flexible, no sería posible asegurar este contacto.

Para solucionar este problema se ha diseñado una estructura de conexión que reviste algunos de los componentes, protegiéndolos y aislandolos para evitar contactos indeseados con los terminales del sensor y que sirve de tope para los Pogo Pins.

La estructura de conexión ha resultado complicada de diseñar debido a las dimensiones tan pequeñas de los componentes; a la colocación de los mismos y a la necesidad de que el sensor encaje en la posición correcta.

3.2.1. Diseño

El diseño de la estructura de conexión se ha realizado con el software SketchUp [16], que es un programa de diseño gráfico y modelado en tres dimensiones (3D) basado en caras. Es muy utilizado en diseño industrial, diseño escénico, videojuegos o películas, planificación urbana, arquitectura e ingeniería civil.

La estructura está compuesta por dos piezas, una para cada placa de circuito impreso y para su diseño, se han tenido muy en cuenta:

- El tamaño y la disposición de los terminales piezoeléctricos del sensor, para asegurar el contacto correcto de los mismos con los Pogo Pins.
- La longitud máxima y mínima de los Pogo Pins, para que la anchura de la estructura no sobrepase la longitud de los mismos.
- Las dimensiones y colocación de todos los componentes, para que el sensor encaje de forma correcta.
- La colocación de los taladros de unión, para garantizar fijación.
- La forma de las placas de circuito impreso, para hacer lo más compacto posible el dispositivo.

El diseño final de la estructura de conexión puede verse en la Figura 3.11 y en el Plano 01.

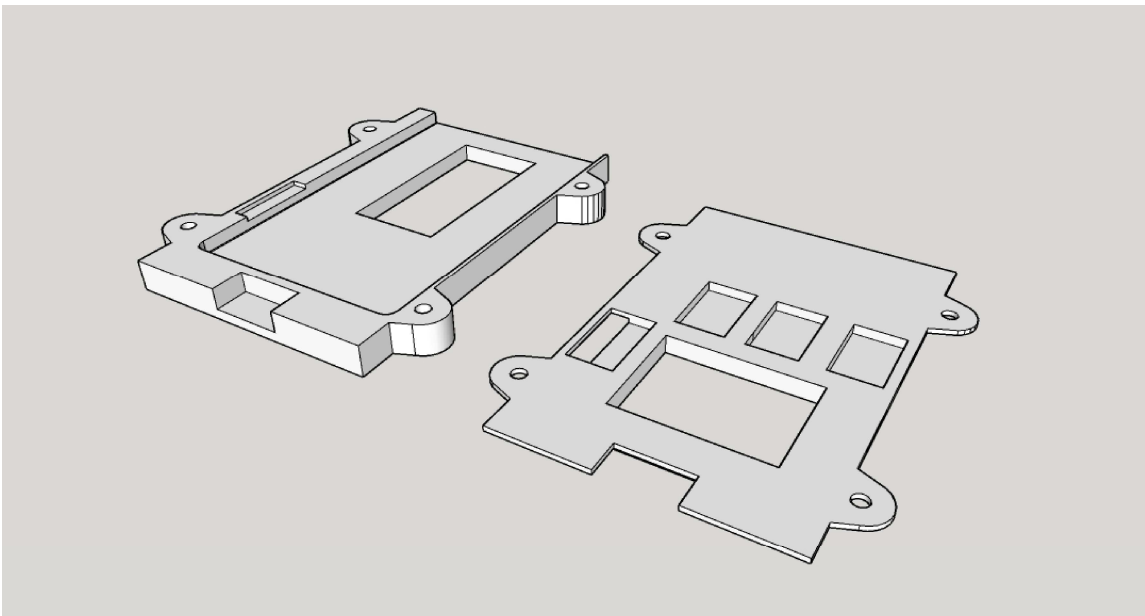


Figura 3.11: Diseño 3D realizado mediante el software SketchUp

3.2.2. Fabricación

Las dos piezas de la estructura se han fabricado mediante impresión 3D, debido a la rapidez, bajo coste, y facilidad de uso en comparación con otras tecnologías de fabricación.

En las Figuras 3.12, 3.13, 3.14 y 3.15 pueden verse ambas piezas para sendas PCBs.

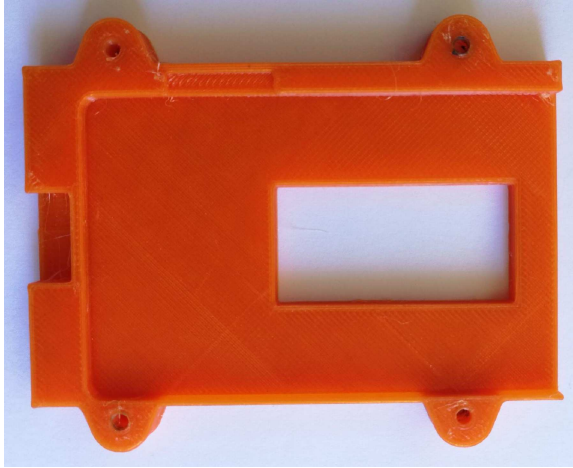


Figura 3.12: Pieza para PCB columnas. Cara Superior

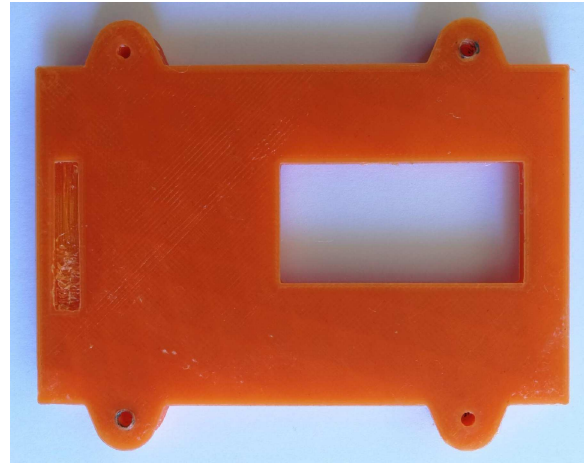


Figura 3.13: Pieza para PCB columnas. Cara Inferior

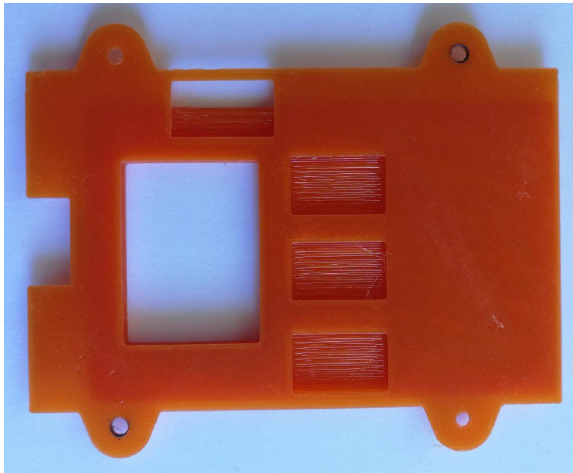


Figura 3.14: Pieza para PCB filas. Cara Superior

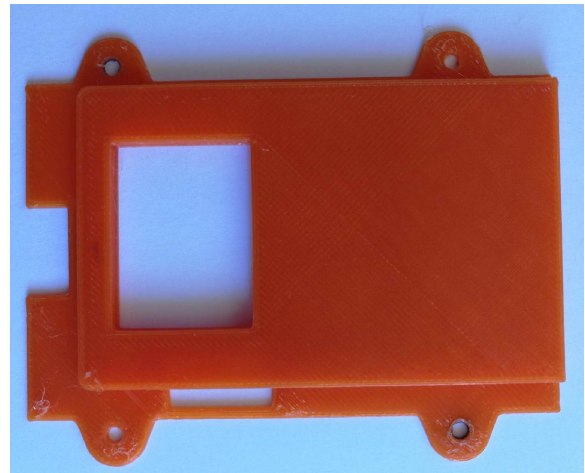


Figura 3.15: Pieza para PCB filas. Cara Inferior

3.3. Dispositivo completo

Una vez se tienen las placas de circuito impreso y las piezas que forman la estructura de conexión, se procede al montaje de las mismas. En las Figuras 3.16, 3.17 y 3.18 se observa el dispositivo final.

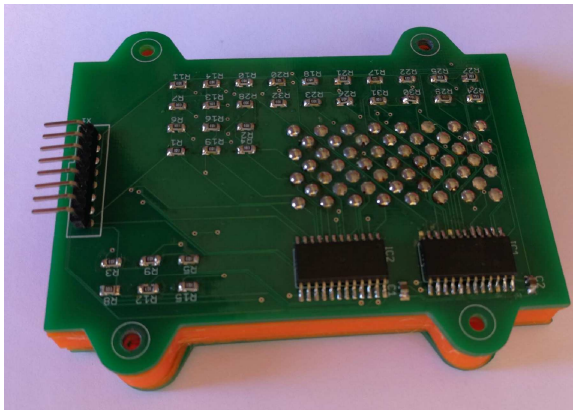


Figura 3.16: Montaje del dispositivo visto desde arriba

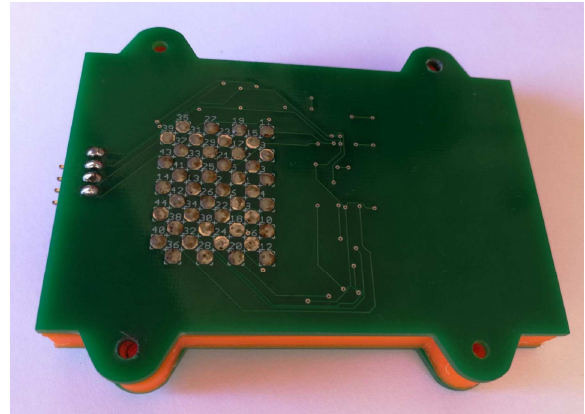


Figura 3.17: Montaje del dispositivo visto desde abajo

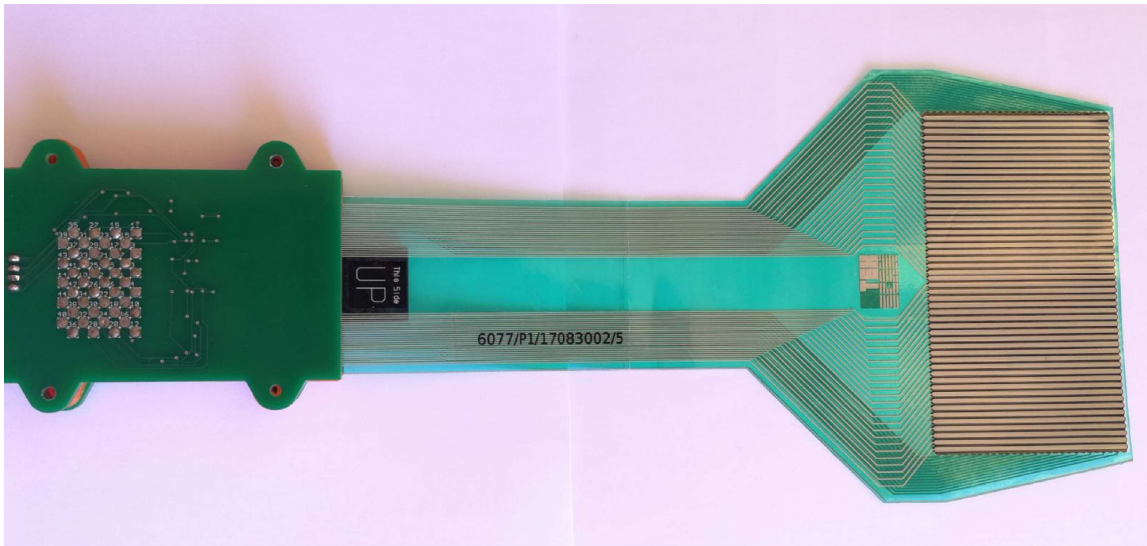


Figura 3.18: Montaje del dispositivo con el sensor acoplado

3.4. Conclusiones

Se ha desarrollado un sistema capaz de excitar las filas del sensor y leer los datos de cada columna a bajo coste; sobre todo en relación al precio del sistema comercial del que se disponía.

El diseño de las placas de circuito impreso ha resultado complicado debido a la cantidad de elementos de pequeñas dimensiones que hay colocados tan cerca unos de otros, lo que dificulta el ruteo de los circuitos impresos y la soldadura de los componentes; no obstante, el diseño final y el tamaño del dispositivo se considera adecuado; y la soldadura, a pesar de ser mejorable, resulta correcta teniendo en cuenta los medios con los que se contaba para ella.

El diseño de la estructura de conexión es correcto y garantiza el contacto del sensor con los Pogo Pins, aunque podría ser mejorado para hacer más fácil la colocación y extracción del mismo.

Capítulo 4

Firmware

4.1. Software de adquisición de datos

El Software de adquisición de datos tiene como objetivo garantizar la lectura correcta de la presión ejercida por un objeto cualquiera en cada punto del sensor y relacionando esos datos, ser capaz de transformarlos en una imagen de colores, mediante la cual, sea fácil interpretar la forma del objeto. Para ello, es necesario excitar ordenadamente cada fila del sensor y leer el dato correspondiente a cada columna. Seguidamente, hay que enviar los datos al ordenador para poder interpretarlos y transformarlos en la imagen de presión.

El Firmware completo se ha desarrollado utilizando Arduino y Matlab. En el esquema de la figura 4.1 se explican los pasos a seguir para la adquisición de las imágenes de presión; y, en los próximos apartados, se explican los códigos desarrollados en Arduino y Matlab, así como el protocolo de comunicaciones entre ambos.

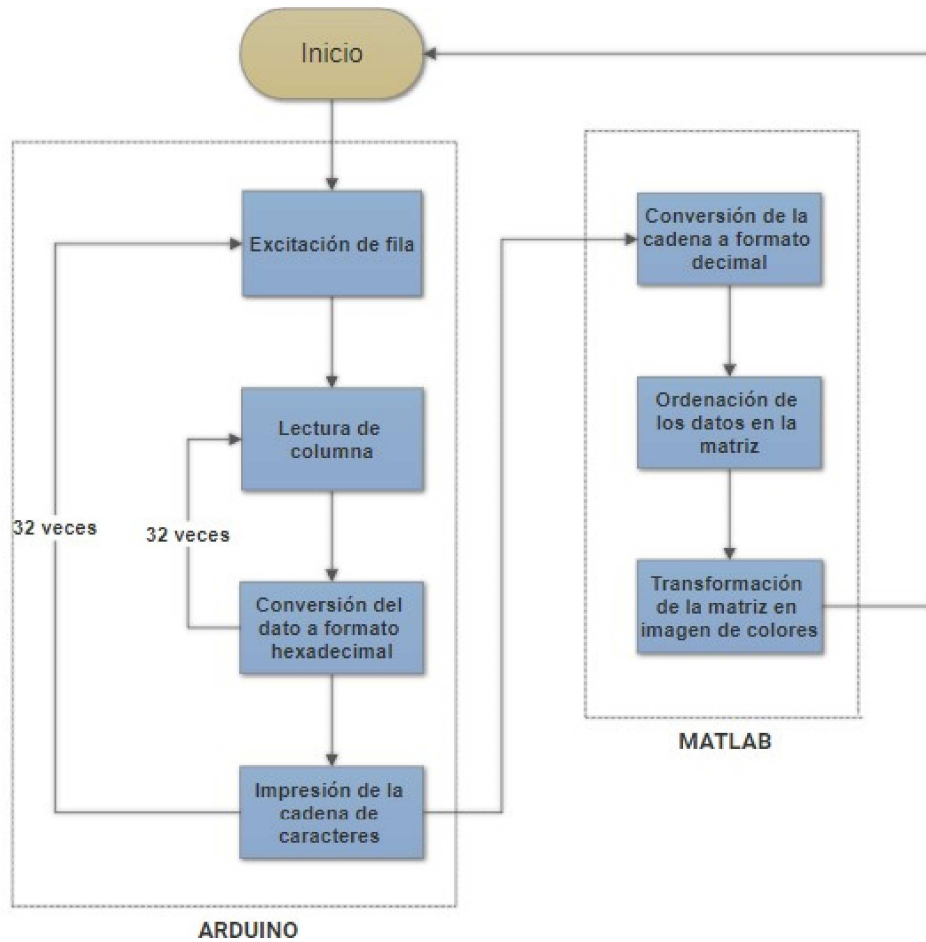


Figura 4.1: Esquema de actuación del Firmware

4.2. Protocolo de comunicaciones

Debido a que es necesario comunicar Arduino y Matlab para la adquisición de las imágenes de presión, es necesario definir un protocolo de comunicaciones para la transmisión de datos entre ellos:

- **Forma:** los datos se reciben a través del puerto serie. Consultar Apartado 4.4.1.
- **Formato:** los datos se envían en formato hexadecimal para poder interpretarlos de forma más fácil. Consultar Apartado 4.3.3.
- **Velocidad de transmisión:** debe ser igual tanto en Arduino como en Matlab, para que la excitación de filas y columnas y el envío de datos por el puerto serie pueda realizarse de manera correcta.

4.3. Código Arduino

Arduino dispone de un entorno de desarrollo constituido por un editor de texto donde se programa el código, un área de mensajes y una consola (Serial Monitor). Permite, además, la conexión con el hardware de Arduino, previa definición del mismo (Arduino UNO, MEGA, NANO, etc.) para poder cargar los programas en el microcontrolador y recibir los datos.

A continuación, se explican los códigos, o partes de código, desarrollados y utilizados en este proyecto.

Para la consulta del código Arduino completo, consultar Apéndice A.

4.3.1. Código de prueba

El siguiente código permite obtener el voltaje, la intensidad y la resistencia variable del sensor al ejercer fuerza sobre un punto del mismo.

```
int sensorPin = A0;
int R1 = 4700;
void setup() {
  Serial.begin(9600);
}
void loop() {
  int sensorVal = analogRead(sensorPin);
  Serial.print("Valor del sensor: ");
  Serial.print(sensorVal);

  //Lectura del ADC a voltaje
  float VR = (sensorVal / 1024.0) * 5.0;
  float Vsensor = 5 - VR;
  Serial.print(", Vsensor: ");
  Serial.print(Vsensor);

  float Rvar = (R1 * Vsensor) / VR;
  Serial.print(", Resistencia: ");
  Serial.print(Rvar);

  float intensidad = (Vsensor / Rvar) * 1000000;
  Serial.print(", Intensidad : ");
  Serial.print(intensidad);
  Serial.println(" uA");
  delay(1250);
}
```

4.3.2. Bucle principal

Se recorren las filas y columnas del sensor activando para ello cada canal de multiplexor y Shift Register que corresponda.

Para el manejo de los Shift Registers (excitación de filas) se utiliza la librería “ShiftRegister74HC595.h” que simplifica el código significativamente.

Para los multiplexores (columnas) se han realizado dos funciones para su activación y lectura de cada dato.

4.3.3. Formato Hexadecimal

Debido a que los datos tomados por el sensor varían de 0 a 1024, es decir, de cero a cuatro dígitos, esta variación de la longitud de la cadena puede hacer tediosa la interpretación de la matriz de datos en Matlab. Por tanto, para que los datos sean de fácil interpretación, se han decidido transmitir en formato hexadecimal con precisión de dos dígitos. Para convertir los datos de formato decimal a hexadecimal se han estudiado dos posibilidades:

- El uso de una función que reciba el dato en decimal y la precisión deseada y devuelva el número en hexadecimal.

```
void printHex(int num, int precision) {
    char tmp[16];          // bits
    char format[128];     // Los primeros 128 puntos de código son los
                          // mismos que los definidos por los caracteres ASCII; Del 128
                          // al 255 son caracteres especiales que no necesitamos
    sprintf(format, "%%.%dX", precision); // Retorna el número de
    // caracteres escritos al array, sin contar el carácter nulo al
    // final, con la precisión que especifiquemos (en nuestro caso
    // será de 2)
    sprintf(tmp, format, num);
    Serial.print(tmp);
}
```

- La conversión directa de cada dato a dos dígitos hexadecimales utilizando, para ello, el código ASCII, teniendo en cuenta que el rango de caracteres válidos es: [48,57]U[97,102]. Números y letras minúsculas de la 'a' a la 'f'.

```
[...]
cl = 48 + (dato & 0x0F); // El carácter ASCII 48: '0'
ch = 48 + ((dato >> 4) & 0x0F);
cadena[i * 2] = ch;
cadena[i * 2 + 1] = cl;
// Si el carácter es mayor al 57, tendríamos caracteres
// especiales,
```

```
// sumamos 18 para llegar a la 'a' -> 97.
if (ch > 57) cadena[i * 2] += 39;
if (cl > 57) cadena[i * 2 + 1] += 39;
[...]
```

La llamada a una función cada vez que se recibe un dato no garantiza la velocidad que consideramos necesaria para la transmisión de los datos, por tanto, se ha optado por la segunda opción, que además puede ser modificada más fácilmente para elevar aún más esa velocidad de transmisión.

4.4. Código Matlab

Para la consulta del código Matlab completo, consultar Apéndice B.

4.4.1. Elección y apertura del puerto serie

Para hacer más eficiente y rápida la adquisición de imágenes, los datos se reciben con Matlab a través del puerto serie. Para ello, es necesario asegurarse de que se está utilizando el puerto correcto antes de ejecutar el código. El puerto al que se conecta, en este caso, 'COM4', depende de cada ordenador y puede variar. Lo óptimo es borrar cualquier objeto que esté conectado al puerto requerido antes de asociarlo. Hay que definir la velocidad de transmisión de los datos, que debe coincidir con la de la configuración de Arduino, y abrir el puerto serie desde Matlab.

```
% Borrar previos
delete(instrfind({'Port'},{'COM4'}));
% Puerto serie. Comprobar el actual COM
s = serial('COM4');
% Velocidad de transmisión. Debe coincidir con la de la configuración
  n de Arduino
set(s, 'BaudRate', 19200);
get(s);
% Abrir puerto serie
fopen(s);
```

4.4.2. Ordenación de la matriz de datos

Los canales de los multiplexores y shift registers han sido conectados a los pogo pins desordenadamente para facilitar el ruteo de las placas de circuito impreso. Por tanto, no han sido conectados consecutivamente a las filas y columnas del sensor. Por consiguiente, la matriz resultante de la toma de datos no está ordenada y es necesario indexarla antes de su transformación en mapa de colores.

Debido a la limitación de memoria que posee Arduino, no es posible guardar los datos en una matriz de 32x32 para su ordenación correcta antes de imprimirla a través del puerto serie. Por tanto, primero, se han transmitido los datos tomados del sensor y, posteriormente, se han reordenado en Matlab; lo que, además, reduce significativamente el número de líneas de código necesarias. En el Cuadro 4.1 se muestra la equivalencia.

Cuadro 4.1: Equivalencia entre filas y columnas del sensor y canales de medida

| FILAS | | | COLUMNAS | |
|--------|----------|----------------------|-------------------|-------------------|
| Sensor | Pogo Pin | Canal Shift Register | Sensor y Pogo Pin | Canal Multiplexor |
| 1 | 8 | 25 | 1 | 16 |
| 2 | 7 | 1 | 2 | 15 |
| 3 | 10 | 32 | 3 | 14 |
| 4 | 9 | 2 | 4 | 13 |
| 5 | 12 | 31 | 5 | 12 |
| 6 | 11 | 5 | 6 | 11 |
| 7 | 13 | 16 | 7 | 10 |
| 8 | 14 | 17 | 8 | 9 |
| 9 | 16 | 30 | 9 | 8 |
| 10 | 15 | 4 | 10 | 7 |
| 11 | 18 | 27 | 11 | 6 |
| 12 | 17 | 3 | 12 | 5 |
| 13 | 20 | 29 | 13 | 4 |
| 14 | 19 | 6 | 14 | 3 |
| 15 | 22 | 26 | 15 | 2 |
| 16 | 21 | 7 | 16 | 1 |
| 17 | 24 | 28 | 17 | 32 |
| 18 | 23 | 8 | 18 | 31 |
| 19 | 26 | 18 | 19 | 30 |
| 20 | 25 | 11 | 20 | 29 |
| 21 | 28 | 23 | 21 | 28 |
| 22 | 27 | 9 | 22 | 27 |
| 23 | 30 | 24 | 23 | 26 |
| 24 | 29 | 10 | 24 | 25 |
| 25 | 32 | 21 | 25 | 24 |
| 26 | 31 | 12 | 26 | 23 |
| 27 | 34 | 19 | 27 | 22 |
| 28 | 33 | 13 | 28 | 21 |
| 29 | 36 | 22 | 29 | 20 |
| 30 | 35 | 14 | 30 | 19 |
| 31 | 38 | 20 | 31 | 18 |
| 32 | 37 | 15 | 32 | 17 |

4.4.3. Lectura de datos

Se reciben las cadenas de caracteres equivalentes a cada fila mediante la función: “fget1(puertoserie)” ; se recorre esa fila y cada dato se va guardando en una matriz de 32x32 hasta rellenarla completamente.

```
for i=1:32 %% Supongo que hay 32 lineas
    linea=fget1(s);
    disp(linea);
    for j=1:32 %% y que cada linea contiene 32 bytes en Hexadecimal
        A_recibida(i,j)=(linea((j-1)*2+1)-48)*16+linea((j-1)*2+2)
            -48;
    end
end
```

Los datos se reciben a través del puerto serie en formato hexadecimal, por lo que es necesario transformarlos a formato decimal, utilizando el código ASCII para deshacer la transformación realizada en el código Arduino.

4.4.4. Imágenes de presión

Para la adquisición de imágenes de presión se ha utilizado un código que convierte cada punto del sensor en un pixel y cada valor numérico en un color, de azul oscuro a rojo (de menor a mayor presión).

```
[nf,nc] = size(A); % siendo A la matriz de datos reordenada
figure
hold on
for(i=1:1:nc)
    for(j=1:1:nf)
        C = [0 1; 0 1];
        F = [nf-1 nf-1; nf nf];
        C = C + (i-1);
        F = F - (j-1);
        B = ones(2,2)*A(j,i);
        surface(C,F,B);
    end
end
axis([0 nc 0 nf])
colormap(jet) % Crea la imagen de colores
set(gca,'XTick',[0:1:nc])
set(gca,'YTick',[0:1:nf])
set(gca,'FontSize',6)
```

Para ello, se utiliza la función de Matlab: “colormap(jet)” [10]. A cada valor numérico no le corresponde un color concreto, sino que dependiendo del rango de valores que se

obtienen, el valor más bajo se corresponde con el primer color del array de colores y el valor más alto con el último color del array, que puede verse en la Figura 4.2



Figura 4.2: Array de colores de tipo JET

4.5. Conclusiones

Se ha desarrollado un Firmware sencillo y que utiliza muy poca memoria del micro-controlador.

La comunicación de Arduino a Matlab se realiza de forma simple y eficaz y, con el código programado, se consigue una velocidad de transmisión de 19200 baudios, que debería aumentarse, mejorando el código de conversión de datos de formato decimal a hexadecimal en Arduino.

Resulta complicada la ordenación de los datos de la matriz de resultados. Hubiera sido más fácil diseñar las placas de circuito impreso de manera que el primer canal de la cascada de Shift Registers excitara la primera fila y el primer canal del primer multiplexor leyera el dato de la primera columna; y continuara sucesivamente. De este modo, aunque el ruteo de las placas de circuito impreso hubiera resultado ser más complicado, se hubiera simplificado el código, además de resultar más intuitivo y poder localizar los errores con mayor facilidad.

Para demostrar el funcionamiento del sistema se han desarrollado diferentes pruebas que se muestran en el Capítulo 5.

Capítulo 5

Pruebas y Resultados

5.1. Puerto serie de Arduino

5.1.1. Voltajes e Intensidades para un punto del sensor

Los primeros resultados se obtienen al conectar a la placa Arduino un divisor de tensión formado por un punto del sensor y una resistencia de 4.7kOhmios y programando el código de prueba explicado en el apartado 4.3.1. Los resultados se muestran en la Figura 5.1 y se observa que hay que ejercer excesiva fuerza para que el sensor sea capaz de medir valores. Debido a las futuras aplicaciones de este proyecto, no es conveniente que sea necesario aplicar mucha fuerza; por ello, se decide aumentar la resistencia del divisor de tensión a un valor entre 20k y 30kOhmios, aunque eso va a aumentar el ruido de la señal al disminuir considerablemente la intensidad del circuito. Los nuevos resultados obtenidos al cambiar la resistencia se muestran en la Figura 5.2.

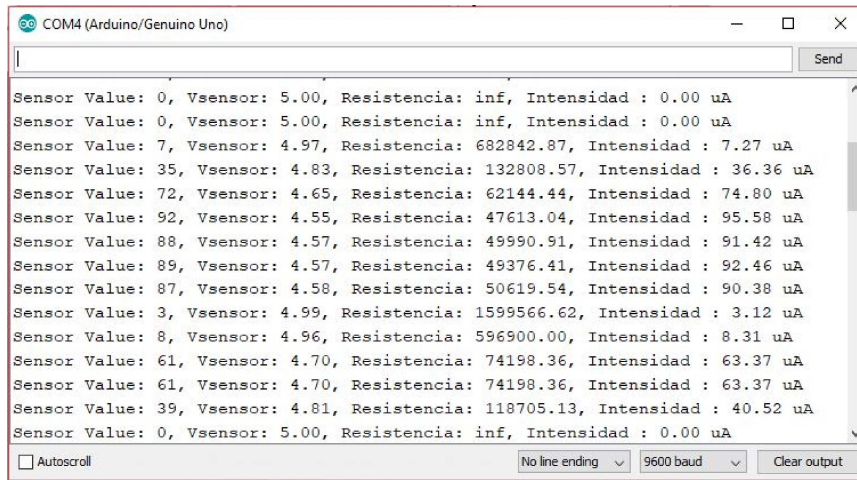


Figura 5.1: Resultados obtenidos al ejercer fuerza en un punto del sensor, conectado a una resistencia de 5kOhmios.

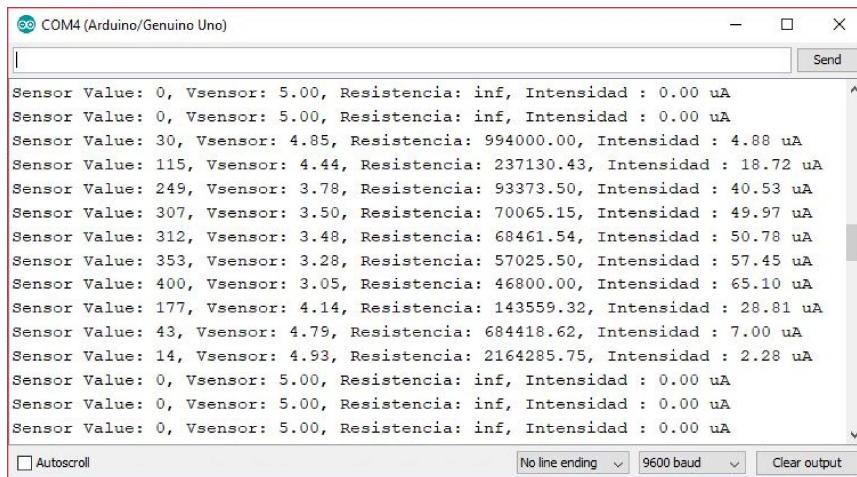


Figura 5.2: Resultados obtenidos al ejercer fuerza en un punto del sensor, conectado a una resistencia de 30kOhmios.

5.1.2. Multiplexores

Se conecta ahora un multiplexor a la placa Arduino y a los puntos correspondientes del sensor de presión. Para mitigar el ruido se sueldan los conectores al módulo del multiplexor y se conectan los condensadores de desacoplo. Se ejerce fuerza solo en el punto correspondiente al Canal 12. Los resultados obtenidos se observan en la Figura 5.3, de donde se concluye que la programación de los mismos es correcta.

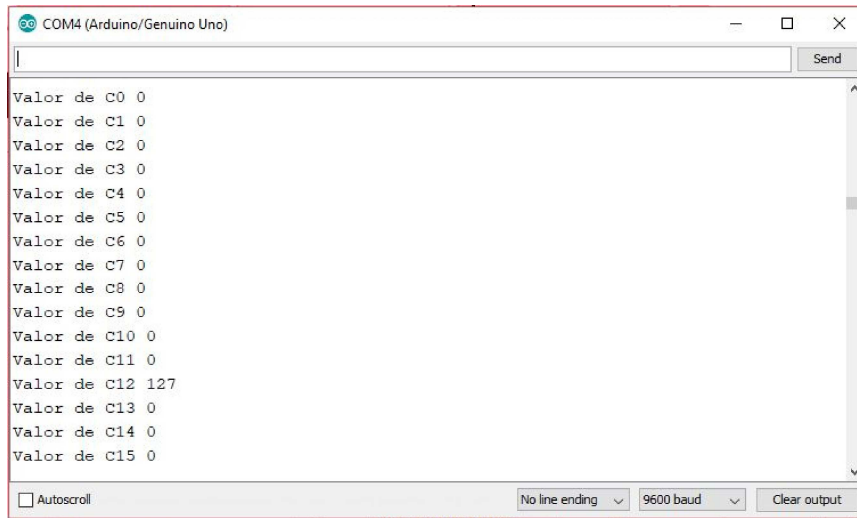


Figura 5.3: Resultados obtenidos al ejercer fuerza solo en el punto conectado al Canal 12 de un multiplexor.

5.1.3. Montaje completo

Una vez se han desarrollado las pruebas básicas, se diseña y monta el circuito completo.

En la Figura 5.4 se comprueba que los datos son recibidos en formato hexadecimal a través del puerto serie, con una longitud de la cadena de 64 caracteres.

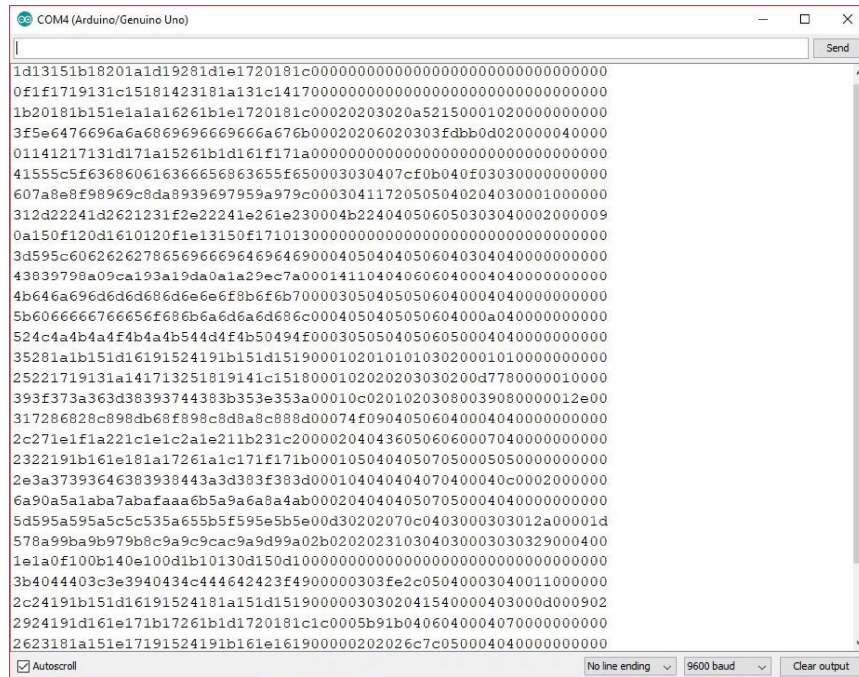


Figura 5.4: Resultados en formato hexadecimal obtenidos al ejercer fuerza sobre el sensor.

Debido a la difícil interpretación de estos datos, y la imposibilidad de comprobar si son correctos o no, se procede a comprobar la exactitud de los mismos de forma más visual, mediante las imágenes de presión en Matlab.

5.2. Imágenes de presión

A continuación se muestran los resultados obtenidos con Matlab al ejercer presión sobre un sensor de 28 filas x 32 columnas con diferentes objetos. Por tanto, en las cuatro primeras filas de la imagen de colores no debería aparecer ninguna medida debido a que el sensor utilizado en la prueba tiene 28 filas en lugar de 32, como estaba previsto.

5.2.1. Sin ejercer presión

Primero, se obtienen los datos que transmite el sensor sin ejercer fuerza, para tener una primera idea de los posibles errores. Se realiza la prueba en dos ocasiones. El resultado puede observarse en la Figura 5.5.

- En ambos casos, los resultados obtenidos, a pesar de no ser exactos, son precisos.
- Existe ruido en la señal, de la columna 1 a la 16 y en la 26. El primer intervalo de columnas se corresponde con el primer multiplexor.
- Se observan valores especialmente altos en las filas 22, 23 y 31, que se corresponden con los canales 9, 24 y 20 del Shift Register o los Pogo Pins 27, 30 y 38, respectivamente. Estos valores elevados también son visibles en la columna 26 (canal 9 del segundo multiplexor).
- Se observan, también, datos erróneos en otros puntos de la imagen. Estos datos puntuales podrían corregirse incorporando un programa inicial para la calibración del sensor antes de la toma de datos.

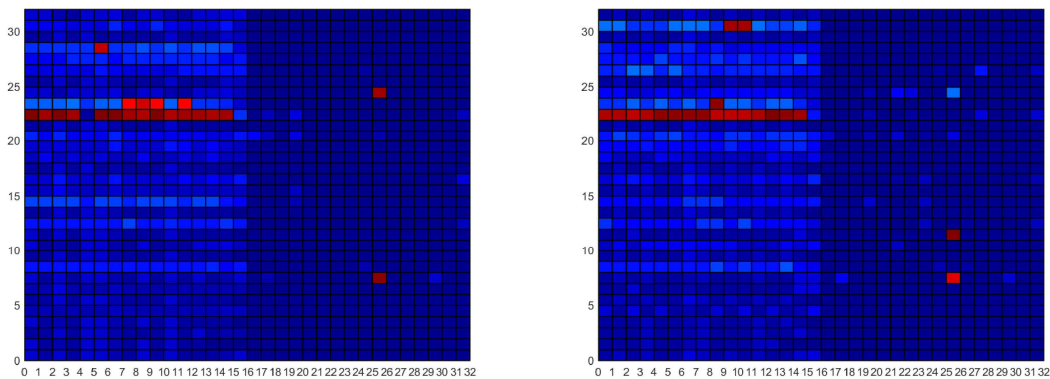


Figura 5.5: Imágenes de presión del sensor sin ejercer presión

5.2.2. Lápiz

A continuación, se coloca un lápiz sobre el sensor en posición diagonal. El resultado se observa en la Figura 5.6.

- Se aprecia la forma del lápiz, en tonos del mismo color, lo que es correcto, dado que se ha ejercido la misma presión en todos los puntos del objeto.
- Existe ruido en la señal en las columnas correspondientes al primer multiplexor, como en la prueba anterior.
- Se observa ausencia de medidas en las filas 6,14,22 y 29.

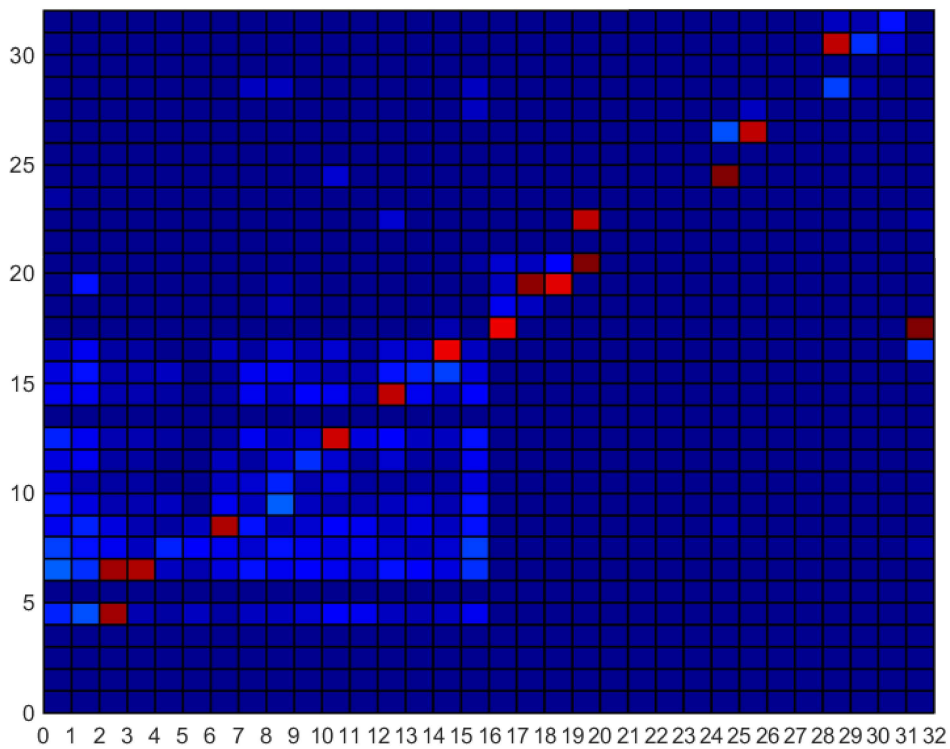


Figura 5.6: Imagen de presión de un lápiz

5.2.3. Pelota de gomaespuma

A continuación, se coloca una pelota de gomaespuma de 6cm de diámetro en el lado izquierdo del sensor. El resultado se observa en la Figura 5.7.

- Se aprecia con facilidad la forma de la pelota.
- Existe ruido en la señal, especialmente en la última columna.
- Se aprecian dos puntos en los que la presión es excesivamente elevada, posiblemente, debido a algún fallo en la medida. Estos valores desproporcionados con respecto a los de su alrededor, hacen que el resto de la imagen se vea en colores bastante más claros.
- Se observa ausencia de medidas en las filas 10 y 22.

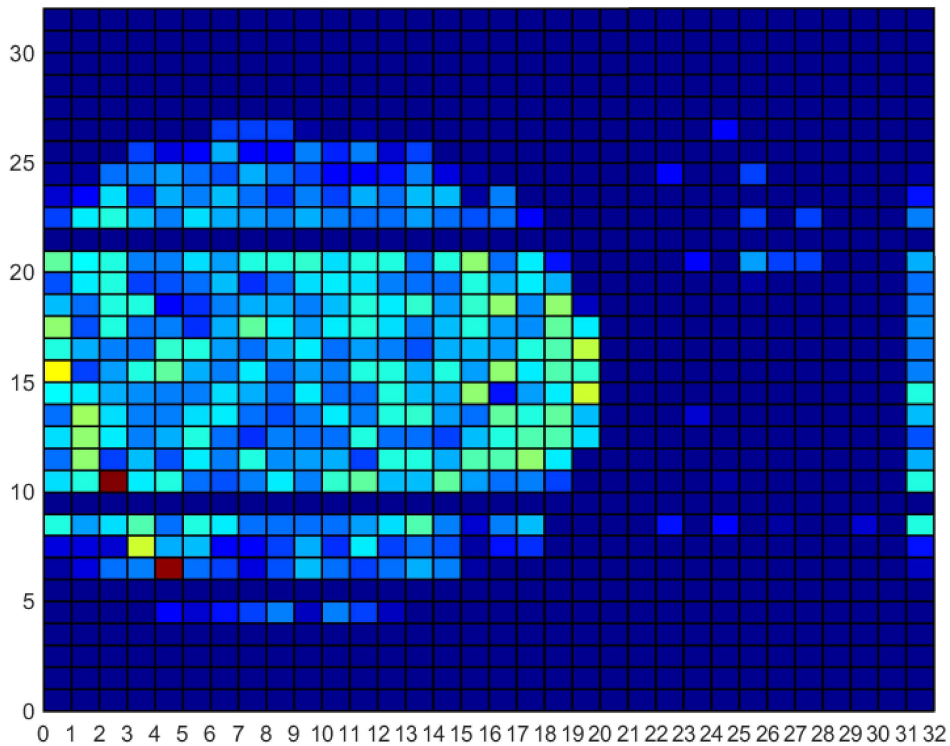


Figura 5.7: Imagen de presión de una pelota

5.2.4. Mano

Por último, se coloca la mano en posición lateral con el puño cerrado, tal y como se colocaría para agarrar un vaso. El resultado se observa en la Figura 5.8.

- Se aprecia la forma de la mano, así como las zonas en las que se ejerce más presión, como son, el nudillo del dedo índice (columnas de la 21 a la 26, filas de la 20 a la 29) y la zona de la muñeca (columnas de la 1 a la 4, filas de la 5 a la 15).
- Existe ruido en la señal, especialmente en la última columna, como en el caso anterior.
- Se aprecian datos erróneos en la columna 16, última columna del primer multiplexor.
- No se transmiten datos de la fila 22, como en los casos anteriores.

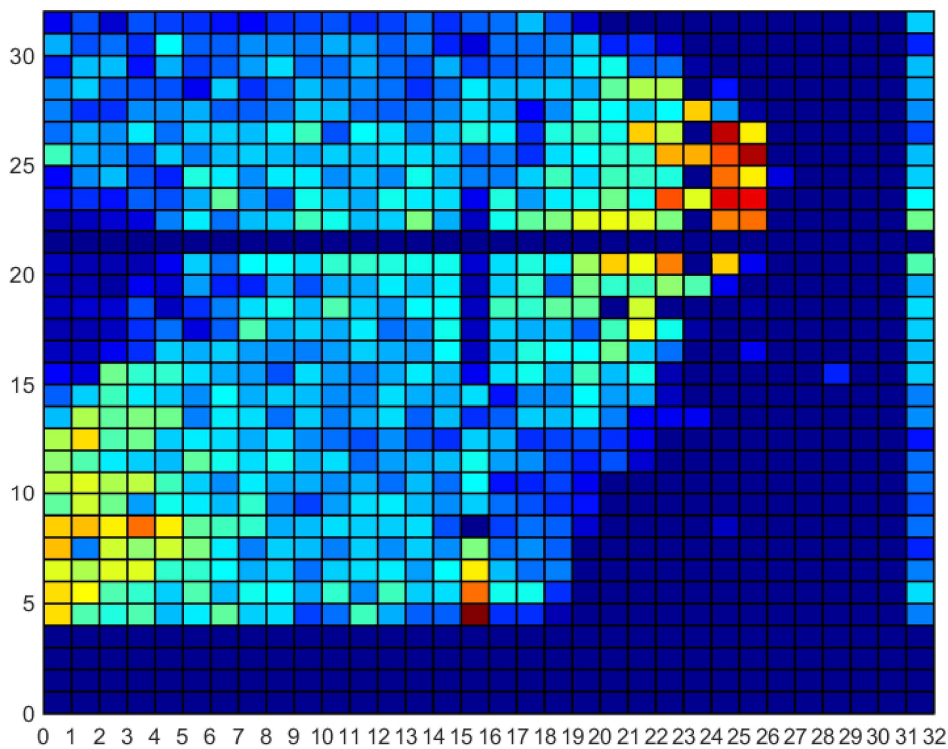


Figura 5.8: Imagen de presión de una mano

5.3. Conclusiones

Una vez se han obtenido los resultados, podemos concluir que el sistema desarrollado cumple su función principal: es capaz de generar imágenes de la presión ejercida por un objeto en el sensor táctil, mediante las cuales puede interpretarse la forma del objeto; lo que, para tratarse de un primer prototipo, resulta satisfactorio.

Al obtener los resultados, se observa que la velocidad de transmisión debería aumentarse, y que sería adecuado modificar el divisor de tensión para que el sensor sea capaz de medir presiones bajas, ya que hay que ejercer una presión considerable para que los resultados sean fiables. Además, sería adecuado calibrar el sensor al inicio de cada lectura de datos.

Capítulo 6

Conclusiones y trabajos futuros

6.1. Conclusiones

En este proyecto se ha presentado el diseño y fabricación de un sistema capaz de adquirir imágenes de presión a través de un sensor táctil resistivo.

- Se ha programado un Firmware básico capaz de transmitir los datos medidos y convertirlos en imágenes de presión que, sin embargo, necesita ser mejorado para su optimización.
- Se ha conseguido un Hardware que permite la adquisición de los datos de forma correcta, sin errores importantes en el diseño y de bajo coste.
- Se observan, sin embargo, errores de soldadura, abundante ruido en la medida y datos erróneos.
- El diseño de la estructura de conexión realizada con la impresora 3D resulta adecuado, ya que asegura el contacto del sensor con los conectores y compacta el dispositivo.
- No se consigue la misma precisión y velocidad que con el modelo comercial del que se disponía.

6.2. Trabajos futuros

Se ha realizado un proyecto mecatrónico completo que, a pesar de haber obtenido buenos resultados aun tratándose de un primer prototipo, necesita de mejoras para poder desempeñar la actividad para la que ha sido diseñado. Esos objetivos de mejora son:

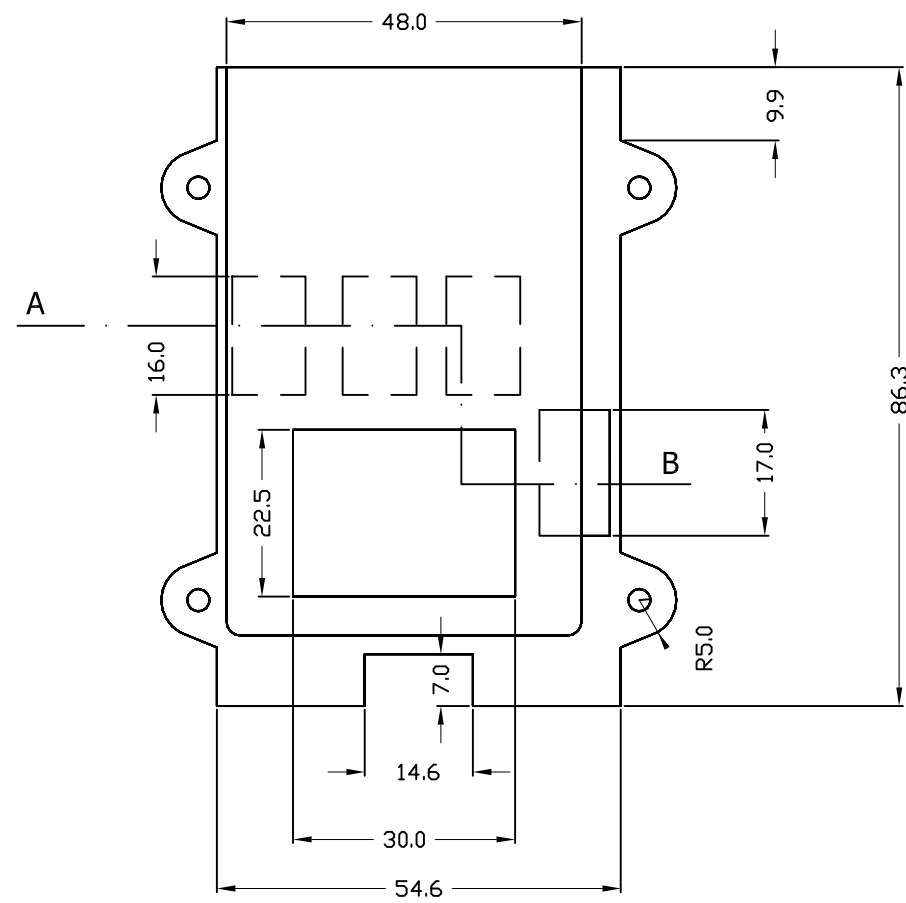
Firmware

- Optimizar el código Arduino para conseguir una mayor velocidad de transmisión de los datos.
- Incorporar un programa para la calibración del sensor.
- Mejorar las imágenes de Matlab, mediante interpolación de los datos para corregir posibles datos incorrectos y suavizar el pixelado.

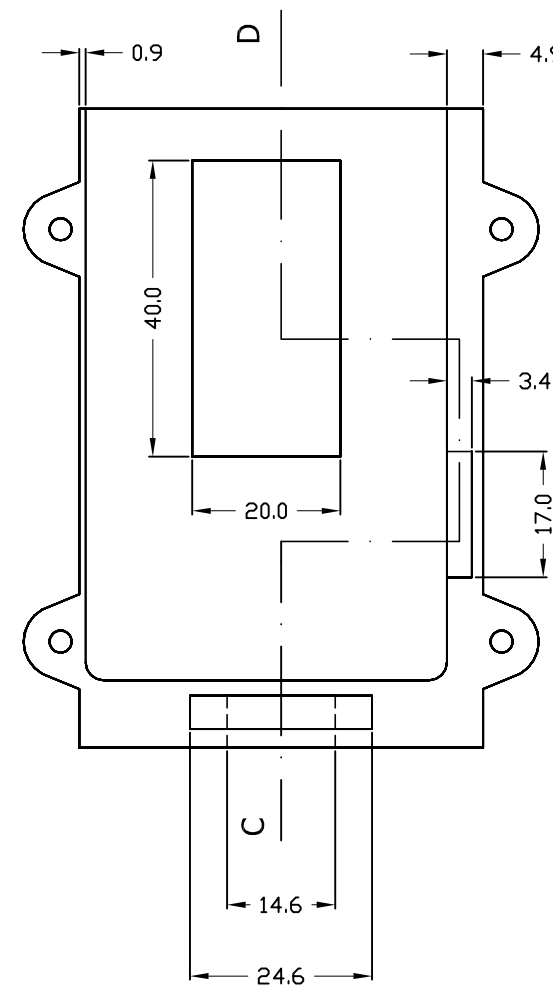
Hardware

- Minimizar los errores de soldadura con el objetivo de minimizar el ruido de la señal.
- Mejorar el diseño del dispositivo para hacer más fácil la apertura del mismo, y por tanto, la introducción del sensor en él.
- Compactar el dispositivo, incorporando el microcontrolador en una de las PCBs, o utilizando un Arduino Nano que se pueda acoplar encima de las mismas.
- Diseñar y fabricar mediante impresión 3D una carcasa que revista y proteja todo el dispositivo (las placas ya existentes y el microcontrolador).

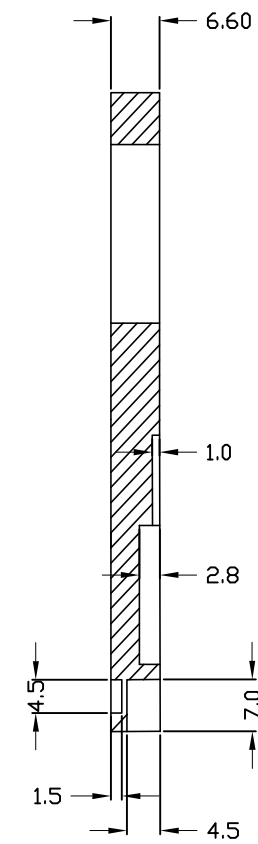
Estructura para cara superior



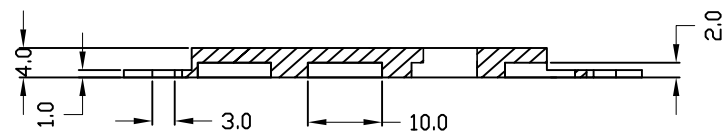
Estructura para cara inferior



Corte CD



Corte AB



Cotas en mm



UNIVERSIDAD DE MÁLAGA

SISTEMA DE ADQUISICIÓN DE IMÁGENES DE PRESIÓN PARA UN SENSOR TÁCTIL MATRICIAL RESISTIVO

María Fernández Hijano

PLANO ESTRUCTURA DE CONEXIÓN

PLANO

01

ESCALA:
1/1
GITI

Fdo. María Fernández Hijano

SEP-2018

Bibliografía

- [1] Arduino.
<https://www.arduino.cc/>
- [2] Autodesk; “Eagle”.
<https://www.autodesk.com/products/eagle/overview>
- [3] Brimacombe JM, Wilson DR, Hodgson AJ, Ho KC, Aanglin C.; “Effect of Calibration Method on Tekscan Sensor Accuracy.” *ASME. J Biomech Eng.* 2008;131(3):034503-034503-4.
- [4] ControlReal; “Sensores. Definición y características.”
<http://controlreal.com/es/sensores-definicion-y-caracteristicas/>
- [5] P. Dario; “Tactile sensing: Technology and applications, Sensors and Actuators”. *A: Physical, Volume 26, Issues 1–3, 1991, Pages 251-256, ISBN 0924-4247.*
- [6] Datalight; “Análisis del funcionamiento del sensor de fuerza resistivo (FSR) con LabView”.
http://www.datalights.com.ec/site2/images/stories/robotica/nap/nap_fsr.pdf
- [7] D. De Rossi and C. Domenici, “Piezoelectric properties of dry human skin,” *IEEE Trans. Electr. Insul.*, vol. EI-21, no. 3, pp. 511–517, Jun. 1986.
- [8] J.M. Gandarias Palacios y J.M. Gómez de Gabriel, “Aplicaciones de Sensores Táctiles a la Robótica de Rescate” *Universidad de Málaga, 2017*
- [9] R. S. Johansson and A. B. Vallbo, “Tactile sensibility in the human hand: Relative and absolute densities of four types of mechanoreceptive units in glabrous skin,” *J. Physio.*, vol. 286, pp. 283–300, 1979.
- [10] Mathworks; “Matlab Answers: colormap for a matrix”
<https://es.mathworks.com/matlabcentral/answers/372161-colormap-for-a-matrix>
- [11] Murin, e.l., et al; “Calibration issues of Tekscan systems for human pressure assessment.” *Queen’s Univ. Kingston (Ontario), 2001.*

- [12] National Instruments; “¿Qué es Adquisición de Datos?”
<https://www.ni.com/data-acquisition/what-is/esa/>
- [13] L. Paredes Madrid, “Modelado de sensores piezoresistivos y uso de una interfaz basada en guantes de datos para el control de impedancia de manipuladores robóticos,” *Universidad Complutense de Madrid*, 2014
<https://eprints.ucm.es/25352/1/T35345.pdf>
- [14] Ravinder S. Dahiya, Giorgio Metta, Maurizio Valle, and Giulio Sandini. “Tactile sensing from humans to humanoid.” *IEEE Transactions on Robotics*, 26(1):1–20, 2 2010.
- [15] Sensitronics; “Tutorials”.
<https://www.sensitronics.com/tutorials.php>
- [16] SketchUp.
<https://www.sketchup.com/es>
- [17] Tekscan; “I-Scan Product Selection Guide”.
<https://www.tekscan.com/resources/product-guide/i-scan-product-selection-guide>
- [18] Wikipedia; “Sensor”.
<https://es.wikipedia.org/wiki/Sensor>
- [19] Wikipedia; “Adquisición de datos”.
https://en.wikipedia.org/wiki/Data_acquisition
- [20] GitHub; “Librería Arduino ShiftRegister74HC595.h”.
<https://github.com/Simss0/ShiftRegister74HC595>
- [21] 74HC4067 DataSheet.
https://assets.nexperia.com/documents/data-sheet/74HC_HCT4067.pdf
- [22] 74HC595D DataSheet.
https://assets.nexperia.com/documents/data-sheet/74HC_HCT595.pdf

Apéndice A

Código Arduino

```
/*
   Programa Arduino para la adquisición de datos de un sensor tactil
   matricial de 32x32 puntos.

   Autor: María Fernández Hijano
*/
#include <ShiftRegister74HC595.h>

int Baud_Rate = 19200;
//Asignación de las entradas ADCs / COLUMNAS
int SIG1 = A0;
int SIG2 = A1;
int dato = SIG1;
byte cl, ch;
char cadena[65]; //La cadena tiene 65 caracteres por fila. 2
                 //caracteres por dato + el caracter de salto de línea ('\0')

int SER = 2; //Shift Register Data
int RCK = 3; //Shift Register Clock

//Asignación de pines de salida del multiplexor
int S0 = 8;
int S1 = 9;
int S2 = 10;
int S3 = 11;
int num = 0;

//Declaración
ShiftRegister74HC595 sr (4, 2, 3, 3); //(int numberOfShiftRegisters,
                                       int serialDataPin, int clockPin, int latchPin)
int CantidadDeShiftRegisters = 4;
int FilasPorSR = 8;
int NumeroDeFilas = 32;
int CantidadDeMultiplexores = 2;
int ColumnasPorMux = 16;
int NumeroDeColumnas = 32;
```

```

void setup() {
  Serial.begin(Baud_Rate);
  pinMode(SIG1, INPUT);
  pinMode(SIG2, INPUT);
  pinMode(SER, OUTPUT);
  pinMode(RCK, OUTPUT);
  pinMode(S0, OUTPUT);
  pinMode(S1, OUTPUT);
  pinMode(S2, OUTPUT);
  pinMode(S3, OUTPUT);
}

// Bucle principal para la lectura de datos de la matriz
void loop() {
  for (int j = 0; j < NumeroDeFilas; j++) {
    sr.set(j, HIGH);
    for (int i = 0; i < NumeroDeColumnas; i++) {
      activarColumna(i);
      //Convertimos los datos a hexadecimal
      //RANGO VÁLIDO [48,57]U[97,102]
      cl = 48 + (dato & 0x0F); // El caracter ASCII 48: '0'
      ch = 48 + ((dato >> 4) & 0x0F);
      cadena[i * 2] = ch;
      cadena[i * 2 + 1] = cl;
      // Si el caracter es mayor al 57, tendríamos caracteres
      especiales, sumamos 18 para llegar a la 'a' -> 97.
      if (ch > 57) cadena[i * 2] += 39;
      if (cl > 57) cadena[i * 2 + 1] += 39;
    }
    sr.set(j, LOW);
    cadena[65] = '\0'; //Salto de línea
    Serial.print(cadena);
    Serial.println();
  }
  Serial.end();
}

void activarColumna(int columna) {
  if (columna < 16) {
    dato = analogRead(SIG1);
    readMux(columna);
  }
  if (columna >= 16) {
    dato = analogRead(SIG2);
    readMux(columna);
  }
}

// Función que lee el dato correspondiente
int readMux(byte channel) {
  digitalWrite(S0, bitRead(channel, 0)); // bitRead(x,n) reads a bit
  of a number.
  digitalWrite(S1, bitRead(channel, 1));
}

```

```
digitalWrite(S2, bitRead(channel, 2));  
digitalWrite(S3, bitRead(channel, 3));  
}
```

Apéndice B

Código Matlab

```
%% Lectura de datos por el puerto serie
% Borrar previos
delete(instrfind({'Port'},{'COM4'}));
% Puerto serie. Comprobar el actual COM
s = serial('COM4');
% Velocidad de transmisión. Debe coincidir con la de la configuración
  n de Arduino
set(s, 'BaudRate', 19200);
get(s);
% Abrir puerto serie
fopen(s);

%% Inicialización
A_recibida = zeros(32);
A_ordenada = zeros(32,32);

%% Lectura de datos
for i=1:32 %% Supongo que hay 32 líneas
    linea=fgetl(s);
    disp(linea);
    for j=1:32 %% y que cada línea contiene 32 bytes en Hexadecimal
        A_recibida(i,j)=(linea((j-1)*2+1)-48)*16+linea((j-1)*2+2)-48;
    end
end

%% Ordenación de la matriz
A_ordenada([2 4 12 10 6 14 16 18 22 24 20 26 28 30 32 7 8 19 27 31
    25 29 21 23 1 15 11 17 13 9 5 3],[16 15 14 13 12 11 10 9 8 7 6 5
    4 3 2 1 32 31 30 29 28 27 26 25 24 23 22 21 20 19 18 17])=
    A_recibida(:,:);
A_ordenada=flipud(A_ordenada)

%% Imagen de colores
figure
```

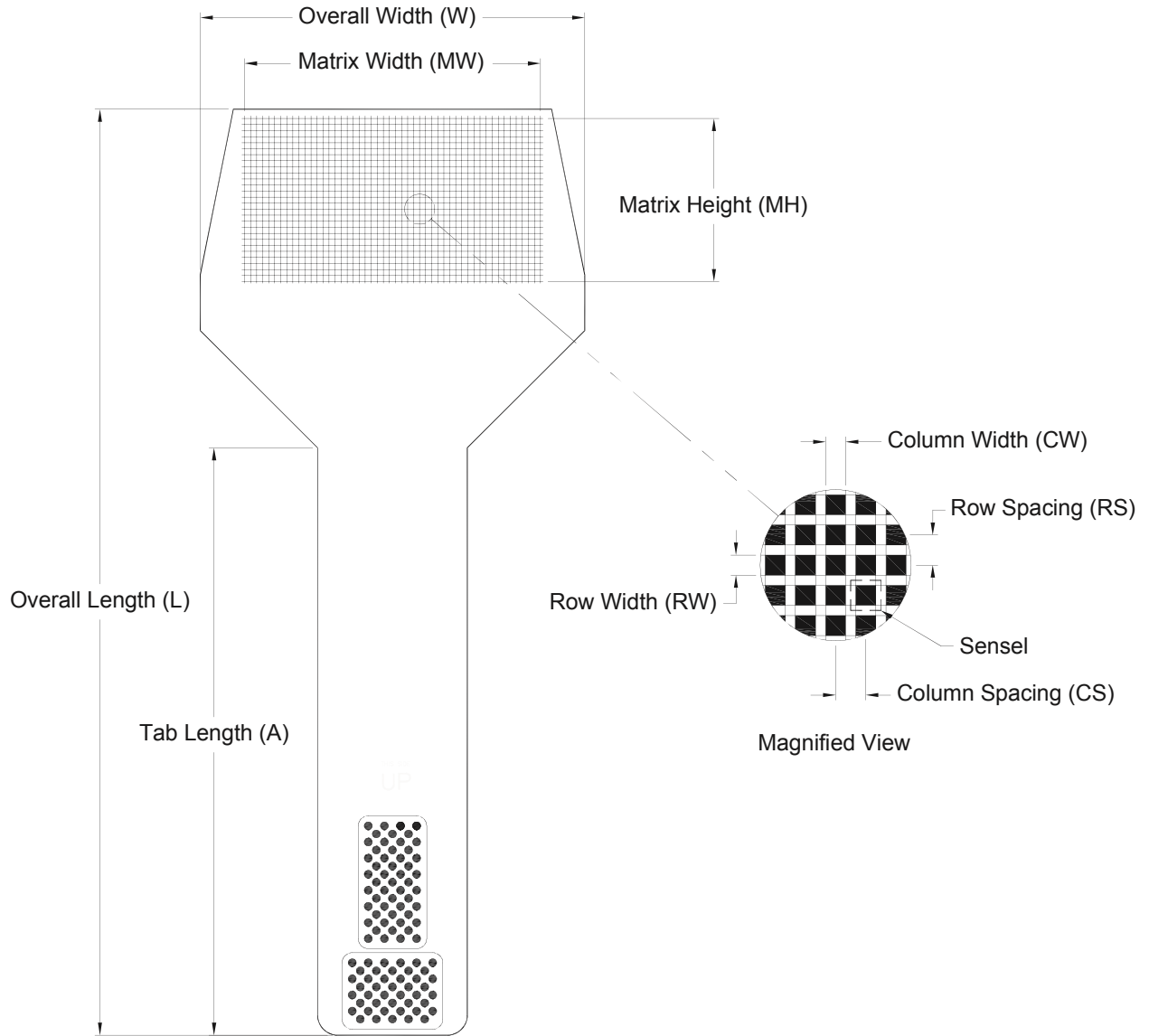


```
[nf,nc] = size(A_ordenada);
hold on
for i=1:1:nc % Bucle necesario para crear los píxeles y la
    superficie con los datos recibidos y ordenados
    for j=1:1:nf
        C = [0 1; 0 1];
        F = [nf-1 nf-1; nf nf];
        C = C + (i-1);
        F = F - (j-1);
        B = ones(2,2)*A_ordenada(j,i);
        surface(C,F,B);
    end
end
axis([0 nc 0 nf])
colormap(jet) % Se crea la imagen de colores
set(gca, 'XTick', 0:1:nc)
set(gca, 'FontSize', 6)
```

Apéndice C

Hoja de especificaciones Sensor Tekscan 6077

PRESSURE MAPPING, FORCE MEASUREMENT, AND TACTILE SENSORS



| General Dimensions | | | | | Sensing Region Dimensions | | | | | | Summary | |
|----------------------------|---------------------------|------------------------|---------------------------|----------------------------|---------------------------|--------------------|-------------|-----------|--------------------|-------------|----------------------|---------------------------|
| Overall Length <i>L</i> | Overall Width <i>W</i> | Tab Length <i>A</i> | Matrix Width <i>MW</i> | Matrix Height <i>MH</i> | Columns | | | Rows | | | Total No. of Sensels | Sensel Spatial Resolution |
| | | | | | <i>CW</i> | Pitch <i>CS</i> | <i>Qty.</i> | <i>RW</i> | Pitch <i>RS</i> | <i>Qty.</i> | | |
| (mm) | (mm) | (mm) | (mm) | (mm) | (mm) | (mm) | | (mm) | (mm) | | | (sensel per sq-cm) |
| 292.1 | 121.3 | 186.8 | 95.3 | 53.3 | 1.3 | 1.9 | 50 | 1.3 | 1.9 | 28 | 1,400 | 27.6 |
| (in) | (in) | (in) | (in) | (in) | (in) | (in) | | (in) | (in) | | | (sensel per sq-in) |
| 11.50 | 4.78 | 7.35 | 3.75 | 2.10 | 0.050 | 0.075 | 50 | 0.050 | 0.075 | 28 | 1,400 | 177.8 |

| Pressure Ranges | |
|-----------------|----|
| kPa | 34 |
| psi | 5 |